Data Analytics and Smart City Operations

by

Sheng Liu

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Industrial Engineering and Operations Research

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Zuo-Jun (Max) Shen, Chair
Professor Dorit Hochbaum
Professor Phil Kaminsky
Professor Terry Taylor

Summer 2019

Data Analytics and Smart City Operations

Abstract

Data Analytics and Smart City Operations

by

Sheng Liu

Doctor of Philosophy in Engineering - Industrial Engineering and Operations Research

University of California, Berkeley

Professor Zuo-Jun (Max) Shen, Chair

This thesis presents models and algorithms that leverage data analytics and optimization approaches to address the challenges in traditional manufacturing systems as well as emerging smart city contexts. The increasing availability of data sources and computing power has created the need for smarter data-driven decision making models, which motivates us to explore various ways of integrating statistics and optimization tools. In particular, we focus on analyzing practical operational problems with real-world data sets. Chapter 2 is devoted to improving yield prediction accuracy in integrated circuit manufacturing using the wafer map data. We propose an innovative yield prediction model, called adjacency-clustering, to capture the neighborhood effect that is often present in the wafer map. The adjacency-clustering model can be solved efficiently and deliver superior prediction performance than the state-of-the-art methods. Chapter 3 studies the order assignment problem in urban last-mile delivery services. We propose a framework to integrate travel time predictors with assignment optimization models to capture the drivers' practical routing behaviors. The proposed framework yields tractable formulations compatible with the existing stochastic and robust optimization tools. We further develop a branch-and-price algorithm to facilitate its real-time application. The real-world case study demonstrates the substantial benefits of applying our framework. Chapter 4 addresses the urban bike lane planning problem based on the real-world GPS trajectory data. We present a flexible optimization model based on the cyclists' utility functions. We analyze the problem structure and propose efficient algorithms to solve the bike lane planning model. We perform extensive numerical experiments on a real-world trajectory data set to validate the performance of our models and algorithms. The numerical study also generates managerial insights to help the city managers improve their bike lane planning decisions.

To Liang and Yaping.

Contents

List of Figures

List of Tables

Acknowledgments

I am grateful to my advisor, Professor Zuo-Jun Max Shen, for his continuing support of my research and career development. His guidance is invaluable and motivates me to study impactful and important problems. I am also indebted to Professor Dorit Hochbaum for her advice in my doctoral studies. Through many commmunications with her, I have developed important research skills and critical thinking abilities. I would also like to thank Professor Phil Kaminsky and Professor Terry Taylor for their incredible mentorship and services on my committee. Furthermore, I want to thank Professor Candace Yano, Professor Paul Grigas, Professor Anil Aswani, and Professor Zeyu Zheng for the insightful conversations.

Studying in Berkeley is a challenging but rewarding experience. I feel very fortunate to have many good friends there who accompanied and helped me during this great journey. In the collegial and warming research lab, I have many fun memories with Wei Qi, Siyuan Sun, Min Zhao, Jiung Lee, Qiaochu He, Auyon Siddiq, Yanqiao Wang, Ying Cao, Chao Mao, Junyu Cao, Meng Qi, Mengxin Wang, Cristobal Pais, and Hongcai Zhang. Within the department, I also spent memorable times with Nan Yang, Hao Fu, Renyuan Xu, Quico Spaen, Rebecca Sarto-Basso, Cheng Lyu, Shiman Ding, Dan Bu, Erik Berteli, Yonatan Mintz, Jiaying Shi, Haoyang Cao, Xu Rao, Darren Lin, and Heyuan Liu. Outside the department, I would like to thank Zhe Ji, Yulin Liu, Yujia Wu, Yao Cai, Manfei Wu, Xiaoxuan Hou, Mogeng Yin, Wei Ni, Xiaofei Zhou, Letian Wang, and Hong Shang for their support.

I also appreciate the help and advice from my coauthors/coworkers inside and outside Cal: Long He, Tianhu Deng, Shixuan Zhang, Deepak Rajan, Muhong Zhang, Felix Cheng, Siyang Xie, Kaibo Wang, Jeremy Karp, Chris Sholley, Xiang Ji, and among others. Also, I feel lucky to have long-time friends standing behind me: Wenzhe Li, Ye Zhang, Weixiang Yuan, Tianjun Chen, and Tuo Wu.

I would like to thank my girlfriend, Yuanyuan Pan, for her understanding and support during my PhD study and academic job search. She is caring and intelligent.

Lastly, I want to thank my parents and grandparents for their endless love.

Chapter 1

Introduction

Recent years have witnessed the rapid development of data analytics technologies thanks to the accelerating advances in data storage and computing power. The widespread use of mobile devices and sensors has enabled collection of a huge amount of data about machines and human, which creates many challenging and interesting research questions regarding making better decisions from those massive data sources. In particular, it is critical for the operations research and operations management scholars to develop a more data-driven way of solving operational problems in contexts such as manufacturing and logistics systems.

One of the many emerging research directions driven by the data analytics is smart city operations. As more and more people live in cities, city residents are facing severe challenges in their living environment such as air pollution, traffic congestion, among others. Since data alone can not solve these problems, developing a data-driven optimization framework to better integrate data with the decision-making process for city management is of great importance.

This thesis presents three papers that explore different data-driven optimization approaches to address the challenges in traditional manufacturing systems and emerging smart city operations.

In Chapter 2, we study the yield prediction problem in integrated circuit manufacturing using the wafer map data. It is known that defects tend to be clustered and a chip is likely to be defective if its neighbors are defective. This neighborhood effect is not well captured in traditional yield modeling approaches. We propose a new yield prediction model, called adjacency-clustering which addresses, for the first time, the neighborhood effect, and delivers prediction results that are significantly better than state-of-the-art methods. Adjacency-clustering (AC) model is a form of the Markov random field minimum energy model (MRF) that is primarily known in the context of image segmentation. AC model is a novel use of MRF for identifying defect patterns that enable diagnosis of failure causes in the manufacturing process. In this paper we utilize the defect patterns obtained by the AC model for yield prediction. We compare the performance of the AC model to that of leading yield prediction models including the Poisson, the negative binomial, the Poisson regression and negative binomial regression models, on real datasets and on simulated datasets. The results demonstrate that the adjacency-clustering model captures the neighborhood effect and delivers superior prediction accuracy. Moreover, the concept and methodology of adjacency-clustering are not limited to integrated circuit manufacturing. Rather, it is applicable in any context where a neighborhood effect is present, such as disease risk mapping and energy consumption prediction. The contents of this chapter correspond to the material in our paper (Hochbaum and S. Liu, 2018).

In Chapter 3, we study how delivery data can be applied to improve the on-time performance of urban last mile delivery services. Motivated by the delivery operations and data of a food delivery service provider, we discuss a framework that integrates the travel time predictors with the order assignment optimization. Such integration enables us to capture the driver's routing behavior in practical urban environment, where the driver's decision-making process is often unobservable or intricate to model. Focusing on the order assignment problem as an example, we discuss the classes of tractable predictors and prediction models that are highly compatible with the existing stochastic and robust optimization tools. We further provide reformulations of the integrated models, which can be efficiently solved with the proposed branch-and-price algorithm. Moreover, we propose two simple heuristics for the multiperiod order assignment problem, which are built upon the single-period solutions. Using the delivery data, our numerical experiments on a real-world application not only demonstrate the superior performance of our proposed order assignment models with travel time predictors, but also highlight the importance of learning the behavioral aspects from the operational data. We find that large sample size does not necessarily compensate for the misspecification of the driver's routing behavior. The contents of this chapter correspond to the material in our paper (**liu2018data**).

In Chapter 4, we study the urban bike lane planning problem based on the bike trajectory data from bike sharing systems. The key decision is where to build bike lanes in the existing road network. As bike sharing systems are widespread in the metropolitan areas across the world, bike lanes are being planned and constructed by many city governments to promote cycling and protect cyclists. Traditional bike lane planning approaches often rely on surveys and heuristics. We develop a general and novel optimization framework to guide the bike lane planning from bike trajectories. We formalize the bike lane planning problem from the view of the cyclists' utility functions and then derive an integer optimization model to maximize the utility. We offer several structural results about our model, and prove the Lagrangian dual is polynomial-time solvable. We then develop tractable formulations and efficient algorithms to solve the large-scale optimization problem. Via a real-world case study with a city government, we demonstrate the efficiency of the proposed algorithms and quantify the trade-off between the coverage of bike trips and the continuity of bike lanes. We show how the topology would evolve according to the utility functions and highlight the importance of understanding cyclists' responsive routing behaviors. The proposed framework drives the data-driven urban planning scheme in smart city operations management.

Chapter 2

Adjacency-Clustering for Yield Prediction

2.1 Background and Motivation

Integrated circuit manufacturing is a highly complex, costly process that involves hundreds of chemical or physical processing steps (Yuan, Ramadan, and Bae 2011). The key processes include wafer fabrication, wafer probe, assembly or packaging and final test. The degree of manufacturing success is measured by *yield*, which is defined as the average ratio of the number of usable devices that pass tests after completing processes to the number of potential usable devices before starting processes (T. Kim and Kuo, 1999; Ferris-Prabhu, 1992). Accurate yield prediction is critical for managers to estimate productivity, production cost and make scheduling decisions. Moreover, yield prediction helps to detect processing problems in an early production stage, which is crucial to quality improvement.

In semiconductor manufacturing, there are four components to the yield: wafer process yield, wafer probe yield, assembly yield and final test yield (Milor, 2013). Among these, wafer process yield (also known as line yield) and wafer probe yield (also known as die yield) are considered to be the major cost determining factors (Cunningham, 1990). Wafer probe defects found in integrated circuits (also called chips) include shorts, opens, misalignment, photoresist splatters and flakes, and pinholes (Charles H Stapper, Armstrong, and Saji 1983). A chip containing at least one fatal defect is considered defective, and "good" otherwise.

Yield prediction based on defect data from sampled wafers is to estimate the ratio of non-defective (good) chips to the total number of chips. Till now only statistical models have been utilized for this purpose. The classical yield model assumes the number of defects on a chip follows Poisson distribution with density λ, taken to be the average number of defects on a chip. It is assumed that the value of λ is uniform across all chips and wafers. The yield is then estimated as the probability that no defects occurs on a chip. In later work researchers relax the assumption of the constant λ and assume λ itself follows a specific distribution. Two popular such models are Murphy's model and Seeds' model proposed in

Murphy (1964) and Seeds (1967), respectively. In addition, negative binomial distribution and variants of Poisson distribution have been applied to improve yield prediction in recent years (see, e.g., Bae, Hwang, and Kuo 2007).

Existing models assume that the distribution of defects is identical for all chips on the wafer. Yet this is not the case in practice where defects are known to be clustered in contiguous groups (Bae, Hwang, and Kuo 2007; M. H. Hansen, Nair, and D. J. Friedman 1997). Indeed, various mechanisms causing defects tend to only affect certain regions of the wafer (Hwang and Kuo 2007; Y.-S. Jeong, S.-J. Kim, and M. K. Jeong 2008; Charles H Stapper, Armstrong, and Saji 1983). Chips in close vicinity of each other are more likely to be affected by the same defect generating mechanism, and therefore the number of defects on a chip is correlated with the number of defects on its neighbors.

The adjacency-clustering approach introduced here is to partition the set of chips on the wafer to subsets, referred to as clusters, so that each cluster contains chips with similar defect level, which also tend to be adjacent to each other. This is attained by minimizing a combination of two objective functions, one that penalizes deviation from the priors (observed number of defects), and the second that penalizes the separation of adjacent chips to different clusters. The resulting clusters tend to contain chips with the same defect distribution since, because they reside in the same neighborhood, they are likely to be caused by the same mechanism. The wafer yield prediction is then attained from a combination of the individual cluster yields. This approach is in contrast to existing yield prediction methods that predict the wafer yield without differentiating among clusters. The performance of our approach is demonstrated via an empirical study on real data sets and on simulated data sets. The results show that the adjacency-clustering approach improves the prediction accuracy by a factor between 3 and 15 as compared to the use of Poisson and Poisson regression model for the real data sets. This superior performance of adjacency-clustering over the state-of-the-art methods is further validated on simulated data.

The success of the adjacency-clustering approach for yield prediction bodes well to its potential applicability to other contexts where the neighborhood effect is an important factor in clustering. This is the case for disease mapping where spatially correlated disease data are utilized to identify high-risk areas (clusters) and predict risk levels (see Charras-Garrido et al. 2012 for more details). Another case is that of energy consumption prediction for households where high consumption households tend to be in the vicinity (see, e.g., Baker and Rylatt 2008).

2.2 Related Literature

Relevant literature is reviewed here within three streams: (1) Advanced statistical yield prediction models; (2) Methods for measuring the extent of spatial aggregation of defects on wafers given defect counts observations; (3) Identifying and classifying defect spatial patterns in a wafer.

Advanced statistical yield prediction models. Among statistical yield models the Poisson model is most widely used. A drawback of the Poisson model is that it is known to considerably underestimate the yield for wafers with defects that are aggregated non-uniformly (see, e.g., Charles H Stapper, Armstrong, and Saji 1983 and Charles H. Stapper 1989). To overcome this limitation, Stapper (1983) derives a negative binomial model by assuming the probability that a defect occurs in a chip depends on the number of faults already on the chip, which is equivalent to assuming that λ follows a gamma distribution. Albin and D. J. Friedman (1989) introduce an alternative distribution, Neyman distribution, to fit the defect data. I. Koren, Z. Koren, and Stepper (1993) add a new parameter, block size, to the negative binomial model to account for the aggregation effects of defects. Although the negative binomial model and the Neyman model capture the defect aggregation, they fail to model the spatial information of chips and relationship between adjacent chips. For instance, these models ignore a common defect pattern where defects tend to be aggregated on the periphery of wafers, called *radial loss* (Ferris-Prabhu et al. 1987). To account for such spatial position effects, regression models (generalized linear models) are introduced: Bae, Hwang, and Kuo (2007) propose Poisson, negative binomial and zero-inflated Poisson regression model. Yuan, Ramadan, and Bae (2011) introduce zero-inflated binomial negative model. Among these regression models, negative binomial regression model yields the lowest prediction error in general. More recently, Krueger and Montgomery (2014) introduce generalized linear mixed models for yield modeling to capture longitudinal correlation between and within batches of samples. However, they only explore the longitudinal correlation but ignore neighborhood effect within wafer. Although these regression models improve yield prediction accuracy, estimation issues remain challenging. Two main estimation methods, Bayesian method and maximum likelihood method, are employed in the parameter estimation of regression models. Bayesian methods based on Markov chain Monte Carlo (MCMC) are time consuming and unstable for small-size samples while maximum likelihood estimation methods may not provide tight interval estimate for parameters (Ghosh, Mukhopadhyay, and J.-C. J. Lu 2006). In addition, for samples showing complicated spatial patterns, it is challenging to choose appropriate covariates and set up the linear relationship in regression models.

Measuring spatial clustering. M. H. Hansen, Nair, and D. J. Friedman (1997) introduce a monitoring statistic to test the significance of spatial clustering based on Markov random field. Fellows, Mastrangelo, and White Jr (2009) study the empirical performance of Hansen et al.' method on real data sets. Hansen et al. (1997) also propose the join-count statistics in order to measure the spatial randomness and the degree of clustering, where join is formed with two neighboring chips and join counts are measures of the adjacencies between different levels of a variable. Taam and Hamada (1993) utilize the join-count to propose the log odds ratio as a measure of spatial clustering. Y.-S. Jeong, S.-J. Kim, and M. K. Jeong (2008) further generalize join-count based statistics with optimal weights, and introduce the spatial correlogram to detect the presence of spatial autocorrelation. There are other statistics known that can be used as defect clustering indices, as reviewed in Tsai, Tong, and C.-H. Wang (2008). All studies that measure spatial clustering are based on binary defect data, where chips are differentiated only in whether or not they contain defects. These methods highlight the importance of spatial clustering but do not apply for the yield prediction tasks addressed here.

Classifying defect patterns. Classifying defect patterns is important for the purpose of diagnosis of failure causes. F.-L. Chen and S.-F. Liu (2000) employ neural networks in order to recognize spatial defect patterns. Di Palma et al. (2005) test the approach of Chen and Liu on simulated and real data set. White Jr, Kundu, and Mastrangelo (2008) develop a procedure to detect different arrangements and shapes of defect aggregations (clusters). Recently, several recognition techniques based on support vector machines (SVM) have been tested on wafer defect data to identify different defect patterns (e.g. see T.-S. Li and Huang 2009, Chao and Tong 2009, Yuan, Bae, and J. I. Park 2010 and M.-J. Wu, Jang, and J.-L. Chen 2015). Ooi et al. (2013) develop an automatic defect pattern recognition system integrating feature extraction, selection and classification techniques. These methods are helpful in diagnosis but they do not explore how defect patterns can help yield prediction. To the best of our knowledge, our paper is the first attempt to utilize defect clustering pattern to improve yield prediction result.

2.3 Our Approach

The defect data available in semiconductor manufacturing is in the form of wafer maps. A wafer map example is given in Figure 1, where the number of defects on each chip is indicated at the chip position on the grid. Let the defect data for a wafer map on n chips be represented by the array $(d_1, d_2, \ldots, d_n)$, where d_i is the observed number of defects at location i or the i-th chip. The wafer map is formalized as a graph $G = (V, E)$ where each node in V represents a chip and each pair, i, j, of neighboring chips is associated with an edge $[i, j] \in E$. There are several alternatives for neighborhood relationship, e.g. 4-neighbor system (rook-move neighborhood) and 8-neighbor system (king-move neighborhood). We select here the 4-neighbor system.

The goal of adjacency-clustering is to partition the set of chips into clusters, so that the chips that belong to the same cluster tend to have similar defect levels as well as tend to be adjacent to each other. These two goals are balanced by a parameter that weighs one goal versus the second. The clustering is represented by cluster label x_i assigned to chip i. One goal is to require that the chip label, x_i for chip i, deviates as little as possible from the observed value d_i, under a penalty called *deviation cost*. For the second goal there is a penalty associated with the difference in assigned labels for neighboring chips. This penalty, called *separation* penalty, is associated with each pair of adjacent chips, or nodes in $G = (V, E)$ that are linked with an edge of E, that differ in their labels. The goal is to attain a solution that minimizes a combination of the two objectives of deviation and separation

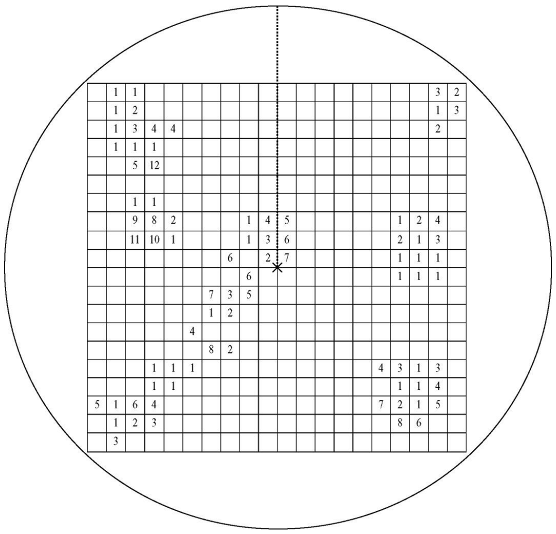


Figure 2.1: A wafer map example used by Bae, Hwang, and Kuo (2007)

penalties. Let $f_i(x_i, d_i)$ be deviation functions associated with node $i \in V$ and $g_{ij}(x_i - x_j)$ be separation functions associated with every edge $[i,j] \in E$. Let X be a set of cluster label values. The adjacency-clustering model (AC) is formulated as a *deviation-separation* optimization problem as follows:

$$\min \sum_{i \in V} f_i(x_i, d_i) + \sum_{[i,j] \in E} g_{ij}(x_i - x_j) \tag{2.1}$$

$$s.t. \quad x_i \in X \quad \forall\, i \in V. \tag{2.2}$$

This deviation-separation formulation arises in contexts such as computer vision and statistics, where it is referred to as Markov Random Field (MRF) (see, e.g., Blake and Zisserman 1987; Ishikawa and Geiger 1998; Hochbaum 2001).

All nodes that share the same label are considered to be a single cluster. The optimal cluster partition depends on the tradeoff between the deviation and separation penalties. The larger the separation penalty functions, the more contiguous the resulting clusters. In contrast, relatively large deviation penalties render clusters that group together objects with similar or identical observation values regardless of their spatial positions.

The label of a cluster corresponds to the *yield level* of the cluster. We choose integer cluster labels in $\{0, 1, \ldots, k\}$ where a higher label value indicates a greater likelihood of having large number of defects and thus a lower yield level. For example, we may choose the labels $\{0, 1, 2\}$, with the interpretation that the model predicts no defects for chips in the cluster labeled 0, moderate number of defects for chips in the cluster labeled 1, and high number of defects for chips in the cluster labeled 2. Another example is for the binary labels $\{0, 1\}$ implying a distinction between a cluster that tend to contain chips with very small number of defects or are surrounded by such chips, and a cluster that tend to contain chips with large number of defects. Such clusters are interpreted as non-defective versus defective clusters.

Our yield prediction model works by first generating the adjacency-clustering. The output of AC is a partition of the wafer's set of chips into $\{V_0, V_1, \ldots, V_k\}$, where V_0 is the cluster of chips that are labeled non-defective and V_j for $j = 1, \ldots, k$ are clusters that for larger label values are increasingly likely to contain larger number of defects. In the second stage a yield model is applied to each cluster, and the weighted average of cluster yields ($\hat{y}_j$ for $j = 0, \ldots, k$) is the reported wafer yield prediction $\hat{y}$:

$$\hat{y} = \frac{\sum_{j=0}^{k} |V_j| \hat{y}_j}{\sum_{j=0}^{k} |V_j|}.$$

For this second stage we use Poisson model and negative binomial model, or a mixture of the two, as a yield model applied to each cluster. We use the notation AC-Poisson, AC-NB, AC-PNB, and AC-NBP to indicate adjacency-clustering followed by: Poisson model, negative binomial (NB) model, a combination of Poisson for the non-defective clusters and NB for defective ones, and a combination of NB for the non-defective clusters and Poisson for defective ones, respectively. Table 2.1 lists the nomenclature for the models, whether using AC, and the respective yield model.

Table 2.1: Model names, clusters generated, if any, and yield models.

Model	Clusters generated	Yield model
AC-Poisson	$(V_0, V_1, \ldots, V_k)$	Poisson
AC-NB	$(V_0, V_1, \ldots, V_k)$	Negative binomial
AC-NBP	$(V_0, V_1, \ldots, V_k)$	V_0: negative binomial $V_1, \ldots, V_k$: Poisson
AC-PNB	$(V_0, V_1, \ldots, V_k)$	V_0: Poisson $V_1, \ldots, V_k$: negative binomial
Poisson	V	Poisson
Poisson regression	V	Poisson regression
Negative binomial	V	Negative binomial
Negative binomial regression	V	Negative binomial regression

2.4 The Adjacency-Clustering Model and Its Solution Technique

We first discuss the use of adjacency-clustering for yield prediction in the case of binary labels. We then discuss the choice of penalty functions for multi-label instances and present the solution technique for the resulting multi-label adjacency-clustering model.

2.4.1 Binary Adjacency-Clustering Model

For binary labels $\{0, 1\}$ the wafer is partitioned into only two clusters, one representing a set of defective chips that are labeled 1, and the second representing nondefective chips that are labeled 0. Let $V_0 = \{i \in V : d_i = 0\}$, $V_+ = \{i \in V : d_i > 0\}$. For chip $i \in V_0$ assigning $x_i = 0$ imposes no deviation cost whereas assigning $x_i = 1$ imposes a penalty of $w_{i0} > 0$. Likewise, for chip $i \in V_+$ assigning $x_i = 1$ imposes no deviation cost whereas assigning $x_i = 0$ incurs a penalty of $w_{i+} > 0$. With this notation, the total deviation cost (penalty) of assigning the x_i labels is,

$$\sum_{i \in V_+} w_{i+} + \sum_{i \in V_0} w_{i0} x_i - \sum_{i \in V_+} w_{i+} x_i. \tag{2.3}$$

Let the separation cost function be $g_{ij}(|x_i - x_j|)$, which equals to $u_{ij} > 0$ if $x_i \neq x_j$ and 0 otherwise. The optimization problem that minimizes the sum of these deviation and separation costs is (omitting the constant term $\sum_{i \in V_+} w_{i+}$),

$$\min \sum_{i \in V_0} w_{i0} x_i - \sum_{i \in V_+} w_{i+} x_i + \sum_{[i,j] \in E} u_{ij} z_{ij} + \sum_{[i,j] \in E} u_{ij} z_{ji}, \tag{2.4}$$

$$s.t.\ z_{ij} \geq x_i - x_j \ \forall\ [i,j] \in E, \tag{2.5}$$

$$z_{ji} \geq x_i - x_j \ \forall\ [i,j] \in E, \tag{2.6}$$

$$x_i \in \{0,1\} \ \forall\ i \in V,\ z_{ij} \in \{0,1\} \ \forall\ [i,j] \in E. \tag{2.7}$$

Here the constraints ensure that $z_{ij} = 1$ for adjacent nodes i and j if $x_i \neq x_j$ and 0 otherwise.

The above problem is the *minimum s-excess problem* (Hochbaum 2001), which is solved in polynomial time by applying a minimum-cut procedure on an associated graph. The optimal solution is a partition to two clusters, one of nodes of label 0, and the other of nodes of label 1. To allow for higher levels of differentiation between yield levels of clusters we present next the multi-label version of AC.

2.4.2 Multi-label Adjacency-Clustering Model

In the multi-label case we let the set of labels be $\{0, 1, \ldots, k\}$, where k is a parameter specified by the user. The choice of k implies there are $(k + 1)$ potential labels that characterize the yield level of each chip. For instance, if $k = 2$, a wafer is partitioned into three types of clusters: non-defective ($x_i = 0$), medium defective ($x_i = 1$) and highly defective ($x_i = 2$).

Selecting deviation and separation functions. For non-binary labels, the deviation and separation functions must be specified. For deviation functions we consider quadratic functions that correspond to Gaussian distribution in Bayesian estimation. Gaussian distribution is commonly used to approximate many distributions and the corresponding quadratic deviation functions are widely applied in image segmentation and spatial statistics (Panjwani and Healey 1995; K. Held et al. 1997; Håvard Rue 2001). When the observation is not Gaussian, batching and averaging observations can lead to an approximately Gaussian sample (Law 2014). Since we impose no restrictions on the probabilistic relationship between the number of defects and yield level, such quadratic functions are suitable. It is noted that the common use of quadratic deviation functions in the literature is due in part to the existence of well known algorithms, e.g. based on KKT conditions, that can be used for solving quadratic minimization problems. This is not the motivating reason in our case, for choosing quadratic deviation functions.

In terms of the separation functions, absolute value penalty $u_{ij}|x_i - x_j|$ (ℓ_1 norm) is commonly used to penalize the difference of neighboring labels. Occasionally, in image segmentation context, the truncated form $g_{i,j}(x_i - x_j) = u_{i,j} \cdot \min\{|x_i - x_j|, M\}$ for a positive value M, is considered desirable (Veksler 2007; Szeliski et al. 2008). This form avoids the over-smoothness associated with the absolute value penalty, which occurs for large gaps in label values. However, these truncated separation functions are nonlinear and the respective problem is NP-hard, and challenging even to approximate. Furthermore, in our set-up the label values are small integers and hence over-smoothness is not a concern.

We select here quadratic deviation functions and absolute value separation functions. For these functions the adjacency-clustering formulation is,

$$\text{(AC)} \quad \min \sum_{i \in V} (x_i - d_i)^2 + \sum_{i \in V} \sum_{j:[i,j] \in E} u_{ij}|x_i - x_j| \tag{2.8}$$

$$s.t. \quad x_i \in \{0, 1, \dots, k\} \quad \forall\ i \in V. \tag{2.9}$$

This AC problem is a MRF on convex separation and deviation functions. Such convex MRF is solved in polynomial time with the algorithm of Ahuja, Hochbaum, and Orlin (2003). For separation functions that are of the form $g_{i,j}(x_i - x_j) = u_{ij}|x_i - x_j|$ and convex deviation functions, the algorithm of Hochbaum (2001) is very efficient and provably fastest possible. The special structure of AC, with quadratic deviation functions, is shown next to be solved with a yet more efficient algorithm.

2.4.3 An Efficient Solution Technique for the Multi-label Adjacency-Clustering Model

We now describe a particularly efficient algorithm for solving AC with quadratic deviation functions and absolute value separation functions. The key to the efficiency of the algorithm is the threshold theorem that links a minimum cut in an associated graph, G_α, with the optimal values of the variables.

For α a scalar in the range of the variables, the graph associated with $G = (V, E)$ is an s,t-graph G_α constructed as follows: We add to the graph $G = (V, E)$ a source node s and a sink node t; for each node $i \in V$ we add an arc from s of capacity $\max\{\frac{\partial f_i}{\partial x_i}(\alpha), 0\}$ and an arc to t of capacity $\max\{-\frac{\partial f_i}{\partial x_i}(\alpha), 0\}$, where $\frac{\partial f_i}{\partial x_i}(\alpha) = 2(\alpha - d_i)$. Note that at least one of these arcs must have capacity of 0, thus a node can be connected either to source or to sink but not to both. Each edge $[i, j] \in E$ is replaced by a pair of arcs (i, j) and (j, i) both of capacity u_{ij}. An example of a G_α graph on 4 nodes and $\alpha = 3$ is illustrated in Figure 2.2. Note that node 1 (the top left node in the graph) has neither an arc from the source nor an arc to the sink since the respective derivative is $\frac{\partial f_1}{\partial x_1}(\alpha) = 2(\alpha - d_1) = 0$.

Because of the convexity of the functions $f_i()$, the graph G_α has the property that the source adjacent capacities are monotone increasing (more generally, monotone non-decreasing) in α, and the sink adjacent capacities are monotone decreasing (more generally, monotone non-increasing) in α. Such graphs are called parametric flow graphs. Let a minimum cut in G_α be $(S_\alpha \cup \{s\}, \bar{S}_\alpha \cup \{t\})$, where $S_\alpha \cup \{s\}$ is the *source set* of the minimum cut. If there are multiple minimum cuts, we select the one where $S_\alpha \cup \{s\}$ is *minimal* (contained in all sources sets of minimum cuts). It is well known that the source sets of minimum cuts in parametric flow graphs are *nested*: For $\alpha_1 < \alpha_2$, $S_{\alpha_1} \subseteq S_{\alpha_2}$. The nestedness is also a corollary of the following threshold theorem which states the relationship between the optimal solution to AC, $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$, and the source set of a minimum cut in G_α (Hochbaum 2001):

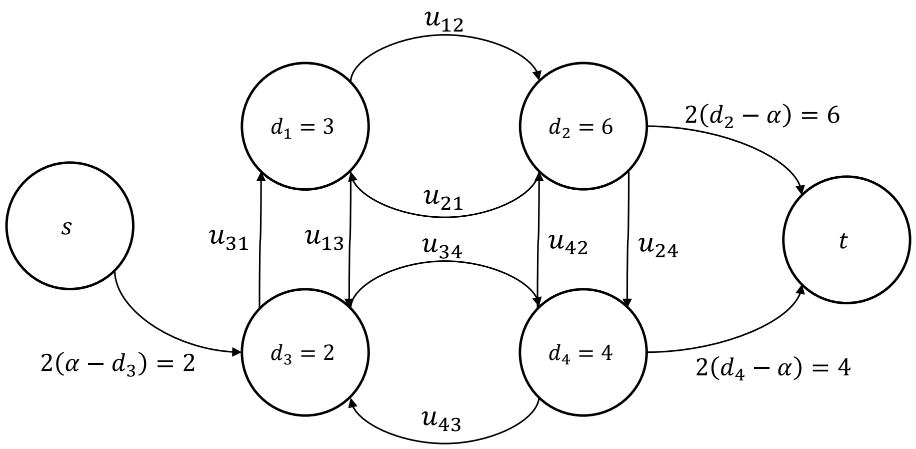


Figure 2.2: An example of G_α on a 4-node graph and $\alpha = 3$.

Theorem 2.1 (Threshold theorem). *For S_α the minimal source set of a minimum cut in G_α, the optimal solution x^* to AC satisfies $x_i^* < \alpha \ \forall i \in S_\alpha$ and $x_i^* \geq \alpha \ \forall i \in \bar{S}_\alpha$.*

With the threshold theorem, the following algorithm is used to solve AC: Call for a minimum cut procedure in the graphs G_α for $\alpha = 1, \ldots, k$ resulting in the sequence of nested source sets, $\{s\} = S_0 \subseteq S_1 \subseteq \ldots \subseteq S_k \subseteq S_{k+1} = V \cup \{s\}$. Let $\Delta_\alpha = S_\alpha \setminus S_{\alpha-1}$, then the optimal solution x^* is determined as follows:

$$\text{if } \ i \in \Delta_{\alpha+1} \text{ then } \ x_i^* = \alpha.$$

Let $T(n, m)$ be the complexity of a minimum cut algorithm on a graph with n nodes and m arcs, then this algorithm requires $O(kT(n, m))$ steps to solve AC.

To improve on the complexity we notice that because of the "nestedness" property, for $\alpha_1 < \alpha_2$, once the maximum flow in G_{α_1} is found, we can "shrink" the source set S_{α_1} with the source node s as it is guaranteed that S_{α_1} is part of the source set of a minimum cut in G_{α_2}. Once the arcs adjacent to source and sink are adjusted for the new parameter value α_2 the previous maximum flow is feasible except possibly for the arcs adjacent to sink where their capacities have gone down. A "parametric flow algorithm" can warm-start from such a solution and solve the entire sequence of k parametric flows and cuts in the complexity of a single maximum flow (Gallo, Grigoriadis, and Tarjan 1989). The push-relabel algorithm (Goldberg and Tarjan 1988) or Hochbaum's Pseudo-flow (HPF) algorithm (Hochbaum 2008) are both known to have this capability, and both run, for k parameter values, in $O(mn \log \frac{n^2}{m} + kn)$ steps on a graph with n nodes and m arcs. As a result, solving AC, where m is $O(n)$ for the graph G, can be accomplished in $O(n^2 \log n + kn)$.

Theorem 2.2. *The time complexity of an algorithm solving AC with a parametric minimum cut HPF or push-relabel is $O(n^2 \log n + kn)$.*

2.5 Empirical Results on Real Data Sets

We analyze four real wafer maps in this section. The first one appears in Tyagi, Bayoumi, et al. (1994) and the other three are presented by Yuan, Ramadan, and Bae (2011). The first wafer map has $20 \times 20 = 400$ chips and the other three have each contained 473 chips.

For all four wafer maps, we choose separation penalties $g_{ij}(x_i - x_j) = u \cdot |x_i - x_j|$, with u a factor that is common for all wafer maps and uniform for all pairs of chips. This is because there is no ex-ante information to differentiate between different pairs. In case there is a reason to differentiate, or to stress the neighborhood effect in some areas of the wafer more than in others, one can select a non-uniform value of u. We test 3 different values of k $(= 1, 2, 3)$ combined with 26 different values of u $(= 0.5, 0.6, \ldots, 3)$. The selection of u is used to balance the separation versus the deviation penalties.

The experiments in this Section and Section 2.6 are performed on a Lenovo X1 computer running the Windows 10 64-bit operating system with a Intel Core i5-5200U 2.20 GHz processor and 8.0 GB RAM. The adjacency-clustering problem for $k = 1$ is solved with Hochbaum's Pseudo-flow (HPF) algorithm (available at `http://riot.ieor.berkeley.edu/Applications/Pseudoflow/maxflow.html`, see Hochbaum 2008, Chandran and Hochbaum 2009). The adjacency-clustering problem for $k \geq 2$ is solved with parametric maximum flow using parametric HPF (the source code used is at `http://riot.ieor.berkeley.edu/Applications/Pseudoflow/parametric.html`).

The results are presented in the following subsections: In subsection 2.5.1 we illustrate the qualitative effect of changing the two parameter values, k and u, on the resulting clustering. Subsection 2.5.2 describes the application of the AC-Poisson model and the evaluation of the choice of the parameters in terms of the prediction error. The prediction error is measured by *relative absolute bias*, defined as:

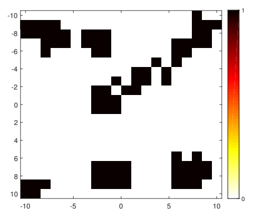
$$\frac{|\text{True yield} - \text{Estimated yield}|}{\text{True yield}}.$$

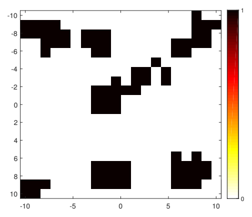
The lowest prediction errors lead to a choice of parameters for AC-Poisson, which is used afterward. In subsection 2.5.3 the AC-Poisson model, with the specific selection of parameters, is compared to the Poisson model and the Poisson regression model in terms of the relative absolute bias. Finally, in subsection 2.5.4, we test various yield models for the clusters generated by AC: First we test the negative binomial yield model, and then a combination of two different yield models (Poisson and negative binomial) for the no-defects cluster (of label 0), and the defective clusters, of positive label. These results are then compared with the negative binomial and negative binomial regression prediction models (that apply to the entire wafer).

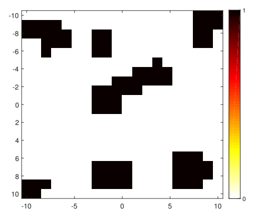
2.5.1 Visual Clustering Results for Varying Parameters

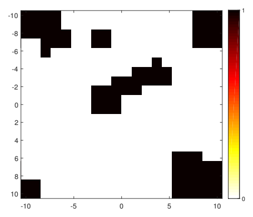
We present first, visually, the clustering results for the first wafer map with different choices of the parameters. As shown in Figure 2.3, as the value of k increases (going down the rows of images), the clusters corresponding to positive values of the label are becoming more differentiated into small contiguous groups. As for the value of u, that increases for the columns of images from left to right, the effect is to create more contiguous clusters, since a higher value of u causes higher penalty for non-contiguity. Indeed Figure 2.3(a) consists of many small contiguous groups while in Figure 2.3(d) there are only a few large groups. It should be noted that clusters generated by AC models are not necessarily contiguous. That is, there is a trade-off between contiguity, that implies contiguous chips should fall in the same cluster, and clustering chips with similar number (or density) of defects. For instance, adjacent chips tend to belong to the same cluster, unless there is a substantial gap between their numbers of defects. On the other hand, non-adjacent chips that have the same number of defects, may well fall in the same cluster, resulting in a cluster by a collection of non-contiguous groups. We observe that for this first wafer map, the groups of positive labeled chips are positioned near the center and the four corners of the wafer, implying potential manufacturing problems.

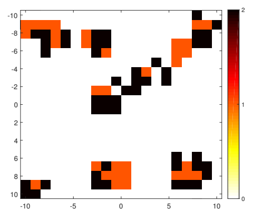
The effects illustrated in Figure 2.3 on changing the values of u and k indicate a general trend. When u is small, the separation cost is low, and the adjacency effect is not playing a role. In contrast, for u that is very large, the separation cost dominates the objective function and the tendency is to group many of the chips in a single contiguous cluster, even

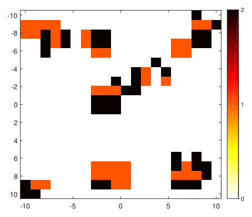

(a)

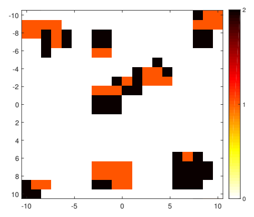

(b)

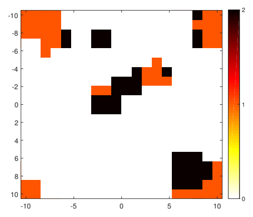

(c)

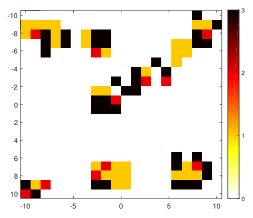

(d)

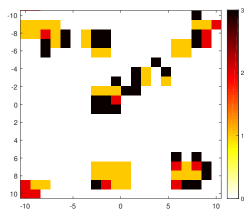

(e)

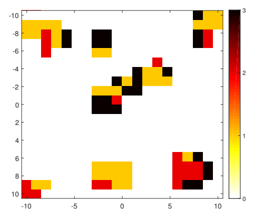

(f)


(g)


(h)


(i)


(j)


(k)

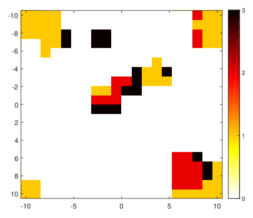
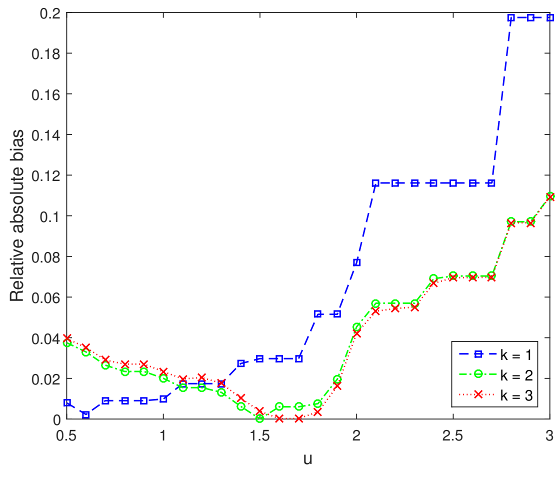

(l)

Figure 2.3: Adjacency-clustering results for sample wafer map 1 where the first row presents the results for $k = 1$, the second row is for $k = 2$ and the third row is for $k = 3$; From left to right, the four columns correspond to increasing values of $u = 0.1, 0.5, 1, 2$. Different clusters are differentiated based on the colors.

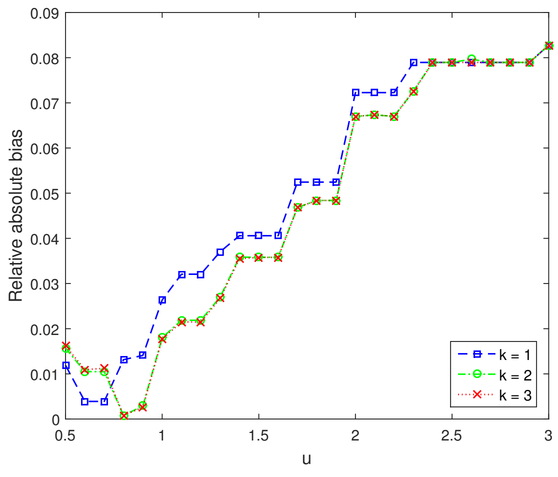
if they differ substantially in their numbers of defects. In the extreme case where $u = +\infty$, the entire wafer forms a single cluster. Similarly for parameter k: when k is small, e.g. the binary case, groups of high number of defects may be clustered together with groups of low number of defects. If k is large, then the clusters tend to have small number of objects which may result in poor prediction performance. Next we study the effects of the parameters selection on the prediction error, for the AC-Poisson model.

2.5.2 Parameter Selection for AC-Poisson

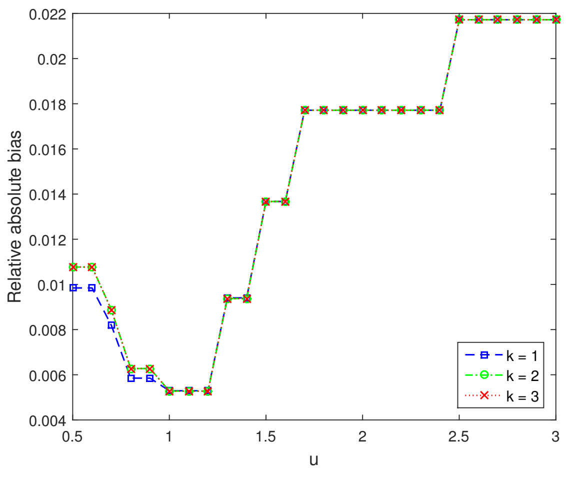
AC-Poisson generates $k + 1$ clusters, the yield of each of which is then computed with a Poisson yield model. The yield for each cluster j is estimated as $\hat{y}_j = \exp(-\lambda_j)$, where λ_j is the average number of defects for the cluster. These estimates are then used to predict the



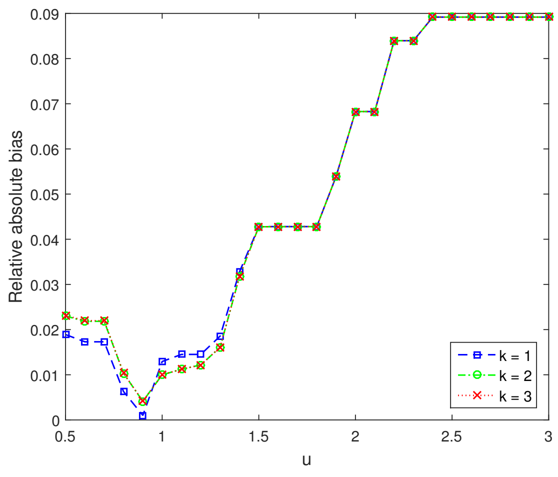
(a) Wafer Map 1 (gap: 0.020)



(b) Wafer Map 2 (gap: 0.017)



(c) Wafer Map 3 (gap: 0.000)



(d) Wafer Map 4 (gap: 0.009)

Figure 2.4: Relative absolute bias error of AC-Poisson model for u in $\{0.5, 0.6, \ldots, 3\}$ and k in $\{1, 2, 3\}$ on four real data sets. The gap value is the difference between the minimum error across combinations attained and the error for $u = 1$ and $k = 2$.

yield for the whole wafer.

To determine which parameters to select and how their choice affects the relative absolute bias error, we apply AC-Poisson to different combinations of u and k for the four datasets. Figure 2.4 presents the relative absolute bias of AC-Poisson model for each dataset with the choice of values of u in $\{0.5, 0.6, \ldots, 3\}$ and values of k in $\{1, 2, 3\}$. The results indicate that the choice of the combination of $u = 1$ and $k = 2$ is very close to the best combination of the two parameters. Indeed, as will be shown, in all our experiments this combination is close to the best combination. We therefore refer to it as the *default setting*. Here, the gaps between the minimum error across all combinations and the error attained for the default setting of $u = 1$ and $k = 2$, are 0.020, 0.019, 0.000, 0.009 for wafer maps 1, 2, 3, 4 respectively.

A known statistical method of selecting a value such as k is the Bayesian information criterion (BIC). BIC is a trade off between an increase in the number of parameters and an increase in the likelihood of all observations that results from finer distributions with larger number of parameters. In the context of the adjacency-clustering model, the number of clusters increases with k, and for each cluster the yield estimation requires the estimation of the cluster's distribution parameters. For instance, using Poisson model for each cluster, the total number of estimated parameters is $(k + 1)$. Therefore the number of parameters, denoted by h_k, grows here linearly with k. For $\mathcal{L}$ the likelihood of all observations on the wafer, the BIC score is defined as:

$$BIC = h_k \ln n - 2 \ln \mathcal{L}.$$

The lower the BIC score the better. Setting $u = 1$ and the AC-Poisson model, we compute the BIC scores for $k = \{1, 2, 3, 4\}$ on the four real wafer maps, as shown in Table 2.2. For each wafer map, the value of k corresponding to the lowest BIC is assigned a rank of 1 and the second lowest is assigned a rank of 2, etc. The average rank across the four samples is presented in the last column of Table 2.2. From Table 2.2, $k = 2$ has the lowest average rank, which is one of the reasons why we select this value of k in our default setting.

Table 2.2: BIC of AC-Poisson.

Wafer Map	1	2	3	4	Average Rank
$k = 1$	478.73	498.42	348.55	568.08	2.75
$k = 2$	384.19	476.97	354.63	564.92	1.75
$k = 3$	355.14	480.33	360.79	571.08	2.5
$k = 4$	351.51	486.35	366.95	577.24	3

We discuss further issues concerning the choice of u and k in the section on simulated data, Section 2.6. For the comparison with other prediction models we are selecting the default setting of $u = 1$ and $k = 2$ as the one for AC-Poisson.

2.5.3 Performance Comparison of AC-Poisson with Poisson and Poisson Regression Models

Table 2.3 provides the comparison of the relative absolute bias for AC-Poisson model ($u = 1$ and $k = 2$), Poisson model, and Poisson regression model. In Poisson regression model, the covariate vector is selected as $\{r, \cos\phi, \sin\phi, r\cos\phi, r\sin\phi\}$, as suggested by Bae, Hwang, and Kuo (2007) (using the center of the wafer as the reference point, r and ϕ denote the radial coordinate and angular coordinate in the polar coordinate system). We estimate the corresponding coefficients by maximum likelihood method with *glm()* function in R (see `https://stat.ethz.ch/R-manual/R-devel/library/stats/html/glm.html` for details). The adjacency-clustering prediction results of AC-Poisson improve, by a factor that varies between 3 to 15, on the Poisson regression model, and by a larger factor on the Poisson model. n particular, the improvement for wafer map is remarkably large. This may be the case since the defects in wafer 1 are heavily clustered and exhibit a clear pattern (as shown in Figure 2.3), which can be easily captured by AC-Poisson.

It is noted, that even though the Poisson regression model is intended to incorporate the spatial positioning that is absent in the Poisson model, our results indicate that it is only minimally effective. In addition, our results show that AC-Poisson model dominates the Poisson regression for any choice of $u \leq 2.3$ and $k \in \{1, 2, 3\}$. The results reported in Table 2.3 provide evidence that strongly supports the capability of adjacency-clustering to introduce major improvements in yield prediction.

In terms of running time, solving the adjacency-clustering model (with parametric maximum flow using HPF algorithm) requires 0.07 seconds for sample 1 and an average of 1.11 seconds for samples 2, 3, 4.

Table 2.3: Yield prediction comparison results: AC-Poisson with Poisson model and Poisson regression model. The best results are given in **boldface**.

Wafer Map	1	2	3	4
True yield	79.50%	84.36%	89.85%	79.28%
Poisson model	52.33%	74.85%	87.90%	72.21%
Relative Absolute Bias	34.18%	11.27%	2.17%	8.92%
Poisson regression model	55.12%	76.13%	88.16%	72.34%
Relative Absolute Bias	30.67%	9.76%	1.88%	8.75%
AC-Poisson model ($u = 1, k = 2$)	81.09%	82.84%	89.38%	78.49%
Relative Absolute Bias	**2.00%**	**1.80%**	**0.52%**	**1.00%**

2.5.4 Testing AC with Different Yield models: Comparison of AC-NB, AC-NBP and AC-PNB

In addition to the Poisson yield model, the negative binomial model is also widely used in yield prediction. Compared with Poisson model, it is less likely to underestimate the yield (K. O. Kim 2011). Following negative binomial model, the yield for cluster j is given by $\hat{y}_j = (1 + \lambda_j/\gamma_j)^{-\gamma_j}$, where γ_j is called the cluster parameter. There are multiple ways of determining γ_j (see Cunningham 1990 for details), and we adopt the method of moments as

$$\gamma_j = \frac{\lambda_j^2}{\sigma_j^2 - \lambda_j}. \tag{2.10}$$

Here σ_j^2 is the variance of the number of defects per chip for the cluster, which is estimated by the sample variance. Three different prediction models combined with adjacency-clustering are used here: (1) AC-NB model: negative binomial yield model is fitted to each cluster; (2) AC-NBP model: negative binomial yield model is fitted to non-defective clusters (cluster with "0"s) while Poisson yield model is fitted to defective clusters (of label > 1); (3) AC-PNB model: Poisson yield model is fitted to non-defective clusters while negative binomial yield model is applied to defective clusters. These three models are tested on the four wafers for different combinations of u and k. We select the parameter values that yield the lowest prediction errors (AC-NB: $u = 0.7$, $k = 3$; AC-NBP: $u = 0.6$, $k = 1$; AC-PNB: $u = 0.7$, $k = 3$). Experimental results for the choice of these parameter values are provided in Appendix A. It should be noted that the choice of $u = 1$ and $k = 2$ achieves similar results to the above parameters with average gaps of 0.0036, 0.0019 and 0.0083 for AC-NB, AC-NBP and AC-PNB respectively.

We compare the prediction results of these three AC models with negative binomial model and negative binomial regression model. In negative binomial regression model, we choose the same covariates as in Poisson regression model, and the coefficients are estimated using maximum likelihood method, which is implemented in *glm.nb()* in R (see `https://stat.ethz.ch/R-manual/R-devel/library/MASS/html/glm.nb.html` for details).

The results of comparing the performance of AC-NB, AC-PNB, AC-NBP, negative binomial and negative binomial regression models are given in Table 2.4. The results indicate that AC-NB is the best model to use uniformly. Specifically, AC-NB model outperforms other models for wafer 3 and wafer 4, AC-NBP model yields the best result for wafer 1 and AC-PNB model gives the best result for wafer 2. Compared with the negative binomial model, AC-NB model improves the prediction result by a factor between 2 and 14. Compared with negative binomial regression model, the error of AC-NB model is lower on wafers 2 and 4 and about the same for wafer 3, and a bit worse for wafer 1.

The reason why in some of the cases AC-PNB and AC-NBP perform better than AC-NB is that improved yield prediction can be achieved by fitting different yield models to different clusters. Combining different yield models works better in cases of unstable manufacturing processes that render different defect behaviors in different areas on the wafer. Still, AC-NB model is uniformly the most robust and therefore it is our recommended choice.

2.6 Simulated Data Study

To further compare the AC model with existing models, we generate simulated wafer maps with different degrees of clustering and radial loss. Wafer maps have been simulated using scattering scheme or superposition of different defect patterns (see, e.g., Yuan, Ramadan, and

Table 2.4: Yield prediction comparison between: AC-NB, AC-PNB, AC-NBP , negative binomial and negative binomial regression models. The best results are given in **boldface**.

Wafer Map	1	2	3	4
True yield	79.50%	84.36%	89.85%	79.28%
AC-NB ($u = 0.7$, $k = 3$)	79.82%	84.12%	89.94%	79.19%
Relative Absolute Bias	0.40%	0.27%	**0.10%**	**0.12%**
AC-NBP ($u = 0.6$, $k = 1$)	79.33%	84.47%	90.54%	80.06%
Relative Absolute Bias	**0.22%**	0.14%	0.76%	0.98%
AC-PNB ($u = 0.7$, $k = 3$)	79.83%	84.34%	90.18%	79.78%
Relative Absolute Bias	0.41%	**0.02%**	0.37%	0.63%
Negative binomial model	76.31%	83.09%	89.71%	78.07%
Relative Absolute Bias	4.01%	1.51%	0.16%	1.53%
Negative binomial regression model	79.28%	84.12%	89.76%	79.07%
Relative Absolute Bias	0.28%	0.28%	**0.10%**	0.26%

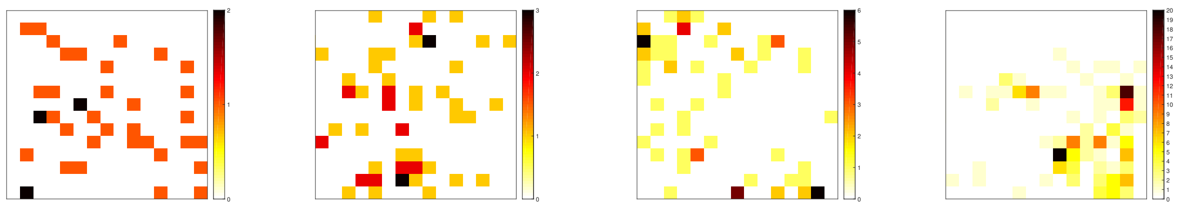
Bae 2011, Bae, Hwang, and Kuo 2007 and M. H. Hansen, Nair, and D. J. Friedman 1997). We adopt here the Generalized Linear Mixed Model (GLMM) to generate simulated samples because it easily captures both the inhomogeneity and spatial dependence of defects, which is difficult to model using either scattering or superposition method. GLMM has been applied in the analysis of spatial correlated count data, as shown in Christensen and Waagepetersen (2002), E. Park and Lord (2007) and Chib and Winkelmann (2012).

In the simulation, the number of defects on chip i follows Poisson distribution with density λ_i, which is related to covariates with the canonical logarithmic function. As radial loss is significant in semiconductor manufacturing, the radial distance, r_i, is used as a covariate while no other spatial variables are considered:

$$\log(\lambda_i) = \beta_0 + \beta_1 r_i + s_i. \tag{2.11}$$

Here $\mathbf{s} = (s_1, \dots, s_n)$ is a random vector that follows a Gaussian distribution with a specified covariance matrix Σ: $\mathbf{s} \sim N(0, \Sigma)$. In order to model the spatial dependency between chips and thus generate defect clusters, Σ is designed with the conditional autoregressive model (CAR), resulting in a type of Gaussian Markov Random Field (GMRF). Under the four-neighborhood system, the inverse of covariance matrix $Q = \Sigma^{-1}$ has the following structure:

$$Q_{ij} = \begin{cases} 4p & i = j \\ -p & j \in N(i) \\ 0 & \text{otherwise,} \end{cases} \tag{2.12}$$



(a) $p = 1$ (b) $p = 0.5$ (c) $p = 0.3$ (d) $p = 0.2$

Figure 2.5: Simulated wafer maps without radial loss for varying values of p.

where p is called precision parameter. Smaller p indicates greater neighborhood effects and thus generates more clustered defects. Additional details about GMRF and CAR can be found in Lichstein et al. (2002) and Havard Rue and L. Held (2005). Our simulation test consists of two parts. The first part simulates wafer maps with different degrees of clustering but with no radial loss, i.e. $\beta_1 = 0$, while the second part simulates samples with radial loss, i.e. $\beta_1 > 0$. The values of β_0 and β_1 are selected such that the simulated wafer maps have similar number of defects to real wafer maps.

Each simulated wafer contains $15 \times 15 = 225$ chips and parameters are chosen to be $(\beta_0, \beta_1) = (-2, 0)$ for cases with radial loss and $(\beta_0, \beta_1) = (-2, 0.1)$ for cases without radial loss. Figure 2.5 demonstrates four wafer maps simulated by GLMM with different values of p, in which smaller p implies greater spatial correlation and thus higher degree of clustering. For each p, we generate 100 simulated wafer maps.

It is noted that our adjacency-clustering algorithm is scalable for large wafer maps. For instance, for a wafer map containing $100 \times 100 = 10000$ chips, the AC model is solved to optimality within 3.36 seconds. For the purpose of generating insights from the simulation it is sufficient to use simulated wafers containing small number of chips, e.g. $15 \times 15 = 225$ chips.

Parameter selection: The AC model requires the setting of the two parameters, u and k. Similar to other data analytics methods and machine learning techniques, the combination of u and k can be selected through training on given training datasets (samples produced in the initial manufacturing stage or representative historical samples). Specifically, we identify the best combination of u and k that minimizes the training error. In order to mimic the practical prediction tasks in integrated circuit manufacturing, we perform two different training procedures to select u and k on simulated wafer maps: (1) **One-sample training:** Among 100 simulated wafer maps, we pick as the training set one wafer map at a time and the other 99 maps as the test set. The AC model is fitted to the training set to find the best combination of u and k which is then applied to the test set and the mean absolute percentage error (MAPE) is calculated. We report as the error the average MAPE across these 100 runs. (2) **Two-fold training:** We select a random subset of size 50 out of the 100 simulated wafer maps to serve as training set and the complement set serves as test set. Then the MAPE is evaluated on the test set. Next the roles of the same two sets are reversed with the second one serving as training and the first as test set. The average of these two MAPE values is then reported. These two training procedures are two different types of cross-validation designed to measure the actual prediction performance of AC models. It is noted that the reported errors are test errors instead of training errors.

In the next two subsections we evaluate the performance of the AC models with the two training procedures as compared to the default setting of $u = 1$ and $k = 2$ on the simulated data. AC with these three parameter selection procedures are compared to Poisson model, Poisson regression and negative binomial models. In subsection 2.6.1 we analyze AC-Poisson with parameters selected by training and the default setting, and compare their performance with Poisson and Poisson regression model. In subsection 2.6.2 we test the performance of AC with other yield models.

2.6.1 Testing AC-Poisson on Simulated Data

In this subsection we compare AC-Poisson to Poisson on simulated maps. We are not testing the regression model here because the regression is on the radial distance and in the simulated data here $\beta_1 = 0$, which means that there is no radial effect.

Table 2.5: Mean absolute percentage error comparison results between AC-Poisson and Poisson model for simulated wafer maps ($\beta_0 = -2, \beta_1 = 0$). The best results are given in **boldface**.

Precision (p)	0.2	0.3	0.5	1
AC-Poisson model (one-sample training)	1.81%	1.64%	1.22%	0.99%
AC-Poisson model (two-fold)	**0.81%**	**0.98%**	**0.84%**	**0.68%**
AC-Poisson model ($u = 1$, $k = 2$)	**0.81%**	**0.98%**	0.85%	**0.68%**
Poisson model	40.42%	19.27%	6.93%	1.81%

The results given in Table 2.5 are in terms of MAPE. The two training procedures are performed and the corresponding cross-validation errors are presented for AC-Poisson. We also report the prediction error of the default setting. As shown in Table 2.5, AC-Poisson model provides significantly better prediction results than the Poisson model. As expected, the gap between the two narrows as the clustering becomes less pronounced in the simulated data, which is measured by the increasing value of the precision parameter p. For highly clustered wafer maps ($p = 0.2$), our model reduces the prediction error by a factor of 50, as compared with Poisson model. In addition, two-fold training provides lower errors than one-sample training, which is explained by the larger size of the training data set. Compared with the two-fold training, the default setting gives almost the same prediction results, which provides additional evidence to support the use of this combination in the context of integrated circuit manufacturing.

Next we consider simulated data with radial loss, i.e. $\beta_1 = 0.1$. Here we compare AC-Poisson with both Poisson and Poisson regression models, where the only covariate for Poisson regression model is $\{r\}$, the radial distance of a chip (and no angle-dependent variables). Table 2.6 displays the comparison results, in terms of the error measure MAPE, of AC-Poisson and Poisson and Poisson regression models.

Table 2.6: Mean absolute percentage errors comparison results between AC-Poisson, Poisson and Poisson regression model for simulated wafer maps with radial loss ($\beta_0 = -2, \beta_1 = 0.1$). The best results are given in **boldface**.

Precision (p)	0.2	0.3	0.5	1
AC-Poisson model (one-sample training)	1.97%	2.39%	2.49%	1.70%
AC-Poisson model (two-fold)	1.27%	**1.29%**	**1.43%**	**1.18%**
AC-Poisson model ($u = 1$, $k = 2$)	**1.23%**	1.41%	1.60%	1.36%
Poisson model	62.21%	41.87%	18.90%	5.77%
Poisson regression model	45.82%	31.53%	14.88%	4.75%

As seen in Table 2.6, AC-Poisson outperforms Poisson and Poisson regression model with the two training procedures and the default setting. As expected, Poisson regression model provides better prediction accuracy than Poisson model, which can be explained by the fact that Poisson model does not relate the defect density to radial distance. Both models, however, are significantly inferior to AC-Poisson, in terms of the error. Overall, two-fold training leads to the best prediction results, and the default setting gives very close results to the two-fold training. This validates the setting of $u = 1$ and $k = 2$ as a good choice in the absence of training data.

2.6.2 Testing AC-NB, AC-NBP and AC-PNB on Simulated Data

In this subsection, we extend our discussion to AC models with the negative binomial yield model. The MAPE of AC-NB, AC-NBP and AC-PNB as well as the negative binomial model for datasets with and without radial loss are presented in Table 2.7 and Table 2.8, respectively. Similarly, we consider three parameter settings for AC models: two with our training procedures and one with the default setting. It should be mentioned that for the simulated wafer maps with radial effect, we do not construct a negative binomial regression model as the iteratively reweighted least squares (IRLS) algorithm fails to converge for many simulated maps. The lack of convergence has been noted previously in the literature (for more details see Marschner et al. 2011). This phenomenon worsens as the neighborhood effect becomes more prominent in our simulation.

From the comparison results we conclude that both AC-NB and AC-NBP models outperform the AC-PNB model and negative binomial model for the simulated data set without radial effects. For the simulated wafer maps with radial effects, AC-NB, AC-NBP and AC-PNB all provide smaller prediction errors than the negative binomial model. On both simulated data sets, AC-NB model is the leading model, in terms of the lowest errors for

Table 2.7: Mean absolute percentage errors comparison results between AC-NB, AC-NBP, AC-PNB and negative binomial model for simulated wafer maps ($\beta_0 = -2, \beta_1 = 0$). The best results are given in **boldface**.

Precision (p)	0.2	0.3	0.5	1
AC-NB (one-sample training)	1.50%	0.87%	0.45%	0.22%
AC-NB (two-fold)	0.92%	**0.37**%	**0.23**%	**0.18**%
AC-NB ($u = 1, k = 2$)	1.64%	0.58%	0.28%	0.19%
AC-NBP (one-sample training)	1.69%	1.53%	0.81%	0.44%
AC-NBP (two-fold)	0.74%	0.69%	0.49%	0.21%
AC-NBP ($u = 1, k = 2$)	**0.69**%	0.71%	0.60%	0.42%
AC-PNB (one-sample training)	2.42%	1.83%	1.02%	0.78%
AC-PNB (two-fold)	1.26%	0.74%	0.58%	0.68%
AC-PNB ($u = 1, k = 2$)	1.85%	0.98%	0.77%	0.63%
Negative binomial model	8.42%	4.93%	1.33%	0.28%

Table 2.8: Mean absolute percentage errors comparison results between AC-NB, AC-NBP, AC-PNB and negative binomial model for simulated wafer maps ($\beta_0 = -2, \beta_1 = 0.1$). The best results are given in **boldface**.

Precision (p)	0.2	0.3	0.5	1
AC-NB (one-sample training)	2.86%	1.38%	1.01%	0.51%
AC-NB (two-fold)	2.10%	**0.94**%	**0.48**%	**0.32**%
AC-NB ($u = 1, k = 2$)	3.74%	1.91%	0.73%	0.34%
AC-NBP (one-sample training)	1.78%	1.93%	1.97%	1.01%
AC-NBP (two-fold)	1.16%	1.02%	1.03%	0.63%
AC-NBP ($u = 1, k = 2$)	**1.03**%	1.23%	1.22%	0.96%
AC-PNB (one-sample training)	3.55%	2.40%	1.96%	0.92%
AC-PNB (two-fold)	2.19%	1.15%	1.04%	0.90%
AC-PNB ($u = 1, k = 2$)	3.66%	1.98%	1.19%	0.92%
Negative binomial model	16.58%	11.93%	4.45%	1.31%

most cases. For these AC models, the default setting of $u = 1$ and $k = 2$ has similar prediction results to the two-fold training results. The default setting provides better results than the one-sample training in most simulations, which implies that it is the combination to select unless sufficient data is available for training.

2.7 Conclusion

We introduce the adjacency-clustering (AC) model for yield prediction that takes into account a neighborhood effect. We demonstrate that this model delivers significant improvements in prediction accuracy as compared to state-of-the-art statistical approaches. The empirical evidence is based on runs for real data sets and simulated data sets. The AC model is parametrized by the selection of two parameters that could be tuned for specific purposes. Nevertheless, we show that even making a default selection of values u=1 and k=2 still delivers high quality prediction results that substantially improve on existing techniques. The AC model applies a polynomial time algorithm to obtain clusters efficiently and thus can be available for online monitoring and other practical uses. Although it fits classical yield model for each cluster, the yield prediction result of adjacency-clustering model exhibits significant improvement in the accuracy compared with classical models that do not differentiate between clusters. We also observe that the scheme of fitting different yield models to clusters with different yield levels can further increase the accuracy. This observation implies that different clusters in a wafer may have different types of mechanisms of generating defects. In practice, historical data can be used as the training set to select the two parameters, u and k. Our simulation results show that AC models work well even with very small training set. And through evaluation on both real and simulated data sets, we find out that the combination of $u = 1$ and $k = 2$ leads to superior prediction performance.

Apart from yield prediction, the adjacency-clustering model can be used to evaluate the extent of clustering of defects on wafer maps, where larger objective values correspond to higher segregation, or separation, of clustering of defects. This may be helpful in the quality control of a manufacturing process.

Compared with existing regression models in the literature, our model presents not only improved prediction accuracy, but also other advantages: First, our model is free from coefficient estimation, which remains challenging for regression models based on complicated distributions or discrete hidden Markov models, especially when handling large-scale data. Second, our model is highly flexible and can be applied to wafers with various spatial patterns since the spatial pattern is naturally captured by the solution to the adjacency-clustering model. In contrast, regression models necessarily require covariate selection and this selection is increasingly difficult as wafers exhibit more complicated spatial patterns. The success of the technique of adjacency-clustering presented in this paper bodes well to its applications to other contexts where a neighborhood effect is manifested, e.g. energy consumption prediction and disease mapping.

Chapter 3

On-Time Last Mile Delivery

3.1 Introduction

As e-commerce booms and customers expect faster delivery, food shopping has recently been shaped into a case in point. The fast-growing online platforms enable convenient food ordering and delivery services for customers. While platforms such as Grubhub and UberEATS deliver food that is prepared and packaged by restaurants, others, for example SpoonRocket and Domino's Pizza, prepare and deliver their own food boxes. In either case, one of the key challenges is how to assign the prepared orders to drivers (or carriers) for on-time delivery.

Motivated by our partner food service provider in China, we aim to improve its on-time delivery performance through better order assignment decisions. This provider prepares the food at its central kitchen, which is referred to as "the depot" hereafter, and delivers to customers within a certain radius from the depot. Customers place orders before each cutoff time, e.g., 10:30 am, are promised to receive the orders by a deadline, e.g., 11:45 am. From its daily operations, the provider found that delays seemed to be inevitable, especially when the orders were poorly assigned to drivers.

In this paper, we highlight two practical complications in order assignment—*the driver's routing behavior* and *uncertain service time*–and tackle them by bridging between the tools in prediction and optimization. First, the drivers do not always follow the planned delivery sequences from the routing tools, if any. While many last mile service providers invest considerably in routing tools, the practitioners realize that significant deviations exist between the drivers' routes and the planned ones. It is said that drivers tend to repeat their familiar routes, and they rely on their past experience and the real-time road conditions. Thus, our partner service provider only assigns orders to drivers without detailed route planning and instead gives them the freedom in choosing their delivery routes. Such a route deviation is not unique to the Asian context. According to a research project conducted in the U.S. and Mexico, using the data on deliveries over a one-year period for a large soft drinks company, it was found that three out of four deliveries did not follow the planned sequence (Y. Li and

Phillips, 2018). Therefore, we aim to construct a travel time prediction model to capture the drivers' behavior, which can be integrated with order assignment optimization.

Second, the time a driver spends at a customer location, which we term as the "service time", is highly uncertain. Because the customer locations are often high rise buildings, the drivers usually need to find parking spaces, navigate to the right floor and meet the customers in person. The service time varies and depends on the customer location and the order size, which are also random from day to day. By leveraging the existing techniques in optimization with uncertainty, we are able to account for uncertain service times when planning the order assignments.

3.1.1 Order Assignment with Travel Time Predictors

We first provide an overview of our model and the general framework for the integration between a prediction model and an optimization formulation in our context. We consider that the order assignment problem consists of two stages: upon receiving the orders from customer locations $\mathcal{I}$, the service provider first assigns the locations $\mathcal{I}_k$ to driver k among a pool of drivers $\mathcal{K}$, and then each driver k responds to the assignments by choosing his preferred route to visit $\mathcal{I}_k$.

Suppose driver k chooses his own route $\mathbf{z}_k$ by optimizing his objective function $g_k(\mathbf{z}_k, \mathcal{I}_k)$. Consequently, the route choice taken by driver k is the optimal solution $\mathbf{z}_k^*(\mathcal{I}_k)$ to the driver's optimization problem $\min_{\mathbf{z}_k} g_k(\mathbf{z}_k, \mathcal{I}_k)$. Moreover, we denote the uncertain service times at all the locations as $\tilde{\mathbf{t}} \in \mathbb{R}^{|\mathcal{I}|}$ that follows a distribution $\mathbb{P}$. Suppose the on-time performance metric is characterized by the function $H\left(\{\mathcal{I}_k, \forall k \in \mathcal{K}\}, \{\mathbf{z}_k^*(\mathcal{I}_k), \forall k \in \mathcal{K}\}, \tilde{\mathbf{t}}\right)$, e.g., the delivery delay. Therefore, the order assignment problem can be generally formulated as a two-stage stochastic program as follows:

$$
\begin{aligned}
\min_{\mathcal{I}_k, \forall k \in \mathcal{K}} \quad & \mathbb{E}_{\mathbb{P}}\left[H\left(\{\mathcal{I}_k, \forall k \in \mathcal{K}\}, \{\mathbf{z}_k^*(\mathcal{I}_k), \forall k \in \mathcal{K}\}, \tilde{\mathbf{t}}\right)\right] \\
\text{s.t.} \quad & \mathbf{z}_k^*(\mathcal{I}_k) = g_k(\mathbf{z}_k, \mathcal{I}_k), \quad \forall k \in \mathcal{K} \\
& \mathcal{I}_k \subseteq \mathcal{I}, \quad \forall k \in \mathcal{K}.
\end{aligned}
\tag{3.1}
$$

In this paper, we focus on the practice that the service provider partitions $\mathcal{I}$ into $\mathcal{I}_k, \forall k \in \mathcal{K}$ for the order assignment, aiming to minimize the total delay of all the drivers as in the following on-time performance metric:

$$
H\left(\{\mathcal{I}_k, \forall k \in \mathcal{K}\}, \{\mathbf{z}_k^*(\mathcal{I}_k), \forall k \in \mathcal{K}\}, \tilde{\mathbf{t}}\right) = \sum_{k \in \mathcal{K}} \left(\sum_{i \in \mathcal{I}_k} \tilde{t}_i + l_k(\mathbf{z}_k^*(\mathcal{I}_k)) - \tau\right)^+, \tag{3.2}
$$

where $\sum_{i \in \mathcal{I}_k} \tilde{t}_i$ is the total service time at the customer locations, $l_k(\mathbf{z}_k^*(\mathcal{I}_k))$ is the total travel time on the road and τ is the target delivery time window.

The main challenge here is to depict $\mathbf{z}_k^*(\mathcal{I}_k)$ for the evaluation of the total travel time $l_k(\mathbf{z}_k^*(\mathcal{I}_k))$. Unfortunately, a driver's preference in route choices is often unobservable to the provider or difficult to be formulated mathematically as elaborated by the driver's routing behavior above. That is, although the driver k's objective function $g_k(\mathbf{z}_k, \mathcal{I}_k)$ is clear to the driver himself, it is unknown to the service provider.

Instead of investigating $g_k(\mathbf{z}_k, \mathcal{I}_k)$ or $\mathbf{z}_k^*(\mathcal{I}_k)$, our approach takes a straightforward step by estimating $l_k(\mathbf{z}_k^*(\mathcal{I}_k))$ as a function of $\mathcal{I}_k$. Using machine learning methods, we generate features based on $\mathcal{I}_k$ as the travel time predictors and develop a prediction model $l(\mathcal{I}_k)$ for the total travel time spent by a driver to visit $\mathcal{I}_k$ locations. We will detail the potential travel time predictors in Section 3.3.1 and discuss how to integrate them with the existing optimization tools for order assignment. Specifically, we identify computationally tractable predictors and prediction models for the total travel time in Sections 3.4.1 and 3.4.2.

The connection between the on-time performance metric and the prediction target variable. Our approach treats the total travel time as the target variable because the considered on-time performance metric in (3.2) emphasizes minimizing the delays in completing all the routes. This measurement is timed from a driver's perspective with respect to his preset target time window. It encourages the drivers to finish their trips on time. Depending on the settings, the travel time measured as $l_k(\mathbf{z}_k^*(\mathcal{I}_k))$ can be of one-way (from the depot to the last visited customer) or round-trip (from the depot to the return). Alternatively, one may propose a more customer-centric metric that includes the delay at every customer location, which highly depends on the actual delivery sequence in a driver's route. Since the on-time performance metric in (3.2) evaluates the delay at the last-visited customer, it is more conservative (i.e., larger) than the average delay experienced by all the customers. While a customer-centric metric is more comprehensive than the metric in (3.2), it inevitably requires the prediction of $\mathbf{z}_k^*(\mathcal{I}_k)$. As we have noted above, due to the higher dimension of $\mathbf{z}_k^*(\mathcal{I}_k)$, learning a driver's exact route choice $\mathbf{z}_k^*(\mathcal{I}_k)$ as a target variable is much more challenging than learning his total travel time $l_k(\mathbf{z}_k^*(\mathcal{I}_k))$. Hence, we will leave it for future research.

3.1.2 Related Literature

We discuss the connections between our study and several streams of literature in terms of the research topic, approach, and framework.

Vehicle Routing Problems and Order Dispatch. The order assignment problem for delivery has been studied extensively in the form of vehicle routing problems (VRPs) in the transportation and operations research literature (see, e.g., Solomon 1987; Laporte 2007). There are various extensions of VRPs developed in deterministic or stochastic contexts, static or dynamic environments, and with or without time window constraints (see D. J. Bertsimas and Van Ryzin 1991; Laporte, Louveaux, and Mercure 1992; D. J. Bertsimas and Van Ryzin 1993; Gendreau, Laporte, and Séguin 1996; A. M. Campbell and Thomas 2008; Erera, Morales, and Savelsbergh 2010; Jaillet, J. Qi, and Sim 2016 for examples). While our problem is related to the routing problems, we aim to explicitly capture the driver's route choice via a travel time prediction model and thus do not solve the driver's routing optimization where its objective is unknown to the service provider.

In the context of last mile delivery, assigning the orders to vehicles is also referred to as dispatching. For example, Klapp, Erera, and Toriello (2016) study the dynamic dispatch problem where the orders arrive dynamically throughout a day. Our problem is also related to the order batching problem in a warehouse. For instance, Gademann and Velde (2005) study the order picking strategy in a parallel-aisle warehouse to minimize the total travel time, where the extraction time, i.e., the time spent at the pick location, is omitted in their optimization model, due to the assumption of a constant total extraction time. In our paper, in addition to the predicted travel time, we also integrate the uncertain service time at the customer locations by employing stochastic and robust optimization tools.

Approximations of the Routes. The VRP tour lengths are often approximated by analytical functions in strategic planning problems. The well-known Beardwood-Halton-Hammersley (BHH) Theorem (Beardwood, Halton, and Hammersley, 1959) allows the traveling salesman problem (TSP) tour length to be expressed in the probability density function of the demand points. By utilizing the asymptotic result in BHH theorem, Carlsson (2012) partitions a service region to balance the workload among vehicles. Alternatively, the approximation functions for TSP tours can also be obtained through simulation (H. Wang and Odoni, 2014). Moreover, the continuum approximation (CA) models are widely used to yield tractable analytical solutions in applications, including terminal design problems (Ouyang and Daganzo, 2006), inventory routing problems (Z.-J. M. Shen and L. Qi, 2007), dynamic facility location (X. Wang, Lim, and Ouyang, 2016), supply chain distribution network design (Lim, Mak, and Z.-J. M. Shen, 2016), and online grocery shopping analysis (Belavina, Girotra, and Kabra, 2016).

The key difference between our paper and the above literature is the starting point of the route approximation—our prediction model directly estimates the total travel time from the historical data without any underlying structural assumptions (e.g., the TSP or CA) for the driver's routing decision. Nevertheless, inspired by these analytical approximations, we construct the relevant predictors for travel time and discuss a class of computationally tractable prediction models for order assignment.

Data-driven modeling and optimization. Our paper is also closely related to the stream of papers that integrate predictive models with optimization. The recent advances include estimating the benefit of potential regions in organ allocation (Kong et al., 2010), resource allocation in a large gas utility (Angalakudati et al., 2014), incorporating demand prediction models into pricing optimization (Ferreira, Lee, and Simchi-Levi, 2015), pricing and selection for wines (Hekimoğlu, Kazaz, and Webster, 2016), analyzing traffic equilibrium (Ahipaşaoğlu, Meskarian, et al., 2015; Ahipaşaoğlu, Arıkan, and Natarajan, 2016), improving the assignment strategy in HIV early infant diagnosis supply chains (Jónasson, Deo, and

Gallien, 2017), as well as the feature-based machine learning algorithms for the newsvendor problem (Ban and Rudin, 2018).

Furthermore, Elmachtoub and Grigas (2017) propose a new predictive framework to incorporate the structure of the optimization problem, which enables robust performance against model misspecification. In recent revenue management research, there are also non-parametric choice modeling approaches that improve the prediction accuracy and assortment decision (e.g., Farias, Jagabathula, and Shah 2013; D. Bertsimas and Mišic 2015). Our paper follows a similar journey in data-driven modeling, where several model components for order assignment are identified and formulated based on data analytics. Nonetheless, our idea of developing a travel time prediction model and integrating it with existing optimization tools is novel and practical for delivery operations.

The closest study to our paper is that of Zheng, Natarajan, and Teo (2016) where least squares linear and quadratic estimators are proposed to approximate the distribution of project completion. Specifically, the authors solve for the least squares normal approximation to obtain the best parameters consist of two parts: the intercept and coefficients associated with individual activities. Similar in concept, we consider the total delivery time consists of the travel time and the uncertain service time at the customer locations, which is analogous to the intercept and random activity durations in Zheng, Natarajan, and Teo (2016). We further develop the prediction model for the travel time by utilizing the spatial information of the customer locations as predictors in the order assignment context.

3.1.3 Our Contributions

Our main contributions include the integration of predictors with optimization, modeling, and algorithms, as well as the insights from the numerical experiments.

The integration of predictors with optimization: We propose the framework in Section 3.1.1 that integrates travel time predictors with order assignment optimization with the objective of improving the on-time performance. The use of predictors enables us to account for driver's routing choice, where the decision-making process is unobservable to the service provider or intricate to model if not impossible. This idea can also be generalized to other applications that involve optimization components not directly representable by mathematical programming but can be approximated by prediction.

Models, reformulations, and algorithms: Using the order assignment problem as an example, we demonstrate the detailed development and implementation of the proposed framework. In particular, we identify the classes of tractable predictors and prediction models that are highly compatible with the existing stochastic and robust optimization tools. We also provide reformulations for the integrated models that can be efficiently solved by optimization solvers with the proposed branch-and-price algorithm. While our main discussion focuses on the order assignment problem for a single-period setting, two simple heuristics for multiperiod order assignment are built upon the single-period solutions, whose effectiveness is shown in the computational experiments.

Insights from a real-world application: We approach the order assignment problem from the analysis of the delivery data, where we observe the driver's routing behavior that deviates from the theoretical shortest-distance tours. Such an observation motivates us to predict a driver's travel time and integrate it with the ultimate order assignment. Using the delivery data, our numerical experiments not only demonstrate the superior performance of our proposed order assignment models with travel time predictors, but also highlight the importance of capturing the driver's routing behavior—a large sample size does not necessarily compensate for the misspecification of the driver's routing behavior.

3.2 A Food Delivery Service: Data and Observations

We acquire an operational data set that contains the detailed ordering and delivery information for a 2-month period in 2015, from a food delivery service provider that operates in Shanghai, China. The data set records the following information: 1) the order time: the time when the order is received by the provider; 2) the quantity: the number of items ordered; 3) the time window: the guaranteed delivery time; 3) the pick-up time: the time when the order is collected by a driver; 4) the delivery time: the actual time when the order is delivered to the customer; 5) the longitude and latitude: the customer location; 6) the cutoff time: it identifies the order batch and all orders with the same cutoff time will be dispatched together. The provider has determined a sequence of cutoff times $\{t_1, t_2, \dots\}$ and all orders placed in $(t_n, t_{n+1}]$ will be guaranteed to be delivered by $t_{n+1} + 75$ minutes.

There are 839 customer locations identified in the data set. In terms of the delivery performance, 20.4% of the total orders were not delivered on time, and 13.9% of the orders were delayed by more than 10 minutes. We summarize the number of orders and delay time in Table 3.1.

Table 3.1: Statistics of demand and delay (in minutes) during the 2-month period

	Min	Median	Max	Mean
Number of orders per location	1	2	503	10.3
Delay per order (minutes)	0	0	120	4.6

This paper aims to improve the order assignment by reducing inefficient routes taken by drivers, while leaving other potential causes such as food preparation operations for future exploration. When a driver picks up the batch of assigned orders, he is notified with the remaining time window for the delivery, i.e., τ in the on-time performance metric (3.2). In the provider's current practice, orders are assigned manually. Figure 3.1 demonstrates a set of orders in one dispatch batch, and how they were delivered in sequence by different drivers (we only show 3 drivers for clarity). We observe that driver 1 only carried one order and visited one customer location while driver 3 made deliveries to more than 5 locations. In addition, driver 2 was assigned to locations in opposite directions and consequently took long detours. Such an assignment decision led to an inefficient utilization of the drivers and poor on-time performance. In fact, driver 3 failed to deliver on time.

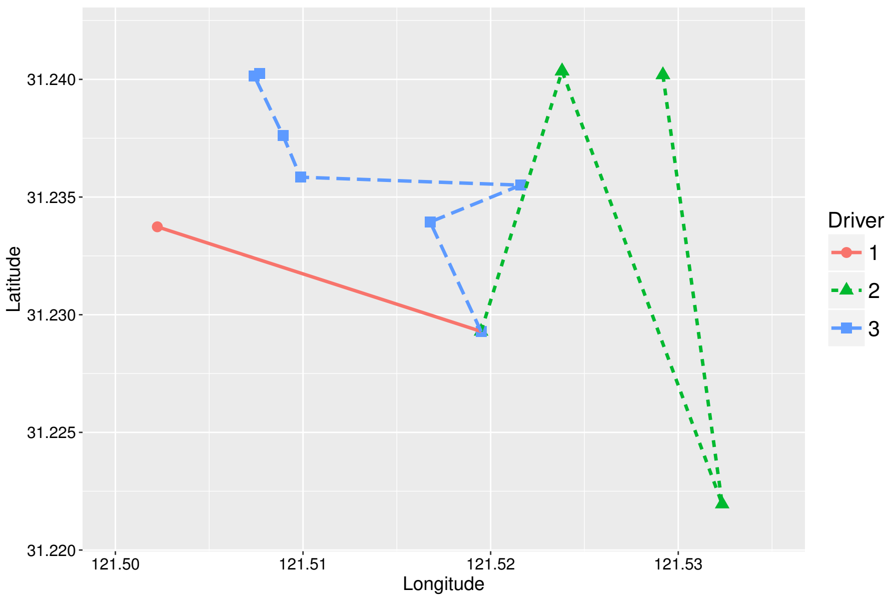


Figure 3.1: Delivery routes by 3 drivers (the red dot represents the depot).

Furthermore, driver 2 did not follow the shortest path by traveling back and forth. It is because the drivers have freedom in planning their own routes—they are not informed about or required to follow any planned routes. The underlying motivation is that the service provider believes that the drivers are familiar with the local area, but the information may not be represented in its database, e.g., the real-time road conditions, biking friendliness, and hidden paths. Hence, to improve the order assignment, it is critical to understand how the drivers would deliver a set of assigned orders.

3.2.1 Driver's Routing Behavior

In theory, it is optimal for a single driver to deliver the assigned orders based on the tour derived from a traveling salesman problem (TSP). However, as elaborated in Section 3.1, the driver's routes often deviate from the planned ones. This subsection examines the driver's routing behavior in comparison to the theoretical TSP routes. Since the comparison is only meaningful when the number of locations is more than one, we exclude the one-location routes in the following analysis.

To both the provider and drivers, the key objective is to minimize delays: the provider receives complaints from customers, and drivers receive penalties on delayed orders. Since the final segment from the last visited customer to the depot does not influence the on-time performance in this batch, the drivers tend to minimize the one-way delivery travel time, starting from the depot and ending at the last visited customer location. Using a standard TSP formulation with a dummy node (see the details in the Appendix), one can find the shortest one-way delivery tour to visit all the assigned customers.

Moreover, based on the drivers' routes reconstructed from the data, we are able to obtain their actual delivery routes, i.e., the travel sequences from the depot to the last visited customer. To compare the TSP routes with the actual ones, we calculate the total travel time of both routes based on the travel time matrix calibrated from *RouteMatrix API* (2019). To account for the speeding effect from electric bikes, we scale down the estimated travel time by a factor of 4/3 as suggested by the literature (Langford, J. Chen, and Cherry, 2015; A. A. Campbell et al., 2016).

Figure 3.2 presents the histogram of the differences between the actual delivery travel time and that from the TSP solution for all the observed routes visiting more than one location. We find that the actual delivery travel time is consistently greater or equal to the delivery travel time from the TSP solution. The long tail in this histogram highlights the instances where the actual delivery tour largely deviates from the shortest TSP route. The difference in travel time is 3.8 minutes on average and its average relative difference is 17.9%.

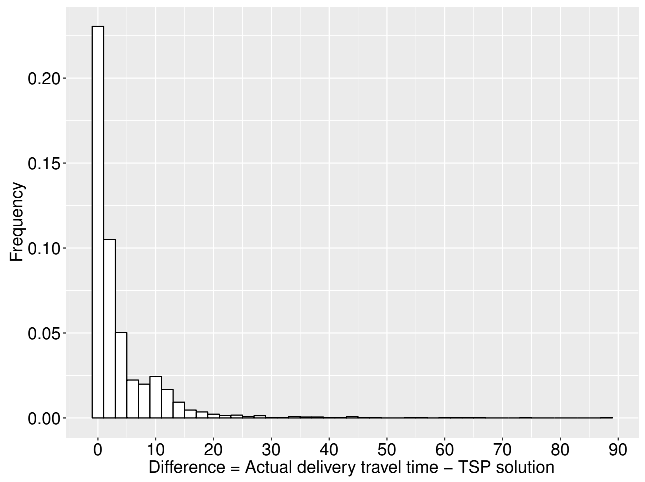


Figure 3.2: The travel time differences between the actual delivery and TSP routes (in minutes).

The above result implies that drivers often deviate from the theoretical shortest routes. This phenomenon is also confirmed in empirical studies on drivers' routing behavior (e.g., Lima et al. 2016 and Y. Li and Phillips 2018). There are several possible explanations.

First, the TSP formulation does not consider many practical constraints faced by drivers. For example, some road intersections have limited left turn flows and drivers may choose to avoid these intersections. Second, the drivers may prefer some travel patterns over others. For instance, zigzagging routes are found to be undesirable because of the increased possibility of accidents on busy streets (Holland et al. 2017). Third, delivery drivers may adjust the routes based on real-time traffic conditions, weather conditions and customers' updated locations.

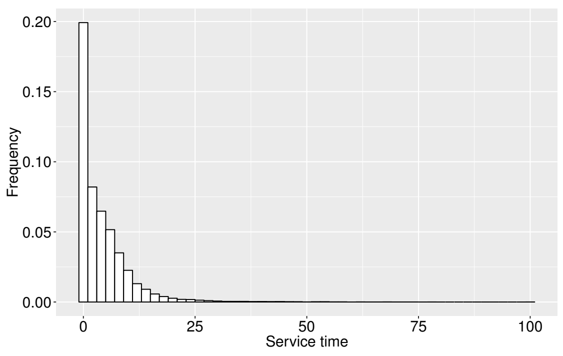
Modeling all the practical constraints and behavior considerations is difficult if not impossible and incorporating them as numerous constraints into a complicated VRP model is almost intractable. Therefore, the driver's routing problem denoted as $g_k(\mathbf{z}_k, \mathcal{I}_k)$ in the general framework 3.1 is difficult to represent in mathematical formulation. To overcome this difficulty, we will utilize machine learning techniques to predict the total travel time for visiting a set of customer locations without specifying the visiting sequence.

3.2.2 Uncertain Service Time

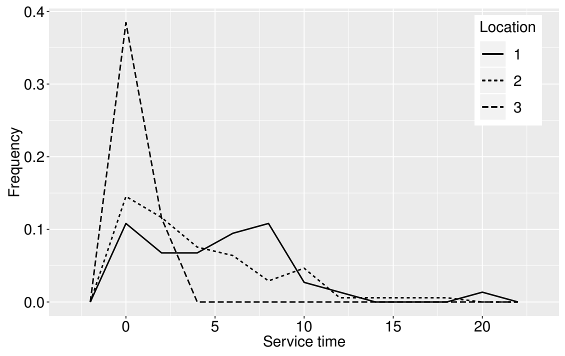
The food orders are usually delivered to customers in person. As many customer locations are high rise buildings in urban areas, the drivers often need to find a parking space, navigate to the right floor and meet the customers to hand over the orders. To differentiate this time from the travel time in the delivery tour, we use the term "service time" to denote the time a driver spends at a customer location. Since the service time is not measured explicitly in the data set, we estimate the service time spent at each customer location as follows. We first measure the biking travel time between two consecutively visited customer locations from the travel time matrix. The travel time estimation does not vary according to the traffic conditions, because electric bikes are allowed to take the bike lanes to avoid traffic congestions. The service time is therefore estimated as the difference between the observed delivery time and the estimated arrival time that is based on the departure time from the previous location and the estimated travel time between the two locations from the query. We note that by using a different travel time estimation tool, the estimated values of the service time may differ. Nevertheless, in this section, we intend to investigate the existence of uncertainty in the service time, as highlighted by the practitioners in the delivery service. In theory, an accurate measure of the service time can be obtained, if the provider tracks the more detailed vehicle location information.

The distribution of the service time estimated across all the customer locations is presented in Figure 3.3 (a), where its mean is 4.16 minutes. We observe that the variability of service time at the same location is significant. Moreover, we notice that the service time is heterogeneous across different locations. The mean service time of a customer location varies from 0 to 101 minutes. Furthermore, the distributions of the service time at various locations are different. Figure 3.3 (b) displays the histograms of service time at three different customer locations with a similar number of orders.

Furthermore, the service times across different customer locations are mostly uncorrelated. We apply Hoeffding's independence test (Hoeffding, 1948) to all location pairs. The results show that only 0.14% of them reject the null hypothesis of independence with a sig-



(a) All customer locations.



(b) Three selected customer locations.

Figure 3.3: Distribution of the service time (in minutes) at customer locations.

nificance level of 0.05 (out of the pairs that share more than 4 observations the percentage is less than 9.6%). This observation facilitates the analysis of the worst-case expected total delay in Section 3.4.3.

3.3 Travel Time Prediction

We dedicate this section to discuss predictors and prediction models for travel time $l_k(\mathbf{z}_k^*(\mathcal{I}_k))$ as a function of orders $\mathcal{I}_k$ assigned to driver k. The proposed prediction methods not only embed driver's routing behavior highlighted above but also contribute to the literature in approximating theoretical TSP and VRP tour lengths.

Previous research has proposed approximate formulas for the TSP (i.e., a special case of the VRP with a single driver) and VRP with asymptotic results under various scenarios. Assuming demand locations are independently and uniformly distributed in a square service region of area A, the optimal TSP tour length TSP^* satisfies (Beardwood, Halton, and Hammersley, 1959):

$$\lim_{n\to\infty} \frac{TSP^*}{\sqrt{n}} = \varphi\sqrt{A}, \tag{3.3}$$

where n is the number of locations and φ is a constant. Similar results for the VRP are also available, e.g., in Daganzo (2005). However, such approximate formulas require strong stochastic assumptions and can only yield good results when n is large (Z.-J. M. Shen and L. Qi, 2007). For food delivery, however, a driver may visit less than 10 locations, which makes the approximate formulas inappropriate. Furthermore, the above formula ignores the drivers' routing behavior discussed in Section 3.2.1.

Therefore, in the following, we explore several classes of predictors and prediction models for the travel time while maintaining the computational tractability in the order assignment problem. The integration of prediction and optimization for order assignment will be discussed in Section 3.4.

3.3.1 Predictors for Travel Time

We first review the promising predictors (features) for TSP tour estimation in the literature and then extend them to tractable predictors for the travel time. The general classes of travel time predictors are based on the number of locations, visiting area, distance, dispersion, and their interactive terms as illustrated in Figure 3.4.

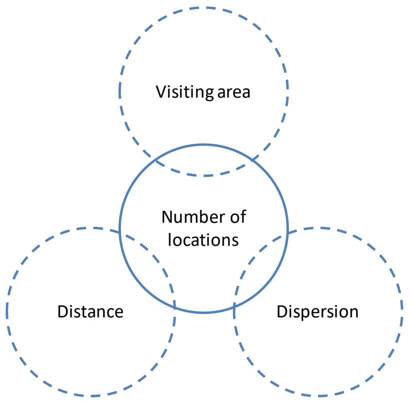


Figure 3.4: Classes of predictors for travel time.

Researchers have proposed simple regression models based on the approximation in (3.3) to estimate the TSP length. Typically, these regression models involve one predictor for the distance between the depot and customers, and the other one for the visiting area. For instance, Chien (1992) suggests $\overline{d}$ (the average distance between the depot and customers) and $\sqrt{R(n-1)}$, where R is the smallest rectangular area to cover the n locations. Moreover, Çavdar and Sokol (2015) propose using the standard deviation of the customers' coordinates to account for the impact of dispersion. In Table 3.2, we introduce the definitions of the graph attributes involved in travel time predictors.

However, not all the predictors can yield tractable representations for computation. For instance, R can be calculated as $a \times b$, where a is the maximum latitudinal difference between a pair of customer locations and b is the maximum longitudinal difference between a pair of customer locations. While a and b (depending on the order assignment decision) can be represented using linear constraints, R does not yield linear or convex representations, which raises tractability challenges for the order assignment problem. As a result, we propose a number of new predictors that can potentially capture the practical delivery travel time and enable tractable representations with linear constraints. We list the predictors surveyed from the literature as well as the tractable ones we identified and extended for our problem in Table 3.3 .

Note that although predictors such as $a\sqrt{n}$ are not linear or convex in the assignment decisions, we can represent them as piecewise linear functions by utilizing the fact that n is an integer variable. More details about the reformulations are provided in Section

Table 3.2: Definitions of the graph attributes in travel time predictors

Attribute	Definition
$\bar{d}$	The average distance between the depot and customer locations
d	The smallest distance between the depot and customer locations
D	The largest distance between the depot and customer locations
R	The area of the smallest rectangle covering the customer locations
R'	The area of the smallest rectangle covering both the depot and the customer locations
L	The area of the smallest lune (determined by two sectors with the same central angle originated from the depot) covering the customer locations
a	The maximum latitudinal difference between a pair of customer locations
b	The maximum longitudinal difference between a pair of customer locations
a'	The maximum latitudinal difference between a pair of customer locations (including the depot)
b'	The maximum longitudinal difference between a pair of customer locations (including the depot)
$cstdev_a$	The standard deviation of the latitudinal differences between the depot and customer locations
$cstdev_b$	The standard deviation of the longitudinal differences between the depot and customer locations
s_a	The average latitudinal difference between a pair of customer locations
s_b	The average longitudinal difference between a pair of customer locations

Table 3.3: Potential predictors for the delivery travel time

Class	Features in the literature	Tractable features
Distance	$\bar{d}$ (Chien 1992)	$\bar{d}$
	D	D
	d	d
Number of locations	n	n
Visiting area $\times$ Number of locations	$\sqrt{Rn}$, $\sqrt{R'(n+1)}$, $\sqrt{Ln}$ (Chien, 1992)	$a\sqrt{n}$, $b\sqrt{n}$, $a'\sqrt{n+1}$, $b'\sqrt{n+1}$
	$\sqrt{R'(n+1)^3}$, $\sqrt{\frac{R'}{n+1}}$ (Kwon, Golden, and Wasil, 1995)	$a'(n+1)^{\frac{3}{2}}$, $b'(n+1)^{\frac{3}{2}}$, $\frac{a'}{\sqrt{n+1}}$, $\frac{b'}{\sqrt{n+1}}$
	$n\sqrt{R}$	an, bn
Dispersion	$\sqrt{n(cstdev_a cstdev_b)}$ (Çavdar and Sokol, 2015)	s_a, s_b

3.4.1. In addition, measuring a and b in the latitudinal and longitudinal distances aligns approximately with the road network in the studied area in Shanghai. For cities with different road orientations, one can apply rotation transformations to establish proper orientation in the measurement of a and b.

3.3.2 Optimization-Compatible Prediction Models

After extracting the predictors, the second step is to select the appropriate prediction model that is compatible with the order assignment optimization. More specifically, the learned prediction model should yield explicit and tractable representations in the assignment decision variables. We discuss here a wide range of machine learning models including linear models, e.g., the least absolute shrinkage and selection operator (LASSO) and ridge regression, and nonlinear models, e.g., support vector regression and random forest. The more complex machine learning models, e.g., neural networks, are not compatible due to their complicated nonlinear structures.

The linear models assume a linear relationship between the target variable and predictors, e.g., the ordinary least squares (OLS). Let $\boldsymbol{\beta}$ denote the vector of coefficients for all the predictors. The LASSO approach extends the OLS regression by adding $p||\boldsymbol{\beta}||_1$, an ℓ_1-norm penalty on $\boldsymbol{\beta}$, into the loss minimization problem. The shrinkage parameter $p > 0$ is used to control the model sparsity—the greater the value of p is, the sparser the resulting coefficients will be. By contrast, ridge regression instead imposes $p||\boldsymbol{\beta}||_2^2$, which is a squared ℓ_2-norm penalty on $\boldsymbol{\beta}$. The ridge penalty (squared ℓ_2-norm) encourages highly correlated features to be averaged (J. Friedman, Hastie, and Tibshirani 2001). Moreover, the elastic net combines both types of penalties with $p(\alpha||\boldsymbol{\beta}||_1 + (1-\alpha)||\boldsymbol{\beta}||_2^2)$, where $0 < \alpha < 1$, and enjoys the properties of both LASSO and ridge regressions. These shrinkage methods are able to reduce the undesired over-fitting of the OLS approach. Once the coefficients $\boldsymbol{\beta}$ are derived, we can represent the total travel time as a linear function of the predictors, which is compatible with the order assignment optimization.

Support vector regression (SVR) stems from the support vector machine (SVM) for classification and uses the ϵ-sensitive error function as its objective (J. Friedman, Hastie, and Tibshirani 2001). In its simplest form, a linear SVR still assumes a linear relationship between the prediction variable and features. As a result, a linear SVR is also compatible with the order assignment optimization. Using kernel functions such as the polynomial and radial basis functions, SVR can map data to a higher-dimension feature space (possibly infinite-dimension), which facilitates capturing the more complicated nonlinear relationships between the target variable and predictors. However, the kernel representation makes it difficult to obtain a tractable prediction model for the following optimization procedure.

The decision tree method is another popular class of machine learning models with great prediction power. In decision trees, the feature space is divided into a set of subspaces where a simple model is fitted to each subspace. More specifically, a decision tree makes a binary partition in each split based on a tree structure. Each split can be represented by linear constraints with indicator variables. Hence, a decision tree is also optimization-compatible.

Nevertheless, a single decision tree may suffer from high variance and poor quality of out-of-sample performance. To overcome these issues, random forest runs a number of decision trees by applying random sampling to average out the noises. While the variance of the prediction shrinks, as the number of trees increases, the number of indicator variables required for the linear representation grows. Consequently, the resulting increased computational burden for the random forest may outweigh the improved prediction performance.

In summary, we identify the optimization-compatible prediction models with linear representations, including the linear models (e.g., OLS, LASSO, ridge regression, elastic net, and linear SVR) and the piecewise linear models (e.g., decision trees). For the model selection, we are able to tune the hyper-parameters as well as calculate the accuracy based on cross-validation. We apply three selection criteria: accuracy, tractability, and interpretability. In the case study of Section 3.5, we detail the selection process considering the above three criteria.

3.4 Integrating Predictors with Optimization Tools

We first detail the general framework proposed in Section 3.1.1 by formulating the order assignment problem as a stochastic optimization problem. We then discuss tractable reformulations for the travel time predictors as well as the prediction models that can be integrated with the optimization formulation. The robust optimization formulations are further developed to address the uncertainty in service time and travel time, followed by a discussion on multiperiod order assignment.

For the ease of exposition, we assume that the orders from the same location are delivered by the same driver. Subsequently, the firm's decision is to partition the set of customer locations with orders denoted by $\mathcal{I}$ into $\mathcal{I}_k \in \mathcal{I}$ for driver $k \in \mathcal{K}$. Let q_i be the associated order quantity of location $i \in \mathcal{I}$. The total number of orders served by a driver cannot exceed his capacity C. After the order assignment, the drivers freely design their routes to visit the assigned customer locations. As elaborated in the previous sections, we predict the total travel time by driver k as $l(\mathcal{I}_k)$ with the travel time predictors from $\mathcal{I}_k$. Let $y_{ik} \in \{0, 1\}$ be the binary decision variable that indicates whether customer location i is served by driver k: 1 if i is served by driver k and 0 otherwise. Subsequently, the vector $\mathbf{y}_k = (y_{ik}, \forall i \in \mathcal{I})$ defines $\mathcal{I}_k$, i.e., $\mathcal{I}_k = \{i \in \mathcal{I} : y_{ik} = 1\}$. Hence, the travel time prediction model $l(\mathcal{I}_k)$ can be represented as a function $l(\mathbf{y}_k)$ in $\mathbf{y}_k$. Note that one can also make the prediction model $l(\cdot)$ depend on k to account for the heterogeneous behaviors among the drivers.

Let the joint distribution of $\tilde{t}_i$ for all $i \in \mathcal{I}$ as $\mathbb{P}$. The order assignment problem is then formulated as the following stochastic program under the on-time performance metric (3.2) from Section 3.1.1:

$$\begin{aligned}
\min_{y_{ik}} \quad & \mathbb{E}_{\mathbb{P}}\left[\sum_{k\in\mathcal{K}}\left(\sum_{i\in\mathcal{I}_k}\tilde{t}_i + l_k - \tau\right)^+\right] \\
\text{s.t.} \quad & \sum_{k\in\mathcal{K}} y_{ik} = 1, \quad \forall i \in \mathcal{I}, \\
& \sum_{i\in\mathcal{I}} q_i y_{ik} \le C, \quad \forall k \in \mathcal{K}, \\
& l_k = l(\mathbf{y}_k), \quad \forall k \in \mathcal{K}, \\
& y_{ik} \in \{0,1\}, \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K},
\end{aligned}$$

where the first constraint ensures that location i is served and the second constraint accounts for the delivery capacity C of each driver, and the third constraint simply replaces the predicted travel time with an auxiliary decision variable.

Since evaluating the objective in this stochastic program requires full knowledge of the joint distribution $\mathbb{P}$ as well as an integration operation, one may deploy the sample average approximation (SAA) scheme. Given a set of historical samples $\mathcal{S}$, let t_i^s be the service time at location i in sample $s \in \mathcal{S}$. The SAA formulation for the order assignment is provided as follows, where we omit the constant multiplier $1/|\mathcal{S}|$ for the sample average:

$$\begin{aligned}
\min_{y_{ik}} \quad & \sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}}\left(\sum_{i\in\mathcal{I}} t_i^s y_{ik} + l_k - \tau\right)^+ \\
\text{s.t.} \quad & \sum_{k\in\mathcal{K}} y_{ik} = 1, \quad \forall i \in \mathcal{I}, \\
& \sum_{i\in\mathcal{I}} q_i y_{ik} \le C, \quad \forall k \in \mathcal{K}, \\
& l_k = l(\mathbf{y}_k), \quad \forall k \in \mathcal{K}, \\
& y_{ik} \in \{0,1\}, \quad \forall i \in \mathcal{I}, k \in \mathcal{K}.
\end{aligned}$$

The nonlinear objective function can be linearized by introducing nonnegative auxiliary variables, e.g., $\sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}}\omega_k^s$ with constraints $\omega_k^s \ge \sum_{i\in\mathcal{I}} t_i^s y_{ik} + l_k - \tau$ and $\omega_k^s \ge 0$. The challenge in solving this problem lies in obtaining a tractable reformulation of $l(\mathbf{y}_k)$ for the travel time prediction, which is detailed in the following subsections.

3.4.1 Linearization of Travel Time Predictors

We first discuss the linear representation for each class of tractable predictors summarized in Table 3.3. We denote the predictors with subscript k for driver $k \in \mathcal{K}$.

Number of locations: n_k.

We can express the number of customer locations assigned to driver k as follows:

$$n_k = \sum_{i \in \mathcal{I}} y_{ik}.$$

Distance: d, D and $\bar{d}$.

For d, which is the shortest distance between the depot and the customer locations, we can construct the following constraints:

$$d_k = \sum_{i \in \mathcal{I}} \hat{d}_i \vartheta_{ik}, \tag{3.4}$$

$$\sum_{i \in \mathcal{I}} \vartheta_{ik} = 1, \tag{3.5}$$

$$\vartheta_{ik} \leq y_{ik}, \quad \forall i \in \mathcal{I}, \tag{3.6}$$

$$d_k \leq \hat{d}_i y_{ik} + M^d(1 - y_{ik}), \quad \forall i \in \mathcal{I}, \tag{3.7}$$

where $\hat{d}_i$ is the distance between the depot and location i and ϑ_{ik}'s are positive auxiliary variables. The big number M^d can be chosen as $\max_{i \in I} \hat{d}_i$. Note that some of the above constraints may be redundant for certain prediction models. For example, with a linear prediction model, constraint (3.7) is not necessary if the coefficient of d is positive.

For D, which is the longest distance between the depot and customer locations, we can adopt constraints similar to (3.4)-(3.6) combined with

$$D_k \geq \hat{d}_i y_{ik}, \quad \forall i \in \mathcal{I}.$$

For $\bar{d}$, the average distance between the depot and customer locations, it can be expressed as

$$\bar{d}_k = \frac{\sum_{i \in \mathcal{I}} \hat{d}_i y_{ik}}{n_k} = \frac{\sum_{i \in \mathcal{I}} \hat{d}_i y_{ik}}{\sum_{i \in \mathcal{I}} y_{ik}}.$$

However, it is not yet tractable due to its nonlinearity. Observe that n_k is an integer variable with a limited range, we can linearize the above nonlinear (and non-convex) expression by using binary variables. In practice, the number of locations assigned to a driver can not be arbitrarily large and should be no larger than C. Therefore, we can safely restrict n_k to be within a certain range $\{0, 1, \ldots, N_k\}$, where $N_k \leq C$. One can interpret N_k as the maximum number of locations a driver can visit, which can be determined by the practitioner or estimated from the data. We can thus express n_k as a piecewise function with a set of auxiliary binary variables:

$$n_k = \sum_{j=0}^{N_k} j \cdot u_{kj}, \tag{3.8}$$

$$\sum_{j=0}^{N_k} u_{kj} = 1, \tag{3.9}$$

$$u_{kj} \in \{0,1\}, \quad \forall j \in \{0,1,...,N_k\}. \tag{3.10}$$

Based on the constraints introduced by Glover (1975), $\bar{d}_k$ can then be represented as

$$\bar{d}_k = \sum_{j=0}^{N_k} f_{kj}, \tag{3.11}$$

where $f_{k0} = 0$ and

$$M_{kj}^{d+} u_{kj} \geq f_{kj} \geq M_{kj}^{d-} u_{kj}, \tag{3.12}$$

$$\frac{\sum_{i\in\mathcal{I}} \hat{d}_i y_{ik}}{j} - M_{kj}^{d-}(1-u_{kj}) \geq f_{kj}, \tag{3.13}$$

$$f_{kj} \geq \frac{\sum_{i\in\mathcal{I}} \hat{d}_i y_{ik}}{j} - M_{kj}^{d+}(1-u_{kj}), \tag{3.14}$$

where M_{kj}^{d+} and M_{kj}^{d-} can be set as the upper and lower bounds on $\frac{\sum_{i\in\mathcal{I}} \hat{d}_i y_{ik}}{j}$ from the data.

Visiting area × Number of locations: a, b, a', b', an^r, bn^r, $a'n^r$ and $b'n^r$ for $r \in \{-0.5, 0.5, 1, 1.5\}$.

We start with a, the maximum latitudinal difference between a pair of customer locations. A similar discussion naturally applies to b, a' and b', which we will omit for conciseness. Note that we can express a_k as

$$a_k = \bar{a}_k + \underline{a}_k \geq 0, \tag{3.15}$$

where $\bar{a}_k$ is the maximum latitude of the assigned customer locations and $\underline{a}_k$ is the negative of the minimum latitude of the assigned customer locations. Moreover, $\bar{a}_k$ and $\underline{a}_k$ can be represented by

$$\bar{a}_k = \sum_{i \in \mathcal{I}} lat_i x'_{ik}, \tag{3.16}$$

$$\underline{a}_k = -\sum_{i \in \mathcal{I}} lat_i x''_{ik}, \tag{3.17}$$

$$\bar{a}_k \geq lat_i \cdot y_{ik}, \quad \forall i \in \mathcal{I}, \tag{3.18}$$

$$\underline{a}_k \geq -lat_i + M_i^{\underline{a}}(y_{ik} - 1), \quad \forall i \in \mathcal{I}, \tag{3.19}$$

$$\sum_{i \in \mathcal{I}} x'_{ik} = 1, \quad \sum_{i \in \mathcal{I}} x''_{ik} = 1, \tag{3.20}$$

$$x'_{ik} \leq y_{ik}, \quad x''_{ik} \leq y_{ik}, \quad \forall i \in \mathcal{I}, \tag{3.21}$$

where x'_{ik} and x''_{ik} are positive auxiliary variables, and $M_i^{\underline{a}}$ is a big number set to be $\max_{i' \in \mathcal{I}} lat_{i'} - lat_i$. Again, some of the above constraints may be redundant for the linear prediction models. For instance, if the coefficients for the predictors involving a are all positive, constraints (3.16)-(3.17) and (3.20)-(3.21) can be omitted.

We then reformulate the predictors in the form of an^r, where $r \in \{-0.5, 0.5, 1, 1.5\}$, as shown in Table 3.3. Following the discussion on $\bar{d}$, we can express n_k as a piecewise function with binary variables u_{kj} and use constraints similar to (3.11)-(3.14) for a linear representation of an^r:

$$a_k n_k^r = \sum_{j=0}^{N_k} f_{kj}^r \tag{3.22}$$

$$M_{kj}^{r+} u_{kj} \geq f_{kj}^r \geq M_{kj}^{r-} u_{kj}, \tag{3.23}$$

$$a_k j^r - M_{kj}^{r-}(1 - u_{kj}) \geq f_{kj}, \tag{3.24}$$

$$f_{kj} \geq a_k j^r - M_{kj}^{r+}(1 - u_{kj}). \tag{3.25}$$

The same reformulation procedures are applicable to bn^r, $a'n^r$ and $b'n^r$.

Dispersion: s_a and s_b.

By definition, we see that

$$s_{ak} = \frac{\sum_{i \in \mathcal{I}} \sum_{i' \in \mathcal{I}} |lat_i - lat_{i'}| y_{ik} y_{i'k}}{n_k(n_k - 1)},$$

where the product term $y_{ik} y_{i'k}$ can be linearized with a binary auxiliary variable. We can further repeat the procedures in the linearization of $\bar{d}$ and an^r to obtain the set of linear constraints for s_a and s_b.

3.4.2 Linearization of Prediction Models

As we discussed in Section 3.3.2, in this paper, we consider the optimization-compatible prediction models that are linearly representable, i.e., the linear models (e.g., OSL, LASSO, ridge regression, elastic net and linear SVR) and piecewise linear models (e.g., decision trees). With the linear prediction models, $l(\mathbf{y}_k)$ is a linear function of the predictors, i.e.,

$$l(\mathbf{y_k}) = \boldsymbol{\beta}^\top \mathbf{x}(\mathbf{y_k}),$$

where $\mathbf{x}(\mathbf{y_k})$ is the predictor vector and the coefficient vector $\boldsymbol{\beta}$ is learned from the prediction algorithm. Based on the linearization of the predictors above, we can then obtain a mixed integer linear program (MILP) for the order assignment model under the SAA scheme. The resulting optimization problem be solved by commercial optimization solvers such as CPLEX and Gurobi.

For the decision trees, each split in a tree (assuming it is applied to a single predictor) checks the branching condition on whether $x_l(\mathbf{y}_k) \geq B_l$ for a specific feature $x_l(\mathbf{y}_k)$. To track the split, binary indicators can be introduced in the disjunctive inequalities with a big number M^B:

$$\begin{aligned} x_l(\mathbf{y}_k) &\geq B_l - M^B \nu_l, \\ x_l(\mathbf{y}_k) &\leq B_l + M^B(1 - \nu_l), \end{aligned}$$

where $\nu_l = 0$ indicates $x_l(\mathbf{y}_k) \geq B$ and $\nu_l = 1$ indicates $x_l(\mathbf{y}_k) \leq B$. For a set of predictor values, we can utilize these indicator variables to locate the corresponding leaf node of a decision tree to retrieve the prediction. As a result, we can represent $l(\mathbf{y}_k)$ as a linear function of indicator variables, which also yields a MILP formulation. Note that the number of the indicator variables required grows linearly with respect to the number of nodes in a decision tree. When running multiple decision trees in the random forest model, the total number of nodes grows, so does the total number of indicator variables. More discussion about the MILP formulation of decision trees can found in Biggs and Hariss (2018) and D. Bertsimas and Dunn (2017).

In conclusion, the order assignment problem with a linear or piecewise linear prediction model can be formulated as an MILP under the SAA scheme. We will refer to this formulation as DOA-SAA (data-driven order assignment with SAA).

3.4.3 A Robust Optimization Formulation

For the DOA-SAA model to perform well, it requires a large number of samples of service times at customer locations. However, the observations of the service times at many customer locations are sparse, since the drivers do not visit every customer location every day. Moreover, its computation time grows as more samples for the SAA scheme are generated, e.g., via bootstrapping. To deal with such challenges, we develop a distributionally robust optimization model that utilizes limited distributional information and is independent of sample size in computation. More importantly, we aim to demonstrate that the predictors above-mentioned can also be integrated with the robust optimization approach, which is also widely used in the decision making under uncertainty.

Without assumptions on the specific probability distributions, the distributionally robust optimization (DRO) approach is widely used in the contexts in which the empirical data are not sufficient to calibrate the full distributional information. DRO approaches have been developed in the applications such as territory partitioning for VRPs and TSPs (Carlsson and Delage, 2013; Carlsson, Behroozi, and Mihic, 2018), bin packing problems (Y. Zhang, Jiang, and S. Shen, 2016), and fleet repositioning in vehicle sharing (He, Hu, and M. Zhang, 2019), where various ambiguity sets are proposed for the studied contexts.

Suppose the joint distribution $\mathbb{P}$ of service time $\tilde{t}_i, \forall i \in \mathcal{I}$ lies in an ambiguity set $\mathbb{F}$ that is specified by partial distributional information, i.e., the mean and covariance matrix. In particular, we construct the following moment ambiguity set with zero covariance:

$$\mathbb{F} = \left\{ \mathbb{P} \in \mathcal{P}_0(\mathbb{R}^{|\mathcal{I}|}) \;\middle|\; \begin{array}{ll} \mathbb{E}_{\mathbb{P}}(\tilde{t}_i) = \mu_i, & \forall i \in \mathcal{I} \\ \mathbb{E}_{\mathbb{P}}\left[\left(\tilde{t}_i - \mu_i\right)^2\right] = \sigma_i^2, & \forall i \in \mathcal{I} \\ \mathbb{E}_{\mathbb{P}}\left[\left(\tilde{t}_i - \mu_i\right)\left(\tilde{t}_j - \mu_j\right)\right] = 0, & \forall i \neq j \in \mathcal{I} \end{array} \right\}.$$

Note that the computational efficiency and solution performance from a robust optimization model depend on the choice of the ambiguity set. Here, we choose to use the ambiguity set $\mathbb{F}$ because it leads to a decomposable and computationally efficient optimization formulation. Thus, it serves the demonstration purpose of bridging the prediction model and a robust optimization approach. Specifically, the zero covariance estimation is supported by the data analysis in Section 3.2.2. Moreover, it allows us to decompose the objective by the drivers. It is possible to refine the ambiguity set at the expense of a larger problem size and higher computational burden. For instance, with a positive support and marginal first and second moments (e.g., Delage and Ye, 2010; Wiesemann, Kuhn, and Sim, 2014), one can show that the resulting second-order cone program formulation will involve $2|\mathcal{I}|$ cones, instead of $|\mathcal{K}|$ cones under our chosen ambiguity set $\mathbb{F}$.

Therefore, we consider the distributionally robust optimization (DRO) model that minimizes the worst-case expected total delay as follows:

$$\begin{aligned} \min_{Y} \max_{\mathbb{P} \in \mathbb{F}} \quad & \mathbb{E}_{\mathbb{P}} \sum_{k \in \mathcal{K}} \left(\sum_{i \in \mathcal{I}} \tilde{t}_i y_{ik} + l_k - \tau \right)^+ \\ \text{s.t.} \quad & \text{Constraints in DOA-SAA.} \end{aligned} \tag{DOA-DRO}$$

We then show in Proposition 3.1 that the DOA-DRO model can be reduced to having an optimization problem with an objective of a sum of K separable worst-case expected delays.

Proposition 3.1. *Let $\tilde{T}_k = \sum_{i \in \mathcal{I}_k} \tilde{t}_i$, where $\mathcal{I}_k = \{i \in \mathcal{I} : y_{ik} = 1\}$, be the projected random variable, i.e., the total service time of driver k. The DOA-DRO model is equivalent to:*

$$\begin{aligned} \min_{Y} \quad & \sum_{k \in \mathcal{K}} \max_{\mathbb{Q}_k \in \mathbb{G}_k} \mathbb{E}_{\mathbb{Q}_k} \left(\tilde{T}_k + l_k - \tau \right)^+ \\ \text{s.t.} \quad & \text{Constraints in DOA-SAA,} \end{aligned} \tag{3.26}$$

where $\mathbb{Q}_k$ is the distribution of $\tilde{T}_k$ and its ambiguity set $\mathbb{G}_k$ is given by

$$\mathbb{G}_k = \left\{ \mathbb{Q}_k \in \mathcal{P}_0(\mathbb{R}) \;\middle|\; \begin{array}{l} \mathbb{E}_{\mathbb{Q}_k}(\tilde{T}_k) = \sum_{i \in \mathcal{I}_k} \mu_i \\ \mathbb{E}_{\mathbb{Q}_k} \left[\left(\tilde{T}_k - \sum_{i \in \mathcal{I}_k} \mu_i \right)^2 \right] = \sum_{i \in \mathcal{I}_k} \sigma_i^2 \end{array} \right\}.$$

Proof. We first show that the worst-case expected total delay in the DOA-DRO model can be separable. Let $\mathbb{P}_k$ be the joint distribution of the service time at the locations assigned to driver k, i.e., $\{\tilde{t}_i\}_{\mathcal{I}_k}$ where $\mathcal{I}_k = \{i \in \mathcal{I} : y_{ik} = 1\}$. Correspondingly, we consider the ambiguity set $\mathbb{F}_k$ as the projection of $\mathbb{F}$ on $\mathbb{P}_k$. That is, $\mathbb{F}_k$ is specified as

$$\mathbb{F}_k = \left\{ \mathbb{P}_k \in \mathcal{P}_0(\mathbb{R}^{|\mathcal{I}_k|}) \;\middle|\; \begin{array}{ll} \mathbb{E}_{\mathbb{P}_k}(\tilde{t}_i) = \mu_i, & \forall i \in \mathcal{I}_k \\ \mathbb{E}_{\mathbb{P}_k} \left[\left(\tilde{t}_i - \mu_i \right)^2 \right] = \sigma_i^2, & \forall i \in \mathcal{I}_k \\ \mathbb{E}_{\mathbb{P}} \left[\left(\tilde{t}_i - \mu_i \right) \left(\tilde{t}_j - \mu_j \right) \right] = 0, & \forall i \neq j \in \mathcal{I}_k \end{array} \right\}.$$

Note that each location is assigned to a single driver. The collection of $\mathcal{I}_k, \forall k \in \mathcal{K}$ is a partition of $\mathcal{I}$, i.e., $\cup_{k \in \mathcal{K}} \mathcal{I}_k = \mathcal{I}$ and $\mathcal{I}_k \cap \mathcal{I}_{k'} = \emptyset$ for $k \neq k' \in \mathcal{K}$. Therefore, for any given $\mathbb{P}_k \in \mathbb{F}_k$ for all $k \in \mathcal{K}$, we can construct the joint distribution of service times at all locations as $\mathbb{P} = \Pi_{k \in K} \mathbb{P}_k$. It is straightforward to see that the constructed $\mathbb{P}$ belongs to the ambiguity set $\mathbb{F}$. Hence, the inner maximization problem in the DOA-DRO model can be reformulated in the following steps:

$$\begin{aligned} & \max_{\mathbb{P} \in \mathbb{F}} \mathbb{E}_{\mathbb{P}} \sum_{k \in \mathcal{K}} \left(\sum_{i \in \mathcal{I}} \tilde{t}_i y_{ik} + l_k - \tau \right)^+ \\ & = \max_{\mathbb{P} \in \mathbb{F}} \sum_{k \in \mathcal{K}} \mathbb{E}_{\mathbb{P}} \left(\sum_{i \in \mathcal{I}_k} \tilde{t}_i + l_k - \tau \right)^+ \\ & = \max_{\mathbb{P}_k \in \mathbb{F}_k, \forall k \in \mathcal{K}} \sum_{k \in \mathcal{K}} \mathbb{E}_{\mathbb{P}_k} \left(\sum_{i \in \mathcal{I}_k} \tilde{t}_i + l_k - \tau \right)^+ \\ & = \sum_{k \in \mathcal{K}} \max_{\mathbb{P}_k \in \mathbb{F}_k} \mathbb{E}_{\mathbb{P}_k} \left(\sum_{i \in \mathcal{I}_k} \tilde{t}_i + l_k - \tau \right)^+ . \end{aligned}$$

Here, the first equality follows from the linearity of expectation and the definition of $\mathcal{I}_k = \{i \in \mathcal{I} : y_{ik} = 1\}$. The second equality is established based on the previous argument—for any given $\mathbb{P}_k \in \mathbb{F}_k$, we can construct $\mathbb{P} = \Pi_{k\in K}\mathbb{P}_k \in \mathbb{F}$ (the direction "$\leq$" holds), while for any given $\mathbb{P} \in \mathbb{F}$, we can construct $\mathbb{P}_k \in \mathbb{F}_k$ by projection (the direction "$\geq$" holds). The third equality is based on the separability of the objective function and the ambiguity sets $\mathbb{F}_k$ by the decision variables $\mathbb{P}_k$.

Let $\tilde{T}_k = \sum_{i\in\mathcal{I}_k} \tilde{t}_i$, be the projected random variable. From Proposition 1 in Popescu (2007), we have

$$\max_{\mathbb{P}_k\in\mathbb{F}_k} \mathbb{E}_{\mathbb{P}_k}\left(\sum_{i\in\mathcal{I}_k}\tilde{t}_i + l_k - \tau\right)^+ = \max_{\mathbb{Q}_k\in\mathbb{G}_k} \mathbb{E}_{\mathbb{Q}_k}\left(\tilde{T}_k + l_k - \tau\right)^+,$$

where

$$\mathbb{G}_k = \left\{ \mathbb{Q}_k \in \mathcal{P}_0(\mathbb{R}) \;\middle|\; \begin{array}{l} \mathbb{E}_{\mathbb{Q}_k}(\tilde{T}_k) = \sum_{i\in\mathcal{I}_k}\mu_i \\ \mathbb{E}_{\mathbb{Q}_k}\left[\left(\tilde{T}_k - \sum_{i\in\mathcal{I}_k}\mu_i\right)^2\right] = \sum_{i\in\mathcal{I}_k}\sigma_i^2 \end{array} \right\}.$$

Therefore, we have the DRO model provided in (3.26) with the univariate distributions in $\mathbb{G}_k$. □

To derive a tractable formulation to problem (3.26), we deal with $\max_{\mathbb{P}_k\in\mathbb{F}_k}\mathbb{E}_{\mathbb{P}_k}\left(\tilde{T}_k + l_k - \tau\right)^+$ in the following lemma.

Lemma 3.1. *The inner problem* $\max_{\mathbb{Q}_k\in\mathbb{G}_k}\mathbb{E}_{\mathbb{G}_k}\left(\tilde{T}_k + l_k - \tau\right)^+$ *in* (3.26) *can be solved by the following formulation with the second-order cone constraints:*

$$\begin{aligned}
\min_{\lambda_k,\eta_k,\theta_k} \quad & \lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i + \theta_k\sum_{i\in\mathcal{I}_k}\sigma_i^2 \\
s.t. \quad & \lambda_k + (\eta_k - 1)\sum_{i\in\mathcal{I}_k}\mu_i - l_k + \tau + \theta_k \geq \left\|\begin{pmatrix} \eta_k - 1 \\ \lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}_k}\mu_i - l_k + \tau - \theta_k \end{pmatrix}\right\|_2, \\
& \lambda_k + (\eta_k - 1)\sum_{i\in\mathcal{I}_k}\mu_i - l_k + \tau \geq 0, \\
& \lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i + \theta_k \geq \left\|\begin{pmatrix} \eta_k \\ \lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i - \theta_k \end{pmatrix}\right\|_2, \\
& \theta_k \geq 0.
\end{aligned}$$

Based on Lemma 3.1, using the KKT conditions, we can formulate the DOA-DRO model as a tractable second-order cone program:

Proposition 3.2. *The DOA-DRO model can be solved by the following mixed-integer second order cone program (MISOCP):*

$$\begin{aligned} \min_{\boldsymbol{Y},\boldsymbol{\rho}} \quad & \sum_{k\in\mathcal{K}} \left(\rho_k + \sum_{i\in\mathcal{I}} \mu_i y_{ik} + l_k - \tau \right) \\ \text{s.t.} \quad & \rho_k \geq \left\| \begin{pmatrix} \sigma_1 y_{1k} \\ \vdots \\ \sigma_I y_{Ik} \\ \sum_{i\in\mathcal{I}} \mu_i y_{ik} + l_k - \tau \end{pmatrix} \right\|_2, \forall k \in \mathcal{K} \\ & \text{Constraints in DOA-SAA.} \end{aligned} \tag{3.27}$$

Problem (3.27) has $|\mathcal{K}|$ second-order cone constraints. Since we can decompose the problem by drivers under the ambiguity set $\mathbb{F}$, we are able to derive an efficient formulation for the DOA-DRO model. For the general distributionally robust optimization model without zero correlation, it is still possible to have a MISOCP formulation but with potentially more conic constraints.

Uncertainty in the Travel Time

Thus far, we have demonstrated that the prediction model for travel time can be integrated with stochastic and robust optimization tools, which consider the uncertainty in service time. Recall that the prediction model $l(\mathcal{I}_k)$ provides a point estimate on the average total travel time. Therefore, it may be necessary to account for the actual uncertainty in routing.

As the deviation in the travel time often increases for a longer tour, we incorporate the uncertainty via an error term in a multiplicative model—the actual travel time is $(1+\tilde{e}_k)l(\mathcal{I}_k)$ where $\tilde{e}_k$ is the error in the travel time for driver k. Indeed, the extension for an additive travel time model $l(\mathcal{I}_k) + \tilde{e}_k$ can be treated in a similar manner. For the DOA-SAA model, it is straightforward to utilize the empirical distribution of $\tilde{e}_k$ to generate the samples. Here, we will focus on the extension in the DOA-DRO model.

We construct the following ambiguity set $\bar{\mathbb{F}}$ with a zero mean and a variance of s_k^2 for $\tilde{e}_k$ and assume that $\tilde{e}_k$ is uncorrelated with the service time and other driver's travel time:

$$\bar{\mathbb{F}} = \left\{ \bar{\mathbb{P}} \in \mathcal{P}_0(\mathbb{R}^{|\mathcal{I}|+|\mathcal{K}|}) \;\middle|\; \begin{array}{ll} \mathbb{E}_{\bar{\mathbb{P}}}(\tilde{t}_i) = \mu_i, & \forall i \in \mathcal{I} \\ \mathbb{E}_{\bar{\mathbb{P}}}\left[\left(\tilde{t}_i - \mu_i\right)^2\right] = \sigma_i^2, & \forall i \in \mathcal{I} \\ \mathbb{E}_{\bar{\mathbb{P}}}\left[\left(\tilde{t}_i - \mu_i\right)\left(\tilde{t}_j - \mu_j\right)\right] = 0, & \forall i \neq j \in \mathcal{I} \\ \mathbb{E}_{\bar{\mathbb{P}}}(\tilde{e}_k) = 0, & \forall k \in \mathcal{K} \\ \mathbb{E}_{\bar{\mathbb{P}}}\left(\tilde{e}_k^2\right) = s_k^2, & \forall k \in \mathcal{K} \\ \mathbb{E}_{\bar{\mathbb{P}}}\left[\tilde{e}_k\left(\tilde{t}_i - \mu_i\right)\right] = 0, & \forall i \in \mathcal{I}, k \in \mathcal{K} \\ \mathbb{E}_{\bar{\mathbb{P}}}\left(\tilde{e}_k \tilde{e}'_k\right) = 0, & \forall k \neq k' \in \mathcal{K} \end{array} \right\}.$$

The DOA-DRO problem with travel time ambiguity is thus formulated as:

$$\begin{aligned}\min_{Y}\max_{\bar{\mathbb{P}}\in\bar{\mathbb{F}}} \quad & \mathbb{E}_{\bar{\mathbb{P}}}\sum_{k\in\mathcal{K}}\left(\sum_{i\in\mathcal{I}}\tilde{t}_i y_{ik}+(1+\tilde{e}_k)l_k-\tau\right)^+ \\ \text{s.t.} \quad & \text{Constraints in DOA-SAA.}\end{aligned} \tag{DOA-DROt}$$

Consequently, we can have the MISOCP reformulation shown in Proposition 3.3 below. Similar to Proposition 3.2, we have $|\mathcal{K}|$ conic constraints in (3.28). The only difference is that the term $s_k l_k$ is involved in the conic constraint for driver k.

Proposition 3.3. *The DOA-DROt model with the ambiguity set $\bar{\mathbb{F}}$ is equivalent to:*

$$\begin{aligned}\min_{\boldsymbol{Y},\boldsymbol{\rho}} \quad & \sum_{k\in\mathcal{K}}\left(\rho_k+\sum_{i\in\mathcal{I}}\mu_i y_{ik}+l_k-\tau\right) \\ \text{s.t.} \quad & \rho_k \geq \left\|\begin{pmatrix}\sigma_1 y_{1k}\\ \vdots \\ \sigma_I y_{Ik}\\ s_k l_k \\ \sum_{i\in\mathcal{I}}\mu_i y_{ik}+l_k-\tau\end{pmatrix}\right\|_2, \forall k\in\mathcal{K} \\ & \text{Constraints in DOA-SAA.}\end{aligned} \tag{3.28}$$

3.4.4 Multiperiod Order Assignment

Our discussion thus far has focused on a single-period setting for a batch of orders with the same delivery deadline, in which we put aside the impact of the current order assignment decision on the future delivery on-time performance. When the drivers are employees of the service provider (as opposed to freelance drivers for an on-demand platform), they need to return to the depot for the next batch of orders. As a result, our current order assignment decision affects when they will return to the depot, which in turn affects the on-time performance for the next batch. For instance, dispatching more drivers improves the on-time performance for the current batch, but it will make fewer drivers available for the next batch. To capture these dynamics, we can formulate a dynamic program for multiperiod order assignment.

For the ease of presentation, we highlight the key trade-offs in the dynamic model and leave out the less important operational details. Let $\mathcal{K}$ be the set of all drivers, including both the idle and en route drivers. The provider operates in $\mathcal{N}=(1,\cdots,N)$ periods with length Δ per period, e.g., $\Delta=15$ minutes for lunch delivery. In the beginning of period $n\in\mathcal{N}$, the system state is described as: 1) the order location and quantity information $(\mathcal{I}^n,\mathbf{q}^n)$ of the current batch; and 2) the driver information $\boldsymbol{\zeta}^n=(\zeta_k^n,\forall k\in\mathcal{K})$, where ζ_k^n is the remaining time for driver k to return to the depot. For example, ζ_k^n can be known by tracking the driver's real-time location information, e.g., $\zeta_k^n=5$ indicates that the estimated time for the driver's return to the depot is 5 minutes. Furthermore, τ_k^n denotes the available time window for driver k to deliver batch n. When the driver is at the depot, i.e., $\zeta_k^n = 0$, we set $\tau_k^n = \bar{\tau}_k^n$ which is the required time window for on-time delivery; otherwise, for $\zeta_k^n > 0$, we have $\tau_k^n = 0$, as the driver is not immediately available. Indeed, by introducing an indicator variable $z_k^n \in \{0,1\}$ for driver dispatch, we can simply write $\tau_k^n = \bar{\tau}_k^n z_k^n$.

Let $\mathbb{P}^n$ be the joint distribution of service times in period n and $\mathbb{Q}^{n+1}$ be the joint distribution of the customer locations and order quantity in period $n+1$. The service provider then makes the order assignment decisions $\{y_{ik}^n \in \{0,1\} : i \in \mathcal{I}^n, k \in \mathcal{K}\}$, by solving the following finite-horizon stochastic dynamic program:

$$\mathcal{H}_n(\mathcal{I}^n, \mathbf{q}^n, \boldsymbol{\zeta}^n) = \min_{\mathbf{Y}^n} \left\{ \mathbb{E}_{\mathbb{P}^n} \left\{ \sum_{k \in \mathcal{K}} \left(\sum_{i \in \mathcal{I}^n} \tilde{t}_i^n y_{ik}^n + l_k^n(\mathbf{Y}^n) - \tau_k^n \right)^+ + \mathbb{E}_{\mathbb{Q}^{n+1}} \left[\mathcal{H}_{n+1}(\mathcal{I}^{n+1}, \mathbf{q}^{n+1}, \boldsymbol{\zeta}^{n+1}) \right] \right\} \right\},$$

$$\mathcal{H}_N(\mathcal{I}^N, \mathbf{q}^N, \boldsymbol{\zeta}^N) = \min_{\mathbf{Y}^N} \left\{ \mathbb{E}_{\mathbb{P}^N} \sum_{k \in \mathcal{K}} \left(\sum_{i \in \mathcal{I}^N} \tilde{t}_i^N y_{ik}^N + l_k^N(\mathbf{Y}^N) - \tau_k^N \right)^+ \right\},$$

with the transition constraints

$$\begin{aligned}
&\zeta_k^{n+1} = \max \left\{ 0, \zeta_k^n - \Delta, \sum_{i \in \mathcal{I}^n} \tilde{t}_i y_{ik}^n + l_k^n(\mathbf{Y}^n) + r_k^n(\mathbf{Y}^n) - \Delta \right\}, \forall k \in \mathcal{K}, n \in \mathcal{N} \setminus \{N\} \\
&\tau_k^n = \bar{\tau}_k^n z_k^n, \forall k \in \mathcal{K}, n \in \mathcal{N} \\
&z_k^n \le 1 - \frac{\zeta_k^n}{M}, \forall k \in \mathcal{K}, n \in \mathcal{N} \\
&y_{ik}^n \le z_k^n, \forall k \in \mathcal{K}, n \in \mathcal{N} \\
&z_k^n \in \{0,1\}, \forall k \in \mathcal{K}, n \in \mathcal{N} \\
&\zeta_k^1 = 0, \forall k \in \mathcal{K}.
\end{aligned}$$

The term $r_k^n(\mathbf{Y}^n)$ represents the travel time from the last visited customer back to the depot, which exists only if $l_k^n(\mathbf{Y}^n)$ is measured for an open route; if $l_k^n(\mathbf{Y}^n)$ is measured to include the return trip, then the term $r_k^n(\mathbf{Y}^n)$ can be eliminated.

The first constraint updates the remaining time for the driver's return to the depot, after an elapsed time of Δ from the previous period. There are two possible scenarios:

1. $\zeta_k^n > 0$: the driver is not available for batch n. Therefore, we will not dispatch driver k, i.e., $z_k^n = 0$ as implied by the third constraint and subsequently $y_{ik}^n = 0$ according to the fourth constraint. The remaining time for the driver's return to the depot becomes $\zeta_k^{n+1} = \max\{0, \zeta_k^n - \Delta\}$.

2. $\zeta_k^n = 0$: the driver is available for batch n. Therefore, the provider might dispatch the driver for batch n. Hence, the remaining time for the driver's return to the depot is updated as $\zeta_k^{n+1} = \max\left\{0, \sum_{i \in \mathcal{I}^n} \tilde{t}_i y_{ik}^n + l_k^n(\mathbf{Y}^n) + r_k^n(\mathbf{Y}^n) - \Delta\right\}$, where $\sum_{i \in \mathcal{I}^n} \tilde{t}_i y_{ik}^n + l_k^n(\mathbf{Y}^n) + r_k^n(\mathbf{Y}^n)$ is the total delivery time for batch n. If the driver is not dispatched

in period n, then $\sum_{i\in\mathcal{I}^n}\tilde{t}_i y_{ik}^n + l_k^n(\mathbf{Y}^n) + r_k^n(\mathbf{Y}^n) = 0$ and the remaining time to return is thus $\zeta_k^{n+1} = \max\{0, -\Delta\} = 0$, as the driver is still available for period $n+1$.

The state transition is stochastic due to the uncertain service times, which can only be revealed when visiting the customers in later periods. Nevertheless, the driver information can be updated at the beginning of each period. The resulting dynamic program is complicated due to its combinatorial and stochastic nature, as well as the high dimensional state space. While it is possible to apply approximate dynamic programming or deep reinforcement learning techniques for its solution, the computational burden would make using those algorithms prohibitive. Indeed, even for a single vehicle with orders on a line, Klapp, Erera, and Toriello (2016) show the dynamic order dispatch problem "has exponentially many variables, constraints, and terms in the expectations" as an MDP with a standard LP dual reformulation. Therefore, we leave the dynamic order assignment model and solution algorithms for future research.

Instead, we propose two simple heuristics built upon the single-period optimization models to balance the on-time performance of the current and future deliveries. Note that the last period problem $\mathcal{H}_N(\mathcal{I}^N, \mathbf{q}^N, \boldsymbol{\zeta}^N)$ is exactly a single-period problem that can be solved by the DOA-SAA or DOA-DRO model. For the earlier periods, we discuss the static manpower planning and the adaptive order assignment heuristics in the following subsections.

Static Manpower Planning

This heuristic decomposes the multiperiod order assignment problem into a series of single-period order assignment problems, by planning the number of drivers K^n for each period n one day ahead.

Suppose there are N planning periods, e.g., 6 periods of 15 minutes each for lunch delivery. Let N_0 periods be the pre-allocated usage for each driver. That is, each driver will be busy for N_0 periods in one dispatch. With the target delivery time window τ (which may include the return trip to the depot), we can convert τ into a corresponding N_0 with a buffer time for the driver to comply with the labor rules as well as to include potential delays. It greatly simplifies the multiperiod problem as we disregard the dynamics of $\boldsymbol{\zeta}^n$ by setting a deterministic usage of N_0 periods for a driver in each dispatch.

With customer locations $\mathcal{I}^n$ to be visited in period n, the expected total delay with K^n drivers is $\mathcal{H}^s(K^n, \mathcal{I}^n, \mathbf{q}^n)$ by solving the single-period problem, e.g., in the DOA-SAA and DOA-DRO models. Subsequently, the expected total delay in period n, as evaluated one day ahead before the realization of $\mathcal{I}^n$, is $\mathbb{E}_{\mathbb{Q}^n}\left[\mathcal{H}^s(K^n, \mathcal{I}^n, \mathbf{q}^n)\right]$ where Q^n is the joint distribution of $(\mathcal{I}^n, \mathbf{q}^n)$. Let $\bar{K}^n$ be the total number of drivers working in period n, according to their working schedule. The static manpower planning problem is thus formulated as

$$\begin{aligned}\min_{K^n\in\mathbb{N}} \quad & \sum_{n=1}^{N} \mathbb{E}_{\mathbb{Q}^n}\left[\mathcal{H}^s(K^n,\mathcal{I}^n,\mathbf{q}^n)\right] \\ \text{s.t.} \quad & \sum_{m=n-N_0+1}^{n} K^m \le \bar{K}^n, \quad \forall n=1,\cdots,N \\ & K^n = 0, \quad \forall n = 1-N_0,\cdots,0,\end{aligned}$$

where the first constraint accounts for the driver availability—once a driver departs from the depot, he will only be available for his next dispatch after N_0 periods.

Note that K^n can only take integer values. The expected delay before the demand realization $\mathbb{E}_{\mathbb{Q}^n}\left[\mathcal{H}^s(K^n,\mathcal{I}^n,\mathbf{q}^n)\right]$ can be evaluated and stored offline. Therefore, we parametrize it as $\pi_k^n = \mathbb{E}_{\mathbb{Q}^n}\left[\mathcal{H}^s(k,\mathcal{I}^n,\mathbf{q}^n)\right]$ to denote the expected delay with k drivers in period n. With binary indicator variables x_k^n, we can rewrite $K^n = \sum_{k=1}^{\bar{K}^n} kx_k^n$. Thus, the manpower planning stage problem becomes an integer assignment problem:

$$\begin{aligned}\min_{x_k^n\in\{0,1\}} \quad & \sum_{n=1}^{N}\sum_{k=0}^{\bar{K}^n} \pi_k^n x_k^n \\ \text{s.t.} \quad & \sum_{m=n-N_0+1}^{n}\sum_{k=1}^{\bar{K}^m} kx_k^m \le \bar{K}^n, \quad \forall n=1,\cdots,N \\ & \sum_{k=1}^{\bar{K}^n} x_k^n = 1, \quad \forall n=1,\cdots,N \\ & \sum_{k=1}^{\bar{K}^n} x_k^n = 0, \quad \forall n=1-N_0,\cdots,0.\end{aligned}$$

Adaptive Order Assignment

An immediate improvement over the static manpower planning heuristic is to assign the orders adaptively on the currently realized order information. When assigning the batch of orders for $\mathcal{I}^n$ in period n, the provider weighs the trade-off between minimizing the delay in the current period and a potential shortage of drivers in the future. By looking ahead, we determine the immediate manpower planning decisions for the current period and the expected usage of drivers for future periods.

Recall that $\mathcal{H}^s\left(k,\mathcal{I}^n,\mathbf{q}^n\right)$ is the expected delay with k drivers to visit $\mathcal{I}^n$ in period n. After seeing $(\mathcal{I}^n,\mathbf{q}^n)$, it can be solved in parallel for all k using the single-period order assignment models. That is, $\mathcal{H}^s\left(k,\mathcal{I}^n,\mathbf{q}^n\right)$ can also be evaluated and parametrized for all k with $\mathcal{I}^n$ realized, together with $\pi_k^n = \mathbb{E}_{\mathbb{Q}^n}\left[\mathcal{H}^s(k,\mathcal{I}^n,\mathbf{q}^n)\right]$ computed one day ahead. Hence, at the beginning of period n, the provider can determine the number of drivers to serve $\mathcal{I}^n$ by solving:

$$\begin{aligned}
\min_{x_k^n \in \{0,1\}} \quad & \sum_{k=0}^{\bar{K}^n} \mathcal{H}^s\left(k, \mathcal{I}^n, \mathbf{q}^n\right) x_k^n + \sum_{n'=n+1}^{N} \sum_{k=0}^{\bar{K}^{n'}} \pi_k^{n'} x_k^{n'} \\
\text{s.t.} \quad & \sum_{m=n'-N_0+1}^{n'} \sum_{k=1}^{\bar{K}^m} k x_k^m \le \bar{K}^{n'}, \quad \forall n' = n, \cdots, N \\
& \sum_{k=1}^{\bar{K}^{n'}} x_k^{n'} = 1, \quad \forall n' = n, \cdots, N
\end{aligned}$$

where $x_k^{n'}$ for $n' \le n-1$ are the parameters traced back to the historical assignment decisions in the previous periods. The first constraint suggests that in each period n', the total number of busy drivers cannot exceed $\bar{K}^{n'}$. The remaining constraints define the integer value selection for the number of drivers to start their delivery in each period.

Compared to the static manpower planning heuristic, the adaptive order assignment fully utilizes the real-time information on $(\mathcal{I}^n, \mathbf{q}^n)$ as well as the available number of drivers (i.e., $\bar{K}^n$ can be iteratively updated). With x_k^n solved in this optimization formulation, the provider is able to identify the number of drivers to dispatch in the current period and obtain the corresponding order assignment decision from the solution of $\mathcal{H}^s\left(k, \mathcal{I}^n, \mathbf{q}^n\right)$. This adaptive order assignment heuristic is implemented in a rolling horizon manner—we only execute the current order assignment decision for period n and are not obliged to follow the future staffing decisions as the decision for period $n+1$ will be resolved in the next period.

3.4.5 The Branch-and-Price Algorithm and Its Computational Performance

The DOA models formulated above can be solved by commercial mixed integer solvers such as Gurobi and CPLEX. As the number of locations ($|\mathcal{I}|$) and the number of drivers ($|\mathcal{K}|$) grow, the assignment problem becomes increasingly difficult to solve. However, the order assignment decision for food delivery service needs to be obtained in real-time, e.g., within the 20-minute food preparation time window. Thus, the standard MILP/MISOCP formulation may fail to deliver good-quality solutions in time. To overcome this issue, we utilize the structure of the order assignment problem and formulate it as a set partitioning problem, which allows us to develop an efficient branch-and-price algorithm.

Set Partitioning Formulation

Let the subset of locations assigned to a driver be associated with a cost, i.e., the expected delay. The order assignment problem is to find the partitioning of the locations with the minimum total cost (expected total delay). As a result, the order assignment problem can be cast as the set partitioning master problem (MP):

$$\begin{aligned}
\min_{\boldsymbol{z}} \quad & \sum_{j\in\mathcal{J}} c_j z_j && \text{(MP)}\\
\text{s.t.} \quad & \sum_{j\in\mathcal{J}} \delta_{ij} z_j = 1, \quad \forall i \in \mathcal{I}, && (3.29)\\
& \sum_{j\in\mathcal{J}} z_j = K, && (3.30)\\
& z_j \in \{0,1\}, \quad \forall j \in \mathcal{J},
\end{aligned}$$

where $\mathcal{J}$ is the set of all the possible subsets that satisfy the cardinality constraints. δ_{ij} is 1 if location i belongs to subset j, and 0 otherwise. Constraints (3.29) ensure that every location is covered by a subset and constraint (3.30) ensures the number of selected subsets equal to K. For SAA (or DRO), the subset cost c_j is the sample average delay (or the worst-case expected delay) of subset j.

MP involves an exponential number of variables (columns) since the number of possible subsets grows exponentially in the number of locations. Instead of enumerating all the possible subsets and solve the entire MP, column generation provides a way to generate the candidate subsets and can be used to solve the LP relaxation of MP when a stop routine is invoked. Specifically, column generation solves the following pricing subproblem at each iteration:

$$\begin{aligned}
\min_{\bar{\boldsymbol{y}}} \quad & -\pi_o - \sum_{i\in\mathcal{I}} \pi_i \bar{y}_i + c(\bar{\boldsymbol{y}}) && \text{(SP)}\\
\text{s.t.} \quad & \text{Constraints in DOA-SAA or DOA-DRO,}
\end{aligned}$$

where π_o is the dual value for constraint (3.30) and π_i's are the dual values for constraints (3.29) in MP. The constraints in SP are similar to those in DOA-SAA and DOA-DRO except that the subscript k is dropped.

Note that in the case of DOA-DRO, the subproblem is also a MISOCP but with fewer binary variables and only one conic constraint. Thus, the subproblem can be solved efficiently. If the optimal objective of SP is negative, we add to MP the subset corresponding to $\bar{\boldsymbol{y}}^*$, i.e., $\{i \in \mathcal{I} : \bar{\boldsymbol{y}}_i^* = 1\}$, and then solve the new MP. We repeat this process until the optimal objective of SP is nonnegative, implying that the LP relaxation of MP is solved. For more details about set partitioning and column generation, we refer readers to Barnhart et al. (1998).

Algorithm Overview

We start the algorithm from an initial pool of subsets that is a feasible partitioning of the customer locations. The initial partitioning is found by solving the order assignment problem by replacing the delivery travel time with its lower bound $(a + b + d)$, i.e., the minimum travel time required to cover all the customer locations. This simpler problem removes many hard constraints and allows us to find a good feasible partitioning quickly.

The optimal solution from solving the LP relaxation of MP with column generation is not necessarily integral and applying a branching rule is required. However, a standard branching rule that sets a fractional variable to $\{0, 1\}$ can fail, since preventing a subset j from reappearing in SP (corresponding to setting $z_j = 0$) is difficult. To resolve this issue, we adopt the branching rule introduced by Ryan and Foster (1981) that can be added to SP in branches. Basically, this branching rule detects a pair of subsets, S_1 and S_2, with fractional values in the solution, and two locations, i and j, with one in $S_1 \cap S_2$ and the other in S_1/S_2. In the two branches, the left branch adds constraints such that i and j can only be in the same subset while the right branch adds constraints such that i and j can not be in the same subset.

Following the suggestions of Mehrotra, Johnson, and Nemhauser (1998), we use a depth-first-search (DFS) to select the next node to solve. Additionally, for all the nodes except the root node, we do not solve the LP relaxation to optimality. Instead, we stop the column generation once the objective value of the LP relaxation is less than the objective value at the root node.

Comparison to the MILP and MISOCP Formulations

To test the efficiency of using branch-and-price algorithm to solve the DOA models, we generate random instances with different characteristics as follows: (a) n customer locations are drawn uniformly from a square plane $[0, 10] \times [0, 10]$; (b) The depot location is set to be at the center $(5, 5)$; (c) The number of orders from each location is drawn from a Poisson distribution with $\lambda = 2$ and the service time at each location follows an exponential distribution with Beta $\in \{3, 5\}$. For the SAA model, we choose the sample size to be 20. For both DOA-SAA and DOA-DRO, the travel time prediction model is assumed to be $0.5\bar{d} + 0.5D + 0.1n + a\sqrt{n} + b\sqrt{n} + 0.4an + 0.4bn$, where $\bar{d}$ and D are measured by the Manhattan distance metric. Each driver has a capacity of 30 (orders) and the number of locations assigned to a driver is bounded by 8. The number of drivers is set by the ceiling function $\lceil \kappa \times \underline{I}_C \rceil$ with $\kappa \in \{1.2, 1.5\}$, where $\underline{I}_C$ is the minimum number of drivers needed according to the capacity restriction. We test the computational performance with various combinations of n, Beta, and κ. Ten random instances are generated for each combination of the parameters. We run the experiments in Python 3.6 using Gurobi 8, on a 3.50 GHz Xeon CPU. We choose the termination criteria to be (i) optimality gap is below 1%, or (ii) CPU time exceeds 20 minutes.

Table 3.4 reports the computational comparison. The "Time" column presents the solution time in seconds and the "Gap" column shows the optimality gap at the termination. It is obvious that the branch-and-price algorithm delivers significant improvement in both solution speed and quality. Furthermore, Figure 3.5 provides summary information on how the solution quality evolves over time across different runs. We observe that the solution

Table 3.4: Computational comparison between the branch-and-price algorithm with the MILP and MISOCP (Time in seconds)

			SAA				DRO			
			MILP		Branch-and-Price		MISOCP		Branch-and-Price	
n	β	κ	Time	Gap	Time	Gap	Time	Gap	Time	Gap
30	3	1.2	1113	90.0%	183.7	0.00%	1200	97.2%	465.4	0.00%
	3	1.5	170.5	0.00%	43.1	10.0%	1200	96.6%	319.7	0.07%
	5	1.2	1200	98.9%	166.3	0.00%	1200	97.0%	269.2	0.18%
	5	1.5	1200	98.9%	107.0	0.10%	1200	96.7%	185.3	0.13%
40	3	1.2	1200	100%	311.6	2.23%	1200	99.1%	969.6	7.74%
	3	1.5	1200	100%	618.8	9.52%	1200	98.7%	915.7	5.36%
	5	1.2	1200	99.5%	304.5	2.06%	1200	98.6%	864.8	3.69%
	5	1.5	1200	99.5%	271.4	0.05%	1200	98.4%	933.6	3.74%

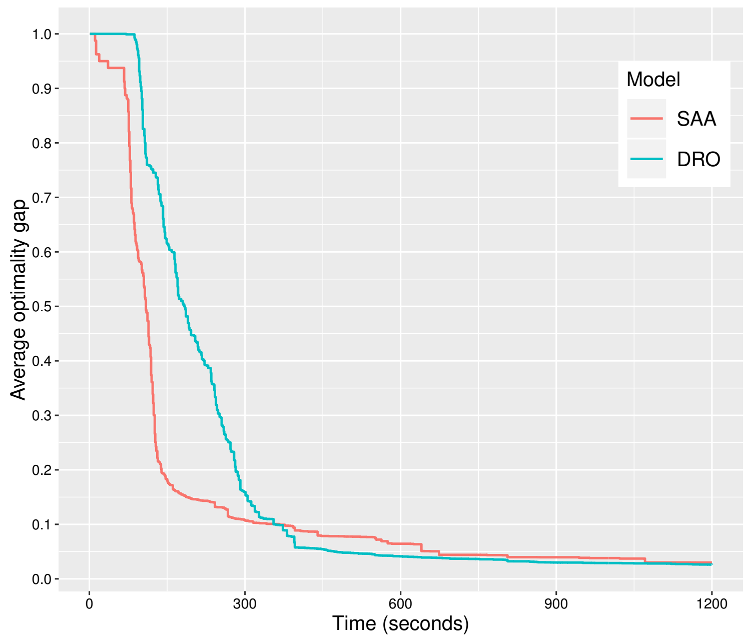


Figure 3.5: Optimality gap in the solution time for the branch-and-price algorithm

quality starts to improve quickly within the first 100 seconds, and the optimality gap drops to below 15% after 300 seconds. Thus, even when a shorter solution time window is desired, the branch-and-price algorithm can still deliver good solutions with an early termination.

3.5 Experiments in a Real-World Case Study

We conduct extensive experiments in this section to comprehensively evaluate the performance of the proposed order assignment models. In this section, we first introduce the setting of our numerical study based on the delivery data set from a food service provider in Shanghai. We then present how we select the prediction model and the detailed evaluation procedures, followed by the evaluation results and managerial insights.

Data: We consider all the batches that contain more than 20 orders during the lunch peak period, with cutoff times in [10:00 am, 10:15 am, . . . , 11:15 am]. We obtain 167 batches in total, of which 134 batches are randomly sampled as the training set and the remaining 33 batches serve as the test set (i.e., an 80/20 split for training and test). The average number of orders per batch is 90.

Preprocessing: We apply a clustering algorithm to each batch of orders in the test data based on their geographical proximity; this is to manage the practical problem size and alleviate the complexity in the actual delivery. For example, the locations in the same building, even if there are slight differences in latitude and longitude measurements, will be considered as one customer location. It is also economical to serve nearby locations by one driver instead of multiple drivers.

Specifically, we apply minimax linkage hierarchical clustering, which is an agglomerative clustering algorithm that builds trees in a bottom-up approach. It starts with the singletons and gradually merges the two closest clusters stage by stage, until only one cluster remains. The resulting binary tree is commonly displayed as a dendrogram, as shown in Figure 3.6. Each leaf node represents a data point and is placed at the bottom of the tree with height 0. Each interior node indicates a merging and the corresponding height is equal to the distance between the clusters merged at that node. A key choice for such clustering algorithms is the distance measure between two clusters. Common measures include the complete, single, average, and centroid linkage (see J. Friedman, Hastie, and Tibshirani 2001 for example). The minimax hierarchical clustering uses a different linkage, defined as follows

$$d(G,H) = \min_{x \in G \cup H} d_{\max}(x, G \cup H),$$

where G and H are clusters and $d_{\max}(x, G \cup H)$ is the maximum distance between the point x and a point in cluster $G \cup H$. This linkage measures the minimax radius of the resulting merged cluster. The most central point of the merged cluster (in terms of minimizing $d_{\max}$) is called the prototype. Minimax hierarchical clustering is proved to have many desirable properties including interpretability and robustness (see Bien and Tibshirani 2011 for more details). In our application, we can restrict the minimax radius of the obtained clusters by cutting the dendrogram at a certain height, so that each resulting cluster contains only the locations within that distance.

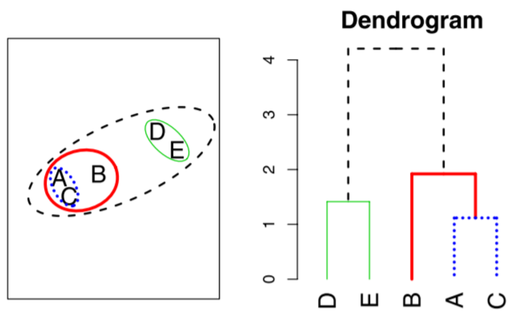


Figure 3.6: A dendrogram with 5 locations (Bien and Tibshirani 2011)

We choose to cut the dendrogram at height 0.15, which implies that the radius of the resulting clusters can not exceed 0.15 km. This choice is made to match the average street block size in the inner ring of Shanghai (Pan and Cao 2015). After clustering, the number of clusters in each batch varies from 15 to 52. The average number of orders from each cluster is 3.0 while the average number of meal boxes ordered from each location is 4.5. In rare cases, a driver may not be able to fully serve a cluster of locations due to the capacity limit. In such a scenario, we split the orders from that cluster into multiple subsets, such that all the subsets except the last one will be served by dedicated drivers to their full capacity. Then, only the last subset will be considered in the order assignment together with other orders.

Training and Testing: The prediction models are trained and selected only using the training data. Then the prediction model estimated from the training data will be integrated into the DOA-SAA and DOA-DRO models. The order assignment decision is derived and evaluated for each batch in the test set.

Sample Generation and Estimation: An important input to the SAA models is the set of sampled service times at different locations. We generate samples from the training set by bootstrapping from the observed set of service times at each (clustered) location. As the distribution of the service time does not exhibit normal behavior, bootstrapping is helpful in approximating the distribution without analytical functional assumptions. In the case of the robust model, sample mean and variance are estimated from the observations to construct the ambiguity set.

SAA Replication and Solution Selection: Following the SAA procedure proposed in Kleywegt, Shapiro, and Homem-de-Mello (2002), we run 20 replications of SAA for each batch of orders in the test set. The sample size in SAA is chosen from $\{10, 15, \ldots, 40\}$ such that a good-quality solution to each replication can be obtained within 20 minutes. The optimality gap is estimated by additional 300 samples drawn independently from the SAA samples. Then we select the best solution from the 20 replications that yields the lowest optimality gap estimate.

3.5.1 Prediction Model Selection

We test 6 different prediction models: LASSO, ridge regression, elastic net, linear SVR, RBF SVR (radial basis function kernel), and random forest. The feature vector contains the tractable features listed in Table 3.3 including $(\bar{d}, D, d, n, a\sqrt{n}, b\sqrt{n}, an, bn, s_a, s_b)$ while some other features are reserved for the evaluation. 5-fold cross-validation is performed to select the best hyper-parameters for all the models (e.g., the shrinkage parameters in LASSO and ridge regression, and the number of trees grown in the random forest). All of the training and validation procedures are implemented in R.

Table 3.5 summarizes the average cross-validation mean squared errors (MSE), which is the average of $\frac{\sum_{s\in\mathcal{S}}(l_s-\hat{l}_s)^2}{|\mathcal{S}|}$, where l_s denotes the actual travel time and $\hat{l}_s$ denotes the predicted travel time for sample s in each fold $\mathcal{S}$ of the training set. In addition, we report the MSE on the test set, which indicates the generalization power of different models. As a benchmark, we also calculate the MSE of a scaled TSP model, wherein the scaling factor is derived by a linear regression.

Table 3.5: Performance evaluation of the prediction models and the TSP solution

Prediction model	Training set cross-validation MSE (min^2)	Test set MSE (min^2)
LASSO	20.2	15.7
Ridge regression	19.7	15.7
Elastic net	19.0	15.8
Linear SVR	19.9	16.4
RBF SVR	21.4	15.1
Random forest	18.8	15.0
TSP solution	63.8	51.5
Scaled TSP solution	46.4	42.5

Based on the cross-validation and test results, all the prediction models achieve significantly lower errors than both the TSP and scaled TSP solutions. In particular, the linear models deliver promising prediction performance, although slightly inferior to random forest. Among the linear models, elastic net has the lowest cross-validation error while LASSO and ridge regression attain the lowest test error. As discussed before, the random forest with a large number of trees does not yield a very efficient linear representation and thus not suitable for our real-time application. By contrast, the linear models with nonlinear features enable tractable representations, and maintain great interpretability.

To further examine the prediction performance, Figure 3.7 presents the mean absolute percentage error (MAPE) of LASSO and the scaled TSP solutions on the test set with a different number of customer locations. MAPE, defined as $\frac{\sum_{s\in\mathcal{S}} |l_s - \hat{l}_s|/l_s}{|\mathcal{S}|} \times 100\%$, is unit free and measures the relative prediction errors. We observe that LASSO has significantly smaller errors than those of the scaled TSP solution, especially when the number of locations exceeds 8. It is worthwhile to note that the MAPE of LASSO is mostly below 25% and varies less compared to that of the scaled TSP solution, while the MAPE of the scaled TSP solution can be relatively large for certain number of locations (e.g., over 50% when the number of locations is 10).

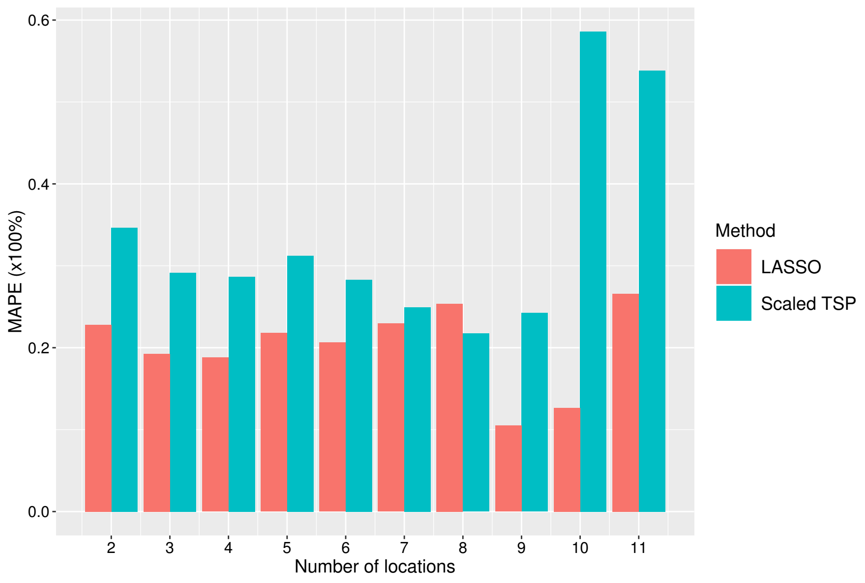


Figure 3.7: MAPE of LASSO versus scaled TSP solution on the test set.

To conclude this subsection, we make two further remarks. First, both elastic net and LASSO can obtain a sparse coefficient vector and screen out the unnecessary features (automatic feature selection by regularization). In particular, LASSO requires the least features, which leads to extraordinarily efficient optimization models. Second, unlike the TSP solution that consistently underestimates the delivery travel time, all the prediction models enable less biased prediction results. Although scaling the TSP solution can reduce the bias, there is still a significant gap between the TSP solution and the actual delivery travel time. In the remaining of this section, we use LASSO as a representative prediction model in our DOA models, to be compared with the benchmark order assignment models introduced in the next subsection.

3.5.2 Order Assignment Model Evaluation

We compare the performance of the assignment decision produced by the DOA models with two benchmark models that assume the drivers follow the TSP routes. With the shortest route assumptions, the benchmark models essentially become stochastic vehicle routing problems with stochastic service times (SVRPs). Similar to the DOA models, we can formulate an SAA model of the SVRP and its distributionally robust counterpart to minimize the worst-case expected delays. We refer to the benchmarks as VRP-based models, VRP-SAA for an SAA model and VRP-DRO for a DRO model respectively. Based on the observation in 3.5.1, we mainly consider the VRP-based models with a scaling factor learned from the training set. We refer to the scaled VRP-based models using SAA and DRO as sVRP-SAA and sVRP-DRO, respectively. Note that our calculation of the delay does not account for the return trip from the last visited location to the depot. As discussed in Section 3.2.1, we introduce a dummy node into the location network, which helps drop the return trip from the route. We employ the commonly used illegal subtour elimination scheme in solving the SVRP (see Laporte, Louveaux, and Mercure 1992 for more details). Both the VRP-SAA and VRP-DRO models are solved with the branch-and-price algorithm, which has been shown to be an efficient exact method of solving VRPs (Fukasawa et al. 2006). Please refer to the detailed formulations of VRP-SAA and VRP-DRO in the Appendix. Additionally, the DRO models with travel time uncertainty can be formulated similar to those in Subsection 3.4.3.

To assess the out-of-sample performance of the order assignment decisions, we construct the validation sets of the service times by drawing independent samples from the test set. We do not estimate the probability density functions because the number of observations at some locations is very limited, which makes many parametric and nonparametric estimation methods inappropriate. For each problem instance in the numerical test, we generate 1,000 validation sets of the service times.

To evaluate the delivery delays from implementing the proposed solutions in the validation sets, we need to estimate the actual delivery travel time for a set of locations, which may not be observed in history. Since the driver behavior is difficult to simulate, we propose a two-step data-driven evaluation procedure. 1) Given a set of assignments, we first search in all the historical delivery routes to see whether the assignment has been observed or not. If yes, the corresponding observed travel time will be returned as the estimate. Otherwise, 2) we use the prediction result from the random forest with an additive perturbation as the estimate where the perturbation is sampled from the residuals of random forest.

3.5.3 Results and Discussion

In this subsection, we first report the performance of the DOA models versus the VRP-based models on the test set. We then compare them to the observed assignment decisions. The impact of the sample size on the solution quality of SAA and implementation of the multiperiod order assignment are further discussed.

Performance Comparison.

We set the delivery time window to 55 minutes, which corresponds to the practical scenario when food preparation takes 20 minutes. Since the studied delivery service provider assigned the orders in a manual way, there is no specific rule to decide the number of drivers. For the purpose of comparison, we set the number of drivers $|\mathcal{K}|$ used for each batch of orders to be $\min\{\underline{I}_C * 1.5, \underline{I}_C + 4\}$, where $\underline{I}_C$ is the minimum number of drivers needed according to the capacity restriction.

The out-of-sample average delay is calculated for each order assignment model. Figure 3.8 presents the relative improvement of the different models (reduction of total delays) on the test set using VRP-SAA as a baseline, where the average improvement is summarized in Table 3.6. We observe that the scaled VRP models improve on VRP-SAA only by a small scale, and the performance gap is not significant. With the DOA models, we get a significant improvement between 35.9% and 38.2% on average. To examine the performance of DOA models more closely, we also calculate the percentage of instances that a DOA model achieves the best performance in all the tested models, which is listed in Table 3.7. The average gap between a DOA model's performance and the best performance is also reported. We find that DOA-DRO achieves the best performance most frequently while DOA-DROt attains the smallest average gap. This observation implies that adding the additional robustness of travel time can help achieve a better overall performance with the price of missing some of the best solutions, potentially due to being overly conservative.

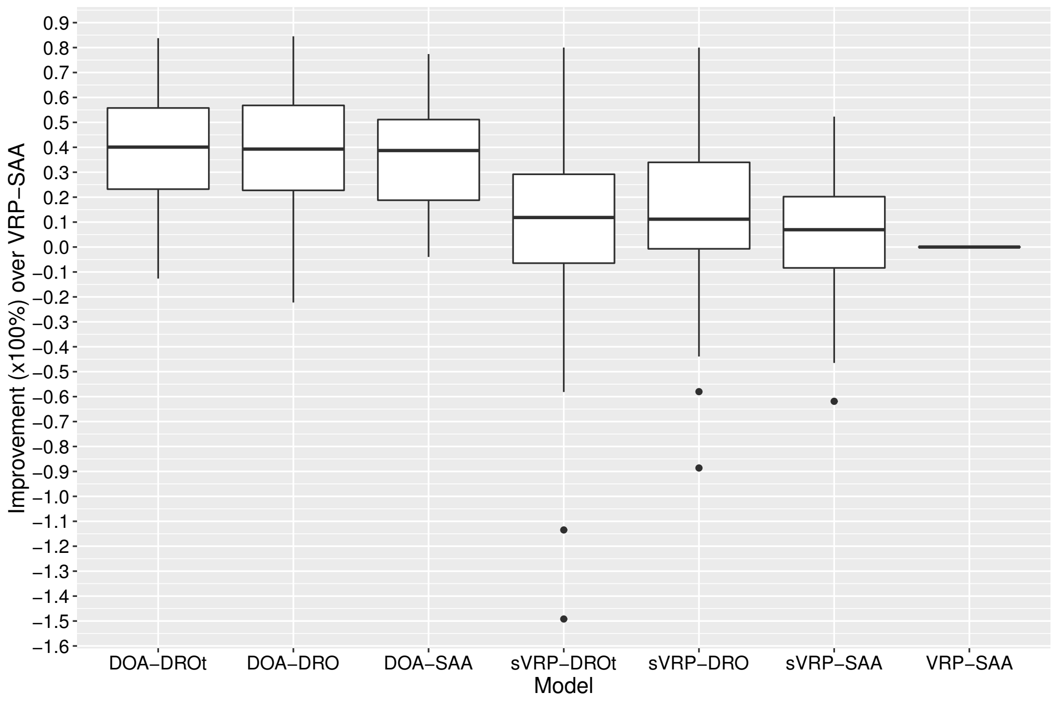


Figure 3.8: Out-of-sample performance (improvement over the VRP-SAA) of the DOA models and VRP models.

We proceed to compare the DOA models to the order assignment decisions observed in the test set (the implemented order assignment decision). The number of drivers is set to be

Table 3.6: Average Improvement of the DOA models and scaled VRP models over the VRP-SAA

	DOA-DROt	DOA-DRO	DOA-SAA	sVRP-DROt	sVRP-DRO	sVRP-SAA
Avg Improvement over VRP-SAA	**38.2%**	36.9%	35.9 %	3.84%	11.6%	6.63%

Table 3.7: Performance comparison of DOA models

	DOA-DROt	DOA-DRO	DOA-SAA
% of instances achieving the best performance	27.3%	**39.4%**	18.2 %
Average gap with the best performance	**9.94%**	12.0%	16.2%

the same as the actual number of dispatched drivers. As described in the evaluation step, the travel times of the currently implemented order assignments are obtained from the historical data directly, while these of the DOA solutions are first checked with the historical records and if no matched record is found, we make a prediction using the random forest (with the added noises). Indeed, for 32.9% of the DOA solutions, we are able to obtain their travel times from the historical records. The out-of-sample evaluation result is shown in Figure 3.9. We observe that all the DOA models deliver significantly better performance than the current manual assignment decision, and the two DRO models show greater improvement than the SAA model. Among the two DRO models, DOA-DROt's improvement is more stable while DOA-DRO delivers high improvements more often. Specifically, DOA-DROt and DOA-DRO reduce the total delivery delay by 75.3% and 74.7%, respectively on average relative to the current assignment.

Moreover, to understand the difference between the generated DOA assignment and the current assignment decision, we present the distribution of the average total delivery time of different drivers for both the DOA-DROt and the current assignment in Figure 3.10. We observe that DOA-DROt balances the delivery tasks between different drivers such that most of them have a similar total delivery time, while the current assignment decision results in a relatively large variation of delivery time among the drivers. The large variation implies that some drivers take too long tours or visit too many customer locations, which contributes to extraordinarily long delays.

The Impact of the Sample Size.

The performance of SAA depends on the number of samples used to approximate the expected objective function value. To obtain a good-quality solution, the number of samples is suggested to be growing logarithmically on the size of the feasible set (Kleywegt, Shapiro,

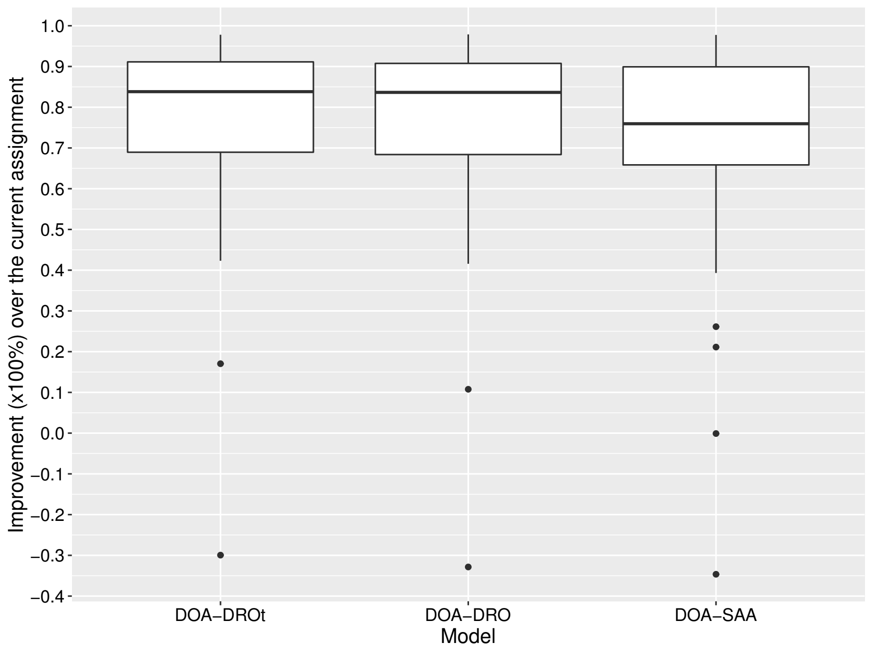


Figure 3.9: Out-of-sample performance comparison of the DOA models versus the current assignment decision.

and Homem-de-Mello 2002). However, the growing number of samples may make the problem computationally inefficient for our application. Furthermore, since the real delivery tour is unknown, our approximated objective function may not approach the true expected objective value even when the number of samples is sufficiently large. To understand the impact of the sample size on the performance of the SAA models, we calculate the average out-of-sample total delay of the SAA models with different sample sizes on a selected set of instances from the test set, as shown in Figure 3.11. We observe that as the sample size grows, the out-of-sample performance of DOA-SAA improves and approaches the reference point of DOA-DRO and DOA-DROt. In contrast, the out-of-sample performance of VRP-SAA and sVRP-SAA can be worse off when a larger sample size is used. In particular, the gap between the performance of VRP-SAA and DOA-SAA reaches the highest when the sample size reaches the maximum of 180.

As discussed before, the objective function of VRP-SAA and sVRP-SAA may not approximate the true objective function well due to their biases in estimating the travel time. When the number of samples increases, the objective function of VRP-SAA and sVRP-SAA may become more biased. As a result, the solution to VRP-SAA and sVRP-SAA can "overfit" to the samples and deviate from the true optimal solution. Instead, the objective function of DOA-SAA has a better approximation performance and is closer to the true objective value. Thus, its solution approaches the true optimal solution for a large sample size.

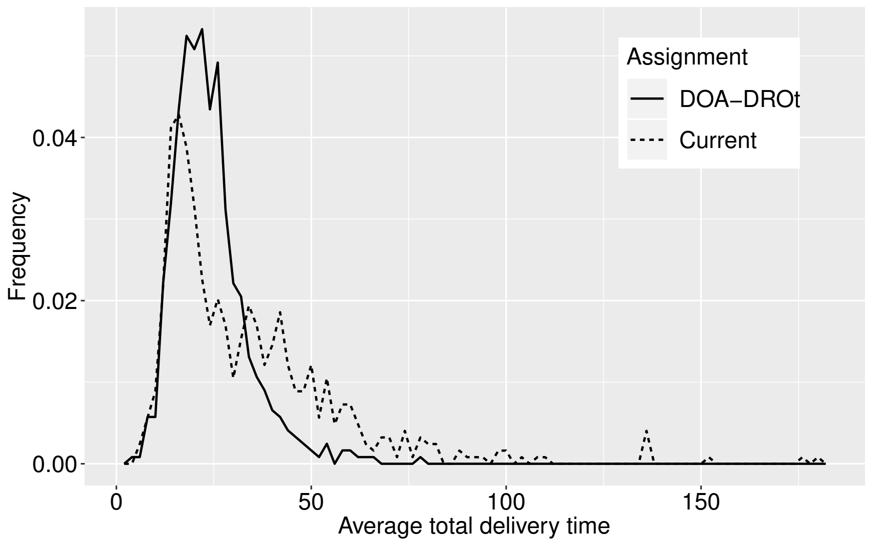


Figure 3.10: Average total delivery time of drivers from the DOA-DROt and current assignments.

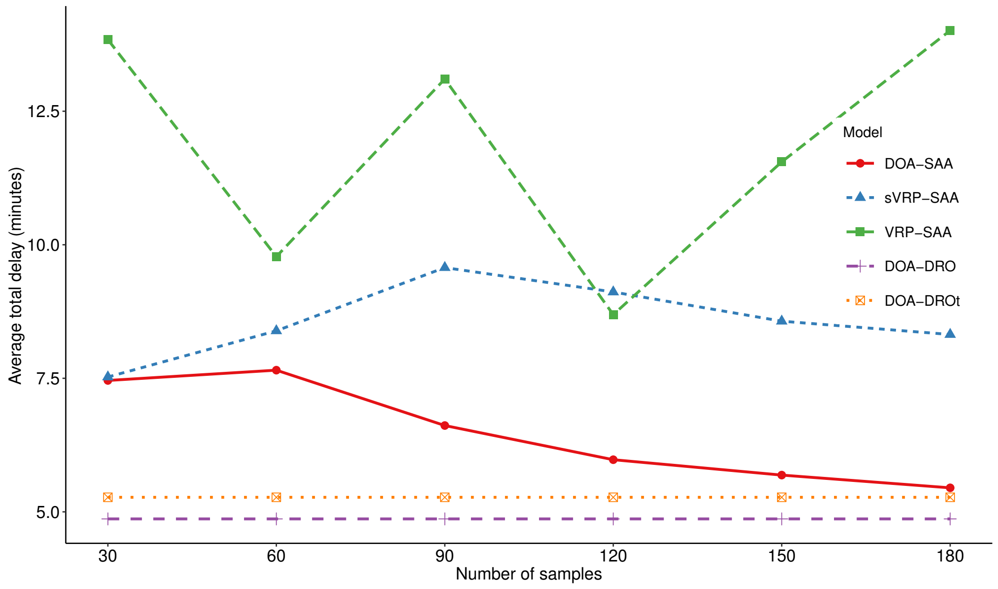


Figure 3.11: Impact of the sample size on the SAA models.

Multiperiod Order Assignment

We also implement both the static manpower planning heuristic as well as the adaptive rolling-horizon heuristic for the peak periods from 10:00 am to 11:15 am, which accounts for more than 85% of the orders for the company. We set $N = 6$ and $N_0 = 5$ correspondingly. The expected delay function $\pi_k^n = \mathbb{E}_{\mathbb{Q}^n}[\mathcal{H}^s(K^n, \mathcal{I}^n, \mathbf{q}^n)]$ is calculated by drawing $(\mathcal{I}^n, \mathbf{q}^n)$

from the observations in the previous week. We use the objective value of DOA-DROt as a proxy for $\mathcal{H}^s(K^n, \mathcal{I}^n, \mathbf{q}^n)$. The two heuristics are compared to a clairvoyant policy that foresees the realizations of $(\mathcal{I}^n, \mathbf{q}^n)$ in later periods and thus provides a lower bound for the expected delay. Table 3.8 presents the average gap between the two heuristics and the clairvoyant policy by varying the total number of drivers $(\bar{K})$. Both the static and adaptive heuristics achieve a satisfactory performance as indicated by the small performance gaps. The adaptive heuristic attains a better performance than that of the static heuristic, reducing the performance gap by more than 50%.

Table 3.8: Performance gap between the two heuristics and the clairvoyant policy for multiperiod order assignment

$\bar{K}$	Adaptive policy	Static Policy
40	2.25%	8.38%
45	2.50%	5.01%
50	1.57%	5.52%
55	0.99%	2.85%

3.6 Conclusion and Future Research

In this study, we propose a framework that integrates travel time predictors with existing optimization techniques to generate the order assignment decisions for last mile delivery. This idea of integrating predictors with optimization can be generalized to other applications where some elements are not directly representable in mathematical programming, e.g., the driver's routing behavior. The use of the prediction model is practical—it captures the driver's routing behavior and avoids an excessively complicated modeling of it.

With the order assignment problem as a showcase, we demonstrate that the linear and piecewise linear prediction models, together with the tractable predictors, can be successfully integrated with stochastic and robust optimization tools. The resulting formulations are an MILP for the stochastic optimization with SAA and an MISOCP for the distributionally robust optimization, which can be efficiently solved via the branch-and-price algorithm proposed by exploiting the problem structure in the order assignment. Furthermore, we provide two simple heuristics, the static manpower planning and adaptive rolling-horizon heuristics, for multiperiod order assignment problems based on the single-period solutions.

Using real-world delivery data from our partner food service provider, we conduct extensive experiments to evaluate the performance of our proposed DOA models, in comparison with the benchmark VRP-based models. The significant on-time performance improvement by the DOA models, compared to the VRP-based models, suggests the importance of considering the driver's routing behavior in making an order assignment. Moreover, the inferior solutions from the VRP-based models, which ignores the driver's routing behavior, cannot be corrected by using a large sample size in their SAA formulations. Finally, in the experiments for multiperiod order assignment, we show the effectiveness of both simple heuristics built on the single-period solutions.

This paper considers that the firm dispatches the orders to the drivers after orders have been prepared. While such an order assignment is responsive to the realized demand and improves the utilization of drivers, it requires extra effort in operations and technology from the service provider, compared to the static order assignment strategy: first divide the service region and allocate dedicated drivers to their own subregions, e.g., delivery zoning and territory partitioning e.g., Carlsson (2012) and Carlsson and Delage (2013). The practical advantage of static order assignment is that drivers become more familiar and effective in visiting their assigned customer locations, which leads to a smoother delivery experience. Therefore, a natural extension is to develop a static order assignment model. A key challenge that arises from the static order assignment is the uncertainty in customer locations to visit in each batch, in addition to the uncertainty in the service times. While our SAA model can be directly extended by using the indicators for the customer location realizations in all the samples, it would be interesting to study a DRO model for static order assignment with both uncertainty in both the customer location and service time.

Chapter 4

Urban Bike Lane Planning

4.1 Introduction

Urbanization is a global trend. More than half of the world's population now lives in towns and cities. It is projected that about 56% of the developing world and 82% of the developed world will be urbanized by 2030, which equals to 5 billion people (The Economist, 2012). Much of this urbanization will unfold in Africa and Asia, bringing huge social, economic and environmental transformations (United Nations Population Fund, 2018). The main challenges brought by the fast urbanization include traffic congestion and air pollution. The growing number of cars exceeds the city's traffic carrying capacity and thus causes severe traffic congestion around the city. In the meanwhile, the vast amount of emissions generated by moving cars worsens the air quality and poses serious public health problems. For Beijing, cars accounted for 30% of city's self-generated pollutants contributing to air pollution (South China Morning Post, 2018).

In addition to driving cars, cycling is a popular urban transit mode for daily commute. For instance, 30% of people go to work by riding bikes in Cambridge, UK (Department for Transport, UK, 2015). According to U.S. Census Bureau (2012), 864,883 people cycled to work in the United States. Cycling is promoted by many countries as it benefits the city residents in many different aspects. First, cycling is free of pollution and the large-scale adoption of cycling can benefit the urban environment. Second, the popularity of cycling could alleviate the traffic congestion and improves the overall traffic condition. Third, cycling is also a more affordable and healthy transit mode than driving cars. As shown by the City of Copenhagen (2010), mortality is reduced by 30% in adults who cycle to and from their workplace on a daily basis, which can translates to huge savings of health care cost. In the recent years, the convenience of using shared bikes also stimulates the popularity of cycling among city residents. It is estimated that more than 75 communities now have bike sharing programs, and the number of shared-bike rides in U.S. cities has achieved 34.6 million in 2017 (The Wall Street Journal, 2018).

To promote cycling and reduce bike crashes, bike lanes have been planned and constructed by city planners. In Amsterdam, for example, there are 250 miles of dedicated bike lanes in use. In the U.S., city governments are rolling out better bike lanes in a large scale. For instance, Chicago has planned to build 50 miles of bike lanes by 2019 (Chicago Tribune, 2015). Cities from developing countries, such as China and India, are also investing heavily in protected bike lanes (The Seattle Times, 2016). The construction of bike lanes improves the safety for cyclists, car drivers as well as pedestrians. It is reported that fatality rate is less than a tenth as high in the countries with well designed cycling road networks as in the countries without cycling friendly infrastructures (Pucker, 2001).

Traditionally, bike lanes are planned based on experience and surveys. As smart phones are widely used, the GPS human mobility data becomes more available and makes it possible to utilize this fine-grained data in bike lane planning. Most recently, dock-less bike sharing systems are expanding quickly across the world. In this system, people can use smart phones to pick up and drop off bikes at arbitrary locations. Bikes in the dock-less system are often embedded with GPS devices so the bike trajectories can be recorded. These bike trajectories record the detailed travel pattern of bike riders and reflect the actual bike travel demand of city residents. Compared to the conventional origin-destination data, the trajectory data reveals the footprints on the road network, which is essential to the bike lane planning as most bike lanes are constructed along the existing road network.

This paper presents and solves the bike lane planning problem using the detailed bike trajectory data. The bike lane planning problem decides on which road segments of the existing road network to construct bike lanes, aiming to balance the two main objectives: 1) *coverage*: cover as many bike trips as possible and 2) *continuity*: build more continuous bike lanes to minimize interruptions. We summarize our main results and contributions as follows.

1. We present the bike lane planning model in view of the cyclist's utility functions based on the trajectory data. We start from a simple adjacency-continuity utility function and then discuss the general class of utility functions. The choice of the utility functions is flexible to characterize the trade-off between the coverage and continuity objectives. To the best of our knowledge, our work is the first to formalize the general bike lane planning model.

2. For the simple adjacency-continuity utility function, we show that the resulting bike lane planning model has a supermodular objective function and admits an efficient mixed integer linear program (MILP) formulation. For the general utility functions, we show that under reasonable conditions, the objective function is also supermodular and the resulting problem yields a polynomial-time solvable Lagrangian dual. Furthermore, we provide a linear programming approach to the Lagrangian relaxation subproblem and propose an efficient algorithm for the general utility functions. We also make the first attempt to account for the cyclists' responsive behaviors in the bike lane planning problem by modeling their route choices.

3. We present a real-word case study based on the collaboration with an urban planning institution and a dock-less bike sharing company. We collect and preprocess a large bike trajectory data set, and test our models and algorithms via extensive numerical experiments. The numerical experiments validate the efficiency of the proposed algorithms and deliver insightful comparison results between different models. We show how the topology of bike lane network would change according to the utility functions and provide quantitative measures to analyze the trade-off between coverage and continuity. We also highlight the importance of understanding cyclists' responsive behaviors in the bike lane planning.

The remainder of the paper is organized as follows. Section 4.2 reviews the related literature and Section 4.3 describes the trajectory data set and provides summary statistics. Section 4.4 develops the bike lane planning model and presents tractable formulations and structural results, which motivates efficient algorithms. Section 4.5 is devoted to the real-world case study and insights derived from numerical experiments. Section 4.6 concludes the paper and presents several future research directions.

4.2 Related Literature

Smart city operations have received growing attention from the operations management community. Different aspects of the smart city movement have been studied, including smart grid, smart transportation, and smart retail. The "smart" parts of these aspects often come from innovative technologies that can disrupt the present urban environment and city operations, or new data collection and data analytics tools to inform better understanding and decisions for city management. For instance, Y. Zhang, M. Lu, and S. Shen (2018) study a promising scheme where vehicle-to-grid (V2G) electricity selling is integrated in electric vehicle sharing systems, as enabled by new technological development. They provide important strategic planning and operational tools to support the advancement of this scheme. Shared mobility, where shared passenger cars are deployed to provide ride and logistics services, has been examined from various perspectives such as pricing (Taylor 2018; Bai et al. 2018), admission control (Afeche, Z. Liu, and Maglaras 2018), repositioning (He, Hu, and M. Zhang, 2019) and last-mile delivery (W. Qi, L. Li, et al., 2018), among others. In retailing, recent development of IT and big data tools have enabled innovative omni-channel strategies (Gao and Su, 2016; Harsha, Subramanian, and Uichanco, 2019) and data-driven pricing and logistics policies (Perakis et al., 2018; S. Liu, He, and Z.-J. M. Shen, 2019). We refer readers to Mak (2018) and W. Qi and Z.-J. M. Shen (2019) for thorough reviews of papers in smart city operations. Our paper is among the first to develop rigorous analytical models to tackle an urgent city planning problem built upon new mobility data sources, and thus promotes the smart city vision.

There has been an increasing focus on the planning and control of bike facilities, in particular, the optimization of bike-sharing systems in recent years. The majority of the existing literature tackles the operational problems of bike sharing systems, such as balancing bike stock levels among bike stations to satisfy temporal and spatial demand. For example, Shu et al. (2013) addresses the bike deployment and redistribution problems with network flow formulations. O'Mahony and Shmoys (2015) study similar rebalancing problems as well as routing and clustering problems with different integer programming techniques. In addition, empirical approaches have been applied to shed lights on the optimal configuration of bike share systems (Kabra, Belavina, and Girotra, 2018). However, few papers consider the roads used by cyclists to travel between bike stations, which is another fundamental deciding factor of a bike-sharing system's success. Noticeably, the above three papers only utilize the origin-destination information of bike rides while our paper incorporates a fine-grained bike trajectory data set, which makes it possible to capture more detailed travel patterns of bike riders.

Traditionally, bike lane planning is not built upon reliable real-world data. A prominent approach of traditional bike-lane planning is to select road segments (on which to construct bike lanes) by evaluating them with respect to a set of predetermined guidelines and criteria. Dondi et al. (2011) propose a context sensitive approach and make the selection based on the analysis of the visual effects, environmental contexts and traffic considerations of road segments. Rybarczyk and C. Wu (2010) select road segments with a modified simple additive weighting method to calculate an overall score for each road segment based on its rankings among all road segments for each criterion in a predetermined set of weighted criteria. Since the guidelines and criteria in these papers are decided based on expertise or specific needs, there is a potential risk of subjective bias. Another classical approach is to select road segments based on cyclists survey results that reveal their preferences. One such approach is built on stated preference surveys, in which respondents are asked to imagine their preferences in some hypothetical cases (Tilahun, Levinson, and Krizek, 2007; Stinson and C. Bhat, 2003; Hunt and Abraham, 2007). However, results from the stated preference surveys could be inaccurate as they do not represent cyclists' actual choices. Researchers also adopted revealed preference surveys, in which respondents reveal their actual choices (Sener, Eluru, and C. R. Bhat, 2009; Howard and Burns, 2001; Hyodo, Suzuki, and Takahashi, 2000). Results from revealed preference surveys are also subject to sampling biases and small sample biases.

In addition to subjective and survey based methods, there are a few papers that develop analytical methods for the bike lane planning problem. Epperson (1994) analyzes the road level of service standard for bike use, which can be then used to guide bike lane design. Lin and Yang (2011) consider a strategic designing problem for bike sharing, in which the location of bike stations and bike lanes are decided to minimize the system-wide operating cost. Their model only accounts for the origins and destinations while ignoring the road network. As a result, their model can not be directly used to guide the practical bike lane construction. Most recently, Bao et al. (2017) propose several heuristics to decide the bike lane locations by maximizing a specific score function under budget and connectivity constraints. Our paper, on the other hand, proposes a general and tractable modeling framework for the bike lane planning, and derives structural results about the resulting planning problem. The structural results motivate efficient algorithms with empirically validated performance.

From a modeling perspective, our paper is related to the general class of facility location problems. While many traditional facility location models are focused on locating warehouses/distribution centers/retailing stores to maximize profits or minimize operational costs (Snyder and Z.-J. M. Shen, 2019), we consider locating bike lanes on the existing road network to maximize riders' utility. Hence our paper echoes the emerging location models in nonprofitable operations and healthcare operations (Cho et al., 2014; Chan, Z.-J. M. Shen, and Siddiq, 2017).

4.3 Data Description and Analysis

We obtain a GPS bike trajectory data set via our collaboration with the urban planning institution of Zhuhai city and a major bike sharing company operating in that city. Zhuhai is a medium-size city of China with a population of 1.67 million, where both station-based and dock-less bike sharing systems have been deployed. Zhuhai residents take hundreds of thousands of rides every day. To promote cycling and improve the safety for cyclists, Zhuhai city government is planning to construct protected/dedicated bike lanes in the following years. We collected the bike trajectories from the bike sharing company in 2017 and 2018, which are then mapped to the road network extracted from OpenStreetMap (Geofabrik, 2018). Each trajectory contains a timestamp that indicates the start time of a trip and a series of GPS coordinates that were recorded every 5 seconds. We loaded and visualized the trajectories in ArcGIS Pro and used its *planarize* feature to split roads into separate road segments at intersections.

Preprocessing. The original data set includes the bike trajectories taken by users who registered in Zhuhai. We first removed the trajectories that started or ended outside the urban area of Zhuhai as well as the trajectories with duration shorter than one minute to obtain representative trajectories.

Since our trajectory data and road network are in different coordinate systems, we also used the Python package *eviltransform* to convert trajectory coordinates from GCJ-02 to WGS-84. To minimize the overlap of road segments and maintain the geometric property of trajectories, we removed short road segments whose lengths are shorter than 35 meters. After preprocessing, the data set has 96,631 bike trajectories in total.

Mapping Trajectory Coordinates to Roads To map the GPS coordinates to the road network, we generate a near table in ArcGIS that finds the nearest road segment to a coordinate point. Then each coordinate is associated with a road segment ID. As a result, each trajectory is transformed to a series of road segments (IDs), which serves as the input to the bike lane planning model.

Summary Statistics. Figure 1 presents the distribution of trajectory duration of our data set. The average duration of the trajectories is 1,043 seconds (17.4 minutes), and the majority of trajectories have duration shorter than 20 minutes. This is because most trajectories are limited to the urban and residential areas of Zhuhai. We identify 3,735 road segments from the trajectories in total. The average length of the road segments is 195.3 m, and more than 60% of road segments have lengths under 200 meters. Most roads are constructed in the urban area of Zhuhai with high density of population, and thus intersections are close to each other in this area.

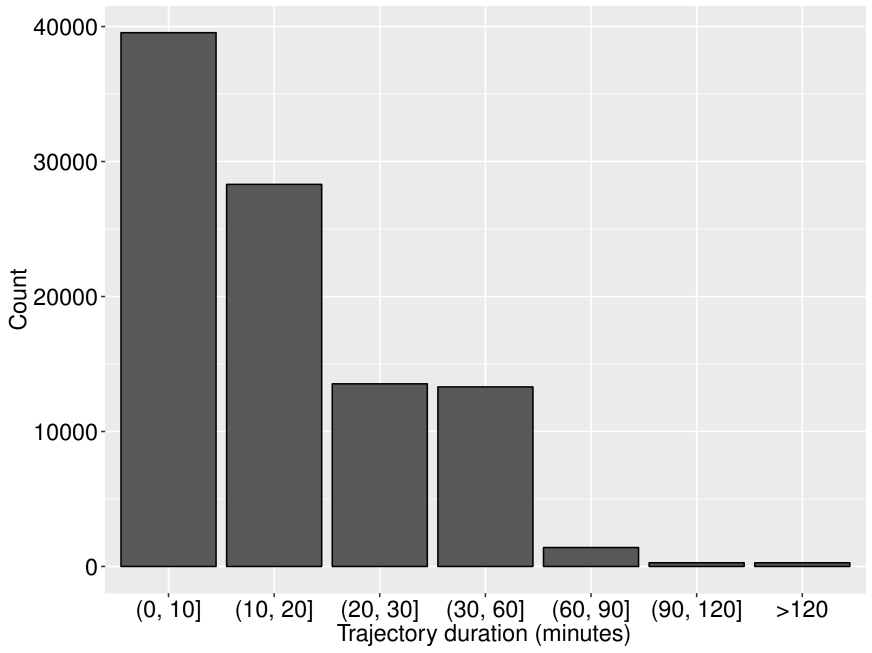
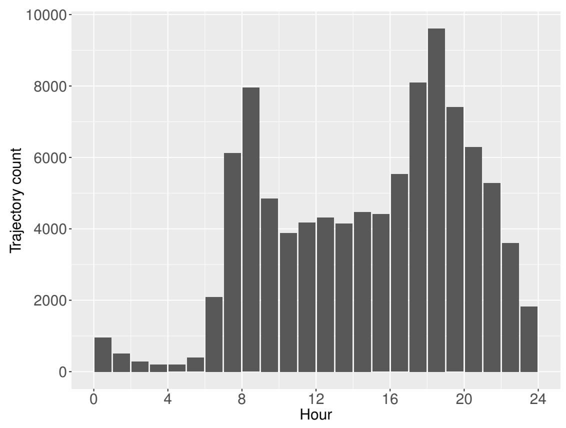


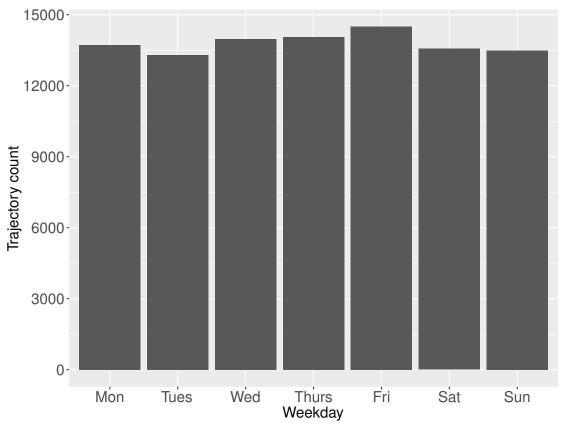
Figure 4.1: Bike trajectory duration distribution.

Temporal Distribution. Figure 4.2a shows the distribution of bike trajectories across different hours of a day. It can be observed that there are two demand peaks: one occurs in the morning (7 to 9:00 am) and the other occurs in the evening (5:00 to 8:00 pm). These two peaks correspond to the commute rush hours. We also note that the bike trip demand falls gradually after the evening peak and still remains substantial until midnight, which can be attributed to people who engage in leisure activities after work. Figure 4.2b shows the distribution of bike trajectories across different days of a week. We observe that the bike trip demand is almost stable throughout a week, while there is a small peak on Fridays. which is possibly due to the rise in leisure activities after work in the advent of weekend.

Spatial Distribution. Figure 4.3a shows the density of the origins of bike trajectories on a map of Zhuhai. We can see that the majority of trajectories begin in four areas in Zhuhai, which include the financial district, shopping district and residential districts of the city. Correspondingly, Figure 4.3b shows the density of the destinations of bike trajectories on a map. It can be observed that popular destinations of trajectories coincide with the four areas that are also popular origins. This is because many people spend a portion of their days staying at these places on a regular basis (e.g. workplaces, shopping malls, homes). So bike lanes should be prioritized in these popular areas.

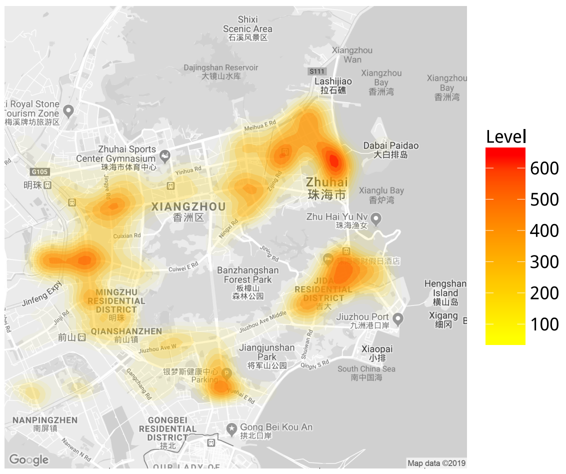


(a) Hour distribution

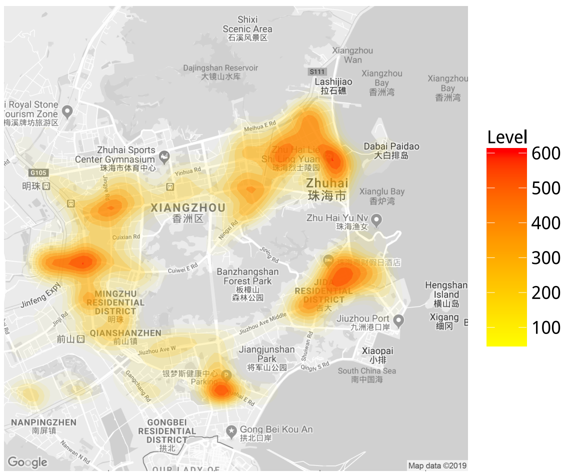


(b) Weekday distribution

Figure 4.2: Bike trajectory temporal distribution.



(a) Origins



(b) Destinations

Figure 4.3: Density heatmap of origins and destinations in the city.

In addition to origins and destinations, our data also reveals the usage of different road segments in bike trajectories. Figure 4.4 shows the spatial distribution of the road segment usage frequency on the street map. It can be observed that the locations of the road segments with high usage match with those of popular trajectory origins and destinations. We also observe that the most popular road segments are spread out over the city, which implies that simply maximizing the coverage of bike trips would result in a highly discontinuous bike lane system.

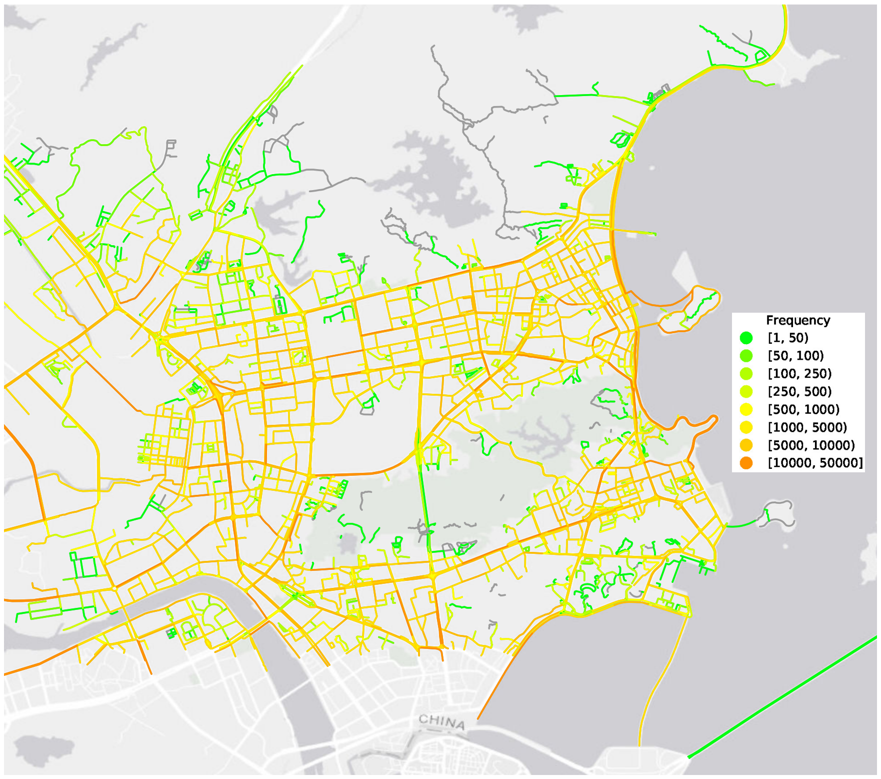


Figure 4.4: Road segment usage spatial distribution.

4.4 Bike Lane Planning Model

We first introduce the bike lane planning model based on the bike trajectory data with a specialized utility function. We then analyze the structural properties of the model and discuss several important generalizations.

4.4.1 Adjacency-Continuity Utility Maximization

Let V be the set of all road segments that have been visited by bike riders in our data set and M be the set of cyclists. We say two road segments i and j are neighbors, $(i, j) \in N$, if they are connected. We can also interpret $(i, j) \in N$ as road intersections. For each road segment $i \in V$, we use d_i to denote the associated number of bike trips. Similarly, each pair of connected road segment $(i, j) \in N$ is also associated with the number of trips that go through the corresponding intersection, d_{ij}.

We consider two main objectives of designing bike lanes inspired by the literature and our communication with the biking community:

1. Constructed bike lanes should be able to cover as many bike trips as possible. (*coverage objective)*

2. Constructed bike lane network should enable continuous and smooth riding experience for cyclists. (*continuity objective*)

Continuity is preferred for both the cyclists and government. Krizek and Roland (2005) show that the discontinuity of bike lanes that can cause great discomfort to cyclists. And the discontinuity at the intersections also generate potentially higher crash risks. For the government, discontinuous bike lanes can pose management challenges as well as construction difficulties. The coverage objective and the continuity objective are often conflicting with each other, as shown by Bao et al. (2017). Maximizing only the coverage may lead to very dispersed bike lanes while maximizing only the continuity can leave many cyclists uncovered. So we need to find the ideal trade-off between the two objectives.

Now we formalize the two objectives from the cyclist's perspective. For a cyclist m, we use $r_m = \{i_m^1, \ldots, i_m^{n_m}\}$ to denote the ordered set of road segments (i.e. trajectory) she traveled through, where $i_m^1, \ldots, i_m^{n_m} \in V$ and n_m is the number of road segments traveled by m ($|r_m|$). The cyclist receives a positive utility if there is a bike lane on the road segment, i.e. $x_i = 1$. Also, she gets an additional λ utility if the bike lanes are continuous along an intersection, i.e. $x_i = x_{i+1}$. So her gained utility of traveling through r_m from the bike lane construction plan x is

$$v_x(r_m) = \sum_{i=1}^{n_m} x_i + \lambda \sum_{i=1}^{n_m - 1} x_i x_{i+1}.$$

Summing over the utility functions of all cyclists gives

$$\sum_{m \in M} v_x(r_m) = \sum_{i \in V} d_i x_i + \lambda \sum_{(i,j) \in N} d_{ij} x_i x_j$$

where d_i is the number of trajectories going through road segment i and d_{ij} is the number of trajectories going through the intersection (i, j). So $\sum_{i \in V} d_i x_i$ stands for the total number of covered road segments (with bike lanes) weighted by the demand, and $\sum_{(i,j) \in N} d_{ij} x_i x_j$ is the additional continuity utility for two adjacent bike lanes weighted by the travel demand. Since the above utility function measures the continuity utility along two adjacent road segments, we call this utility function as the adjacency-continuity (AC) utility function. Note that here we treat each road segment equally regardless of the length for the ease of exposition and the discussion can be easily extended to consider the impact of length on the utility.

Based on the AC utility function, we propose a bike lane planning model that takes into account both the coverage and the continuity requirement. Let $x_i \in \{0,1\}$ denote the bike lane construction decision variable: $x_i = 1$ if a bike lane is planned at road segment i and $x_i = 0$ otherwise. And we use c_i to denote the construction cost of building a bike lane on road segment i. The bike lane planning model (BL) can be formulated as an integer program (IP):

$$\begin{aligned} \max_x \quad & \sum_{i \in V} d_i x_i + \lambda \sum_{(i,j)\in N} d_{ij} x_i x_j, && \text{(BL-AC)} \\ \text{s.t.} \quad & \sum_{e_i \in V} c_i x_i \leq B, && (4.1) \\ & x_i \in \{0,1\}, \quad \forall i \in V. && (4.2) \end{aligned}$$

The objective function measures the cyclists' welfare from bike lanes. The parameter $\lambda \geq 0$ determines the relative continuity benefit to the cyclist. A higher λ means continuity is more desirable and thus a more continuous bike lane network would be proposed. Constraint (4.1) is the budget constraint that ensures the total construction cost does not exceed the allowable government budget B. In practice, there may be other constraints that limit the construction of bike lanes in certain regions, which can be incorporated as needed. Furthermore, it is possible that building bike lanes along certain roads may reduce the capacity for car traffic flows, and hence worsening the traffic condition. To account for this, we can add additional penalty terms to our objective function. The detailed modeling of this traffic impact is out of the scope of this paper and we leave it for future research.

Analysis

When $\lambda = 0$, the problem reduces to the classical Knapsack problem, which is NP-hard. Otherwise, the problem is a special case of the 0-1 quadratic knapsack problem (QKP), where the coefficient matrix for the quadratic terms have a sparse structure. If d_{ij} is separable and can be decomposed as $d_{ij} = \tilde{d}_i \tilde{d}_j$, then the objective function is referred to as the *half-product* function. The maximization of the half-product function over a knapsack polytope is known to admit a *Fully Polynomial-Time Approximation Scheme* (FPTAS). However, no FPTAS is available for the general non-separable objective function.

Nevertheless, similar to the general QKP, it can be shown that the objective function with $d_{ij} \geq 0$ is supermodular.

Lemma 4.1. *When $\lambda \geq 0$ and $d_{ij} \geq 0$ for all $(i,j) \in N$, the objective function of BL-AC is supermodular.*

The proofs of Lemma 4.1 and other results in this section are presented in the Appendix. The supermodularity result implies that the problem can be solved efficiently without the budget constraint. So we can adopt the Lagrangian relaxation methodology to relax the budget constraint. The resulting Lagrangian dual is given as

$$\min_{u\geq 0} \Phi(u), \tag{4.3}$$

where

$$\Phi(u) = \max_{x_i\in\{0,1\}} \quad \sum_{i\in V} d_i x_i + \lambda \sum_{(i,j)\in N} d_{ij} x_i x_j - u(\sum_{i\in V} c_i x_i - B). \tag{4.4}$$

Here $\Phi(u)$ is the Lagrangian relaxation of BL-AC for $u \geq 0$. Based on the result of Gallo and Simeone (1989) and Chaillou, P. Hansen, and Mahieu (1989), $\Phi(u)$ can be solved in polynomial time.

Proposition 4.1. *When $\lambda \geq 0$ and $d_{ij} \geq 0$ for all $(i,j) \in N$, the Lagrangian dual of BL-AC can be solved in polynomial time.*

More specifically, one can show that the Lagrangian dual is a piece-wise linear convex function with at most $|V|$ break points. And each $\Phi(u)$ is equivalent to a maximum flow problem (Chaillou, P. Hansen, and Mahieu, 1989). There are other Lagrangian relaxation methods for QKP that rely on relaxing different sets of constraints (e.g. Caprara, Pisinger, and Toth 1999), we refer readers to Pisinger (2007) for an extensive review. Since the Lagrangian dual only provides an upper bound, we may still need to perform branch and bound to get an exact solution. Alternatively, we can get an equivalent mixed integer linear programming (MILP) formulation to BL-AC, which can deliver satisfactory computational performance with the commercial MILP solvers.

MILP Formulation

Although the objective function of BL-AC is nonlinear, we can linearize the product terms in BL-AC by replacing $x_i x_j$ with y_{ij}, and derive the following MILP formulation for bike lane planning:

$$\max_{x} \quad \sum_{i\in V} d_i x_i + \lambda \sum_{(i,j)\in N} d_{ij} y_{ij}, \tag{BL-AC-MILP}$$

$$\text{s.t.} \quad y_{ij} \geq x_i + x_j - 1, \quad \forall (i,j) \in N, \tag{4.5}$$

$$y_{ij} \leq x_i, \quad \forall (i,j) \in N, \tag{4.6}$$

$$y_{ij} \leq x_j, \quad \forall (i,j) \in N, \tag{4.7}$$

$$\sum_{i\in V} c_i x_i \leq B, \tag{4.8}$$

$$0 \leq y_{ij} \leq 1, \quad \forall (i,j) \in N, \tag{4.9}$$

$$x_i \in \{0,1\}, \quad \forall i \in V. \tag{4.10}$$

Constraints (4.5)-(4.7) ensure that $y_{ij} = 1$ if $x_i = x_j = 1$ and 0 otherwise. Constraints (4.5) are redundant when d_{ij}'s are positive. BL-AC-MILP is ready to be solved using commercial solvers such as Gurobi and CPLEX. Some additional useful constraints can be introduced to tighten the formulation, although they are redundant for the integer program. For instance, the following set of constraints are valid (Helmberg, Rendl, and Weismantel, 2000):

$$\sum_{i \in V/\{j\}} c_i y_{ij} \leq (B - c_j) x_j, \quad \forall j \in V. \tag{4.11}$$

As shown later in the computational experiment, the above formulation can be solved efficiently.

4.4.2 General Utility Functions

The aforementioned utility function assumes the continuity utility only applies to two adjacent road segments. We now discuss a more general class of cyclist utility functions with the consideration of continuity beyond adjacency. In essence, the continuity utility may also depend on the size of continuous bike lanes, which can not be captured by the AC utility function.

Example 1. *Consider a cyclist riding through $r = \{1, 2, 3, 4, 5\}$ and two bike lane construction plans, namely A and B: plan A builds bike lanes on $\{1, 2, 4, 5\}$ and plan B builds on $\{1, 2, 3, 5\}$. Under the AC utility function, the cyclist's utility from both plans are the same: $4 + 2\lambda$. However, plan B may be more preferable to the cyclist if the marginal benefit from the continuity is increasing in the size of continuous bike lanes.*

Given a trajectory r and a bike lane construction plan x, let $S_x(r)$ denote the set of maximal continuous road segments with bike lanes on r. For instance, if $r = \{1, 2, 3, 4, 5\}$ and bike lanes are constructed on $\{1, 3, 4, 5\}$ ($x_1 = x_3 = x_4 = x_5 = 1$ and $x_2 = 0$), then $S_x(r) = \{\{1\}, \{3, 4, 5\}\}$. We define a general utility function as

$$v_x(r) = \sum_{s \in S_x(r)} f(|s|), \tag{4.12}$$

where $f(\cdot)$ is an increasing function. Under the adjacency-continuity utility function, $f(|s|) = |s| + \lambda(|s| - 1) = (\lambda + 1)|s| - \lambda$, which is a linear function of $|s|$. And the score function used by Bao et al. (2017) is a special case of (4.12), wherein $f(|s|) = |s|\alpha^{|s|}$ with $\alpha \geq 1$. We will refer the bike lane planning model with the general utility function as BL-GU. Although maximizing the general utility function is often challenging due to the nonlinearity, we can show that the utility function (4.12) has a desirable structure when $f(\cdot)$ is further assumed to be convex.

Theorem 4.2. *If $f(\cdot)$ is an increasing convex function, $v_x(r)$ defined in (4.12) is supermodular.*

The convex assumption of $f(\cdot)$ is consistent with the notion that cyclists receive additionally more benefits by riding through more continuous bike lanes. For example, utility functions such as $f(|s|) = |s|\alpha^{|s|}$ $(\alpha > 1)$ are supermodular. With supermodular utility functions, the general utility maximization problem over the budget constraint has a polynomial-time solvable Lagrangian dual problem.

Corollary 4.2.1. *If $f(\cdot)$ is an increasing convex function, maximizing the utility function $v_x(r)$ defined in (4.12) over a budget constraint yields a polynomial-time solvable Lagrangian dual.*

However, different from the AC utility case, each iteration of the Lagrangian relaxation under the general utility function is not equivalent to a maximum flow problem. Instead, we can use a general supermodular maximization oracle such as Fujishige's minimum-norm-point algorithm (Fujishige, 2005). Then the computational performance of solving the Lagrangian dual problem heavily depends on the efficiency of the supermodular maximization oracle. In our case, the minimum-norm-point algorithm is not computationally efficient.

Nevertheless, the bike lane planning problem using the general utility function (4.12) can be formulated as a MILP.

Proposition 4.2. *Under the general utility function (4.12), the bike lane planning problem can be solved as a MILP.*

Note that the MILP formulation is attainable without assuming $f(\cdot)$ is convex. Specifically, the general utility function can be represented as

$$v_x(r) = \sum_{l \in L(r)} \beta_l \prod_{i \in l} x_i \tag{4.13}$$

with properly chosen coefficients β_l, where $L(r)$ is defined to include all the possible sets of continuous road segments on r. Each element $l \in L$ is a set of continuous road segments. Here $L(r)$ is different from $S_x(r)$ in the sense that $L(r)$ is independent of x. For instance, given $r = \{1, 2, 3\}$, then $L(r) = \{\{1\}, \{2\}, \{3\}, \{1, 2\}, \{2, 3\}, \{1, 2, 3\}\}$ and $v_x(r) = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_{1,2} x_1 x_2 + \beta_{2,3} x_2 x_3 + \beta_{1,2,3} x_1 x_2 x_3$, where coefficients β's can be calculated as

$$\begin{aligned}
&\beta_1 = \beta_2 = \beta_3 = f(1),\\
&\beta_{1,2} = \beta_{2,3} = f(2) - 2f(1),\\
&\beta_{1,2,3} = f(3) - 2(f(2) - 2f(1)) - 3f(1) = f(3) - 2f(2) + f(1).
\end{aligned}$$

More generally, $\beta_l = f(|l|) - 2f(|l| - 1) + f(|l| - 2)$ for a nonempty l (the proper definition requires $f(-1) = 0$). When $f(\cdot)$ is an increasing convex function, all β's are nonnegative. Since function (4.13) only involves product terms with binary variables, we can linearize them to get a MILP in a similar fashion to BL-AC-MILP. We call this formulation BL-GU-MILP.

$$\max_{x} \quad \sum_{m\in M}\sum_{l\in L(r_m)} \beta_l y_l, \tag{BL-GU-MILP}$$

$$\text{s.t.} \quad y_l \geq \sum_{i\in l} x_i - (|l|-1), \quad \forall l \in L(r_m), m \in M, \tag{4.14}$$

$$y_l \leq x_i, \quad \forall i \in l, l \in L(r_m), m \in M, \tag{4.15}$$

$$\sum_{i\in V} c_i x_i \leq B, \tag{4.16}$$

$$0 \leq y_l \leq 1, \quad \forall l \in L(r_m), m \in M, \tag{4.17}$$

$$x_i \in \{0,1\}, \quad \forall i \in V. \tag{4.18}$$

Again, constraints (4.14) are not necessary if β's are nonnegative. The number of constraints (4.15) may be intimidating, but we can get a simple reduction by utilizing the nested structure of l. For $l \in L(r)$ with $|l| > 2$, two subsets, l_- and l^-, can be obtained by removing the first and the last road segment of l, respectively. Since $l_-, l^- \in L(r)$, we can use two constraints $y_l \leq y_{l_-}$ and $y_l \leq y_{l^-}$ instead of constraints (4.15). For example, given $l = \{1,2,3,4\}$, we can use $l_- = \{2,3,4\}$ and $l^- = \{1,2,3\}$. After the reduction, the constraint matrix of BL-GU-MILP boils down to a totally unimodular matrix along with a budget constraint, as formalized in the following proposition.

Proposition 4.3. *If $f(\cdot)$ is an increasing convex function, BL-GU-MILP with the relaxed budget constraint has a totally unimodular constraint matrix. Then the corresponding Lagrangian relaxation can be solved as a linear program (LP).*

Proposition 4.3 has several important implications. First, it presents an alternative way to prove the polynomial-time solvability result of the Lagrangian dual in Corollary 4.2.1. Since LP is polynomial-time solvable and the Lagrangial dual has a limited number of break points, the Lagrangian dual can be solved in polynomial time. Second, instead of resorting to a general supermodular maximization oracle, the Lagrangian dual can now be solved with a linear programming solver, which tends to deliver significantly better computational performance.

Regarding the size of BL-GU-MILP, both the number of continuous variables and the number of constraints in BL-GU-MILP are $O(\min\{|M|N_e^2, |L(V)|\})$, where N_e is the size of the longest trajectory (in terms of the number of road segments). Although $|M|$ can be arbitrarily large, $L(V)$, the set of all possible continuous road segments on the entire road network, is limited. And in practice, many trajectories are similar so the actual number of variables and constraints is much smaller.

Furthermore, the supermodularity result can be used to strengthen the formulation of BL-GU-MILP. By the supermodularity, our problem can be transformed into a minimization problem with a submodular objective function.

Definition 4.1. *For a submodular function g, the polyhedron*

$$EP_g = \{\pi \in \mathbb{R}^{|V|} : \pi(S) \leq g(S),\ \forall S \subseteq V\}$$

is called an extended polymatroid.

It has been shown by Atamtürk and Narayanan (2008) that inequalities

$$\pi x \leq \gamma,\ \forall \pi \in EP_g$$

are valid for the convex lower envelope of g, defined as

$$Q_g = \text{conv}\{(x, \gamma) \in \{0,1\}^{|V|} \times \mathbb{R} : g(x) \leq \gamma\}.$$

So given a solution x^* and its corresponding objective value γ^*, we can search in EP_g to find π^* that maximizes πx^*. If $\pi^* x^* > \gamma^*$, then a cut can be added as $\pi^* x \leq \gamma$. As shown in Edmonds (2003), π^* can be found by a greedy algorithm.

A Lagrangian Relaxation Based Algorithm.

Solving BL-GU-MILP directly via commercial MILP solvers can be challenging due to the large number of variables and constraints in practical applications. Given that the Lagrangian relaxation subproblem of BL-GU-MILP by relaxing the budget constraint can be solved as a LP, we propose a simple and efficient Lagrangian relaxation based algorithm to solve the large-scale bike lane planning problem under the general supermodular utility functions, which is detailed as follows.

1. The Lagrangian dual of BL-GU-MILP is first solved with a "outer approximation" algorithm, wherein each iteration involves a LP. For the ease of exposition, we use $S(x)$ to denote the objective function of BL-GU-MILP, $c(x)$ for the construction cost $(= \sum_{i \in V} c_i x_i)$, $x(u)$ for the maximizer of the Lagrangian relaxtion at u, and $g(u) = -(c(x) - B)$ for the subgradient. We start with $u' = 0$ and $u'' = \max_i \frac{S(r)}{c_i}$. At these two multiplier values, $g(u') = -(c(e) - B)$ and $g(u'') = B$.
   a) Calculate $u^* = \frac{S(x(u'')) - S(x(u'))}{c(x(u'')) - c(x(u'))}$, and solve for $x(u^*)$ with LP.
   b) If $\Phi(u^*) = \Phi(u'') + (u^* - u'')g(u'')$, then u^* is optimal. Otherwise go to the next step.
   c) If $c(x(u^*)) > B$, set $u' = u^*$; otherwise set $u'' = u^*$. Repeat step (a) and (b).
2. After solving the Lagrangian dual, if $c(x(u^*)) = B$, then $x(u^*)$ is the optimal solution to BL-GU-MILP. Otherwise we find the best feasible solution by removing items from $x(u^*)$. The formulation is the same as BL-GU-MILP except that we drop x_i for $i \in \{j \in V : x_j(u^*) = 0\}$ and y_l for $l \in \{l' \in L(e_m), \forall m \in M : \exists i \in l', x_i(u^*) = 0\}$. The resulting problem often has a much smaller size than the original problem and admits a relatively short solution time.

4.4.3 Cyclist's Response to Bike Lane Plan

Up till now our model makes no assumptions about cyclists' responsive behaviors to the bike lane construction plan. The aforementioned models maximize the cyclists' utility assuming their route choices are fixed (their trajectories will not be impacted by the constructed bike lanes). This assumption may not be valid if cyclists update their route choices based on the constructed bike lanes. Since the utility from riding through a route is determined by the constructed bike lanes, cyclists may choose to take a different route than the observed trajectory if more bike lanes are constructed along that route.

To account for cyclists' responses, we assume for a cyclist m riding from $i \in V$ to $j \in V$, she can choose from a set of candidate routes/trajectories $C_m = \{r_m^1, \ldots, r_m^{t_m}\}$, where each route starts with i and ends with j. In addition to the bike lane utility, cyclists' evaluation of a route also depends on the length, slope, noises, and other physical characteristics. Therefore, we add to the utility function an exogenous utility term $\bar{v}(r)$ for a route r. In practice, $\bar{v}(r)$ can be estimated beforehand and assumed to be known.

Given a bike lane construction plan x, cyclist m chooses the route $r \in C_m$ with probability $p_m(r)$ according to a discrete choice model. Then the objective of the bike lane planning problem is

$$\max_x \quad \sum_{m \in M} \sum_{r \in C_m} p_m(r) v_x(r). \tag{4.19}$$

The bike route choice is often modeled using the Multinomial Logit model (MNL), as shown in Hood, Sall, and Charlton (2011) and Khatri et al. (2016). Under the MNL model, the probability $p_m(r)$ is given by

$$p_m(r) = \frac{\exp(v_x(r) + \bar{v}(r))}{\sum_{r' \in C_m} \exp(v_x(r') + \bar{v}(r'))}. \tag{4.20}$$

Because many alternative routes have overlapping road segments, the assumption of irrelevant alternatives may not be satisfied. To relieve this concern, a correction term called Path Size factor (PS) is introduced as (Broach, Dill, and Gliebe, 2012)

$$PS_m(r) = \sum_{i \in r} \frac{l_i}{L_r} \frac{1}{\sum_{r' \in C_m} \delta_{ir'}}, \tag{4.21}$$

where l_i is the length of the road segment i, L_r is the length of route r, and $\delta_{ir'}$ is a binary variable that equals 1 if $i \in r'$ and 0 otherwise. This correction factor is added to the utility function in the logrithmic form, yielding the choice probability $p'_m(r)$ as

$$p'_m(r) = \frac{\exp(v_x(r) + \bar{v}(r) + \ln(PS_m(r')))}{\sum_{r' \in C_m} \exp(v_x(r') + \bar{v}(r) + \ln(PS_m(r'))}, \tag{4.22}$$

which is often referred to as the basic Path Size Logit (PSL) model (Ben-Akiva and Bierlaire, 1999). Intuitively, the correction term decreases the utility of a route when it overlaps with other alternative routes. And the decrease is proportional to the degree of overlap.

The consequent bike planning problem is similar to an assortment optimization problem as decisions in both problems shape the choice probabilities. However, while the assortment decision influences the choice probabilities by altering the choice set, the bike lane construction decision transforms the choice probabilities by changing the utility values. That being said, the choice set in our problem is invariant to the decision variables, which is a critical difference between our problem and the assortment optimization problem. Furthermore, unlike the assortment optimization where the marginal profit of each product is exogenously given, the "profit" from bike lanes $v_x(r)$ depends on the decision variables. Hence problem (4.19) also shares a similar structure to the joint assortment-pricing optimization problem. However, the decisions here are binary and the analysis in the pricing literature can not carry on, which makes the exact solution to this problem difficult to derive especially for the practical large-scale problem.

Remark. With other alternative assumptions about the cyclist behavior, we may arrive at a more tractable formulation for (4.19). For instance, if the utility function of every cyclist is known exactly and they choose the route with the highest utility, then the problem (4.19) would admit a MILP formulation. Due to the lack of empirical evidence we omit the discussion here. Nevertheless, we want to highlight the need for more empirical research in this behavioral aspect.

Admitting the difficulty in solving the above model exactly, we seek a simple but practical remedy to address the responsive behaviors by using an alternating algorithm. The algorithm alternates between updating the choice probabilities given x between optimizing x with the fixed choice probabilities, as detailed in the following. 1) First solve the bike lane planning problem using the observed routes (BL-GU-MILP) and get the optimal solution x^*; 2) Calculate $p_m^t(x^*)$ according to the choice model; 3) Fixing the choice probability $p_m^t(x^*)$, solve the bike lane planning problem (4.19) to get the new solution and update x^*; repeat 2) and 3) until the termination criteria is met.

The application of the introduced alternating algorithm is not limited to the MNL model, but bodes well to other choice models such as Exponomial choice model and Markov chain choice model. Moreover, when a more refined description of cyclist's behavior is preferred (e.g., through simulators), the alternating algorithm can be easily adapted.

4.5 A Real-World Case Study

We apply the proposed models and algorithms to the real-world trajectory data set from Zhuhai, as described in Section 4.3. First, we discuss the computational performance and solution quality of proposed algorithms, in comparison to a benchmark greedy heuristic. Second, we compare the bike lane construction plans generated by different models with varying parameters, with and without consideration of cyclists' responsive behaviors.

4.5.1 Computational Result

We test the two models, BL-AC and BL-GU, on the practical road network with the use of the trajectory data set. BL-AC is solved by the MILP directly (BL-AC-MILP) and BL-GU is solved by the Lagrangian relaxation based algorithm proposed in Subsection 4.4.2. For BL-GU, the utility function takes the form of $v_x(r) = \sum_{s\in S_x(r)} |s|\alpha^{|s|}$. We vary the choices of λ, α, B, and the number of sampled trajectories (cyclists) m. The experiments were conducted with Gurobi (Gurobi Optimization 2018) and ran on a Windows 10 64-bit machine with a Intel Xeon 4114 2.20 GHz processor and 32.0 GB RAM.

Table 4.1 presents the running time and optimality gap of solving BL-AC-MILP. We can see that the MILP formulation can be solved efficiently and all the instances can be solved within 15 seconds on the given network. We also test its efficiency on randomly generated road networks with 20,000 road segments, and the MILP formulation is able to deliver the global optimal solution within a minute.

Table 4.1: Computational performance of BL-AC-MILP (all trajectories)

	$B = 30$ (km)		$B = 50$ (km)		$B = 100$ (km)	
λ	Time (sec)	Gap	Time (sec)	Gap	Time (sec)	Gap
0	2.49	0.00%	2.50	0.00%	2.58	0.00%
1	3.73	0.00%	4.09	0.00%	6.55	0.00%
2	6.06	0.00%	4.60	0.00%	6.01	0.00%
5	6.66	0.00%	5.49	0.00%	7.28	0.00%
10	7.28	0.00%	7.02	0.00%	13.80	0.00%

Different from BL-AC, the general model BL-GU often involves a much greater number of variables and the corresponding MILP formulation is not efficient. We compare here the efficiency of the proposed Lagrangian relaxation based algorithm (denoted as GU-Lag) versus a benchmark greedy heuristic adapted from Bao et al. (2017), which selects road segments to increase the objective function in a greedy way. The detailed comparison result is given in Table 4.2, in which we test the two algorithms on different sets of trajectories (e.g., $m = 2000$ indicates a random sample of 20,000 trajectories). The reported performance is averaged across five different runs. We observe that GU-Lag delivers superior performance in terms of both the running time and solution quality. In particular, the greedy algorithm does not scale well to the cases with large values of B or α. For example, when $\alpha = 1.1$, the solution derived from the greedy algorithm suffers from the significant suboptimality, as implied by the large optimality gap. By contrast, GU-Lag admits a reliable computational performance across all different combinations of parameters.

Table 4.2: Computational performance of GU-Lag and the greedy algorithm (time is in seconds)

α	B (km)	$m = 20,000$				$m = 50,000$				all trajectories			
		GU-Lag		Greedy		GU-Lag		Greedy		GU-Lag		Greedy	
		Time	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
1.02	30	882	0.0%	3,474	10.8%	2,842	0.0%	25,671	4.9%	6,825	0.0%	38,483	3.6%
1.05	30	2,610	1.2%	2,385	9.2%	11,477	0.1%	17,146	4.7%	26,930	0.0%	28,458	6.0%
1.1	30	570	9.6%	3,613	81.6%	1,790	3.6%	6,543	96.3%	5,815	0.6%	14,479	95.3%
1.02	50	967	0.0%	9,100	8.1%	3,136	0.0%	58,623	2.5%	7,427	0.0%	84,043	2.0%
1.05	50	3,352	0.5%	5,574	10.3%	25,554	1.0%	38,796	3.7%	64,580	1.5%	60,936	3.0%
1.1	50	527	2.9%	3,613	81.6%	3,467	2.3%	14,659	92.6%	6,111	4.9%	28,908	93.6%
1.02	100	1,117	0.0%	35,287	1.6%	4,328	0.0%	161,236	1.1%	8,388	0.0%	208,982	0.9%
1.05	100	6,719	0.4%	19,370	11.0%	30,236	0.4%	108,593	8.4%	113,041	0.6%	156,920	8.0%
1.1	100	4,791	0.4%	10,481	69.3%	4,698	0.2%	37,969	80.5%	16,222	1.3%	71,059	87.6%

4.5.2 Bike Lane Planning Result and Discussion

We compare the bike lane planning solutions generated from BL-AC and BL-GU with quantitative topological measures as well as visualization results. The setup is the same as in Subsection 4.5.1 and we use all the trajectories as the model input. We first focus on the planning model with fixed trajectories, and discuss the impact of cyclists' responsive behaviors at the end.

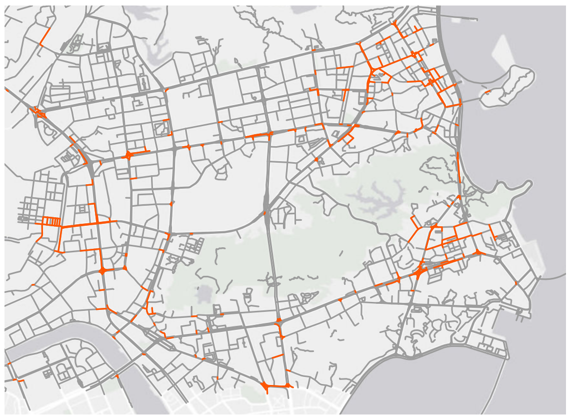
Topological Comparisons.

We consider five relevant topological features: the number of selected bike lanes (road segments), the number of continuous bike lane pairs, the mean number of connections per bike lane, the mean size of continuous bike lanes (along trajectories), and the max size of continuous bike lanes (along trajectories). All the features except the number of selected bike lanes measure the continuity of bike lanes on the road network. Table 4.3 presents the comparison results based on these features for BL-AC and BL-GU with different parameter values. Note that BL-GU with $\alpha = 1$ is equivalent to GL-AC with $\lambda = 0$.

Table 4.3: Topological comparison of BL-AC and BL-GU with $B = 30$ km.

	# of selected bike lanes	# of continuous bike lane pairs	mean # of connections per bike lane	mean size of continuous bike lanes	max size of continuous bike lanes
			BL-AC		
$\lambda = 0$	389	1,871	5.9	4.1	16
$\lambda = 2$	332	2,805	10.3	5.4	21
$\lambda = 10$	312	2,932	11.5	5.9	33
			BL-GU		
$\alpha = 1.02$	381	2,105	6.8	4.7	19
$\alpha = 1.05$	330	2,816	10.7	6.9	37
$\alpha = 1.1$	127	547	5.3	3.1	115

We have several observations. First, in BL-AC and BL-GU, increasing the value of λ and α lead to fewer selected bike lanes. Because when the continuity is less important, the planning model tends to build bike lanes on more road segments to cover more bike trajectories. And when the continuity is more preferred, it is beneficial to build bike lanes in a few areas to make sure the bike lanes are connected to each other. Second, as λ grows in BL-AC, the number of continuous bike lane pairs increases, which is consistent with the objective function of BL-AC that maximizes the adjacency continuity. In the meanwhile, the mean size of continuous bike lanes, the maximum size of continuous bike lanes, and the mean number of connections per bike lane also increase. By contrast, as α increases in BL-GU, we observe that the number of continuous bike lane pairs first increases and then decreases, which also applies to the mean number of connections per bike lane and the mean size of


(a) $\lambda = 0$

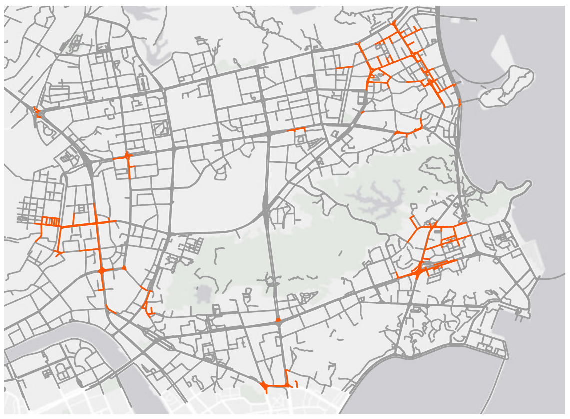
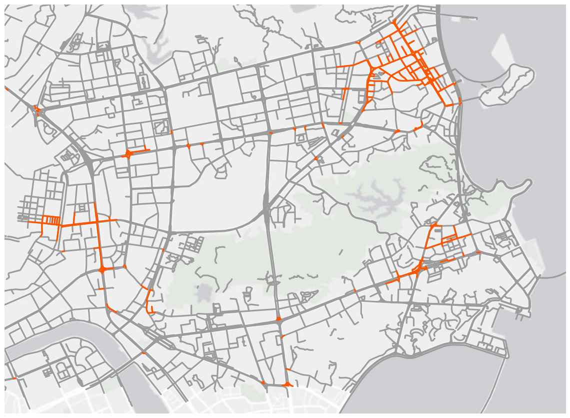

(b) $\lambda = 2$

Figure 4.5: Selected bike lanes (in red color) from BL-AC with $B = 30$ km.

continuous bike lanes. However, the maximum size of continuous bike lanes always grows. This is because that when α is large, the objective function of BL-GU will be dominated by the longest trajectory that has the largest number of road segments. As a result, BL-GU will build bike lanes on the long trajectories. In practice, the mean size of continuous bike lanes is more important than its maximum value and thus we may want to avoid too large values of α. Third, we find that the mean size of continuous bike lanes from BL-AC is often smaller than that from BL-GU. Even when λ is very large, BL-AC can not get the same level as BL-GU in terms of the mean size of continuous bike lanes. This highlights the limitation of BL-AC, of which the objective function does not incorporates the size of continuous bike lanes. BL-GU, however, can overcome this issue by measuring the size of continuous bike lanes explicitly. We observe that choosing $\alpha = 1.05$ achieves desirable continuity measures in the mean number of connections per bike lane and the mean size of continuous bike lanes.

We visualize the bike lane planning results of BL-AC and BL-GU on the road network in Figure 4.5 and Figure 4.6, respectively. It can be shown that when $\lambda = 0$ ($\alpha = 1$), the selected bike lanes are spread out over the city. Consistent with our topological findings, a large value of λ in BL-AC induces fewer and more continuous bike lanes that are mainly built in a few areas (e.g., the top left and bottom right). Increasing the value of α in BL-GU has a similar effect that yields a more continuous bike lane network, as shown in Figure 4.6a. Different from BL-AC, as indicated by our previous discussion, having a large value of α in BL-GU also tends to select continuous road segments on the long trajectories, which leads to a very different bike lane plan in Figure 4.6b. Interestingly, the selected bike lanes from Figure 4.6b align with the main roads of the city road network connecting different districts while the selected bike lanes from Figure 4.5b are along with the secondary roads.


(a) $\alpha = 1.05$

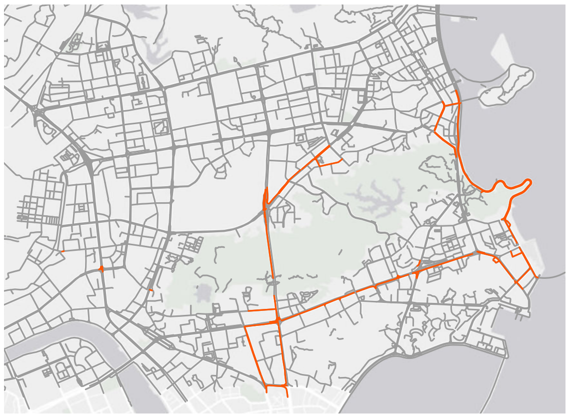

(b) $\alpha = 1.1$

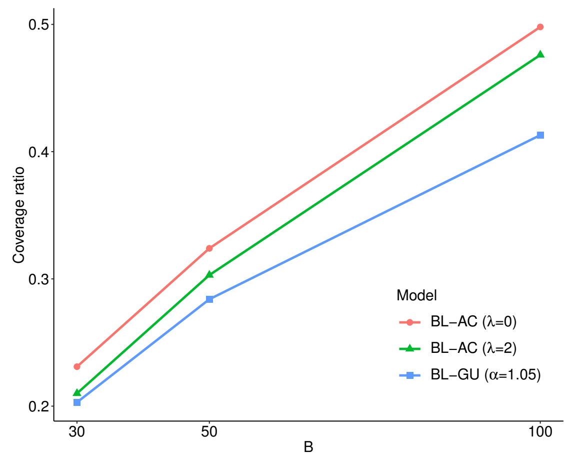
Figure 4.6: Selected bike lanes (in red color) from BL-GU with $B = 30$ km.

Table 4.4: Percentage change in coverage and continuity measures with varying λ and α.

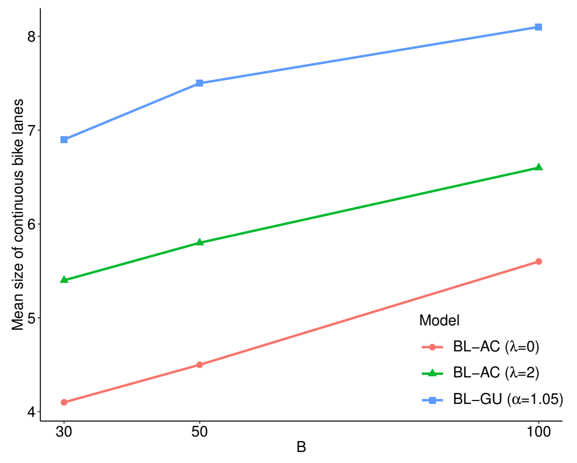
	% change in the coverage ratio	% change in the mean # of connections per bike lane	% change in the mean size of continuous bike lanes
		BL-AC	
$\lambda = 2$	-9.09%	74.58%	31.71%
$\lambda = 10$	-12.99%	94.92%	43.90%
		BL-GU	
$\alpha = 1.02$	-0.43%	15.25%	14.63%
$\alpha = 1.05$	-12.12%	81.36%	68.29%

Coverage-Continuity Trade-Off.

In both BL-AC and BL-GU, we are balancing the coverage objective versus the continuity objective, which are reflected in the utility functions. When increasing the value of λ or α, the bike lane planning model puts more weights on the continuity than the coverage objective. Table 4.4 presents the percentage changes of the coverage and continuity measures using the BL-AC with $\lambda = 0$ as the baseline. The coverage ratio is calculated as the percentage of trajectories (sets of road segments) that are covered by bike lanes. Indeed, we observe that both the mean number of connections per bike lane and the mean size of continuous bike lanes grow at the expense of coverage ratio. Notably, the BL-GU with $\alpha = 1.02$ improves the continuity of bike lanes significantly while only lowering the coverage ratio slightly.



(a) Coverage ratio versus B (km)



(b) Mean size of continuous bike lanes versus B (km)

Figure 4.7: The impact of B on the coverage ratio and mean size of continuous bike lanes.

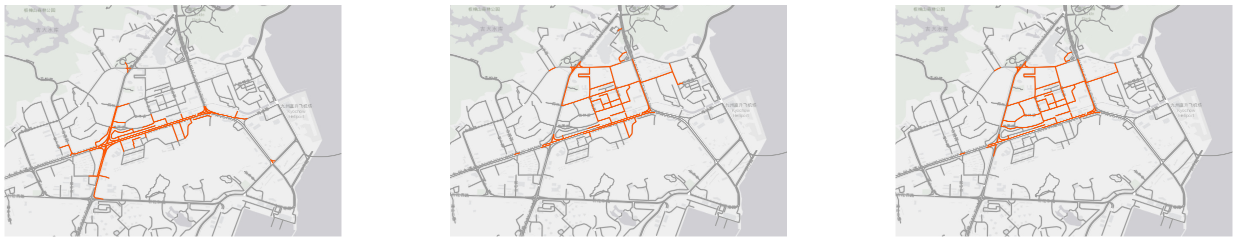
Varying the Budget, B.

Increasing the budget in both BL-AC and BL-GU will improve coverage and continuity. Figure 4.7 shows how the coverage ratio and the mean size of continuous bike lanes evolve with the budget value. As B grows, the two measures increase and the gap between different models remains significant. This implies that even when a large budget is allowed, making a good bike lane planning decision is still critical. Since bike lane construction is often costly, the city government can weigh the benefits from building bike lanes versus the cost using the quantified measures proposed in this paper.

Cyclists' Responsive Behaviors.

The previous discussion is focused on the scenario where the observed bike trajectories are directly fed into the model and no cyclists' responsive behaviors is assumed. Here we investigate how the planning result would change when the cyclists update their route choices following the discussion in Subsection 4.4.3. To better illustrate and compare the result, we focus on a subarea of the city (Jida Residential District), which corresponds to the bottom right area on the map. We apply the alternating algorithm proposed in Subsection 4.4.3 and set the termination criteria to be: 1) the relative difference between the objective values from the last two iterations is less than 1% or 2) the number of iterations exceeds 100. We call the Google Map Directions API (`https://developers.google.com/maps/documentation/directions/start`) to generate the candidate routes for each origin-destination pair in the subarea. Hence the choice set C_m includes both the observed route as well as the routes returned by Google. The utility function is assumed to take the form of $\eta v_x(r) - \ln(L_r)$ for a route r, as motivated by Khatri et al. (2016). The value of η measures the relative importance of bike lanes to the cyclists' routing behaviors. A larger value of η indicates cyclists are more responsive to the constructed bike lanes.

We test the algorithm with $\alpha \in \{1.02, 1.05, 1.1\}, \eta \in \{0.1, 0.5\}, B \in \{5, 10\}$ and the algorithm converges within 10 iterations for all the instances. Figure 4.8 presents the bike lane planning results for $\alpha = 1.05$ and $B = 10$ with different responsive behaviors of cyclists. We find that the responsive behaviors affect the bike lane plan significantly. The selected bike lanes can be very different if we assume cyclists can take other alternative routes. Specifically, with the consideration of cyclists' responsive behaviors, the selected bike lanes tend to spread out to cover more routes other than the observed routes. This highlights the importance of understanding cyclists' routing choices when facing bike lanes (including both coverage and continuity considerations), which remains as an interesting future research direction.



(a) No responsive behaviors (b) $\eta = 0.1$ (c) $\eta = 0.5$

Figure 4.8: Selected bike lanes from BL-GU with different responsive behaviors of cyclists ($\alpha = 1.05,\ B = 10$ km)

4.6 Conclusion

This paper studies the bike lane planning problem with novel formulations and algorithms. Unlike the previous work that mainly builds on surveys and heuristics, we present a modeling framework that directly utilizes the GPS bike trajectory data from the emerging dock-less bike sharing programs. Our model formalizes the main objectives of the bike lane planning in view of the cyclists' utility functions. Depending on the choices of the utility functions, we propose efficient algorithms to solve the corresponding bike lane planning model by exploiting the problem structure. We demonstrate the effectiveness of the models and algorithms on a large-scale real-world data set. We show how the topology of the bike lane network would change with varying choices of the utility functions and demonstrate the tension between coverage and continuity quantitatively. Additionally, our results indicate the importance of incorporating cyclists' potential responsive behaviors to the bike lane planning, which is often ignored in many strategic planning models.

There are several promising future research directions. First, it would be interesting to consider the width of the bike lanes as another decision dimension in the model. The width of the bike lanes can impact the interaction between car flows and bike flows, and eventually change the traffic equilibrium of the whole city environment. Second, since the cyclists' responsive behaviors are often hard to predict before any bike lane is deployed, the city government may dynamically construct bike lanes in the city, e.g., first build a few bike lanes to learn the behaviors and then add more bike lanes. Then the problem becomes a dynamic strategic planning model with behavior learning. Third, jointly designing the bike lanes with other bike facilities such as bike sharing stations and parking areas can be of interest to the city government.

Chapter 5

Conclusion

In this thesis we presented a set of data-driven decision making approaches for solving practical operations research and management problems, with a focus on smart city operations. We started with the yield prediction problem in integrated circuit manufacturing and proposed efficient models and algorithms to improve the prediction accuracy based on the neighborhood effect in the wafer map data. Then we explored the integration of data analytics and optimization tools in the last mile delivery services. We proposed tractable integrated models and leveraged the results from distributionally robust optimization to improve the order assignment efficiency. Lastly, we studied how the bike trajectory data can help better design the urban bike lane system. We proposed a flexible and tractable optimization framework for the bike lane planning problem, and demonstrate its performance on a real-world data set. This work is the first to rigorously analyze the bike lane planning problem with the consideration of the cyclists' utility functions.

Notably, this thesis studies problems on the basis of real-world data sets. We believe this will be the trend of operations research and management. In addition, smart city operations will continue attracting more attention from both inside and outside the community. While we have listed several possible extensions of our work in the previous chapters, we believe there are many other aspects of smart city operations worth investigating in depth. For instance, the deployment of air quality monitoring stations is critical to providing accurate and credible alerts to city residents, and finding a trade-off between cost and accuracy is important. Another example lies in the vision of autonomous vehicles, which will create enormous research opportunities.

Bibliography

Afeche, Philipp, Zhe Liu, and Costis Maglaras (2018). “Ride-hailing networks with strategic drivers: The impact of platform control capabilities on performance”. In: *Columbia Business School Research Paper* 18-19, pp. 18–19.

Ahipaşaoğlu, Selin Damla, Uğur Arıkan, and Karthik Natarajan (2016). “On the flexibility of using marginal distribution choice models in traffic equilibrium”. In: *Transportation Research Part B: Methodological* 91, pp. 130–158.

Ahipaşaoğlu, Selin Damla, Rudabeh Meskarian, et al. (2015). “Beyond normality: A cross moment-stochastic user equilibrium model”. In: *Transportation Research Part B: Methodological* 81, pp. 333–354.

Ahuja, Ravindra K, Dorit S Hochbaum, and James B Orlin (2003). “Solving the convex cost integer dual network flow problem”. In: *Management Science* 49.7, pp. 950–964.

Albin, Susan L and David J Friedman (1989). “The impact of clustered defect distributions in IC fabrication”. In: *Management Science* 35.9, pp. 1066–1078.

Angalakudati, Mallik et al. (2014). “Business analytics for flexible resource allocation under random emergencies”. In: *Management Science* 60.6, pp. 1552–1573.

Atamtürk, Alper and Vishnu Narayanan (2008). “Polymatroids and mean-risk minimization in discrete optimization”. In: *Operations Research Letters* 36.5, pp. 618–622.

Bae, Suk Joo, Jung Yoon Hwang, and Way Kuo (2007). “Yield prediction via spatial modeling of clustered defect counts across a wafer map”. In: *IIE Transactions* 39.12, pp. 1073–1083.

Bai, Jiaru et al. (2018). “Coordinating supply and demand on an on-demand service platform with impatient customers”. In: *Manufacturing & Service Operations Management.*

RouteMatrix API (2019). `http://lbsyun.baidu.com/index.php?title=webapi/route-matrix-api-v2`.

Baker, Keith J and R Mark Rylatt (2008). “Improving the prediction of UK domestic energy-demand using annual consumption-data”. In: *Applied Energy* 85.6, pp. 475–482.

Ban, Gah-Yi and Cynthia Rudin (2018). “The big data newsvendor: Practical insights from machine learning”. In: *Operations Research* Forthcoming.

Bao, Jie et al. (2017). “Planning bike lanes based on sharing-bikes’ trajectories”. In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp. 1377–1386.

Barnhart, Cynthia et al. (1998). “Branch-and-price: Column generation for solving huge integer programs”. In: *Operations research* 46.3, pp. 316–329.

Beardwood, Jillian, John H Halton, and John Michael Hammersley (1959). “The shortest path through many points”. In: *Mathematical Proceedings of the Cambridge Philosophical Society*. Vol. 55. 4. Cambridge University Press, pp. 299–327.

Belavina, Elena, Karan Girotra, and Ashish Kabra (2016). “Online grocery retail: Revenue models and environmental impact”. In: *Management Science* 63.6, pp. 1781–1799.

Ben-Akiva, Moshe and Michel Bierlaire (1999). “Discrete choice methods and their applications to short term travel decisions”. In: *Handbook of transportation science*. Springer, pp. 5–33.

Bertsimas, Dimitris J and Garrett Van Ryzin (1991). “A stochastic and dynamic vehicle routing problem in the Euclidean plane”. In: *Operations Research* 39.4, pp. 601–615.

— (1993). “Stochastic and dynamic vehicle routing in the Euclidean plane with multiple capacitated vehicles”. In: *Operations Research* 41.1, pp. 60–76.

Bertsimas, Dimitris and Jack Dunn (2017). “Optimal classification trees”. In: *Machine Learning* 106.7, pp. 1039–1082.

Bertsimas, Dimitris and Velibor V Mišic (2015). “Data-driven assortment optimization”. In: *working paper*.

Bien, Jacob and Robert Tibshirani (2011). “Hierarchical clustering with prototypes via minimax linkage”. In: *Journal of the American Statistical Association* 106.495, pp. 1075–1084.

Biggs, Max and Rim Hariss (2018). “Optimizing Objective Functions Determined from Random Forests”. In: *Available at SSRN 2986630*.

Blake, Andrew and Andrew Zisserman (1987). *Visual reconstruction.* MIT press.

Broach, Joseph, Jennifer Dill, and John Gliebe (2012). "Where do cyclists ride? A route choice model developed with revealed preference GPS data". In: *Transportation Research Part A: Policy and Practice* 46.10, pp. 1730–1740.

Campbell, Andrew A et al. (2016). "Factors influencing the choice of shared bicycles and shared electric bikes in Beijing". In: *Transportation research part C: emerging technologies* 67, pp. 399–414.

Campbell, Ann M and Barrett W Thomas (2008). "Probabilistic traveling salesman problem with deadlines". In: *Transportation Science* 42.1, pp. 1–21.

Caprara, Alberto, David Pisinger, and Paolo Toth (1999). "Exact solution of the quadratic knapsack problem". In: *INFORMS Journal on Computing* 11.2, pp. 125–137.

Carlsson, John Gunnar (2012). "Dividing a territory among several vehicles". In: *INFORMS Journal on Computing* 24.4, pp. 565–577.

Carlsson, John Gunnar, Mehdi Behroozi, and Kresimir Mihic (2018). "Wasserstein distance and the distributionally robust TSP". In: *Operations Research* 66.6, pp. 1603–1624.

Carlsson, John Gunnar and Erick Delage (2013). "Robust partitioning for stochastic multi-vehicle routing". In: *Operations research* 61.3, pp. 727–744.

Çavdar, Bahar and Joel Sokol (2015). "A distribution-free TSP tour length estimation model for random graphs". In: *European Journal of Operational Research* 243.2, pp. 588–598.

Chaillou, Paul, Pierre Hansen, and Yvon Mahieu (1989). "Best network flow bounds for the quadratic knapsack problem". In: *Combinatorial Optimization.* Springer, pp. 225–235.

Chan, Timothy CY, Zuo-Jun Max Shen, and Auyon Siddiq (2017). "Robust defibrillator deployment under cardiac arrest location uncertainty via row-and-column generation". In: *Operations Research* 66.2, pp. 358–379.

Chandran, Bala G and Dorit S Hochbaum (2009). "A computational study of the pseudoflow and push-relabel algorithms for the maximum flow problem". In: *Operations research* 57.2, pp. 358–376.

Chao, Li-Chang and Lee-Ing Tong (2009). "Wafer defect pattern recognition by multi-class support vector machines by using a novel defect cluster index". In: *Expert Systems with Applications* 36.6, pp. 10158–10167.

Charras-Garrido, Myriam et al. (2012). “Classification method for disease risk mapping based on discrete hidden Markov random fields”. In: *Biostatistics* 13.2, pp. 241–255.

Chen, Fei-Long and Shu-Fan Liu (2000). “A neural-network approach to recognize defect spatial pattern in semiconductor fabrication”. In: *IIEEE Trans. Semicond. Manuf* 13.3, pp. 366–373.

Chib, Siddhartha and Rainer Winkelmann (2012). “Markov chain Monte Carlo analysis of correlated count data”. In: *Journal of Business & Economic Statistics* 19, pp. 428–435.

Chicago Tribune (2015). *Build more and better bike lanes, cycling advocates urge Chicago.* URL: `http://www.chicagotribune.com/news/columnists/ct-bike-lane-network-getting-around-met-1012-20151011-column.html` (visited on 08/30/2018).

Chien, T William (1992). “Operational estimators for the length of a traveling salesman tour”. In: *Computers & operations research* 19.6, pp. 469–478.

Cho, Soo-Haeng et al. (2014). “Simultaneous location of trauma centers and helicopters for emergency medical service planning”. In: *Operations Research* 62.4, pp. 751–771.

Christensen, Ole F and Rasmus Waagepetersen (2002). “Bayesian prediction of spatial count data using generalized linear mixed models”. In: *Biometrics* 58.2, pp. 280–286.

City of Copenhagen (2010). *Copenhagen City of Cyclists.* URL: `https://web.archive.org/web/20120929230131/http://www.kk.dk/sitecore/content/Subsites/CityOfCopenhagen/SubsiteFrontpage/LivingInCopenhagen/CityAndTraffic/~/media/439FAEB2B21F40D3A0C4B174941E72D3.ashx` (visited on 08/30/2018).

Cunningham, James A (1990). “The use and evaluation of yield models in integrated circuit manufacturing”. In: *IIEEE Trans. Semicond. Manuf* 3.2, pp. 60–71.

Daganzo, Carlos F (2005). *Logistics systems analysis.* Springer Science & Business Media.

Delage, Erick and Yinyu Ye (2010). “Distributionally robust optimization under moment uncertainty with application to data-driven problems”. In: *Operations research* 58.3, pp. 595–612.

Department for Transport, UK (2015). *Transport minister encourages people to get on their bike for Cycle to Work Day.* URL: `https://www.gov.uk/government/news/transport-minister-encourages-people-to-get-on-their-bike-for-cycle-to-work-day` (visited on 08/30/2018).

Di Palma, Federico et al. (2005). “Unsupervised spatial pattern classification of electrical-wafer-sorting maps in semiconductor manufacturing”. In: *Pattern recognition letters* 26.12, pp. 1857–1865.

Dondi, Giulio et al. (2011). “Bike lane design: the context sensitive approach”. In: *Procedia engineering* 21, pp. 897–906.

Edmonds, Jack (2003). “Submodular functions, matroids, and certain polyhedra”. In: *Combinatorial Optimization–Eureka, You Shrink!* Springer, pp. 11–26.

Elmachtoub, Adam N and Paul Grigas (2017). “Smart” Predict, then Optimize””. In: *arXiv preprint arXiv:1710.08005*.

Epperson, Bruce (1994). *Evaluating suitability of roadways for bicycle use: Toward a cycling level-of-service standard.* HS-042 007.

Erera, Alan L, Juan C Morales, and Martin Savelsbergh (2010). “The vehicle routing problem with stochastic demand and duration constraints”. In: *Transportation Science* 44.4, pp. 474–492.

Farias, Vivek F, Srikanth Jagabathula, and Devavrat Shah (2013). “A nonparametric approach to modeling choice with limited data”. In: *Management Science* 59.2, pp. 305–322.

Fellows, Heyward H, Christina M Mastrangelo, and K Preston White Jr (2009). “An empirical comparison of spatial randomness models for yield analysis”. In: *Electronics Packaging Manufacturing, IEEE Transactions on* 32.2, pp. 115–120.

Ferreira, Kris Johnson, Bin Hong Alex Lee, and David Simchi-Levi (2015). “Analytics for an online retailer: Demand forecasting and price optimization”. In: *Manufacturing & Service Operations Management* 18.1, pp. 69–88.

Ferris-Prabhu, Albert V (1992). *Introduction to semiconductor device yield modeling.* Artech House on Demand.

Ferris-Prabhu, Albert V et al. (1987). “Radial yield variations in semiconductor wafers”. In: *Circuits and Devices Magazine, IEEE* 3.2, pp. 42–47.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2001). *The elements of statistical learning.* Vol. 1. Springer series in statistics Springer, Berlin.

Fujishige, Satoru (2005). *Submodular functions and optimization.* Vol. 58. Elsevier.

Fukasawa, Ricardo et al. (2006). “Robust branch-and-cut-and-price for the capacitated vehicle routing problem”. In: *Mathematical programming* 106.3, pp. 491–511.

Gademann, Noud and Steef Velde (2005). “Order batching to minimize total travel time in a parallel-aisle warehouse”. In: *IIE transactions* 37.1, pp. 63–75.

Gallo, Giorgio, Michael D Grigoriadis, and Robert E Tarjan (1989). “A fast parametric maximum flow algorithm and applications”. In: *SIAM Journal on Computing* 18.1, pp. 30–55.

Gallo, Giorgio and Bruno Simeone (1989). “On the supermodular knapsack problem”. In: *Mathematical Programming* 45.1-3, pp. 295–309.

Gao, Fei and Xuanming Su (2016). “Omnichannel retail operations with buy-online-and-pick-up-in-store”. In: *Management Science* 63.8, pp. 2478–2492.

Gendreau, Michel, Gilbert Laporte, and René Séguin (1996). “Stochastic vehicle routing”. In: *European Journal of Operational Research* 88.1, pp. 3–12.

Geofabrik (2018). *OpenStreetMap China.* URL: `http://download.geofabrik.de/asia/china.html` (visited on 08/30/2018).

Ghosh, Sujit K, Pabak Mukhopadhyay, and Jye-Chyi JC Lu (2006). “Bayesian analysis of zero-inflated regression models”. In: *Journal of Statistical planning and Inference* 136.4, pp. 1360–1375.

Glover, Fred (1975). “Improved linear integer programming formulations of nonlinear integer problems”. In: *Management Science* 22.4, pp. 455–460.

Goldberg, Andrew V and Robert E Tarjan (1988). “A new approach to the maximum-flow problem”. In: *Journal of the ACM (JACM)* 35.4, pp. 921–940.

Hansen, Mark H, Vijayan N Nair, and David J Friedman (1997). “Monitoring wafer map data from integrated circuit fabrication processes for spatially clustered defects”. In: *Technometrics* 39.3, pp. 241–253.

Harsha, Pavithra, Shivaram Subramanian, and Joline Uichanco (2019). “Dynamic pricing of omnichannel inventories”. In: *Manufacturing & Service Operations Management.*

He, Long, Zhenyu Hu, and Meilin Zhang (2019). “Robust repositioning for vehicle sharing”. In: *Manufacturing & Service Operations Management.*

Hekimoğlu, Mert Hakan, Burak Kazaz, and Scott Webster (2016). “Wine analytics: Fine wine pricing and selection under weather and market uncertainty”. In: *Manufacturing & Service Operations Management* 19.2, pp. 202–215.

Held, Karsten et al. (1997). “Markov random field segmentation of brain MR images”. In: *IEEE transactions on medical imaging* 16.6, pp. 878–886.

Helmberg, Christoph, Franz Rendl, and Robert Weismantel (2000). “A semidefinite programming approach to the quadratic knapsack problem”. In: *Journal of combinatorial optimization* 4.2, pp. 197–215.

Hochbaum, Dorit S (2001). “An efficient algorithm for image segmentation, Markov random fields and related problems”. In: *Journal of the ACM (JACM)* 48.4, pp. 686–701.

— (2008). “The pseudoflow algorithm: A new algorithm for the maximum-flow problem”. In: *Operations research* 56.4, pp. 992–1009.

Hochbaum, Dorit S and Sheng Liu (2018). “Adjacency-Clustering and Its Application for Yield Prediction in Integrated Circuit Manufacturing”. In: *Operations Research* 66.6, pp. 1571–1585.

Hoeffding, Wassily (1948). “A non-parametric test of independence”. In: *The annals of mathematical statistics*, pp. 546–557.

Holland, Chuck et al. (2017). “UPS Optimizes Delivery Routes”. In: *Interfaces* 47.1, pp. 8–23.

Hood, Jeffrey, Elizabeth Sall, and Billy Charlton (2011). “A GPS-based bicycle route choice model for San Francisco, California”. In: *Transportation letters* 3.1, pp. 63–75.

Howard, Charlene and Elizabeth Burns (2001). “Cycling to work in Phoenix: route choice, travel behavior, and commuter characteristics”. In: *Transportation Research Record: Journal of the Transportation Research Board* 1773, pp. 39–46.

Hunt, John Douglas and John E Abraham (2007). “Influences on bicycle use”. In: *Transportation* 34.4, pp. 453–470.

Hwang, Jung Yoon and Way Kuo (2007). “Model-based clustering for integrated circuit yield enhancement”. In: *European Journal of Operational Research* 178.1, pp. 143–153.

Hyodo, Tetsuro, Norikazu Suzuki, and Katsumi Takahashi (2000). “Modeling of bicycle route and destination choice behavior for bicycle road network plan”. In: *Transportation Research Record: Journal of the Transportation Research Board* 1705, pp. 70–76.

Ishikawa, Hiroshi and Davi Geiger (1998). “Segmentation by grouping junctions”. In: *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on.* IEEE, pp. 125–131.

Jaillet, Patrick, Jin Qi, and Melvyn Sim (2016). “Routing optimization under uncertainty”. In: *Operations research* 64.1, pp. 186–200.

Jeong, Young-Seon, Seong-Jun Kim, and Myong K Jeong (2008). “Automatic identification of defect patterns in semiconductor wafer maps using spatial correlogram and dynamic time warping”. In: *IEEE Trans. Semicond. Manuf* 21.4, pp. 625–637.

Jónasson, Jónas Oddur, Sarang Deo, and Jérémie Gallien (2017). “Improving HIV Early Infant Diagnosis Supply Chains in Sub-Saharan Africa: Models and Application to Mozambique”. In: *Operations Research* 65.6, pp. 1479–1493.

Kabra, Ashish, Elena Belavina, and Karan Girotra (2018). “Bike-share systems: Accessibility and availability”. In: *Chicago Booth Research Paper* 15-04.

Khatri, Ranjit et al. (2016). “Modeling route choice of utilitarian bikeshare users with GPS data”. In: *Transportation research record* 2587.1, pp. 141–149.

Kim, Kyungmee O (2011). “Burn-in considering yield loss and reliability gain for integrated circuits”. In: *European Journal of Operational Research* 212.2, pp. 337–344.

Kim, Taeho and Way Kuo (1999). “Modeling manufacturing yield and reliability”. In: *IEEE Trans. Semicond. Manuf* 12.4, pp. 485–492.

Klapp, Mathias A, Alan L Erera, and Alejandro Toriello (2016). “The one-dimensional dynamic dispatch waves problem”. In: *Transportation Science* 52.2, pp. 402–415.

Kleywegt, Anton J, Alexander Shapiro, and Tito Homem-de-Mello (2002). “The sample average approximation method for stochastic discrete optimization”. In: *SIAM Journal on Optimization* 12.2, pp. 479–502.

Kong, Nan et al. (2010). “Maximizing the efficiency of the US liver allocation system through region design”. In: *Management Science* 56.12, pp. 2111–2122.

Koren, Israel, Zahava Koren, and CH Stepper (1993). “A unified negative-binomial distribution for yield analysis of defect-tolerant circuits”. In: *IEEE Transactions on Computers* 42.6, pp. 724–734.

Krizek, Kevin J and Rio W Roland (2005). “What is at the end of the road? Understanding discontinuities of on-street bicycle lanes in urban settings”. In: *Transportation Research Part D: Transport and Environment* 10.1, pp. 55–68.

Krueger, DC and DC Montgomery (2014). “Modeling and analyzing semiconductor yield with generalized linear mixed models”. In: *Applied Stochastic Models in Business and Industry* 30.6, pp. 691–707.

Kwon, Ohseok, Bruce Golden, and Edward Wasil (1995). “Estimating the length of the optimal TSP tour: an empirical study using regression and neural networks”. In: *Computers & operations research* 22.10, pp. 1039–1046.

Langford, Brian Casey, Jiaoli Chen, and Christopher R Cherry (2015). “Risky riding: Naturalistic methods comparing safety behavior from conventional bicycle riders and electric bike riders”. In: *Accident Analysis & Prevention* 82, pp. 220–226.

Laporte, Gilbert (2007). “What you should know about the vehicle routing problem”. In: *Naval Research Logistics (NRL)* 54.8, pp. 811–819.

Laporte, Gilbert, Francois Louveaux, and Hélène Mercure (1992). “The vehicle routing problem with stochastic travel times”. In: *Transportation science* 26.3, pp. 161–170.

Law, Averill M (2014). *Simulation Modeling and Analysis.* McGraw-Hill New York.

Li, Te-Sheng and Cheng-Lung Huang (2009). “Defect spatial pattern recognition using a hybrid SOM–SVM approach in semiconductor manufacturing”. In: *Expert Systems with Applications* 36.1, pp. 374–385.

Li, Yiyao and William Phillips (2018). “Learning from Route Plan Deviation in Last-Mile Delivery”. In: *Master Thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY.*

Lichstein, Jeremy W et al. (2002). “Spatial autocorrelation and autoregressive models in ecology”. In: *Ecological monographs* 72.3, pp. 445–463.

Lim, Michael K, Ho-Yin Mak, and Zuo-Jun Max Shen (2016). “Agility and proximity considerations in supply chain design”. In: *Management Science* 63.4, pp. 1026–1041.

Lima, Antonio et al. (2016). “Understanding individual routing behaviour”. In: *Journal of The Royal Society Interface* 13.116, p. 20160021.

Lin, Jenn-Rong and Ta-Hui Yang (2011). “Strategic design of public bicycle sharing systems with service level constraints”. In: *Transportation research part E: logistics and transportation review* 47.2, pp. 284–294.

Liu, Sheng, Long He, and Zuo-Jun Max Shen (2019). “On-Time Last Mile Delivery: Order Assignment with Travel Time Predictors”. In: *Available at SSRN 3179994*.

Mak, Ho-Yin (2018). “Enabling Smarter Cities with Operations Management”. In: *Available at SSRN 3307458*.

Marschner, Ian C et al. (2011). “glm2: fitting generalized linear models with convergence problems”. In: *The R Journal* 3.2, pp. 12–15.

Mehrotra, Anuj, Ellis L Johnson, and George L Nemhauser (1998). “An optimization based heuristic for political districting”. In: *Management Science* 44.8, pp. 1100–1114.

Milor, Linda (2013). “A Survey of Yield Modeling and Yield Enhancement Methods”. In: *IEEE Trans. Semicond. Manuf* 26.2, pp. 196–213.

Murphy, Bernard T (1964). “Cost-size optima of monolithic integrated circuits”. In: *Proceedings of the IEEE* 52.12, pp. 1537–1545.

O’Mahony, Eoin and David B Shmoys (2015). “Data Analysis and Optimization for (Citi) Bike Sharing.” In: *AAAI*, pp. 687–694.

Ooi, Melanie Po-Leen et al. (2013). “Defect cluster recognition system for fabricated semiconductor wafers”. In: *Engineering Applications of Artificial Intelligence* 26.3, pp. 1029–1043.

Ouyang, Yanfeng and Carlos F Daganzo (2006). “Discretization and validation of the continuum approximation scheme for terminal system design”. In: *Transportation Science* 40.1, pp. 89–98.

Pan, Qisheng and Jason Cao (2015). *Recent Developments in Chinese Urban Planning.* Springer.

Panjwani, Dileep Kumar and Glenn Healey (1995). “Markov random field models for unsupervised segmentation of textured color images”. In: *IEEE Transactions on pattern analysis and machine intelligence* 17.10, pp. 939–954.

Park, Eun and Dominique Lord (2007). “Multivariate Poisson-lognormal models for jointly modeling crash frequency by severity”. In: *Transportation Research Record: Journal of the Transportation Research Board* 2019, pp. 1–6.

Perakis, Georgia et al. (2018). “Joint Pricing and Production: A Fusion of Machine Learning and Robust Optimization”. In: *Available at SSRN 3305039.*

Pisinger, David (2007). “The quadratic knapsack problemfffdfffdfffda survey”. In: *Discrete applied mathematics* 155.5, pp. 623–648.

Popescu, Ioana (2007). “Robust mean-covariance solutions for stochastic optimization”. In: *Operations Research* 55.1, pp. 98–112.

Pucker, J (2001). “Cycling safety on bikeways vs. roads.” In: *Transportation Quarterly* 55.4, pp. 9–11.

Qi, Wei, Lefei Li, et al. (2018). “Shared mobility for Last-Mile delivery: design, operational prescriptions, and environmental impact”. In: *Manufacturing & Service Operations Management* 20.4, pp. 737–751.

Qi, Wei and Zuo-Jun Max Shen (2019). “A Smart-City Scope of Operations Management”. In: *Production and Operations Management* 28.2, pp. 393–406.

Rue, Håvard (2001). “Fast sampling of Gaussian Markov random fields”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.2, pp. 325–338.

Rue, Havard and Leonhard Held (2005). *Gaussian Markov random fields: theory and applications.* CRC Press.

Ryan, DM and EA Foster (1981). “An integer programming approach to scheduling”. In: *Computer Scheduling of Public Transport.*

Rybarczyk, Greg and Changshan Wu (2010). “Bicycle facility planning using GIS and multi-criteria decision analysis”. In: *Applied Geography* 30.2, pp. 282–293.

Seeds, RB (1967). “Yield and cost analysis of bipolar LSI”. In: *Electron Devices Meeting, 1967 International.* Vol. 13. IEEE, pp. 12–12.

Sener, Ipek N, Naveen Eluru, and Chandra R Bhat (2009). “An analysis of bicycle route choice preferences in Texas, US”. In: *Transportation* 36.5, pp. 511–539.

Shen, Zuo-Jun Max and Lian Qi (2007). "Incorporating inventory and routing costs in strategic location models". In: *European journal of operational research* 179.2, pp. 372–389.

Shu, Jia et al. (2013). "Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems". In: *Operations Research* 61.6, pp. 1346–1359.

Snyder, Lawrence V and Zuo-Jun Max Shen (2019). *Fundamentals of supply chain theory*. John Wiley & Sons.

Solomon, Marius M (1987). "Algorithms for the vehicle routing and scheduling problems with time window constraints". In: *Operations research* 35.2, pp. 254–265.

South China Morning Post (2018). *From coal to cars: Beijing moves up a gear in the war against air pollution*. URL: `https://www.scmp.com/news/china/society/article/2131862/coal-cars-beijing-moves-gear-war-against-air-pollution` (visited on 08/30/2018).

Stapper, Charles H. (1989). "Large-area fault clusters and fault tolerance in VLSI circuits". In: *IBM Journal of research and Development* 33.2, pp. 162–173.

Stapper, Charles H, Frederick M Armstrong, and Kiyotaka Saji (1983). "Integrated circuit yield statistics". In: *Proceedings of the IEEE* 71.4, pp. 453–470.

Stinson, Monique and Chandra Bhat (2003). "Commuter bicyclist route choice: Analysis using a stated preference survey". In: *Transportation research record: journal of the transportation research board* 1828, pp. 107–115.

Szeliski, Richard et al. (2008). "A comparative study of energy minimization methods for markov random fields with smoothness-based priors". In: *IEEE transactions on pattern analysis and machine intelligence* 30.6, pp. 1068–1080.

Taam, Winson and Michael Hamada (1993). "Detecting spatial effects from factorial experiments: An application from integrated-circuit manufacturing". In: *Technometrics* 35.2, pp. 149–160.

Taylor, Terry A (2018). "On-demand service platforms". In: *Manufacturing & Service Operations Management* 20.4, pp. 704–720.

The Economist (2012). *Open-air computers*. URL: `https://www.economist.com/special-report/2012/10/27/open-air-computers` (visited on 08/30/2018).

The Seattle Times (2016). *How bicycle lanes are evolving around the world.* URL: `https://www.seattletimes.com/life/outdoors/how-bicycle-lanes-are-evolving-around-the-world/` (visited on 08/30/2018).

The Wall Street Journal (2018). *The Most Dangerous Place to Bicycle in America.* URL: `https://www.wsj.com/articles/the-most-dangerous-place-to-bicycle-in-america-1537867800` (visited on 09/25/2018).

Tilahun, Nebiyou Y, David M Levinson, and Kevin J Krizek (2007). "Trails, lanes, or traffic: Valuing bicycle facilities with an adaptive stated preference survey". In: *Transportation Research Part A: Policy and Practice* 41.4, pp. 287–301.

Tsai, Wen-Jie, Lee-Ing Tong, and Chung-Ho Wang (2008). "Developing a new defect cluster index". In: *Journal of the Chinese Institute of Industrial Engineers* 25.1, pp. 18–30.

Tyagi, Aakash, Magdy Bayoumi, et al. (1994). "The nature of defect patterns on integrated-circuit wafer maps". In: *IEEE Trans. Rel.* 43.1, pp. 22–29.

United Nations Population Fund (2018). *Urbanization.* URL: `https://www.unfpa.org/urbanization` (visited on 08/30/2018).

U.S. Census Bureau (2012). *American Community Survey.* URL: `https://factfinder.census.gov/faces/tableservices/jsf/pages/productview.xhtml?pid=ACS_12_1YR_B08301&prodType=table` (visited on 08/30/2018).

Veksler, Olga (2007). "Graph cut based optimization for MRFs with truncated convex priors". In: *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on.* IEEE, pp. 1–8.

Wang, Hai and Amedeo Odoni (2014). "Approximating the performance of a fffdfffdfffdlast milefffdfffdfffd transportation system". In: *Transportation Science* 50.2, pp. 659–675.

Wang, Xin, Michael K Lim, and Yanfeng Ouyang (2016). "A continuum approximation approach to the dynamic facility location problem in a growing market". In: *Transportation Science* 51.1, pp. 343–357.

White Jr, K Preston, Bijoy Kundu, and Christina M Mastrangelo (2008). "Classification of defect clusters on semiconductor wafers via the Hough transformation". In: *IEEE Trans. Semicond. Manuf* 21.2, pp. 272–278.

Wiesemann, Wolfram, Daniel Kuhn, and Melvyn Sim (2014). "Distributionally robust convex optimization". In: *Operations Research* 62.6, pp. 1358–1376.

Wu, Ming-Ju, Jyh-Shing R Jang, and Jui-Long Chen (2015). “Wafer Map Failure Pattern Recognition and Similarity Ranking for Large-Scale Data Sets”. In: *IEEE Trans. Semicond. Manuf* 28.1, pp. 1–12.

Yuan, Tao, Suk Joo Bae, and Jong In Park (2010). “Bayesian spatial defect pattern recognition in semiconductor fabrication using support vector clustering”. In: *The International Journal of Advanced Manufacturing Technology* 51.5-8, pp. 671–683.

Yuan, Tao, Saleem Z Ramadan, and Suk Joo Bae (2011). “Yield prediction for integrated circuits manufacturing through hierarchical Bayesian modeling of spatial defects”. In: *IEEE Trans. Rel.* 60.4, pp. 729–741.

Zhang, Yiling, Ruiwei Jiang, and Siqian Shen (2016). “Ambiguous Chance-Constrained Bin Packing under Mean-Covariance Information”. In: *arXiv preprint arXiv:1610.00035.*

Zhang, Yiling, Mengshi Lu, and Siqian Shen (2018). “On the Values of Vehicle-to-Grid Electricity Selling in Electric Vehicle Sharing”. In: *Available at SSRN 3172116.*

Zheng, Zhichao, Karthik Natarajan, and Chung-Piaw Teo (2016). “Least squares approximation to the distribution of project completion times with Gaussian uncertainty”. In: *Operations Research* 64.6, pp. 1406–1421.

Appendix A

Parameter Selection in Chapter 2

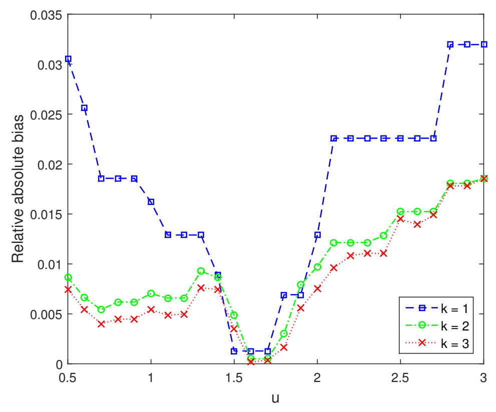
The adjacency-clustering model requires to select the two parameters, u and k, where u determines the balance between the permissible deviation of the priors, and the strength of the neighborhood effect, and k determines the label set. In the main text, it is shown that when using Poisson yield model, the combination of $u = 1$ and $k = 2$ provides superior accuracy in both practical and simulated instances.

In this e-companion, we present the relative absolute bias of prediction results for AC-NB, AC-NBP and AC-PNB in order to determine the uniform selection of u and k for each one of these models. The models are run with $u \in \{0.5, 0.6, \ldots, 3\}$ and $k \in \{1, 2, 3\}$, and the best uniform pair of u and k is selected for each model based on delivering consistently better results than other selections.

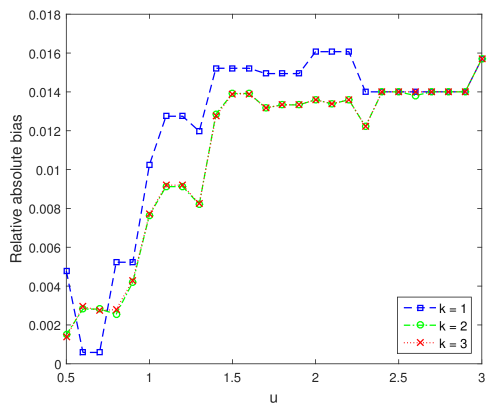
Based on the results for AC-NB in Figure A.1, we choose the uniform best pair of parameters to be $u = 0.7$ and $k = 3$. The average gap between this combination and the choice of $u = 1$ and $k = 2$ is 0.0036.

From the results for AC-NBP in Figure A.2, we choose the uniform best pair of parameters to be $u = 0.6$ and $k = 1$. The average gap between this combination and the choice of $u = 1$ and $k = 2$ is 0.0019.

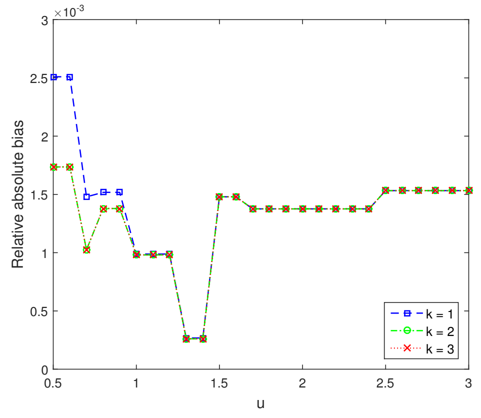
From the results for AC-PNB in Figure A.3, we choose the uniform best pair of parameters to be $u = 0.7$ and $k = 3$. The average gap between this combination and the choice of $u = 1$ and $k = 2$ is 0.0083.



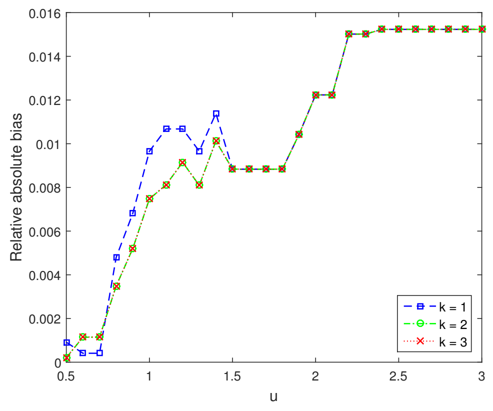
(a) Wafer Map 1



(b) Wafer Map 2

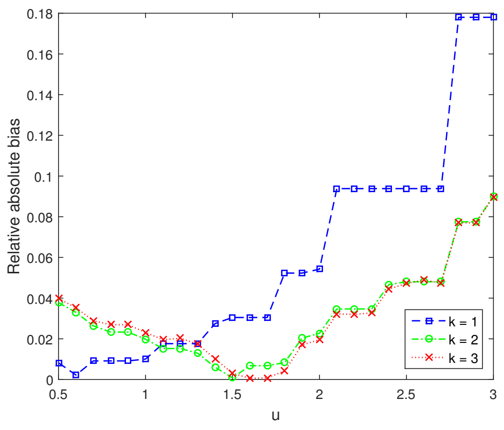


(c) Wafer Map 3

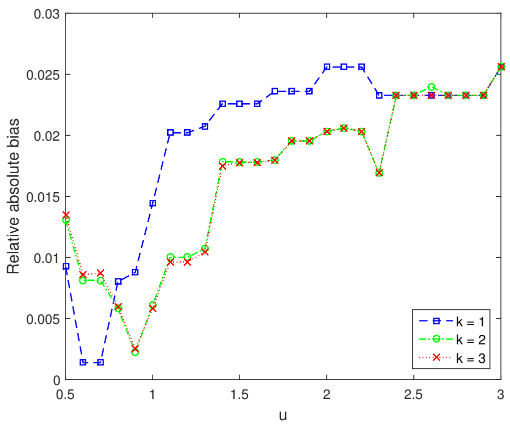


(d) Wafer Map 4

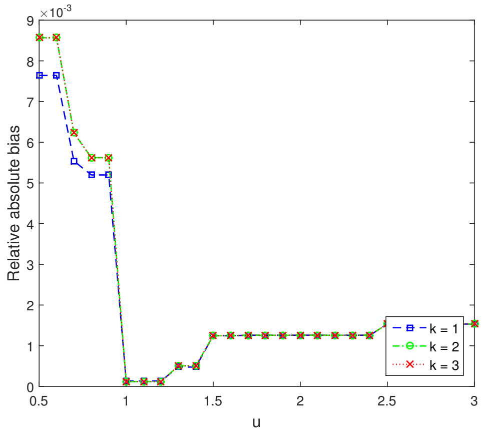
Figure A.1: Relative absolute bias of AC-NB model on four real data sets



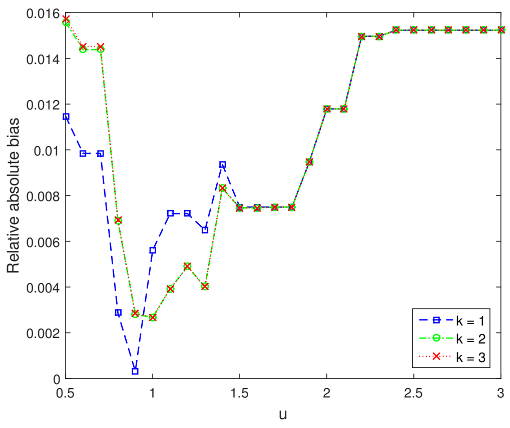
(a) Wafer Map 1



(b) Wafer Map 2

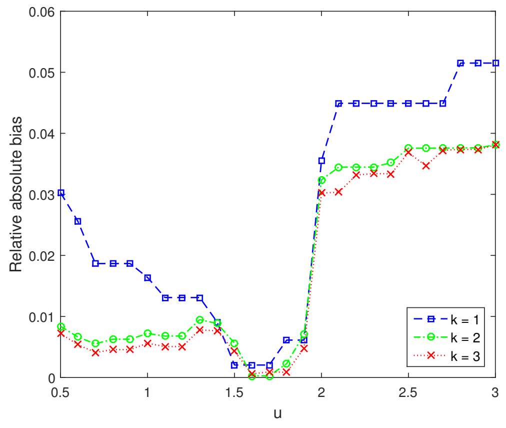


(c) Wafer Map 3

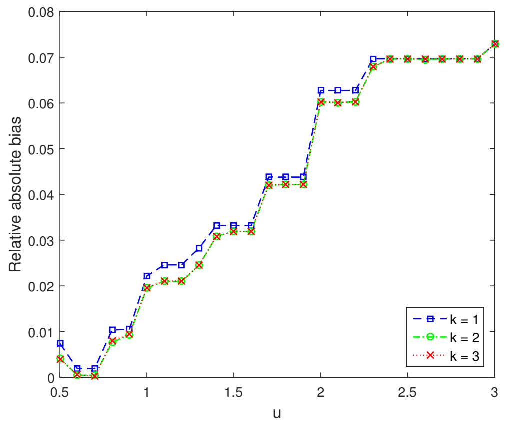


(d) Wafer Map 4

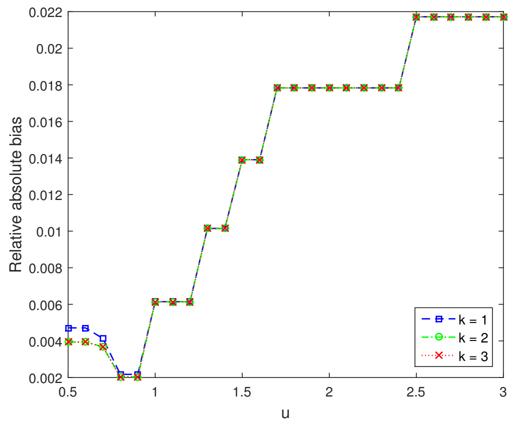
Figure A.2: Relative absolute bias of AC-NBP model on four real data sets



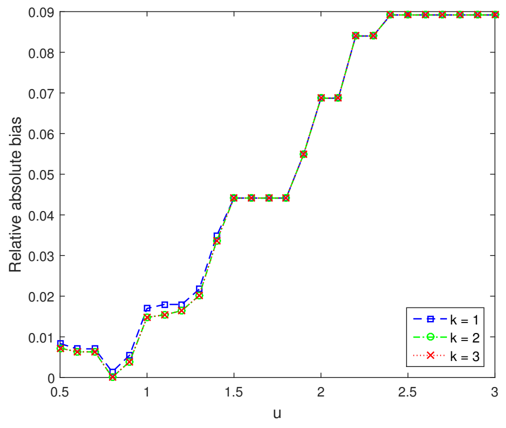
(a) Wafer Map 1



(b) Wafer Map 2



(c) Wafer Map 3



(d) Wafer Map 4

Figure A.3: Relative absolute bias of AC-PNB model on four real data sets

Appendix B

Proofs and Detailed Formulations for Results in Chapter 3

Proofs of Lemma 3.1, Proposition 3.2 and Proposition 3.3

Proof of Lemma 3.1. By strong duality of $\max_{\mathbb{P}_k \in \mathbb{F}_k} \mathbb{E}_{\mathbb{P}_k} \left(\tilde{T}_k + l_k - \tau\right)^+$, we have

$$\begin{aligned} \min_{\lambda_k, \eta_k, \theta_k} \quad & \lambda_k + \eta_k \sum_{i \in \mathcal{I}_k} \mu_i + \theta_k \sum_{i \in \mathcal{I}_k} \sigma_i^2 \\ \text{s.t.} \quad & \lambda_k + \eta_k \tilde{T}_k + \theta_k \left(\tilde{T}_k - \sum_{i \in \mathcal{I}_k} \mu_i \right)^2 \geq \left(\tilde{T}_k + l_k - \tau \right)^+, \quad \forall \tilde{T}_k \in \mathbb{R}. \end{aligned}$$

It is equivalent to

$$\begin{aligned} \min_{\lambda_k, \eta_k, \theta_k} \quad & \lambda_k + \eta_k \sum_{i \in \mathcal{I}_k} \mu_i + \theta_k \sum_{i \in \mathcal{I}_k} \sigma_i^2 \\ \text{s.t.} \quad & \lambda_k + \eta_k \tilde{T}_k + \theta_k \left(\tilde{T}_k - \sum_{i \in \mathcal{I}_k} \mu_i \right)^2 \geq \tilde{T}_k + l_k - \tau, \quad \forall \tilde{T}_k \in \mathbb{R}, \\ & \lambda_k + \eta_k \tilde{T}_k + \theta_k \left(\tilde{T}_k - \sum_{i \in \mathcal{I}_k} \mu_i \right)^2 \geq 0, \quad \forall \tilde{T}_k \in \mathbb{R}. \end{aligned}$$

From the constraints, we observe that θ must be nonnegative. Otherwise, the quadratic constraints will be violated by large values of $\tilde{T}_k$. We rewrite the first constraint as:

$$\min_{\tilde{T}_k \in \mathbb{R}} \quad \lambda_k + (\eta_k - 1)\tilde{T}_k + \theta_k \left(\tilde{T}_k - \sum_{i \in \mathcal{I}_k} \mu_i \right)^2 \geq l_k - \tau.$$

Since the left-hand-side can be solved, it becomes:

$$\lambda_k + (\eta_k - 1)\sum_{i\in\mathcal{I}_k}\mu_i - \frac{(\eta_k-1)^2}{4\theta_k} \geq l_k - \tau$$
$$\equiv \begin{cases} \left(\lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}_k}\mu_i - (l_k-\tau) + \theta_k\right)^2 \geq \\ \qquad \left(\lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}_k}\mu_i - (l_k-\tau) - \theta_k\right)^2 + (\eta_k-1)^2 \\ \lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}_k}\mu_i - (l_k-\tau) \geq 0. \end{cases}$$

Similarly, the second constraint is equivalent to

$$\lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i - \frac{\eta_k^2}{4\theta_k} \geq 0$$
$$\equiv \begin{cases} \left(\lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i + \theta_k\right)^2 \geq \left(\lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i - \theta_k\right)^2 + \eta_k^2 \\ \lambda_k + \eta_k\sum_{i\in\mathcal{I}_k}\mu_i \geq 0. \end{cases}$$

Thus, the inner problem is equivalent to the optimization problem with second-order cone constraints, as provided in the proposition. □

Proof of Proposition 3.2. From the results in Lemma 3.1 and the definition of $\mathcal{I}_k$, the distributionally robust optimization problem (3.26) can be reformulated as

$$\begin{aligned}
\min_{\boldsymbol{Y},\boldsymbol{\lambda},\boldsymbol{\eta},\boldsymbol{\theta}} \quad & \sum_{k\in\mathcal{K}}\left(\lambda_k + \eta_k\sum_{i\in\mathcal{I}}\mu_i y_{ik} + \theta_k\sum_{i\in\mathcal{I}}\sigma_i^2 y_{ik}\right) \\
\text{s.t.} \quad & \left(\lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}}\mu_i y_{ik} - (l_k-\tau) + \theta_k\right)^2 \geq \\
& \qquad \left(\lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}}\mu_i y_{ik} - (l_k-\tau) - \theta_k\right)^2 + (\eta_k-1)^2, \forall k\in\mathcal{K}, \\
& \lambda_k + (\eta_k-1)\sum_{i\in\mathcal{I}}\mu_i y_{ik} - (l_k-\tau) \geq 0, \forall k\in\mathcal{K}, \\
& \left(\lambda_k + \eta_k\sum_{i\in\mathcal{I}}\mu_i y_{ik} + \theta_k\right)^2 \geq \left(\lambda_k + \eta_k\sum_{i\in\mathcal{I}}\mu_i y_{ik} - \theta_k\right)^2 + \eta_k^2, \forall k\in\mathcal{K}, \\
& \lambda_k + \eta_k\sum_{i\in\mathcal{I}}\mu_i y_{ik} \geq 0, \forall k\in\mathcal{K}, \\
& \boldsymbol{\theta} \geq 0, \\
& \text{Constraints in DOA-SAA.}
\end{aligned}$$

We replace $\lambda_k + \eta_k \sum_{i\in\mathcal{I}} \mu_i y_{ik} = \varrho_k$ and simplify the above program as:

$$\min_{\boldsymbol{Y},\boldsymbol{\eta},\boldsymbol{\theta},\boldsymbol{\varrho}} \quad \sum_{k\in\mathcal{K}} \left[\varrho_k + \theta_k \sum_{i\in\mathcal{I}} \sigma_i^2 y_{ik} \right]$$

$$\text{s.t.} \quad \left(\varrho_k - \sum_{i\in\mathcal{I}} \mu_i y_{ik} - (l_k - \tau) + \theta_k \right)^2 \geq \tag{B.1}$$

$$\left(\varrho_k - \sum_{i\in\mathcal{I}} \mu_i y_{ik} - (l_k - \tau) - \theta_k \right)^2 + (\eta_k - 1)^2, \forall k \in \mathcal{K}, \tag{B.2}$$

$$\varrho_k - \sum_{i\in\mathcal{I}} \mu_i y_{ik} - (l_k - \tau) \geq 0, \forall k \in \mathcal{K}, \tag{B.3}$$

$$(\varrho_k + \theta_k)^2 \geq (\varrho_k - \theta_k)^2 + \eta_k^2, \forall k \in \mathcal{K}, \tag{B.4}$$

$$\varrho_k, \theta_k \geq 0, \forall k \in \mathcal{K}, \tag{B.5}$$

Constraints in DOA-SAA.

Now we consider the above minimization problem with a fixed $\boldsymbol{Y}$. After fixing $\boldsymbol{Y}$, the resulting minimization problem boils down to K independent subproblems, which corresponds to K drivers. Let the KKT multipliers be $\alpha_1, \alpha_2, \ldots, \alpha_5 \geq 0$ (the subscript k is dropped here for brevity), corresponding to constraints (B.2)-(B.5). The stationarity conditions can be written as follows:

$$1 = 4\theta_k\alpha_1 + \alpha_2 + 4\theta_k\alpha_3 + \alpha_4, \tag{B.6}$$

$$0 = -2(\eta_k - 1)\alpha_1 - 2\eta_k\alpha_3, \tag{B.7}$$

$$\sum_{i\in I} \sigma_i^2 y_{ik} = 4(\varrho_k - \sum_{i\in I} \mu_i y_{ik} - (l_k - \tau))\alpha_1 + 4\varrho_k\alpha_3 + \alpha_5. \tag{B.8}$$

First, we show that $\varrho_k > 0$ and $\theta_k > 0$. 1) If $\varrho_k = 0$, then from constraints (B.4) we have $\eta_k = 0$. It follows that $\alpha_1 = 0$ based on the stationarity condition (B.7). Then from (B.8), $\alpha_5 = \sum_{i\in I} \sigma_i^2 y_{ik} > 0$. Hence $\theta_k = 0$ by the complementary slackness, which indicates that $\eta_k = 1$ from constraints (B.2). Thus we get a contradiction. 2) If $\theta_k = 0$, constraints (B.2) imply that $\eta_k = 1$ while constraints (B.4) imply $\eta_k = 0$, which generates a contradiction. As a result, ϱ_k and θ_k must be both positive, and $\alpha_4 = \alpha_5 = 0$.

Second, we prove that $\varrho_k > \sum_{i\in I} \mu_i y_{ik} + (l_k - \tau)$ by contradiction. If $\varrho_k = \sum_{i\in I} \mu_i y_{ik} + (l_k - \tau)$, then $\eta_k = 1$ as implied by constraints (B.2). From the stationarity condition (B.7), we have $\alpha_3 = 0$. It follows that the RHS of constraint (B.8) is zero while the LHS is strictly positive. Hence $\varrho_k > \sum_{i\in I} \mu_i y_{ik} + (l_k - \tau)$, and $\alpha_2 = 0$ by the complementary slackness.

We can now rewrite the stationarity conditions as:

$$1 = 4\theta_k(\alpha_1 + \alpha_3), \tag{B.9}$$

$$2\alpha_1 = 2\eta_k(\alpha_1 + \alpha_3), \tag{B.10}$$

$$\sum_{i\in I} \sigma_i^2 y_{ik} = 4(\varrho_k - \sum_{i\in I} \mu_i y_{ik} - h)\alpha_1 + 4\varrho_k \alpha_3. \tag{B.11}$$

Next we can show that $\alpha_1, \alpha_3 > 0$ by contradiction. 1) If $\alpha_1 = 0$, then we have $\theta_k, \alpha_3 > 0$ from (B.9) and $\eta_k = 0$ from (B.10). However, by the complementary slackness, $\alpha_3 > 0$ indicates $(\varrho_k + \theta_k)^2 = (\varrho_k - \theta_k)^2$, which is impossible as both ϱ_k and θ_k are positive. 2) If $\alpha_3 = 0$, stationarity conditions (B.9) and (B.10) imply that $\alpha_1 = 1/2\theta_k > 0$ and $\eta_k = 1$. By the complementary slackness, constraints (B.2) must be satisfied at the equality, i.e., $\rho_k = \sum_{i\in I} \mu_i y_{ik} + (l_k - \tau)$ (recall $\theta_k > 0$). Then the RHS of (B.11) is zero while the LHS is strictly positive, which leads to a contradiction. Consequently, we have both α_1 and α_3 are positive. Then the complementary slackness implies:

$$4(\varrho_k - \sum_{i\in I} \mu_i y_{ik} - h)\theta_k = (\eta_k - 1)^2, \tag{B.12}$$

$$4\varrho_k\theta_k = \eta_k^2. \tag{B.13}$$

Then we can solve for $\varrho_k, \theta_k, \eta_k$ and α_1, α_3 using the above five equations (B.9)-(B.13):

$$\begin{aligned}
\varrho_k &= \frac{\left[\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau) + \sqrt{(\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau))^2 + \sum_{i\in I} \sigma_i^2 y_{ik}}\,\right]^2}{4\sqrt{(\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau))^2 + \sum_{i\in I} \sigma_i^2 y_{ik}}}, \\
\eta_k &= \frac{\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau) + \sqrt{(\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau))^2 + \sum_{i\in I} \sigma_i^2 y_{ik}}}{2\sqrt{(\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau))^2 + \sum_{i\in I} \sigma_i^2 y_{ik}}}, \\
\theta_k &= \frac{\varrho_k - (\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau))\eta_k}{\sum_{i\in I} \sigma_i^2 y_{ik}}.
\end{aligned}$$

As a result, we have

$$\varrho_k + \theta_k \sum_{i\in I} \sigma_i^2 y_{ik} = \sum_{i\in I} \mu_i y_{ik} + (l_k - \tau) + \sqrt{(\sum_{i\in I} \mu_i y_{ik} + (l_k - \tau))^2 + \sum_{i\in I} \sigma_i^2 y_{ik}}.$$

The original robust optimization formulation thus is equivalent to:

$$\begin{aligned}
\min_{\boldsymbol{Y},\boldsymbol{\rho}} \quad & \sum_{k\in\mathcal{K}} \left(\rho_k + \sum_{i\in\mathcal{I}} \mu_i y_{ik} + l_k - \tau\right) \\
\text{s.t.} \quad & \rho_k^2 \geq \sum_{i\in\mathcal{I}} \sigma_i^2 y_{ik} + (\sum_{i\in\mathcal{I}} \mu_i y_{ik} + (l_k - \tau))^2, \forall k \in \mathcal{K}, \\
& \text{Constraints in DOA-SAA.}
\end{aligned}$$

The second constraints can be transformed to

$$\rho_k^2 \geq \sum_{i\in\mathcal{I}} \sigma_i^2 y_{ik}^2 + (\sum_{i\in\mathcal{I}} \mu_i y_{ik} + (l_k - \tau))^2, \forall k \in \mathcal{K},$$

by utilizing the fact that y_{ik} is binary. Thus, the resulting optimization model is a MISOCP. □

Proof of Proposition 3.3. We will follow similar steps in the previous proofs. Let $\bar{T}_k = \sum_{i\in\mathcal{I}_k} \tilde{t}_i + \tilde{e}_k l_k$. With the same argument in Proof of Lemma 3.1, the DOA-DROt model is also decomposable as follows:

$$\begin{aligned} \min_{Y} \quad & \sum_{k\in\mathcal{K}} \max_{\bar{\mathbb{Q}}_k \in \bar{\mathbb{G}}_k} \mathbb{E}_{\bar{\mathbb{Q}}_k} \left(\bar{T}_k + l_k - \tau\right)^+ \\ \text{s.t.} \quad & \text{Constraints in DOA-SAA.} \end{aligned}$$

where $\bar{\mathbb{Q}}_k$ is in the ambiguity set $\bar{\mathbb{G}}_k$, defined by

$$\bar{\mathbb{G}}_k = \left\{ \bar{\mathbb{Q}}_k \in \mathcal{P}_0(\mathbb{R}) \;\middle|\; \begin{array}{l} \mathbb{E}_{\bar{\mathbb{Q}}_k}(\bar{T}_k) = \sum_{i\in\mathcal{I}_k} \mu_i \\ \mathbb{E}_{\bar{\mathbb{Q}}_k}\left[\left(\bar{T}_k - \sum_{i\in\mathcal{I}_k} \mu_i\right)^2\right] = \sum_{i\in\mathcal{I}_k} \sigma_i^2 + s_k^2 l_k^2 \end{array} \right\}.$$

Correspondingly, the inner problem $\max_{\bar{\mathbb{Q}}_k\in\bar{\mathbb{G}}_k} \mathbb{E}_{\bar{\mathbb{Q}}_k} \left(\bar{T}_k + l_k - \tau\right)^+$ can be solved by:

$$\begin{aligned} \min_{\lambda_k,\eta_k,\theta_k} \quad & \lambda_k + \eta_k \sum_{i\in\mathcal{I}_k} \mu_i + \theta_k \left(\sum_{i\in\mathcal{I}_k} \sigma_i^2 + s_k^2 l_k^2\right) \\ \text{s.t.} \quad & \lambda_k + (\eta_k - 1)\sum_{i\in\mathcal{I}_k} \mu_i - l_k + \tau + \theta_k \geq \left\| \begin{pmatrix} \eta_k - 1 \\ \lambda_k + (\eta_k - 1)\sum_{i\in\mathcal{I}_k} \mu_i - l_k + \tau - \theta_k \end{pmatrix} \right\|_2, \\ & \lambda_k + (\eta_k - 1)\sum_{i\in\mathcal{I}_k} \mu_i - l_k + \tau \geq 0, \\ & \lambda_k + \eta_k \sum_{i\in\mathcal{I}_k} \mu_i + \theta_k \geq \left\| \begin{pmatrix} \eta_k \\ \lambda_k + \eta_k \sum_{i\in\mathcal{I}_k} \mu_i - \theta_k \end{pmatrix} \right\|_2, \\ & \theta_k \geq 0. \end{aligned}$$

Compared to Lemma 3.1, the only difference here is that the term $\theta_k \sum_{i\in\mathcal{I}_k} \sigma_i^2$ in Lemma 3.1 is replaced by $\theta_k \left(\sum_{i\in\mathcal{I}_k} \sigma_i^2 + s_k^2 l_k^2\right)$ in the objective function. Thus, by the steps in the proof of Proposition 3.2, we are able to obtain the MISOCP in (3.28). We omit the duplicated steps for brevity. □

The Formulation of One-Way Traveling Salesman Problem

Given the set of realized customer locations $\mathcal{V} \subset \mathcal{I}$ and the depot node 0, we also introduce a dummy node $0'$ to facilitate the one-way travel distance calculation. Define a complete arc set $\mathcal{A}_{\mathcal{V}}$ on the node set $\mathcal{V}' = \mathcal{V} \cup \{0, 0'\}$ and each arc $(i, j) \in \mathcal{A}_{\mathcal{V}}$ is associated with a distance d_{ij}. The distance between the dummy node and all other nodes are 0, i.e. $d_{0'i} = d_{i0'} = 0$ for $i \in \mathcal{V} \cup \{0\}$. The decision variables are binary variables ζ_{ij} that indicate whether the driver travels arc $(i, j) \in \mathcal{A}_{\mathcal{V}}$. The formulation for the one-way traveling salesman problem is

$$\min \quad \sum_{(i,j)\in\mathcal{A}} \zeta_{ij} d_{ij}, \tag{B.14}$$

$$\sum_{j\in\mathcal{I}'} \zeta_{ij} = 1, \quad \forall i \in \mathcal{I}, \tag{B.15}$$

$$\sum_{j\in\mathcal{I}'} \zeta_{ji} = 1, \quad \forall i \in \mathcal{I}, \tag{B.16}$$

$$\sum_{i\in\mathcal{I}} \zeta_{0i} = 1, \tag{B.17}$$

$$\sum_{i\in\mathcal{I}} \zeta_{i0'} = 1, \tag{B.18}$$

$$\zeta_{0'0} = 1, \tag{B.19}$$

$$\sum_{i,j\in\mathcal{S}} \zeta_{ij} \leq |S| - 1, \quad \forall S \subset \mathcal{I}',\ 2 \leq |S| \leq I. \tag{B.20}$$

Constraints (B.15) and (B.16) are degree (flow) constraints for customer nodes. Constraint (B.17) ensures that the route begins from the depot. Constraint (B.18) specifies that the dummy node is entered once from the customer nodes and constraint (B.19) requires the route returns to the depot through the dummy node so no return trip cost is incurred. Constraints (B.20) are subtour elimination constraints that prevents the formation of illegal subtours. We implement the subtour elimination constraints as lazy constraints in Gurobi.

The Detailed Description of SP in the Branch-and-Price Algorithm

Constraints of DOA-SAA

The complete set of constraints in the pricing subproblem of DOA-SAA is:

$$\sum_{i\in\mathcal{I}} \bar{y}_i \leq N \tag{B.21}$$

$$c(\bar{\boldsymbol{y}}) = \sum_{s\in S} w^s, \tag{B.22}$$

$$\omega^s \geq \sum_{i\in\mathcal{I}} t_i^s \bar{y}_i + \frac{l}{v} - \tau, \quad \forall s \in \mathcal{S} \tag{B.23}$$

$$\omega^s \geq 0, \quad \forall s \in \mathcal{S}, \tag{B.24}$$

$$l = \sum_{j=0}^{N} f_j, \tag{B.25}$$

$$D_j^+ u_j \geq f_j \geq D_j^- u_j, \quad \forall j \in \{1,\dots,N\}, \tag{B.26}$$

$$\beta_0 d + \beta_1 a + \beta_2 b + \beta_3 a\sqrt{j-1} + \beta_4 b\sqrt{j-1} + \beta_5 n - D_j^-(1-u_j) \geq f_j, \quad \forall j \in \{1,\dots,N\}, \tag{B.27}$$

$$f_j \geq \beta_0 d + \beta_1 a + \beta_2 b + \beta_3 a\sqrt{j-1} + \beta_4 b\sqrt{j-1} + \beta_5 n - D_j^+(1-u_j), \quad \forall j \in \{1,\dots,N\}, \tag{B.28}$$

$$d = \sum_{i\in\mathcal{I}} \hat{d}_i \bar{x}_i, \quad \forall k \in \mathcal{K}, \tag{B.29}$$

$$\sum_{i\in\mathcal{I}} \bar{x}_i \geq \bar{y}_i, \quad \forall i \in \mathcal{I}, \tag{B.30}$$

$$\bar{x}_i \leq \bar{y}_i, \quad \forall i \in \mathcal{I}, \tag{B.31}$$

$$n = \sum_{j=0}^{N} j \cdot u_j, \tag{B.32}$$

$$\sum_{j=0}^{N} u_j = 1, \tag{B.33}$$

$$a = \bar{a} + \underline{a} \geq 0, \tag{B.34}$$

$$b = \bar{b} + \underline{b} \geq 0, \tag{B.35}$$

$$\bar{a} \geq lat_i \cdot \bar{y}_i, \quad \forall i \in \mathcal{I}, \tag{B.36}$$

$$\underline{a} \geq -lat_i + M(\bar{y}_i - 1), \quad \forall i \in \mathcal{I}, \tag{B.37}$$

$$\bar{b} \geq long_i \cdot \bar{y}_i, \quad \forall i \in \mathcal{I}, \tag{B.38}$$

$$\underline{b} \geq -long_i + M(\bar{y}_i - 1), \quad \forall i \in \mathcal{I}, \tag{B.39}$$

$$u_j \in \{0,1\}, \quad \forall j \in \{0,1,...,N\}, \tag{B.40}$$

$$\bar{y}_i \in \{0,1\}, \quad \forall i \in \mathcal{I}. \tag{B.41}$$

The constraints have similar meanings to those in the original DOA-SAA except that the zone subscript is removed.

Constraints of DOA-DRO

In DOA-DRO, the delay cost $c(\bar{\mathbf{y}})$ has a different form than DOA-SAA but the constraints defining the travel distance l remains the same. So the complete set of constraints in the SP of DOA-DRO can be stated as

$$\sum_{i\in\mathcal{I}} \bar{y}_i \leq N, \tag{B.42}$$

$$c(\bar{\boldsymbol{y}}) = \rho + h + \sum_{i\in\mathcal{I}} \mu_i \bar{y}_i, \tag{B.43}$$

$$h = \frac{l}{v} - \tau, \tag{B.44}$$

$$\rho^2 \geq \sum_{i\in\mathcal{I}} \sigma^2 y_i^2 + \left(\sum_{i\in\mathcal{I}} \mu_i y_i + h\right)^2, \tag{B.45}$$

$$\text{Constraints (B.25)-(B.41).} \tag{B.46}$$

Constraints of VRP-SAA

In addition to the set of customer locations $\mathcal{I}$ and the depot node 0, a dummy node $0'$ is added as in the one-way TSP formulation. Similarly, we define a complete arc set $\mathcal{A}$ on the node set $\mathcal{I}' = \mathcal{I} \cup \{0, 0'\}$ and each arc $(i, j) \in \mathcal{A}$ is associated with a distance d_{ij}. The distance between the dummy node and all other nodes are 0, i.e. $d_{0'i} = d_{i0'} = 0$ for $i \in \mathcal{I} \cup \{0\}$. In the pricing subproblem of VRP-SAA, the decision variables are binary variables ζ_{ij} that indicate whether the driver travels arc $(i, j) \in \mathcal{A}$, as well as the variables $\bar{y}_i$ that indicate whether customer i is covered in this route. The detailed formulation is given as follows:

$$\sum_{i\in\mathcal{I}} \bar{y}_i \leq N, \tag{B.47}$$

$$c(\bar{\boldsymbol{y}}) = \sum_{s\in S} w^s, \tag{B.48}$$

$$\omega^s \geq \sum_{i\in\mathcal{I}} t_i^s \bar{y}_i + \frac{l}{v} - \tau, \quad \forall s \in \mathcal{S}, \tag{B.49}$$

$$l = \sum_{(i,j)\in\mathcal{A}} \zeta_{ij} d_{ij}, \tag{B.50}$$

$$\sum_{j\in\mathcal{I}'} \zeta_{ij} = \bar{y}_i, \quad \forall i \in \mathcal{I}, \tag{B.51}$$

$$\sum_{j\in\mathcal{I}'} \zeta_{ji} = \bar{y}_i, \quad \forall i \in \mathcal{I}, \tag{B.52}$$

$$\sum_{i\in\mathcal{I}} \zeta_{0i} = 1, \tag{B.53}$$

$$\sum_{i\in\mathcal{I}} \zeta_{i0'} = 1, \tag{B.54}$$

$$\zeta_{0'0} = 1, \tag{B.55}$$

$$\sum_{i,j\in\mathcal{S}} \zeta_{ij} \leq |S| - 1, \quad \forall S \subset \mathcal{I}',\ 2 \leq |S| \leq I. \tag{B.56}$$

Constraints (B.51)-(B.56) play the same roles as in the one-way TSP formulation.

Constraints of VRP-DRO

The constraints in the pricing subproblem of VRP-DRO can be obtained by combining the constraints from DOA-DRO and VRP-SAA: (B.42)-(B.45) and (B.50)-(B.56).

Appendix C

Proofs for Results in Chapter 4

Proof of Lemma 4.1. Let two sets of road segments $\mathcal{A}$ and $\mathcal{B}$ satisfy $\mathcal{A} \subseteq \mathcal{B}$. Consider adding a road segment $k \notin B$ to the two sets. In terms of the first part of the objective function, the change caused by adding k is the same for $\mathcal{A}$ and $\mathcal{B}$ because of linearity. For the second part, as $\lambda \geq 0$ and $d_{ij} \geq 0$ for all $(i, j) \in N$, the value of the second part can only increase with the newly added e. Since $\mathcal{B}$ can only contain more road segments than $\mathcal{A}$, there are potentially more neighbors of k that are included in $\mathcal{B}$. As a result, the increase of objective value caused by adding e is greater for $\mathcal{B}$ than $\mathcal{A}$. □

Proof of Proposition 4.1. Directly from Lemma 4.1, BL-AC is essentially a supermodular knapsack problem, which admits a polynomial-time solvable Lagrangian dual, as shown in Gallo and Simeone, 1989. □

Proof of Theorem 4.2. Consider two bike lane construction plans, i.e. sets of road segments $\mathcal{A}$ and $\mathcal{B}$ to build bike lanes satisfying $\mathcal{A} \subseteq \mathcal{B}$, and a road segment $i \notin \mathcal{B}$. We prove the supermularity of $v_x(r)$ by conditioning on i: 1) If $i \notin r$, then building a bike lane on i does not impact the value of $v_x(r)$, so $v_{\mathcal{A}\cup\{i\}}(r) = v_A(r)$ and $v_{\mathcal{B}\cup\{i\}}(r) = v_B(r)$. 2) If $i \in r$, let $i-1$ and $i+1$ denote the road segment visited before and after i on the trajectory r, respectively (when i is at the head or the tail of the trajectory, either $i-1$ or $i+1$ is empty and our analysis can be easily extended). We further consider the following two scenarios:

- If both $i-1$ and $i+1$ do not belong to $\mathcal{A}$, then $v_{\mathcal{A}\cup\{i\}}(r) - v_{\mathcal{A}}(r) = f(1)$. Then if either $i-1$ or $i+1$ belongs to $\mathcal{B}$, we have $v_{\mathcal{B}\cup\{i\}}(r) - v_{\mathcal{B}}(r) = f(|s|+1) - f(|s|)$ for $|s| > 1$. By the assumption that $f(\cdot)$ is increasing convex, we have $f(|s|+1) - f(|s|) > f(1)$. Otherwise, when neither of $i-1$ and $i+1$ belong to $\mathcal{B}$, $v_{\mathcal{B}\cup\{i\}}(r) - v_{\mathcal{B}}(r) = f(1) = v_{\mathcal{A}\cup\{i\}}(r) - v_{\mathcal{A}}(r)$. Hence $v_{\mathcal{B}\cup\{i\}}(r) - v_{\mathcal{B}}(r) \geq v_{\mathcal{A}\cup\{i\}}(r) - v_{\mathcal{A}}(r)$.

- If either or both of $\{i-1, i+1\}$ belong to $\mathcal{A}$, then $v_{\mathcal{A}\cup\{i\}}(r) - v_{\mathcal{A}}(r) = f(|s|+1) - f(|s|)$ for some $|s| > 1$. For $\mathcal{B}$, as $\mathcal{A} \subseteq \mathcal{B}$, we have $v_{\mathcal{B}\cup\{i\}}(r) - v_{\mathcal{B}}(r) = f(|s'|+1) - f(|s'|)$ for $|s'| \geq |s|$. Because $f(\cdot)$ is an increasing convex function, $f(|s'|+1) - f(|s'|) \geq f(|s|+1) - f(|s|)$ and thus $v_{\mathcal{B}\cup\{i\}}(r) - v_{\mathcal{B}}(r) \geq v_{\mathcal{A}\cup\{i\}}(r) - v_{\mathcal{A}}(r)$.

In all the cases we have shown that $v_{\mathcal{B}\cup\{i\}}(r) - v_{\mathcal{B}}(r) \geq v_{\mathcal{A}\cup\{i\}}(r) - v_{\mathcal{A}}(r)$ so $v_x(r)$ is indeed supermodular. □

Proof of Corollary 4.2.1. Based on Theorem 4.2, the utility function $v_x(r)$ is supermodular and maximization of this utility function over a budget constraint is a supermodular knapsack problem, which has a polynomial-time solvable Lagrangian dual due to the results from Gallo and Simeone, 1989 □

Proof of Proposition 4.2. Following the reformulation of $v_x(r)$ as a polynomial function (4.13), we can further apply standard linearization techniques to get a MILP formulation, as shown in BL-GU-MILP. □

Proof of Proposition 4.3. When $f(\cdot)$ is an increasing convex function, the coefficients β_l in BL-GU-MILP are all positive: $\beta_l = f(|l|) - 2f(|l|-1) + f(|l|-2) = f(|l|) - f(|l|-1) - (f(|l|-1) - f(|l|-2)) \geq 0$. Then we can remove constraints (4.14) from the formulation. The remaining constraints are constraints (4.15) and constraints (4.17) in addition to the budget constraints. Our goal is to prove that the constraint matrix consisting of these two types of constraints are totally unimodular. It is well known that matrix $(P|I)^T$ is totally unimodular if P is totally unimodular. Note that constraints (4.17) correspond to an identity matrix and does not influence the totally unimodularity result so we can focus on the matrix of constraints (4.15), denoted by P. With or without using the reduction techniques, each row of P only includes one 1 and one -1. Then we can show P^T is totally unimodular since it satisfies: 1) Each entry of P is $\{-1, 0, 1\}$; 2) Each column contains at most two non-zero coefficients; 3) There exists a partition (M_1, M_2) of the set M of the set of rows of P such that each column j containing two non-zero coefficients satisfies $\sum_{i\in M_1} p_{ij} - \sum_{i\in M_2} p_{ij} = 0$. It is obvious that conditions 1) and 2) hold for P, and the third condition is satisfied by using the partition $(P, \emptyset)$. It follows that P is totally unimodular because the transpose of a totally unimodular matrix is also totally unimodular. As a result, the Lagrangian relaxation of BL-GU-MILP with relaxed budget constraint can be solved as a LP. □