Statistical metamodeling of dynamic network loading

Wenjing Song[a], Ke Han[b,*], Yiou Wang[a], Terry L. Friesz[a], Enrique del Castillo[a]

[a]*Department of Industrial and Manufacturing Engineering, Pennsylvania State University, USA.*
[b]*Department of Civil and Environmental Engineering, Imperial College London, United Kingdom.*

Abstract

Dynamic traffic assignment models rely on a network performance module known as *dynamic network loading* (DNL), which expresses the dynamics of flow propagation, flow conservation, and travel delay at a network level. The DNL defines the so-called network *delay operator*, which maps a set of path departure rates to a set of path travel times. It is widely known that the delay operator is not available in closed form, and has undesirable properties that severely complicate DTA analysis and computation, such as discontinuity, non-differentiability, non-monotonicity, and computational inefficiency. This paper proposes a fresh take on this important and difficult problem, by providing a class of surrogate DNL models based on a statistical learning method known as *Kriging*. We present a metamodeling framework that systematically approximates DNL models and is flexible in the sense of allowing the modeler to make trade-offs among model granularity, complexity, and accuracy. It is shown that such surrogate DNL models yield highly accurate approximations (with errors below 8%) and superior computational efficiency (9 to 455 times faster than conventional DNL procedures). Moreover, these approximate DNL models admit closed-form and analytical delay operators, which are Lipschitz continuous and infinitely differentiable, while possessing closed-form Jacobians. The implications of these desirable properties for DTA research and model applications are discussed in depth.

Keywords:
dynamic traffic assignment, dynamic network loading, delay operator, metamodeling, Kriging, dynamic games, Gaussian processes

1. Introduction

Dynamic traffic assignment (DTA) is the descriptive modeling of time-varying flows on traffic networks consistent with traffic flow theory and travel choice principles. DTA models describe and predict departure rates, departure times and route choices of travelers over a given time horizon. Analytical DTA models consist of two main components: (i) the mathematical expression of trip assignment such as the dynamic extension of the Wardrop's principles (Wardrop, 1952); (ii) the network performance model, which captures the relationships among link entry flow, link exit flow, junction flow, link delay and path delay. The latter is usually referred to as *dynamic network loading* (DNL). The DNL problem gives rise to the *delay operator*, which is interpreted as a mapping from the set of path departure rates to the set of path travel times. Such delay operators will be the main focus of this paper, although other notions of the delay operators, often going by different names, have been invoked in a variety of different contexts (Gentile et al., 2007; Jang et al., 2005; Lo and Szeto, 2002; Perakis and Roels, 2006; Ukkusuri et al., 2012).

Being an integral part of a complete mathematical formulation of DTA problems, the delay operator plays a fundamental role and affects the analytical properties of the DTA models in many different ways. For example, the existence of *dynamic user equilibrium* (DUE), which is the most widely studied form of DTA problems, depends on the continuity of the delay operators (Han et al., 2013c; Smith and Wisten, 1995; Zhu and Marcotte, 2000), while the uniqueness of DUE is guaranteed by the monotonicity of the delay operator (Mounce and Smith, 2007). Moreover,

*Corresponding author. Tel.: +44-020-7594-5682 ; fax:+44-020-7594-6102.
Email addresses: wus145@psu.edu (Wenjing Song), k.han@imperial.ac.uk (Ke Han), yiouwang2012@gmail.com (Yiou Wang), tfriesz@psu.edu (Terry L. Friesz), exd13@psu.edu (Enrique del Castillo)

all computational procedures for DUE problems rely on certain versions of continuity and monotonicity to converge (Friesz et al., 2011; Han et al., 2015; Long et al., 2013; Mounce, 2006; Szeto and Lo, 2004). Furthermore, differentiability is typically required for the delay operator for problems such as sensitivity analysis (Chung et al., 2014; Yang and Bell, 2007) and mathematical programs with equilibrium constraints (Yang et al., 2007).

For large-scale networks, and sophisticated traffic models that capture phenomena such as shock waves and car spillback, it has been recognized that the DNL models are rather complicated, and the corresponding delay operators enjoy very few analytical properties essential for the analysis and computation of DTA models. For instance, DNL with physical-queue models (e.g. Lighthill-Whitham-Richards model, cell transmission model, link transmission model) is known to yield discontinuities in the delay operator for general networks (Szeto, 2003; Han et al., 2016a). As a consequence, the delay operator is non-differentiable. Furthermore, the non-monotonicity of the delay operators on networks has been reported in the literature (Mounce and Smith, 2007) and has been the major hurdle to computing DUE solutions with convergence guarantee.

In general, the delay operator is not available in closed form, and has to be numerically evaluated via the DNL procedure. Such a procedure is based on a series of link, node, and path dynamic models that typically involve solving ordinary or partial differential equations (Friesz et al., 2013; Gentile et al., 2007; Lo and Szeto, 2002; Perakis and Roels, 2006; Szeto and Lo, 2004; Yperman et al., 2005). Expressing the complete DNL model analytically and embedding it into certain math programming formulation is therefore an onerous task, and could severely complicate the DTA computational procedures (Friesz et al., 2013; Szeto and Lo, 2004; Ukkusuri et al., 2012). Moreover, conventional DNL procedures tend to be computationally demanding, and could scale badly as the network size increases.

Given the aforementioned theoretical and computational limitations of the conventional way of exploring the delay operator, this paper proposes a fresh take on this classical problem from the novel perspective of *statistical metamodeling*. Our goal is to provide a class of surrogate DNL models that approximate the exact ones, with considerable benefits including closed-form expression, improved regularity (e.g. continuity, differentiability), and superior computational efficiency, at the expense of minor yet controllable approximation errors.

Metamodeling is the process of building a "model of the model", i.e., a surrogate approximation of the exact models/processes in order to improve the computation efficiency or gain better analytical properties (Wang and Shan, 2007). Specifically, we treat the delay operator as a highly nonlinear mapping from the set of path departure rates to the set of path travel times, and interpret and approximate its inherent input-output mechanism using a statistical learning technique called *Kriging* (Matheron, 1963). Kriging considers the observed input-output functional relationship as a realization of a Gaussian Random Field (GRF), and the resulting estimation corresponds to the posterior predictive density of the function approximation. The proposed framework is general enough to handle a wide range of DNL models with different traffic flow dynamics (i.e. it is a "model of DNL models"). Each output of this metamodel is an approximate DNL model with closed-form expression and superior regularity and computational efficiency.

Statistical/machine learning algorithms learn from and make predictions based on data, sometimes without exploration of the behavioral foundations and plausibility of the learning processes on which they are based. However, this is not the case in our proposed Kriging framework. Firstly, instead of performing unstructured interpretation of the input-output mechanisms, the proposed approach utilizes information on the network structure, path, and time to identify parts of the input variables that are likely to be correlated, and defines the correlation functions accordingly. Secondly, although the path delay operators may be discontinuous, it is understood that the congestion effect, observed at some point in the spatio-temporal domain, tend to progress in a continuous way in space and time due to the finite propagation speeds of traffic characteristics (e.g. kinematic waves). The potential discontinuity of path delay operators may be dealt with in Kriging by making appropriate regions of the spatio-temporal surface very steep yet smooth.

There is a large and rapidly growing literature on machine learning methods (Hastie et al., 2009; James et al., 2013; Murphy, 2012). Among the many choices of statistical/machine learning techniques that can be used for metamodeling, Kriging is the only technique that has *all* the following 3 properties:

- it is an *exact interpolator*. This means that when approximating a function $Y(\boldsymbol{x})$ with a metamodel $\hat{Y}(\boldsymbol{x})$ at a data point $\boldsymbol{x}$ used to fit the model (a point in the *training* dataset), $\hat{Y}(\boldsymbol{x}) = Y(\boldsymbol{x})$ (i.e, the prediction error at $\boldsymbol{x}$ is zero). This is an important property in metamodeling if running the original model to be approximated is very expensive or time consuming. It is therefore convenient that the metamodel mimics the original model exactly at the instances where the original model was already ran. This property is *not* shared by other popular

techniques such as Support Vector Regression (SVR), Neural Network (NN), or Random Forest (RF);

- it provides a predictor that is closed-form and analytic. This is a property shared by Kriging and SVRs, but not by NNs or RFs. A closed-form Kriging predictor has a number of advantages in DTA applications as we have already discussed before;

- it provides a closed-form, analytic expression of the prediction error variance (or standard error). This property is exclusive to Kriging thanks to the interpretation of its predictor as the mean of the predictive posterior density of a Gaussian Process (or Gaussian Random Field). The underlying Gaussianity assumption is used only if prediction intervals are desired, in which case Kriging provides minimum mean square error predictions (Santner et al., 2013). If the Gaussianity assumption does not hold, the Kriging predictor can be shown to be the Best *Linear* Unbiased Predictor (Santner et al., 2013).

In common with other Kernel-based machine learning methods such as SVRs or Smoothing Splines, Kriging is highly flexible. Flexibility refers to its ability to fit functions/mappings with different degrees of smoothness, whether they are differentiable or not (Santner et al., 2013). Kriging does not rely on any given functional form (in contrast to, e.g. polynomial-based fitting), which is suitable for our problem since the delay operator cannot be explicitly expressed and enjoys little regularity.

We perform Kriging on a non-conventional space with network-specific distance metric. The metamodel requires a set of training data consisting of both inputs (i.e. path departure rates) and outputs (i.e. path travel times) of the delay operator to be approximated, which are generated by conducting the conventional DNL procedure; e.g. following Daganzo (1995), Han et al. (2016a) and Yperman et al. (2005). Then, based on statistical learning and parameter estimation, we are able to provide an analytical and closed-form predictor as a surrogate DNL model.

We note that the proposed metamodeling approach is not an attempt to statistically fit a model in an *ad hoc* fashion. Rather, it seeks to represent the complicated nature of the DNL problems in a systematic, efficient, and parsimonious way – in the same way any model tries to describe physical processes with fewer variables and simpler assumptions. Any output of the metamodeling is a new DNL model that strikes a balance between physical accuracy in terms of capturing the network congestion effect, and tractability barely seen in existing DNL models.

In addition, Kriging is capable of handling high dimensionality. As we shall show later, through working on an input space with reduced dimension and customized distance metric, we will be able to handle large-scale dynamic networks quite efficiently. When the problem size further increases, more advanced experimental design and parameter estimation methods are required, and the Kriging framework easily allows these extensions. Some other machine learning methods can handle the high dimensionality of DNL problems, such as Neural Networks (Haykin, 2008). However, NNs can be difficult to implement because the trained neural networks do not provide predictions as closed-form and analytical expressions. Rather, they are expressed in terms of numerous neurons and activation functions without meaningful interpretation of the network structure or traffic dynamics. In contrast, Kriging yields zero error at the training data and provides analytical expressions of the delay operators, which are essential for theoretical investigation of the DUE and DTA problems (e.g. see Section 5). A numerical comparison of the Kriging and Neural Network approaches is presented in Section 6.3.

This paper explores the potential of a new generation of dynamic network performance models which, if utilized properly, could benefit DTA research and bring fundamental changes to the way transportation networks are modeled. The rest of this paper is organized as follows. Section 2 provides a brief review of state of the arts in topics including dynamic network loading, metamodeling, and Kriging. In Section 3 we articulate the mathematical notion of delay operator in both continuous- and discrete-time settings. Section 4 presents the metamodeling framework and technical details of the Kriging method. In Section 5, we discuss in detail the closed-form representations of the proposed delay operators, their analytical properties, and impacts on dynamic traffic assignment problems. In Section 6 we show the efficacy of the proposed surrogate DNL models in approximating the exact ones and their superior computational efficiencies. Finally, Section 7 provides a few concluding remarks and future research directions.

2. Literature review

2.1. Dynamic network loading

In DTA modeling, the DNL sub-problem aims at describing and predicting the spatial-temporal evolution of traffic flows and congestion on a network that is consistent with established route and departure time choices of drivers. This is done by introducing appropriate dynamics to flow propagation, flow conservation, and travel delays on a network level. Any DNL must be consistent with the established path departure rates and link delay model, and is usually performed under the *first-in-first-out* (FIFO) rule. A few link flow models commonly employed for the DNL procedure include the link delay model (Friesz et al., 1993), the point-queue model (Han et al., 2013a,b), the cell transmission model (Daganzo, 1995), and the link transmission model (Yperman et al., 2005; Han et al., 2016b). Studies of the dynamic network loading models date back to the 1990's with a significant number of publications (Friesz et al., 2013; Lo and Szeto, 2002; Szeto, 2003; Szeto and Lo, 2004; Ukkusuri et al., 2012). Among these advancements some have focused on capturing realistic network congestion effects such as the formation, propagation and dissipation of queues and spillbacks (Nie and Zhang, 2010; Han et al., 2016a), while others have managed to reduce the complexity of the network dynamics and improve the computational efficiency (Yperman et al., 2005; Gentile et al., 2007).

2.2. Metamodeling

Metamodeling is first proposed for achieving better analytical insight and computational efficiency by providing a surrogate model of a complex process/simulation procedure/function/computer routine (Kleijnen, 2015; Wang and Shan, 2007). In the age of modern information, massive datasets have become available for interpreting complex systems, which inspire growing interests and research efforts to investigate problems with high dimensionality and nonlinearity. In the past decade, new developments in metamodeling have been continuously coming forth, which have become powerful tools for approximating models, exploring design spaces, formulating problems, and supporting optimization, with good accuracy and performance at a reasonable cost (Wang and Shan, 2007).

In general, metamodeling consists of the following three main steps: (1) experimental design/sampling; (2) metamodel choice; and (3) model learning/fitting. The goal of experimental design is to strategically sample the design space to generate sample dataset for better training of models (Fang et al., 2005). The goodness of a design strategy is assessed by the efficiency of sampling the design region and accuracy of the resulting surrogate models. Santner et al. (2013) provide a systematic introduction on space-filling techniques for computer metamodeling and experiments. Steps (2) and (3) are often coupled and considered in line with each other since learning/fitting techniques have evolved to be highly specific and dependent on the metamodel used. Examples of popular metamodels include polynomials (Montgomery, 2008), Kriging (Sacks et al., 1989), neural networks (Cheng and Titterington, 1994), radial basis functions (Fang and Horstemeyer, 2006; Regis and Shoemaker, 2013), multivariate adaptive regression splines (Friedman, 1991), and inductive learning (Wang and Shan, 2007).

The application of metamodeling in transportation is relatively recent, and can be categorized into (1) traffic prediction and (2) network optimization. Table 1 provides a comparison of several relevant studies. Our literature search reveals the non-existence of studies that try to characterize or approximate the network delay operator using metamodeling. Moreover, all the studies on network design & management rely on simulation-based DTA models at the expense of mathematical rigor and solution optimality. For example, the simulation softwares shown in Table 1 do not yield lower-level DUE in the analytical sense; rather, the simulations terminate when some stationarity conditions are met. To the authors' knowledge, there does not exist a single study that combines metamodeling with analytical DTA problems for the benefit of fast computation and sound theoretical foundation, which this paper tries to achieve. The proposed surrogate delay operator holds much promise in facilitating the analysis and computation of DUE and MPEC problems with analytical insights unavailable in the current literature.

2.3. Kriging

Kriging, named after the South African mining engineer Krige, refers to a class of response surface methods that provide estimation of complex and computationally expensive functions, procedures, or systems. The method was first developed in the 1960's (Matheron, 1963), and extensively studies in late 1980's, which provides solid foundation for using Kriging to build surrogate models for deterministic computer codes or experiments, especially when they are high-dimensional and computationally expensive (see Sacks et al. (1989) for a comprehensive review).

Table 1: Comparison of metamodeling in transportation.

	Metamodel	Application	Underlying Model	Object of Surrogate Model
Ciuffo et al. (2013)	Ordinary Kriging	Sensitivity analysis	Mesoscopic simulation (AIMSUN)	Network-wide density, average flow, average delay, average travel time
Zhang et al. (2014)	Stochastic Kriging	Active traffic management	Simulation-based DTA (DynasT)	Network average trip time
Chen et al. (2015)	Universal Kriging	Bi-level network design	Simulation-based DUE (DTALite)	Network average travel time
Chen et al. (2014)	Polynomial, Gaussian RBF, Kriging	Bi-level road pricing	Static MPEC, AIMSUN simulation	Network average travel time
Idé and Kato (2009)	Ordinary Kriging	Travel time estimation	Agent-based simulation (IBM)	Path travel time
Xie et al. (2010)	Universal Kriging	Short-term traffic flow forecast	Empirical data	Link traffic volume
Sun and Xu (2011)	Mixtures of Gaussian processes	Short-term traffic flow prediction	Empirical data	Link traffic volume
Chan et al. (2012)	Neural networks	Short-term traffic flow forecast	Empirical data	Link traffic volume
Wang and Shi (2013)	Hybrid Support Vector Machine	Short-term traffic speed forecast	Empirical data	Link traffic speed
Ye et al. (2012)	Hybrid ARIMA and NN	Short-term traffic speed forecast	Empirical data	Road segment traffic speed

In recent years, due to its capability to explore implicit relationships among intrinsic variables and its flexibility in approximating nonlinearity, Kriging has become an important class of kernel-based learning and metamodeling algorithms. Kriging is widely used in modeling computer experiments, especially exact and deterministic computer programs or routines (Fang et al., 2005), which is ideal for the analytical DNL models that this paper addresses. For some general introduction to Kriging, we refer the reader to Cressie (1993) and Stein (2012). More advanced application of Kriging with experimental design and statistical learning in metamodeling can be found in Fang et al. (2005); Jones et al. (1998); Santner et al. (2013) and Sacks et al. (1989).

3. Dynamic network loading and the delay operator

The notion of dynamic network loading varies in context and application in the literature. In this section we articulate the delay operator that this paper addresses using precise mathematical languages, while referring the reader to a number of other papers that discuss the DNL procedures as well as their numerical implementations in a wider spectrum (Friesz et al., 2013; Gentile et al., 2007; Han et al., 2016a; Huang and Lam, 2002; Ukkusuri et al., 2012). However, we note that the proposed metamodeling framework is applicable regardless of the shape or form of the underlying dynamic network loading model. The DNL model employed in this paper is meant to provide an illustrative example, and our conclusion is not dependent on such a specific choice. The comparison of surrogate delay operators for different DNL models remains a future study.

3.1. Delay operator as an infinite-dimensional mapping

We consider a general network with a time horizon $[t_0,\, t_f]$. Let $\mathcal{P}$ be the set of all paths employed by travelers. For each $p \in \mathcal{P}$, we define its path departure rate $h_p(\cdot)$ as a function of departure time t. Then, we let $h(\cdot) = (h_p(\cdot) :\ p \in \mathcal{P})$ be the vector of path departure rates. The following constraints on the departure rates are commonly employed for *route choice* DUE problems (Smith and Wisten, 1995; Zhu and Marcotte, 2000):

$$\sum_{p \in \mathcal{P}_{rs}} h_p(t) \;=\; R_{rs}(t) \qquad \forall t \in [t_0,\, t_f] \quad \forall (r,\, s) \in \mathcal{W}, \tag{3.1}$$

where $\mathcal{W}$ is the set of origin-destination (O-D) pairs, $\mathcal{P}_{rs}$ is the set of paths that connect O-D pair $(r,\ s) \in \mathcal{W}$. $R_{rs}(t)$ is the time-varying departure rate between O-D pair $(r,\ s)$ (over all possible paths), and is given *a priori*. Therefore, the set of feasible path departure rates can be defined as

$$\Lambda \ = \ \left\{ h(\cdot) : \ \ h_p(t) \geq 0, \ \ \sum_{p \in \mathcal{P}_{rs}} h_p(t) \ = \ R_{rs}(t) \qquad \forall t \in [t_0,\ t_f], \ \ \forall (r,\ s) \in \mathcal{W} \right\}. \tag{3.2}$$

The delay operator Ψ is a mapping that relates the set of path departure rates h to the set of path travel times:

$$\Psi(h) \ \doteq \ \Big(\Psi_p(t;\ h) : \ t \in [t_0,\ t_f], \ p \in \mathcal{P} \Big) \qquad h \in \Lambda, \tag{3.3}$$

where each $\Psi_p(t;\ h)$ denotes the path travel time experienced by drivers departing at time t and following path p, when the path departure rates of the entire network is given by h. In summary, we define the delay operator, which is viewed as an infinite-dimensional mapping, as follows.

Definition 3.1. (Infinite-dimensional delay operator) *Given a road network and the feasible path flow set* (3.2)*, a delay operator* Ψ *is a mapping from the set of path departure rates* $h \in \Lambda$ *to the set of path travel times* $\Psi(h)$*.*

Remark 3.2. *There are many different ways to perform the DNL procedure in order to evaluate the delay operator* Ψ*, including those mentioned in Section 2.1 and at the beginning of Section 3. Each way can be seen as a DNL model and hence there are many different DNL models. On the other hand, this paper proposes one metamodel that can be applied to these individual DNL models by following the same procedure, which is to be elaborated in the sections below.*

3.2. Delay operator as a finite-dimensional mapping

In order to apply the Kriging technique and facilitate numerical implementation, we need to define the delay operator in a finite-dimensional space. To this end, we let n be an arbitrary positive integer, and partition the time horizon $[t_0,\ t_f]$ into n equal sub-intervals, denoted $\mathcal{I}^i,\ i = 1,\ \ldots,\ n$. We define the discrete path departure rate to be a n-dimensional vector $\boldsymbol{h}_p \in \mathbb{R}^n_+$, and the entire vector of path departure rates to be a $n \times |\mathcal{P}|$-dimensional vector $\boldsymbol{h} = (\boldsymbol{h}_p : p \in \mathcal{P}) \in \mathbb{R}^{n \times |\mathcal{P}|}_+$. Then, we define the corresponding continuous-time path departure rates as:

$$h \ = \ \Big(h_p(\cdot) : \ p \in \mathcal{P} \Big) \quad \text{such that} \quad h_p(t) \ \equiv \ \boldsymbol{h}_{p,i} \quad t \in \mathcal{I}^i, \quad \forall 1 \leq i \leq n, \ \ \forall p \in \mathcal{P},$$

h can be viewed as the continuous-time counterpart of $\boldsymbol{h}$. We then construct the mapping $\boldsymbol{\Psi}$ as follows:

$$\boldsymbol{\Psi}(\boldsymbol{h}) \ \doteq \ \Big(\boldsymbol{\Psi}_{p,i}(\boldsymbol{h}) : \ p \in \mathcal{P},\ i = 1,\ \ldots,\ n \Big) \in \mathbb{R}^{n \times |\mathcal{P}|}_+ \qquad \text{where} \qquad \boldsymbol{\Psi}_{p,i}(\boldsymbol{h}) \doteq \frac{1}{|\mathcal{I}^i|} \int_{\mathcal{I}^i} \Psi_p(t;\ h) dt.$$

Let us also define the discrete version of the flow conservation constraints (3.2), as follows:

$$\boldsymbol{\Lambda} \ = \ \left\{ \boldsymbol{h} \in \mathbb{R}^{n \times |\mathcal{P}|}_+, \quad \sum_{p \in \mathcal{P}_{rs}} \boldsymbol{h}_{p,i} \ = \ \boldsymbol{R}^i_{rs} \qquad \forall i = 1,\ \ldots,\ n, \ \ \forall (r,\ s) \in \mathcal{W} \right\}, \tag{3.4}$$

where

$$\boldsymbol{R}^i_{rs} \ \doteq \ \frac{1}{|\mathcal{I}^i|} \int_{\mathcal{I}^i} R_{rs}(t)\, dt$$

can be interpreted as the average O-D departure rate during the i-th time interval.

Definition 3.3. (Finite-dimensional delay operator) *Given a network and the feasible path flow set* (3.4)*, a delay operator* $\boldsymbol{\Psi}$ *is a mapping from the set of discrete path departure rates* $\boldsymbol{h} \in \boldsymbol{\Lambda}$ *to the set of discrete path travel times* $\boldsymbol{\Psi}(\boldsymbol{h})$*.*

As such, $\boldsymbol{\Psi}$ is a mapping between two subsets of the Euclidean space $\mathbb{R}^{n \times |\mathcal{P}|}$, and is viewed as the discrete counterpart of the delay operator Ψ.

Remark 3.4. *The operator* $\mathbf{\Psi}$ *should not be confused with the delay operators with numerical discretization, such as those based on DNL performed via the cell transmission model or the link transmission model. Here, the selection of* n *is arbitrary, not constrained by the network or the time horizon, nor by the numerical discretization scheme (e.g. the Courant-Friedrichs-Lewy condition). The introduction of* $\boldsymbol{h}$ *and* $\mathbf{\Psi}$ *enables us to effectively trade off temporal granularity with dimensionality. Moreover, a coarse time grid (small* n*) also makes sense in practice: In an actual network the average path departure rates or path travel times may not vary much over a time period (e.g. 30 min) that is significantly longer than a typical time step in numerical computations (e.g. 5 s, 1 min). Thus, the delay operator* $\mathbf{\Psi}$ *with a small* n *is useful for provide average travel times within, say, 30 min.*

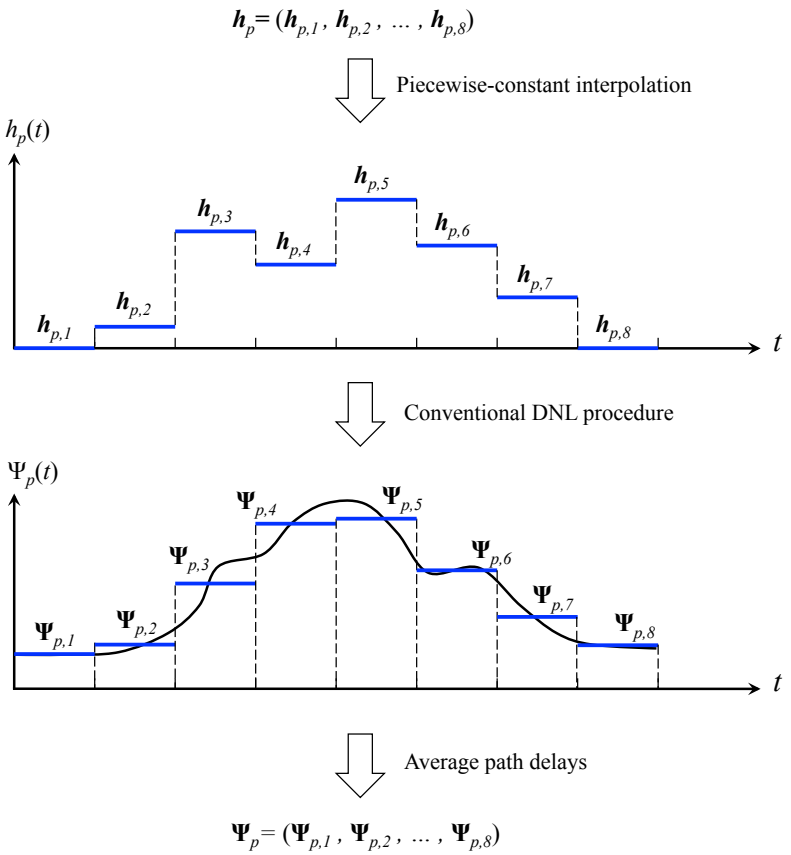


Figure 1: Illustration of the mapping from $\boldsymbol{h}_p$ to $\mathbf{\Psi}_p$ when $n = 8$, which defines the finite-dimensional operator $\mathbf{\Psi}$.

The procedure above defines the delay operator in a finite-dimensional setting, where the vector of path departure rates is represented by the vector $\boldsymbol{h}$ and the corresponding vector of path travel times is $\mathbf{\Psi}(\boldsymbol{h})$. This process is graphically illustrated in Figure 1. The definition of $\mathbf{\Psi}$ allows us to apply Kriging techniques in the finite-dimensional space $\mathbb{R}^{n\times|\mathcal{P}|}$.

4. Methodology

4.1. Statistical model and assumptions

We propose the learning method based on Gaussian processes by making statistical assumptions on the underlying delay operator $\mathbf{\Psi}$, based on the interdependencies across all paths and time steps. In particular, the delays for a set of departure choice pairs (p, i) [1] can be seen as a statistical process, with the dissimilarity between any two such

[1] Here and subsequently, p denotes path and i denotes departure time interval, $i = 1, 2, \ldots, n$.

pairs defined as a vector of aggregated differences in the path departure rates, which we will introduce in detail in subsequent sections. We apply the *Ordinary Kriging* model, which is the most commonly used Kriging model in practice (Fang et al., 2005). For any vector of path departure rates $\boldsymbol{h} \in \mathbb{R}_+^{n\times|\mathcal{P}|}$, we assume that the mapping $\boldsymbol{\Psi}$ can be approximated by a realization of the random process:

$$\tilde{\boldsymbol{\Psi}}_{p,i}(\boldsymbol{h}) \;=\; \mu_{p,i} + \varepsilon_{p,i}(\boldsymbol{h}) \qquad \forall (p,\, i), \quad \forall \boldsymbol{h} \in \mathbb{R}_+^{n\times|\mathcal{P}|}. \tag{4.5}$$

For each pair $(p,\, i)$, $\mu_{p,i}$ is the deterministic mean,[2] $\varepsilon_{p,i}(\cdot)$ is a Gaussian process with zero mean and covariance as follows:

$$\operatorname{cov}(\varepsilon_{p,i}(\boldsymbol{h}^1)\,,\; \varepsilon_{p,i}(\boldsymbol{h}^2)) \;=\; C_{p,i}(\boldsymbol{h}^1,\, \boldsymbol{h}^2;\, \boldsymbol{\theta}^{p,i}) \qquad \forall (p,\, i), \tag{4.6}$$

where $\boldsymbol{\theta}^{p,i}$ is some vector of parameters to be estimated later. We further assume the stationary property. That is,

$$C_{p,i}(\boldsymbol{h}^1,\, \boldsymbol{h}^2;\, \boldsymbol{\theta}^{p,i}) \;=\; \widetilde{C}_{p,i}(\boldsymbol{d}_{p,i}(\boldsymbol{h}^1,\, \boldsymbol{h}^2);\, \boldsymbol{\theta}^{p,i}), \tag{4.7}$$

where $\boldsymbol{d}_{p,i}(\boldsymbol{h}^1,\, \boldsymbol{h}^2)$ is some dissimilarity indicator between $\boldsymbol{h}^1$ and $\boldsymbol{h}^2$.

Remark 4.1. *The stationarity is a common assumption made to characterize the class of Gaussian random fields (Gaussian processes) we are considering. Without it, the statistical inference about the underlying probability law would be almost impossible based on a single realization of the field (Cressie and Wikle, 2011; Fang et al., 2005). Assuming stationarity means that the probabilistic structure of the random process looks similar in different areas of the experimental region (Santner et al., 2013). However, this is not a strong assumption since the Gaussian random field remains sufficiently flexible through anisotropic variance function and unknown parameters.*

It is crucial to determine an appropriate form of the dissimilarity function in order for the statistical model to perform properly. Here, we propose a dissimilarity function that uses different kernels for different paths and time intervals. We first use δ_{pq} defined below to indicate the proximity or similarity between an arbitrary pair of paths p and q:

$$\delta_{pq} \;=\; \frac{\text{\# of shared links between path } p \text{ and path } q}{\text{average \# of shared links betwen } p \text{ and all paths } q' \in \mathcal{P}}\,, \tag{4.8}$$

which gives rise to the proximity matrix $\{\delta_{pq}\}_{p,q\in\mathcal{P}}$.

Remark 4.2. *The distance measure δ_{pq} encapsulate information about the network structure into the Kriging framework. There are a number of other choices for the distance measure, e.g., the portion of overlapping sections, and link/path characteristics such as free-flow travel time and capacity. Further study should compare these different measures in terms of the performance of their respective Kriging predictors.*

Next, we define the dissimilarity (distance) $\boldsymbol{d}_{p,i}(\boldsymbol{h}^1,\, \boldsymbol{h}^2)$ between two path departure rate vectors $\boldsymbol{h}^1$ and $\boldsymbol{h}^2$, pertaining to path p and time interval i:

$$\boldsymbol{d}_{p,i}(\boldsymbol{h}^1,\, \boldsymbol{h}^2) \;=\; \left(\delta_{pq}\left\|w^i \circ (\boldsymbol{h}^1_q - \boldsymbol{h}^2_q)\right\|_2^2 :\; q \in \mathcal{P}\right) \in \mathbb{R}^{|\mathcal{P}|} \qquad \forall p \in \mathcal{P},\; 1 \le i \le n, \tag{4.9}$$

where $\boldsymbol{h}^1_p - \boldsymbol{h}^2_p \doteq (\boldsymbol{h}^1_{p,i} - \boldsymbol{h}^2_{p,i} :\; i = 1, \ldots, n) \in \mathbb{R}^n$, and the operator $\circ$ performs component-wise multiplication. The parameter $w^i \in \mathbb{R}^n$ is defined as

$$w^i \;=\; (w^i_1, \ldots, w^i_n) \quad \text{where } \; w^i_j \;=\; \begin{cases} 1 & \text{if } \; j \le i \\ 0 & \text{if } \; j > i \end{cases} \qquad 1 \le j \le n. \tag{4.10}$$

Remark 4.3. *The similarity indicator δ_{pq} accounts for the topological configuration of any two paths in the network and therefore reflects the potential influences of one another using spatial information. On the other hand, w^i_j accounts for the temporal correlation between the departure rates in two distinct time intervals. The way in which we define*

[2] Ordinary Kriging, which assumes a constant mean, does not imply a flat surface. This assumption often suffices in handling complex situations in practice (Kleijnen, 2009; Chen et al., 2003).

w^i_j implies the assumption that the departure rates at a later time are not considered in defining the dissimilarity at an earlier time. Note that this assumption is not entirely realistic as the departure flow along certain path can indeed affect drivers departing at an earlier time along some other path. But here we choose (4.10) *to balance between sophistication of the model and mathematical simplicity as well as dimensionality. Building on this, the dissimilarity function $\boldsymbol{d}_{p,i}(\cdot,\,\cdot)$ assesses the 'distance' between two feasible path flows comprehensively in both spatial and temporal dimensions.*

Based on the dissimilarity function, we employ an exponential form for the covariance function (4.7):

$$\widetilde{C}_{p,i}\Big(\boldsymbol{d}_{p,i}(\boldsymbol{h}^1,\,\boldsymbol{h}^2);\,\boldsymbol{\theta}^{p,i}\Big)\;=\;\sigma^2_{p,i}\exp\Big(-\boldsymbol{d}^T_{p,i}\cdot\boldsymbol{\theta}^{p,i}\Big), \tag{4.11}$$

where $\boldsymbol{\theta}^{p,i}=(\theta^{p,i}_1,\theta^{p,i}_2,\ldots,\theta^{p,i}_{|\mathcal{P}|})^T\in\mathbb{R}^{|\mathcal{P}|}$, and $\sigma_{p,i}$ is the variance of the response $\tilde{\boldsymbol{\Psi}}_{p,i}$ at the sampled data points.

4.2. Training and predicting procedures

We consider a set of training data $(\boldsymbol{h}^k:\,k=1,\ldots,K)$ and $(\boldsymbol{\Psi}(\boldsymbol{h}^k):\,k=1,\ldots,K)$. We will state how such a set of training data should be generated, namely the design of experiments later in section 4.3. Given these data, we first learn the parameters in the covariance function through *maximum likelihood estimation* (MLE) based on the assumed Gaussian process. Let $\boldsymbol{\theta}^{p,i}$ be the parameters in the covariance function, which is to be learned. The MLE of $\boldsymbol{\theta}^{p,i}$ is given by:

$$\boldsymbol{\theta}^{p,i}\;=\operatorname*{argmin}_x\Big(K\log\sigma^2_{p,i}(x)+\log|\boldsymbol{\Sigma}_{p,i}(x)|\Big), \tag{4.12}$$

where

$$\sigma^2_{p,i}(x)\;=\;\frac{1}{K}\Big(\phi_{p,i}-\hat{\mu}_{p,i}(x)\cdot\mathbf{1}\Big)^T\Big(\boldsymbol{\Sigma}_{p,i}(x)\Big)^{-1}\Big(\phi_{p,i}-\hat{\mu}_{p,i}(x)\cdot\mathbf{1}\Big) \tag{4.13}$$

$$\hat{\mu}_{p,i}(x)\;=\;\frac{\mathbf{1}^T\Big(\boldsymbol{\Sigma}_{p,i}(x)\Big)^{-1}\phi_{p,i}}{\mathbf{1}^T\Big(\boldsymbol{\Sigma}_{p,i}(x)\Big)^{-1}\mathbf{1}} \tag{4.14}$$

$$\boldsymbol{\Sigma}_{p,i}(x)\;=\;\Big\{C_{p,i}(\boldsymbol{h}^j,\,\boldsymbol{h}^k;\,x)\Big\}_{j,k=1,\ldots,K}\in\mathbb{R}^{K\times K},\qquad\phi_{p,i}\;=\;\Big(\boldsymbol{\Psi}_{p,i}(\boldsymbol{h}^1),\ldots,\boldsymbol{\Psi}_{p,i}(\boldsymbol{h}^K)\Big)^T\in\mathbb{R}^K,$$

and $\mathbf{1}$ is the $K\times1$ column vector consisting of one's. The *best linear unbiased predictor*(BLUP) from ordinary Kriging yields the following estimates for a given vector $\boldsymbol{h}^0$ of path departure rates:

$$\boldsymbol{\Psi}_{p,i}(\boldsymbol{h}^0)\;\approx\;\hat{\mu}_{p,i}(\boldsymbol{\theta}^{p,i})+\boldsymbol{c}^T_{p,i}\cdot\Big(\boldsymbol{\Sigma}_{p,i}(\boldsymbol{\theta}^{p,i})\Big)^{-1}\Big(\phi_{p,i}-\hat{\mu}_{p,i}(\boldsymbol{\theta}^{p,i})\cdot\mathbf{1}\Big)\qquad\forall(p,i), \tag{4.15}$$

where $\boldsymbol{c}_{p,i}=\Big(C_{p,i}(\boldsymbol{h}^0,\,\boldsymbol{h}^1;\,\boldsymbol{\theta}^{p,i}),\ldots,C_{p,i}(\boldsymbol{h}^0,\,\boldsymbol{h}^K;\,\boldsymbol{\theta}^{p,i})\Big)^T$, and each $C_{p,i}(\boldsymbol{h}^0,\,\boldsymbol{h}^1;\,\boldsymbol{\theta}^{p,i})$ is given by (4.7)-(4.11).

4.3. Experimental design of training dataset

4.3.1. A space-filling experimental design

In this section we discuss how certain methodology in experimental design should be applied to generate training data for our problem. The goal of experimental design is to uniformly sample the input variables from their respective domains. The uniformity of a design or sampling strategy can be measured by many different criteria including mean square error, discrepancy and so forth. For our specific problem, we apply a space-filling sampling strategy adapted from Latin Hypercube Sampling (LHS) (Tang, 2008; Fang et al., 2005; Santner et al., 2013) The advantage of LHS, compared to simple random sampling or Monte Carlo sampling, is that it has a smaller variance of the sample mean and lower discrepancy, which means better uniformity in the experimental region (Fang et al., 2005; McKay et al., 1979).

Our procedure is to systematically generate the training data on a simplex induced by some flow conservation constraints, which is aligned with the route-choice DUE framework; see (3.4). Specifically, given the O-D departure rate vector $\boldsymbol{R}_{rs}=(\boldsymbol{R}^i_{rs}:\,i=1,\ldots,n)$ for each O-D pair $(r,\,s)\in\mathcal{W}$, we have the following constraints for path-specific departure rates $\boldsymbol{h}_p$:

$$\sum_{p\in\mathcal{P}_{rs}}\boldsymbol{h}_{p,i}\;=\;\boldsymbol{R}^i_{rs}\qquad\forall i=1,\,\ldots,\,n,\quad\forall(r,\,s)\in\mathcal{W}. \tag{4.16}$$

We see that such constraints are decoupled for different time intervals i and different O-D pairs $(r,\ s)$. Taking advantage of such a time-independency, we employ the Latin Hypercube Sampling(LHS) method and map the design onto simplex uniformly, following Fang et al. (2005), Santner et al. (2013) and Fang and Wang (1993). We begin with the following definition (Fang et al., 2005).

Definition 4.4. (Latin Hypercube Design (LHD)) *A LHD design with M runs and N input variables, denoted by LHS*$(M,\ N)$*, is a* $M \times N$ *matrix. In such a matrix, each column is a random permutation of* $\{1,\ 2,\ \cdots,\ M\}$.

The LHS is a well developed design method based on Latin Hypercube Design (LHD) for uniformly sampling a cubic design region. Here, we propose an algorithm based on the LHS (Fang et al., 2005) and a mapping method suggested by Fang and Wang (1993) that converts a unit cube design to a unit simplex design. The following algorithm summarizes the procedure to generate the training data uniformly within the simplex defined based on (4.16).

Algorithm 1: LHS-based generation of training data for OD-pair (r, s) **and time interval** i
For further notation convenience, let $L = |\mathcal{P}_{rs}|$ and $\mathcal{P}_{rs} = \{p^l_{rs} : l = 1, \ldots, L\}$.

(Step 1) For integers $\{1, \ldots, K\}$ where K is the prescribed number of samples, independently generate $L-1$ number of random permutations: $\{\pi_l(1), \ldots, \pi_l(K)\}$ for $l = 1, \ldots, L-1$. These permutations constitute a Latin Hypercube Design $LHD(K,\ L-1)$.

(Step 2) Generate $K \times (L-1)$ i.i.d. uniformly distributed variables $\{U^l_k : k = 1, \ldots, K,\ l = 1, \ldots, L-1\}$ between 0 and 1 and let $\lambda_k = (\lambda_{k1}, \ldots, \lambda_{kL}) \in \mathbb{R}^{L-1}$ where

$$\lambda_{kl} \ = \ \frac{\pi_l(k)}{K} - \frac{U^l_k}{K} \qquad k = 1, \ldots, K,\ \ l = 1, \ldots, L-1.$$

Then $\{\lambda_k, k = 1, \ldots, K\}$ is Latin Hypercube sample of size K from LHS method.

(Step 3) Let $c_k = (c_{k1}, \ldots, c_{kL}) \in \mathbb{R}^L$ and compute c_k for each $k = 1, \ldots, K$ as

$$c_{kl} = (1 - \lambda_{kl}^{\frac{1}{L-l}}) \prod_{j=1}^{l-1} \lambda_{kj}^{\frac{1}{L-j}},\ \ l = 1, \ldots, L-1\,, \qquad c_{kL} = \prod_{j=1}^{L-1} \lambda_{kj}^{\frac{1}{L-j}}.$$

This forms the set $\{c_k, k = 1, \ldots, K\}$ uniformly scattered on unit simplex region.

(Step 4) For each $k = 1, \ldots, K$, the departure rates at time interval i are assigned as:

$$h^k_{p^l_{rs},i} \ = \ \boldsymbol{R}^i_{rs} \cdot c_{kl}.$$

For networks with multiple O-D pairs and time periods, this algorithm takes advantage of the fact that the traffic demands are independent among different O-D pairs and with respect to time. For a target sample size, the training dataset are generated by first carrying out Algorithm 1 independently for all O-D pairs (r, s) and time interval i, and then grouping all the generated O-D specific departure rates into one complete vector of path departure rates.

4.3.2. Adaptive sampling of the experimental domain

The LHS-based algorithm presented above tends to sample the training dataset uniformly from the simplex structure (4.16). However, when the response surface is sufficiently irregular such uniform sampling strategy may not be ideal. To further improve the performance of the training phase we apply an adaptive sampling procedure to iteratively update the training set until a target error rate is achieved. This will result in a better exploration of the experimental domain especially in areas where the response surface has low regularity. The procedure is as follows.

Algorithm 2: Iterative update of the training set

(Step 1) Fix integers $0 < M < K$, $\delta > 0$, and error threshold $\varepsilon > 0$. Generate an initial training set $\mathcal{T}_0$ of size K based on Algorithm 1. Select a target error threshold ε. Set the iteration counter $j = 0$.

(Step 2) Train the model using training set $\mathcal{T}_j$. Generate a uniformly sampled testing dataset $\mathcal{S}_j$ of size M using Algorithm 1.

(Step 3) Test the trained model using the testing dataset $\mathcal{S}_j$. Find the subset $\mathcal{E}_j \subset \mathcal{S}_j$ that contains testing data points with errors above ε.

(Step 4) If $|\mathcal{E}_j| \leq \delta$, stop and output $\mathcal{T}_j$. Otherwise, update training set as $\mathcal{T}_{j+1} = \mathcal{T}_j \cup \mathcal{E}_j$. Set $j = j + 1$ and go to Step 2.

5. Discussion and application

The proposed metamodeling framework is applicable to a wide range of DNL models with a variety of link dynamics, junction models, and path delay models. It underpins a new generation of network performance models that serve as efficient and tractable alternatives to the otherwise *exact* models, and their reliability can be improved with in-depth analysis of the covariance functions, parameters, and methods that generate training data (Li and Sudjianto, 2012). The computational efficiency of the DNL models resulting from the metamodeling approach based on Kriging, which will be demonstrated in our numerical studies in Section 6, could significantly speed up DTA computations for large-scale networks. Moreover, the corresponding delay operators have closed-form expressions, which is potentially beneficial for DTA modeling, and this section will explore these opportunities. Throughout this section, we use $\mathbf{\Phi} : \mathbb{R}^{n\times|\mathcal{P}|} \to \mathbb{R}^{n\times|\mathcal{P}|}$ to denote the surrogate delay operator obtained from Kriging.

5.1. Closed-form expression of the delay operator $\mathbf{\Phi}$

Given a training dataset consisting of K samples, the predictor, expressed in (4.15), can be written in a concise form as:

$$\mathbf{\Phi}_{p,i}(\boldsymbol{h}^0) \;=\; X^{p,i} + c_{p,i}(\boldsymbol{h}^0)^T \cdot Y^{p,i} \tag{5.17}$$

$\forall p \in \mathcal{P},\ i = 1,\dots,n$, where $\boldsymbol{h}^0$ is the input path flow vector. Both $X^{p,i} \in \mathbb{R}$ and $Y^{p,i} = (Y_k^{p,i} : k = 1,\dots,K) \in \mathbb{R}^K$ only depend on the training data and thus can be treated as constant once the training phase is finished. Moreover, the expression for $c_{p,i}(\boldsymbol{h}^0)$ is given by (4.7)-(4.11):

$$c_{p,i}(\boldsymbol{h}^0) \;=\; \sigma_{p,i}^2\Big(\exp\big(-\boldsymbol{d}_{p,i}(\boldsymbol{h}^0,\boldsymbol{h}^1)^T\cdot\theta^{p,i}\big),\ \dots,\ \exp\big(-\boldsymbol{d}_{p,i}(\boldsymbol{h}^0,\boldsymbol{h}^K)^T\cdot\boldsymbol{\theta}^{p,i}\big)\Big) \in \mathbb{R}^K. \tag{5.18}$$

Recall that

$$\boldsymbol{d}_{p,i}(\boldsymbol{h}^0,\boldsymbol{h}^k) \;=\; \Big(\delta_{pq}\big\|w^i\circ(\boldsymbol{h}_q^0-\boldsymbol{h}_q^k)\big\|_2^2 :\ q\in\mathcal{P}\Big)\in\mathbb{R}^{|\mathcal{P}|}\qquad k=1,\dots,K, \tag{5.19}$$

and $w^i \in \mathbb{R}^n$ is such that $w^i = (w_1^i,\dots,w_n^i)$ where $w_j^i = 1$ if $j \leq i$ and $w_j^i = 0$ if $j > i$. Based on (5.17)-(5.19), it is easy to derive the following closed-form expression:

$$\mathbf{\Phi}_{p,i}(\boldsymbol{h}^0) \;=\; X^{p,i} + \sigma_{p,i}^2\sum_{k=1}^{K} Y_k^{p,i}\exp\left(-\sum_{q\in\mathcal{P}}\delta_{pq}\theta_q^{p,i}\sum_{j=1}^{n} w_j^i(\boldsymbol{h}_{q,j}^0-\boldsymbol{h}_{q,j}^k)^2\right) \tag{5.20}$$

$\forall p \in \mathcal{P},\ i = 1,\dots,n$. Here, $\boldsymbol{h}_{q,j}^0$'s are input variables of the delay operator; $X^{p,i}$, $\sigma_{p,i}$, $Y_k^{p,i}$'s, $\theta_q^{p,i}$'s and $\boldsymbol{h}_{q,j}^k$'s are all treated as constants in the prediction of path delays.

5.2. Analytical properties of the surrogate delay operator $\mathbf{\Phi}$

5.2.1. Continuity and smoothness

The delay operator expressed in (5.20) is not only continuous but also smooth (i.e. infinitely differentiable), since it involves only elementary summation, multiplication, and exponential operations. Unlike conventional DNL models for which the discontinuity may fail when spillback occurs (we refer the reader to Szeto (2003) and Han et al. (2016a) for some examples), the continuity and smoothness of the proposed surrogate delay operator $\mathbf{\Phi}$ will always hold regardless of the underlying link, junction, or network flow dynamics. The continuity of $\mathbf{\Phi}$ can be used to prove the existence of solutions to DTA problems when the DNL problem is replaced with the approximate surrogate model we have presented so far.

5.2.2. Differentiability and Lipschitz continuity

We can also easily differentiate the delay operator and obtain its Jacobian matrix as follows. Recall that the surrogate delay operator is a mapping from the $n \times |\mathcal{P}|$-dimensional Euclidean space into itself:

$$\mathbf{\Phi}(\boldsymbol{h}^0) \;=\; \Big(\mathbf{\Phi}_{p,i}(\boldsymbol{h}^0) :\; \forall (p,\, i)\Big).$$

Therefore, individual entries of the Jacobian matrix can be calculated as:

$$\left.\frac{\partial \mathbf{\Phi}_{p,i}}{\partial (q,\, j)}\right|_{\boldsymbol{h}^0} \;=\; -\,2\sigma^2_{p,i}\delta_{pq}\theta^{p,i}_q w^i_j \sum_{k=1}^{K} Y^{p,i}_k \exp\left(-\sum_{\hat{q}\in\mathcal{P}} \delta_{p\hat{q}}\theta^{p,i}_{\hat{q}} \sum_{\hat{j}=1}^{n} w^i_{\hat{j}} (\boldsymbol{h}^0_{\hat{q},\hat{j}} - \boldsymbol{h}^k_{\hat{q},\hat{j}})^2\right)(\boldsymbol{h}^0_{q,j} - \boldsymbol{h}^k_{q,j}). \tag{5.21}$$

Due to the finite-dimensional nature of this problem, it can be easily shown that the partial derivatives (5.21) are uniformly bounded. Therefore the delay operator $\mathbf{\Phi}$ is in fact Lipschitz continuous.

The differentiability and closed-form Jacobian of the delay operator $\mathbf{\Phi}$ will have a number of important applications including, but not limited to, the following.

- All gradient-based methods can directly benefit from the explicit Jacobian of the delay operator. These include single-level problems (e.g. dynamic user equilibrium, dynamic system optimal), and bi-level problems (e.g. dynamic mathematical program with equilibrium constraints) that can be reduced to single-level problems (e.g. via the KKT conditions). All these problems can be directly solved by commercial solvers such as GAMS (e.g. see Friesz and Kwon (2007)).
- The explicit differentiation of the delay operator also facilitates sensitivity analysis of dynamic network traffic equilibria (Chung et al., 2014), and benefits network design heuristics based on it. For example, the heuristic network design based on sensitivity analysis from Suwansirikul et al. (1987) can be adapted to treat dynamic traffic networks.

5.2.3. Generalized monotonicity

Generalized monotonicity, such as monotonicity, strong monotonicity, pseudo monotonicity, quasi monotonicity, and dual solvability, is necessary for the convergence of computational algorithms for DUE problems (see Han et al. (2015) for an overview of these notions). However, conventional DNL models do not allow insights regarding generalized monotonicity for general networks due to the lack of analytical representations of the delay operators. With the closed-form expression of the delay operator $\mathbf{\Phi}$, it is possible to conduct rigorous analysis regarding monotonicity.

We take (strong) monotonicity as an example. An operator $\mathbf{\Phi}$ is said to be (strongly) monotone if

$$\left\langle \mathbf{\Phi}(\boldsymbol{h}^1) - \mathbf{\Phi}(\boldsymbol{h}^2)\,,\; \boldsymbol{h}^1 - \boldsymbol{h}^2 \right\rangle \;\geq\; 0 \quad (\geq\; \lambda\|\boldsymbol{h}^1 - \boldsymbol{h}^2\| \text{ for some } \lambda > 0) \quad \forall \boldsymbol{h}^1,\, \boldsymbol{h}^2 \in \mathbf{\Lambda}. \tag{5.22}$$

Although it is not likely that the operator $\mathbf{\Phi}$ satisfies these conditions in its entire domain, its closed-form expression allows us to identify regions where it is (strongly) monotone. For instance, thanks to the closed-form representation of $\mathbf{\Phi}$ the left hand side of (5.22) can be also expressed in closed form, making it sufficient to solve an algebraic equation in order to identify the (strongly) monotone region of $\mathbf{\Phi}$. In addition, due to the closed-form expression and differentiability, methods based on first-order expansion of $\mathbf{\Phi}$ can be similarly applied to draw insights into its generalized monotonicity.

The closed-form expressions of $\mathbf{\Phi}$ and its Jacobian allow us to inspect, in a quantitative way, a wide spectrum of convergence conditions proposed in the literature (Han and Lo, 2002; Konnov, 2003; Long et al., 2013), and to devise tailored numerical schemes that take advantage of such information, which is unavailable through conventional DNL models. This approach holds much promise in tackling the convergence issue that have stymied researchers for decades, and will be pursued in a future study as it is out of the scope of this paper.

6. Numerical studies

The goal of this section is to numerically evaluate the performance of the surrogate DNL models obtained from the proposed metamodeling approach with Kriging, in terms of approximation accuracy and computational efficiency. For the conventional and exact DNL model used in generating both the training and testing datasets, we employ the discretized Lighthill-Whitham-Richards network model equivalent to the link transmission model (Yperman et al., 2005). The detailed DNL procedure based on this model is presented in Han et al. (2016a). The link transmission model is among the most efficient computational algorithms for propagating flow and congestion on a network level, and captures realistic phenomena such as physical queues and spillback.

For the training phase of Kriging, we adapt the well-known DACE Matlab toolbox (Lophaven et al., 2002) by incorporating the customized distance and correlation functions defined in (4.9) and (4.11). The maximum likelihood estimation (4.12) is performed with an iterative pattern search algorithm. Moreover, the generation of the training data follows Algorithms 1 and 2 presented in Section 4.3.

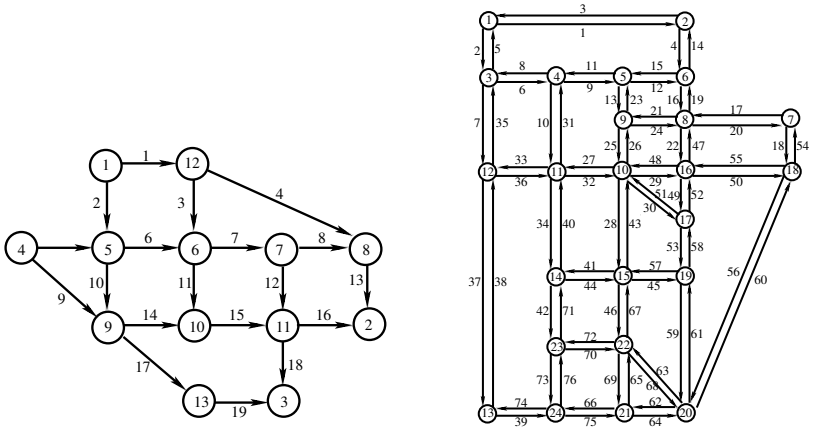
6.1. Test scenarios

We test the performance of the Kriging approach based on five network scenarios:

(1) The 19-arc network with 4 OD pairs and 24 paths, which is studied in Nguyen (1984) and Nie and Zhang (2010) and hereafter referred to as the Nguyen network; and

(2) The Sioux Falls network (Friesz et al., 2011; Suwansirikul et al., 1987) with 60, 119, 201, and 501 paths, respectively.

The Nguyen network and the Sioux Falls network are shown in Figure 2. The time horizon of all DNL models is [0, 5] (in hour). Two values $n = 5$ and $n = 10$ are considered where n is the number of time intervals used in the construction of the finite-dimensional delay operator $\mathbf{\Psi}$ (see Section 3.2).



Nguyen network Sioux Falls network

Figure 2: The test networks

In order to evaluate the accuracy of the delay operators $\mathbf{\Phi}$ obtained from Kriging, we compare their outputs (i.e. predicted path travel times) with those from the exact delay operator $\mathbf{\Psi}$. Specifically, the prediction error corresponding to one test data $\boldsymbol{h}^0$ is calculated as

$$\text{Prediction error} \;=\; \frac{\left\|\mathbf{\Phi}(\boldsymbol{h}^0) - \mathbf{\Psi}(\boldsymbol{h}^0)\right\|_2}{\left\|\mathbf{\Psi}(\boldsymbol{h}^0)\right\|_2} \times 100\%. \tag{6.23}$$

Moreover, the *speed-up index* for one instance of prediction is defined as the ratio between the computational time of the exact delay operator $\mathbf{\Psi}$ and the prediction time of $\mathbf{\Phi}$ (not including training time). The larger the speed-up index, the more computationally efficient our surrogate model is compared to the conventional DNL. For each test scenario, 100 testing data are randomly generated, over which the prediction errors and speed-up indices are averaged.

6.2. Test results for the Kriging-based delay operator

The performance of the Kriging-based delay operators $\mathbf{\Phi}$ is summarized in Table 2 where, for each test scenario, a smaller and a larger dataset are used to train $\mathbf{\Phi}$. It can be seen that the accuracy of the approximate DNL models increases at some expense of decreased efficiency when the training dataset becomes larger. Such a trend is further illustrated in Figure 3 for the Nguyen network and Figure 4 for the Sioux Falls network (with 119 paths), where results based on a wider range of training dataset sizes are available. It is also obvious from Table 2 and Figures 3-4 that the surrogate DNL models yield accurate prediction of the path travel times, with errors typically below 8% for the Nguyen network and, interestingly, below 3.4% for the Sioux Falls networks. The lower errors for larger networks is likely to be caused by the selection of training data; that is, Algorithms 1 & 2 sample the design domain more exhaustively for the Sioux Falls networks than the Nguyen network. One can adaptively reduce the training datasets for the Sioux Falls networks to gain more computational efficiency in the training phase without significantly increasing the errors. This highlights the robustness of the Kriging method with flexible trade-offs between model accuracy and complexity.

In addition, the surrogate DNL models yield significant computational savings compared to the exact DNL models, with speed-ups ranging from 9 to 333 (times faster). This far superior computational efficiency is achieved at only very minor prediction error, showing a very effective trade-off between model accuracy and complexity. In addition, the results for the Sioux Falls network with 501 paths also shows that the proposed metamodelling methodology is capable of handling large-scale and high-dimensional problems.

Table 2: Performances of the surrogate delay operator $\mathbf{\Phi}$. The prediction errors and speed-up indices are averaged over 100 randomly sampled testing data.

$n = 5$	Nguyen network (24 paths)		Sioux Falls (60 paths)		Sioux Falls (119 paths)		Sioux Falls (201 paths)		Sioux Falls (501 paths)	
# of training data	100	200	100	200	100	200	100	300	100	300
Training time	30 s	121 s	270 s	1053 s	30 min	151 min	99 min	873 min	32 h	221 h
Prediction error	6.7%	5.3%	2.5%	1.9%	3.3%	2.7%	1.9%	1.7%	3.2%	2.8%
Speed-up index	181	89	143	77	455	159	21	21	91	31
$n = 10$	Nguyen network (24 paths)		Sioux Falls (60 paths)		Sioux Falls (119 paths)		Sioux Falls (201 paths)		Sioux Falls (501 paths)	
# of training data	120	240	120	240	120	240	150	200	200	300
Training time	91 s	351 s	777 s	3116 s	105 min	386 min	458 min	752 min	449 h	497 h
Prediction error	7.1%	5.6%	2.9%	2.2%	3.3%	2.9%	2.2%	2.2%	3.3%	2.9%
Speed-up index	38	20	42	19	132	66	12	9	13	9

In Figures 5 and 6 we show, for some example paths, the exact and predicted path delays on the Nguyen network and the Sioux Falls network, respectively. It is clearly seen that the surrogate DNL models approximate the exact ones quite well, with errors similar to those shown in Tables 2. The situations on the other paths are similar and are not shown exhaustively here.

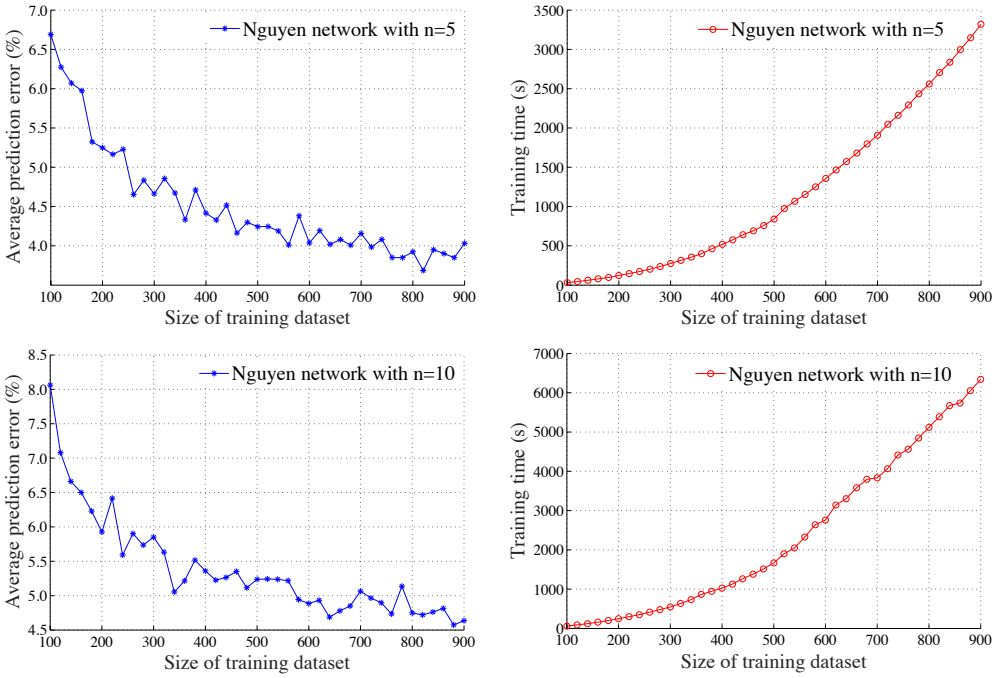


Figure 3: Trends of the prediction errors and training time as the number of training data increases for the Nguyen network.

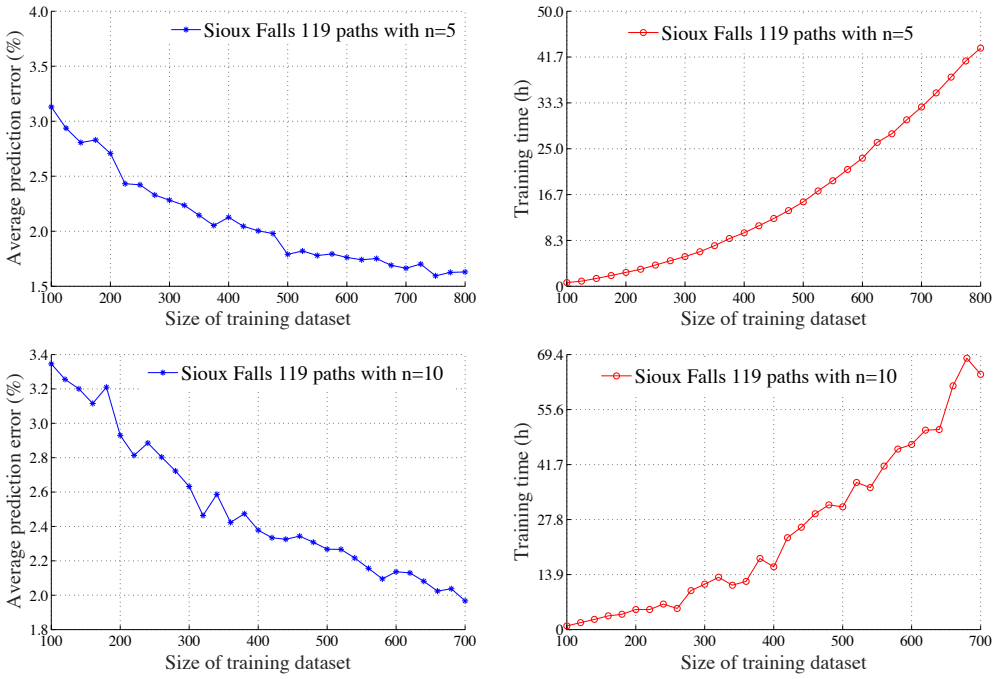


Figure 4: Trends of the prediction errors and training time as the number of training data increases for the Sioux Falls network.

6.3. Comparison with Neural Networks

A comparison of metamodeling performances between Kriging and Neural Networks (NN) is conducted. The test scenario involves the Nguyen network and $n = 5$. Both Kriging and the NN are fitted using the same set of training

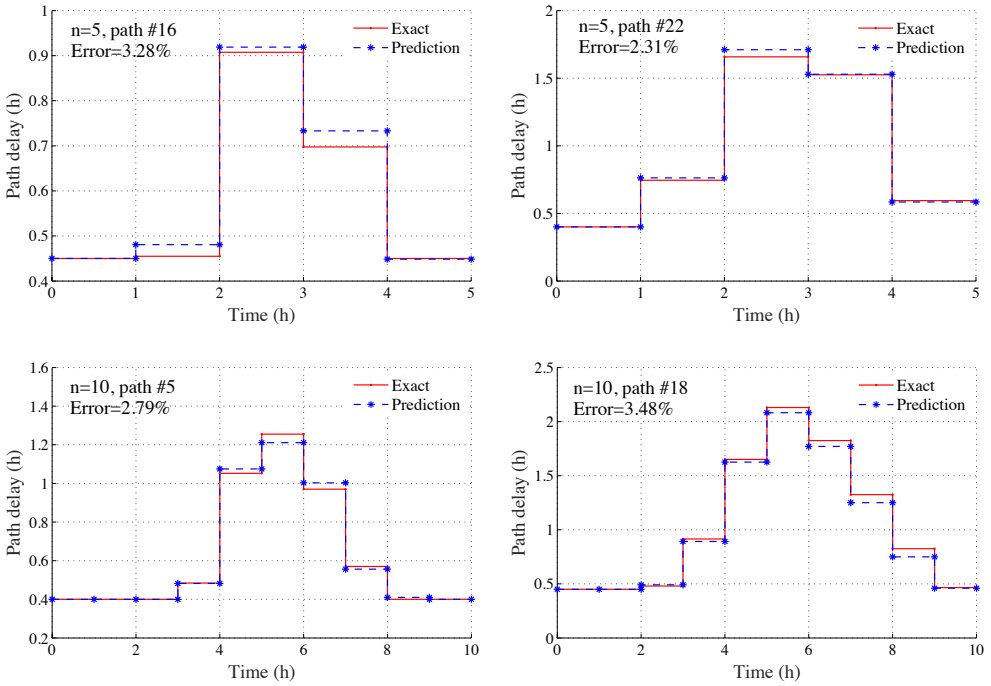


Figure 5: The Nguyen network: Comparison of exact and predicted path delays on example paths.

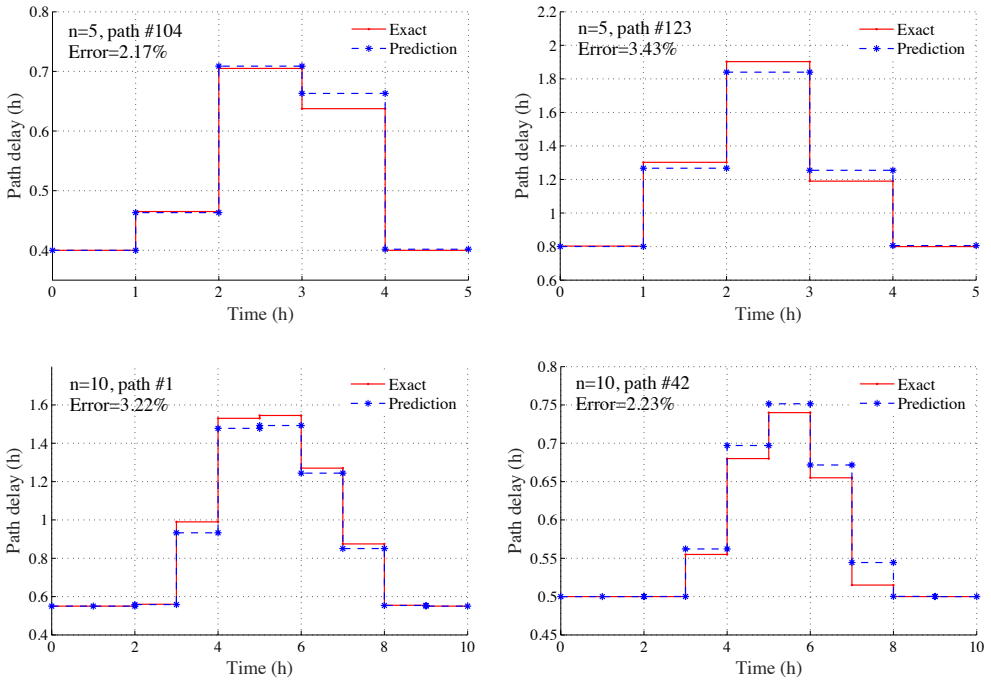


Figure 6: Sioux Falls network with 501 paths: Comparison of exact and predicted path delays on example paths.

datasets, with sizes ranging from 100 to 880. Afterwards, their prediction performances are compared using the same test data consisting of 100 randomly generated samples. The NN has 2 layers and is trained with the Levenberg-Marquardt algorithm. We use Matlab's R2016 Neural Networks toolbox for this task.

Figure 7(left) shows the average prediction error provided by each method on the testing dataset. The bars denote two standard deviations from the prediction means. The overlap of the prediction intervals indicates that differences in prediction error between the two methods are not statistically significant. Figure 7(right) shows the training times to build the surrogate models. Again, no significant different is found between the computational efforts required as a function of the size of the training dataset.

Based on the preliminary comparison on this test case, we can see that for about the same computational training time, Kriging and NN meta-models perform similarly. As mentioned before, Kriging has two main advantages which justify its use: it provides closed form, analytic expressions for the predictions and it is an exact interpolator, which means that predictions at the trained data points are not smoothed but are identical to the observed (i.e., the prediction error for the training dataset is zero). These are not properties shared by Neural Networks.

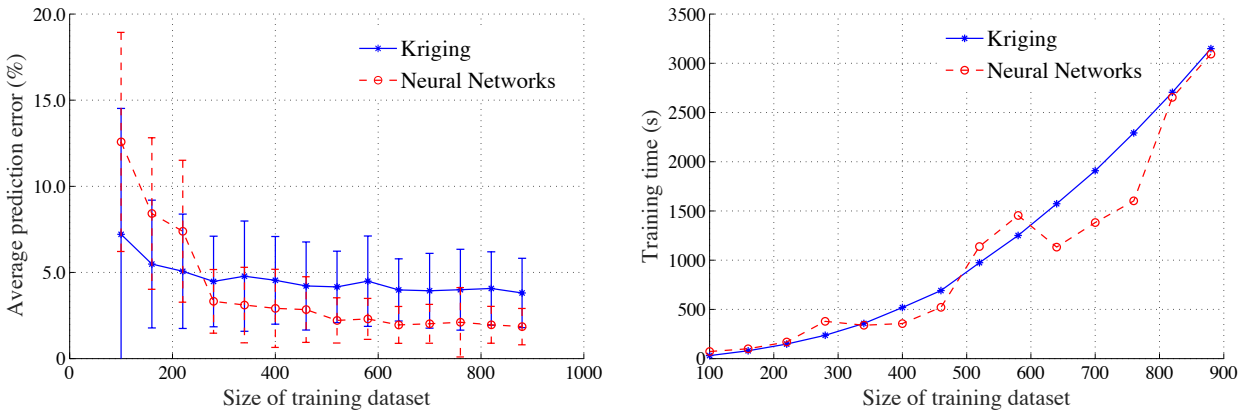


Figure 7: Comparison of Kriging and NN on the Nguyen network. The vertical bars denote two standard deviations from the prediction means.

7. Conclusion and future research

This paper proposes a novel metamodeling approach based on statistical learning for a class of dynamic network loading models. The goal is to provide a class of surrogate DNL models that approximate the exact ones, with considerable benefits including closed-form representation, improved regularity, and superior computational efficiency, at the expense of minor yet controllable approximation errors. In order to achieve this, we employ the Kriging method to interpret and approximate the inherent input-output mechanism of the delay operator. This method considers the observed functional relationship as a realization of a Gaussian Random field, and the Kriging estimate corresponds to the posterior predictive density of the function approximation.

Within the framework of metamodeling, we first articulate the precise definition of delay operators in continuous time, and then define its discrete counterpart in finite-dimensional spaces. Then, we implement the statistical learning methodology by utilizing the network structure and path information to perform Kriging on a non-conventional space with network-specific distance metric. Training of the model parameters is formulated as a maximum likelihood estimation based on the Gaussian process, and yields a closed-form predictor. Following this, a way to systematically generate training data based on Latin Hypercube Design is proposed, and complemented by a second one based on heuristic and adaptive approach.

This metamodeling approach produces, as its output, a family of surrogate DNL models that approximate the exact ones. Compared to the delay operators obtained from conventional DNL procedures, the proposed ones enjoy much improved analytical properties that benefit DTA analysis and computation significantly; these include closed-form expression, (Lipschitz) continuity, differentiability, and closed-form expression for the Jacobians. These advantages are partially discussed in this paper, and further applications are proposed in future studies. On the computational side, it is shown in our numerical study that the surrogate DNL models have far superior computational efficiencies, and are 9 to 333 times faster than conventional ones. Moreover, the approximation errors remain low (less than 8%) throughout our numerical experiment. However, we note that there are many other advanced experimental design

methods that could potentially further improve the performance of our metamodels. These include the penalty method for solving the likelihood maximization problem (Li and Sudjianto, 2012), and dimension reduction techniques based on the clustering of the paths.

It is widely known that the occurrence of vehicle spillback may cause the delay operator to be discontinuous (Han et al., 2016a; Szeto, 2003). Such a fact does not render our metamodel ineffective as Kriging is a highly flexible metamodeling method in that it can fit functions/mappings with different degrees of smoothness (including discontinuities). However, the discontinuity in the delay operator could affect the error estimate of the Kriging method. Further study is needed to distinguish the performance of Kriging by different regularities of the delay operator; i.e. the spillback vs. non-spillback cases.

The proposed statistical metamodeling approach opens the pathway to a family of new network performance models with tractability barely seen in existing ones. Research is underway to explore its impact on dynamic traffic assignment by (1) incorporating the surrogate DNL models into the computation of DUE and DSO problems; (2) devising gradient-based methods for solving bi-level problems on large-scale networks; and (3) testing the possibility of incorporating real-world measurements into the delay operator through statistical learning of similar kind.

8. Acknowledgement

The authors are grateful for the helpful comments and suggestions regarding statistical learning provided by Professor Runze Li from Penn State University.

References

Chan, K.Y., Dillon, T.S., Singh, J., Chang, E., 2012. Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg-Marquardt algorithm. IEEE Transactions on Intelligent Transportation Systems 13(2), 644-654.

Chen, V.C., Tsui, K.L., Barton, R.R., Allen, J.K. 2003. Ch. 7. A review of design and modeling in computer experiments. Handbook of Statistics, 22, 231-261.

Chen, X., Zhang, L., He, X., Xiong, C., Li, Z., 2014. Surrogate-based optimization of expensive-to-evaluate objective for optimal highway toll charges in transportation network. Computer-Aided Civil and Infrastructure Engineering, 29(5), 359-381.

Chen, X., Zhu, Z., He, X., Zhang, L., 2015. Surrogate-based optimization for solving a mixed integer network design problem. Transportation Research Record: Journal of the Transportation Research Board, 2497, 124-134.

Cheng, B., Titterington, D.M., 1994. Neural networks: A review from a statistical perspective. Statistical science, 2-30.

Chung, B.D., Cho, H. J., Friesz, T. L., Huang, H., Yao, T. 2014. Sensitivity analysis of user equilibrium flows revisited. Networks and Spatial Economics, 14(2), 183-207.

Ciuffo, B., Casas, J., Montanino, M., Perarnau, J., Punzo, V., 2013. Gaussian process metamodels for sensitivity analysis of traffic simulation models: Case study of AIMSUN mesoscopic model. Transportation Research Record, 2390(1), 87-98.

Cressie, N. 1993. Statistics for spatial data: Wiley series in probability and statistics. Wiley-Interscience New York, 15, 16.

Cressie, N., Wikle, C.K., 2011. Statistics for spatio-temporal data. John Wiley & Sons.

Currin, C., Mitchell, T., Morris, M., Ylvisaker, D., 1991. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. Journal of the American Statistical Association, 86 (416), 953-963.

Daganzo, C.F., 1995. The cell transmission model, part II: network traffic. Transportation Research Part B 29 (2), 79-93.

Fang, K.T., Li, R., Sudjianto, A., 2005. Design and modeling for computer experiments. CRC Press.

Fang, K.T., Wang, Y., 1993. Number-theoretic methods in statistics (Vol. 51). CRC Press.

Fang, H., Horstemeyer, M.F., 2006. Global response approximation with radial basis functions. Engineering Optimization, 38 (04), 407-424.

Friedman, J.H., 1991. Multivariate adaptive regression splines. The annals of statistics, 1-67.

Friesz, T.L., Bernstein, D., Smith, T., Tobin, R., Wie, B., 1993. A variational inequality formulation of the dynamic network user equilibrium problem. Operations Research 41 (1), 80-91.

Friesz, T.L., Han, K., Neto, P.A., Meimand, A., Yao, T., 2013. Dynamic user equilibrium based on a hydrodynamic model. Transportation Research Part B 47, 102-126.

Friesz, T.L., Kim, T., Kwon, C., Rigdon, M.A., 2011. Approximate dynamic network loading and dual time scale dynamic user equilibrium. Transportation Research Part B 45 (1), 176-207.

Friesz, T.L., Kwon, C., Mookherjee, R., 2007. A computable theory of dynamic congestion pricing. Transportation and traffic theory, 1-26.

Gentile, G., Meschini, L., Papola, N., 2007. Spillback congestion in dynamic traffic assignment: A macroscopic flow model with time-varying bottlenecks. Transportation Research Part B 41, 1114-1138.

Han, D., Lo, H.K., 2002. Two new self-adaptive projection methods for variational inequality problems. Computers and Mathematics with Applications 43, 1529-1537.

Han, K., Friesz, T.L., Szeto, W.Y., Liu, H., 2015. Elastic demand dynamic network user equilibrium: Formulation, existence and computation. Transportation Research Part B 81, 183-209.

Han, K., Friesz, T.L., Yao, T., 2013a. A partial differential equation formulation of Vickrey's bottleneck model, part I: Methodology and theoretical analysis. Transportation Research Part B 49, 55-74.

Han, K., Friesz, T.L., Yao, T., 2013b. A partial differential equation formulation of Vickrey's bottleneck model, part II: Numerical analysis and computation. Transportation Research Part B 49, 75-93.
Han, K., Friesz, T. L., Yao, T., 2013c. Existence of simultaneous route and departure choice dynamic user equilibrium. Transportation Research Part B 53, 17-30.
Han, K., Piccoli, B., Friesz, T.L., 2016a. Continuity of the path delay operator for dynamic network loading with spillback. Transportation Research Part B 92, 211-233.
Han, K., Piccoli, B., Szeto, W.Y., 2016b. Continuous-time link-based kinematic wave model: formulation, solution existence, and well-posedness. Transportmetrica B: Transport Dynamics 4 (3), 187-222.
Hastie, T., Tibshirani, R., Friedman, J., 2009. *The elements of Statistical Learning*, 2nd. ed., NY: Springer.
Haykin, S.O., 2008. Neural Networks and Learning Machines, 3rd ed. NY: Pearson-Prentice Hall.
Huang, H.J., Lam, W.H.K., 2002. Modeling and solving the dynamic user equilibrium route and departure time choice problem in network with queues. Transportation Research Part B 36 (3), 253-273.
Idé, T., Kato, S., 2009. Travel-time prediction using Gaussian process regression: A trajectory-based approach. In Proceedings of the SIAM International Conference on Data Mining, 1183-1194.
James, G., Witten, D., Hastie, T., Tibshirani, R., 2013. *An Introduction to Statistical Learning*, NY: Springer.
Jang, W., Ran, B., Choi, K., 2005. A discrete time dynamic flow model and a formulation and solution method for dynamic route choice. Transportation Research Part B 39 (7), 593-620.
Jones, D.R., Schonlau, M., Welch, W.J., 1998. Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13(4), 455-492.
Kleijnen, J.P.C., 2009. Kriging metamodeling in simulation: A review. European Journal of Operational Research 192 (3), 707-716.
Kleijnen, J.P.C., 2015, Design and Analysis of Simulation Experiments, 2nd ed., NY, Springer.
Koehler, J., Owen, A., 1996, Computer experiments, in S. Ghosh and C.R. Rao (eds.), Handbook of Statistics, 13: Design and Analysis of Experiments, pp. 261-308, Elsevier, Amsterdam.
Konnov, I.V., 2003. Application of the proximal point method to non monotone equilibrium problems. Journal of Optimization Theory and Applications 119, 317-333
Li, R., Sudjianto, A., 2005. Analysis of computer experiments using penalized likelihood in Gaussian Kriging models. Technometrics 47 (2), 111-120.
Lo, H., Szeto, W., 2002. A cell-based variational inequality formulation of the dynamic user optimal assignment problem. Transportation Research Part B 36 (5), 421-443.
Long, J.C., Huang H.J., Gao, Z.Y., Szeto, W.Y., 2013. An intersection-movement-based dynamic user optimal route choice problem. Operations Research 61 (5), 1134-1147.
Lophaven, S.N., Nielsen, H.B., Søndergaard, J., 2002.Aspects of the matlab toolbox DACE. Informatics and Mathematical Modelling, Technical University of Denmark, DTU.
Lophaven, S. N., Nielsen, H. B., Søndergaard, J. 2002.DACE-A Matlab Kriging toolbox, version 2.0.
McKay, M.D., Beckman, R.J., Conover, W.J., 1979. Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. Technometrics, 21 (2), 239-245.
Matheron, G., 1963. Principles of geostatistics. Economic Geology, 58 (8), 1246-1266.
Mounce, R., 2006. Convergence in a continuous dynamic queuing model for traffic networks. Transportation Research Part B 40 (9), 779-791.
Montgomery, D.C., 2008. *Design and analysis of experiments*. John Wiley & Sons.
Mounce, R., Smith, M., 2007. Uniqueness of equilibrium in steady state and dynamic traffic networks. In: Transportation and Traffic Theory, 281-299, Elsevier, Eds: R.E.Allsop, M.G.H.Bell and B.G.Heydecker.
Murphy, K.P., 2012. *Machine Learning, a Probabilistic Perspective*, Cambridge, MA: The MIT Press.
Nguyen, S., 1984. Estimating origindestination matrices from observed flows. In: Florian, M. (Ed.), Transportation Planning Models. Elsevier Science Publishers, Amsterdam, 363-380.
Nie, Y.M., Zhang, H.M., 2010. Solving the dynamic user optimal assignment problem considering queue spillback. Networks and Spatial Economics, 10 (1), 49-71.
Perakis, G., Roels, G., 2006. An analytical model for traffic delays and the dynamic user equilibrium problem. Operations Research 54 (6), 1151-1171.
Regis, R.G., Shoemaker, C.A., 2013. Combining radial basis function surrogates and dynamic coordinate search in high-dimensional expensive black-box optimization. Engineering Optimization, 45(5), 529-555.
Rokneddin, K., Ghosh, J., Duenas-Osorio, L., Padgett, J.E., 2014. Seismic reliability assessment of bridge networks by statistical learning. Safety, Reliability, Risk and Life-Cycle Performance of Structures and Infrastructures, 613-620.
Sacks, J., Ylvisaker, D., 1970. Statistical designs and integral approximation, in R. Pyke (ed.), Proceedings of the Twelfth Biennial Seminar of the Canadian Mathematical Congress, 115-136, Canadian Mathematical Congress, Montreal.
Sacks, J., Welch, W. J., Mitchell, T. J., Wynn, H.P., 1989. Design and analysis of computer experiments. Statistical science, 409-423.
Santner, T.J., Williams, B.J., Notz, W.I., 2013. The design and analysis of computer experiments. Springer Science & Business Media.
Smith, M.J., Wisten, M.B., 1995. A continuous day-to-day traffic assignment model and the existence of a continuous dynamic user equilibrium. Annals of Operations Research 60 (1), 59-79.
Stein, M.L. 2012. Interpolation of spatial data: some theory for kriging. Springer Science & Business Media.
Sun, S., Xu, X., 2011. Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction. Intelligent Transportation Systems, IEEE Transactions on, 12 (2), 466-475.
Suwansirikul, C., Friesz, T.L., Tobin, R.L., 1987. Equilibrium decomposed optimization: a heuristic for the continuous equilibrium network design problem. Transportation science 21(4), 254-263.
Szeto, W.Y., 2003. Dynamic Traffic Assignment: Formulations, properties, and extensions. PhD Thesis, The Hong Kong University of Science and Technology, China.

Szeto, W.Y., Lo, H.K., 2004. A cell-based simultaneous route and departure time choice model with elastic demand. Transportation Research Part B 38, 593-612.
Szeto, W.Y., Lo, H.K., 2006. Dynamic traffic assignment: properties and extensions. Transportmetrica, 2(1), 31-52.
Tang, B., 2008. Latin Hypercube Designs. Encyclopedia of Statistics in Quality and Reliability. II.
Ukkusuri, S., Han, L., Doan, K., 2012. Dynamic user equilibrium with a path based cell transmission model for general traffic networks. Transportation Research Part B 46 (10), 1657-1684.
Wang, G.G., Shan, S., 2007. Review of metamodeling techniques in support of engineering design optimization. Journal of Mechanical design, 129(4), 370-380.
Wang, J., Shi, Q., 2013. Short-term traffic speed forecasting hybrid model based on chaos-wavelet analysis-support vector machine theory. Transportation Research Part C 27, 219-232.
Wardrop, J., 1952. Some theoretical aspects of road traffic research. In ICE Proceedings: Part II, Engineering Divisions 1, 325-362.
Welch, W.J., Buck, R.J., Sacks, J., Wynn, H.P., Mitchell, T.J., Morris, M.D., 1992. Screening, predicting, and computer experiments. Technometrics, 34(1), 15-25.
Xie, Y., Zhao, K., Sun, Y., Chen, D., 2010. Gaussian processes for short-term traffic volume forecasting. Transportation Research Record: Journal of the Transportation Research Board, 2165 (1), 69-78.
Yang, H., Bell, M.G., 2007. Sensitivity analysis of network traffic equilibrium revisited: the corrected approach. In: Heydecker BG (ed) Mathematics in transport. Elsevier, Netherlands, 373-395.
Yang, H., Zhang, X., Meng, Q., 2007. Stackelberg games and multiple equilibrium behaviors on networks. Transportation Research Part B 41 (8), 841-861.
Ye, Q., Szeto, W.Y., Wong, S.C., 2012. Short-term traffic speed forecasting based on data recorded at irregular intervals. IEEE Transactions on Intelligent Transportation Systems 13(4), 1727-1737.
Yperman, I., Logghe, S., Immers, B., 2005. The Link Transmission Model: An Efficient Implementation of the Kinematic Wave Theory in Traffic Networks, Advanced OR and AI Methods in Transportation, Proc. 10th EWGT Meeting and 16th Mini-EURO Conference, Poznan, Poland, 122-127, Publishing House of Poznan University of Technology.
Zhang, L., He, X., Xiong, C., Zhu, Z., 2014. Bayesian stochastic Kriging metamodel for active traffic management of corridors. In IIE Annual Conference. Proceedings (p. 1790). Institute of Industrial Engineers-Publisher.
Zhu, D.L., Marcotte, P., 2000. On the existence of solutions to the dynamic user equilibrium problem. Transportation Science 34 (4), 402-414.