Philippe Brigger

Data preconditioning for data-driven predictive control in urban traffic control

Master's Thesis

Automatic Control Laboratory
ETH Zurich

Supervision

Alessio Rimoldi
Dr. Carlo Cenedese
Dr. Alberto Padoan
Prof. Dr. John Lygeros

7. 10. 2024

Abstract

Traffic congestion in urban areas presents a significant challenge, often without the possibility of expanding road infrastructure. To address this, we employ Data-Enabl**E**d Predictive Control (DeePC) to dynamically adjust the green-red cycles of traffic lights based on current traffic conditions. Given the nonlinear nature of urban traffic networks, we propose modeling them as Linear Parameter-Varying (LPV) systems.

Our research demonstrates how to adapt the data used by DeePC for decision-making, contingent on known switching times of the LPV system. We establish the equivalence of closed-loop behavior between Model-Predictive Control (MPC) and DeePC during specific intervals of the LPV system. Offline data clustering is utilized to group traffic data from various days into behavior groups, which represent the subsystems of our urban traffic network modeled as an LPV system. The most appropriate data cluster is selected online to match current traffic conditions, and the controller's data is updated online as the urban traffic network transitions between different behaviors when we control the urban traffic network with our proposed data-switching DeePC algorithm.

Numerical simulations illustrate that the proposed algorithm enhances the robustness of traffic congestion prevention and improves computational efficiency compared to a DeePC controller with fixed data.

Keywords: Smart mobility, DeePC, Urban traffic control, LPV systems, Data clustering.

Acknowledgment

The completion of this thesis represents a significant milestone in my academic journey, and I would like to thank many individuals for their support, guidance, and encouragement which have been instrumental along the way.

First and foremost, I would like to express my deepest thanks to my thesis supervisors, Carlo, Alberto, and Alessio. Your unwavering support, insightful ideas, and honest feedback have been invaluable. Your guidance made this journey both enjoyable and intellectually rewarding, helping me grow as a researcher and person. Thank you for your great mentorship!

I am also profoundly thankful to my parents, brothers, and girlfriend for their boundless love, encouragement, and emotional support. You have all been my pillars of strength throughout this endeavor. To my parents, Yannick, and Pascal—your positivity and advice kept me going during the more challenging moments. To Maya, thank you for being my constant source of love and comfort, for listening to my frustrations, and lifting my spirits when things did not go as planned.

A heartfelt thank you to everyone involved at the REAL lab. Claudia, your initiative created an environment in which I have been able to significantly improve my communication, writing, and presentation skills. I hope many more students will benefit as I have.

I would also like to extend a special thanks to all the graduate students at IfA for the laughter, camaraderie, and shared experiences in the student room. To all the brilliant researchers at IfA, thank you for generously sharing your knowledge and expertise with me.

Lastly, to the friends I have made during these six years at ETH—you made this time fly by, and I'm excited to see the remarkable things each of you will accomplish in the years ahead.

Contents

Chapter 1

Introduction

Urbanization is an ongoing global phenomenon, with a steady migration of populations from rural to urban areas projected to continue in the coming decades [1]. This transition fuels a marked increase in vehicular traffic, exemplified by a 41% surge in Switzerland over the past 23 years alone [2]. However, as urban road networks remain largely static, this surge exacerbates issues of traffic congestion, resulting in costs worth billions of dollars annually because of decreased economic productivity and higher fuel consumption [3]. Moreover, the consequent increase in greenhouse gas emissions and air pollution poses significant health hazards [4], particularly in urban areas where factors such as idling at traffic lights and prolonged cold starts further increase emissions [5, 6].

Urban traffic control emerges as a critical domain for intervention, leveraging data-driven methodologies. Building upon the groundwork laid in [7], this thesis increases the computational efficiency and tracking robustness of DeePC for urban traffic control. We model the urban traffic network as a linear parameter-varying (LPV) system, cluster data into behavior groups, and select new data online for the controller from the behavior groups when the system behavior changes.

1.1 Related work

Prior research in urban traffic control advocates the adoption of observation-based over prediction-based models [8]. Macroscopic Fundamental Diagrams (MFDs) play a pivotal role in this paradigm, as they relate critical traffic densities for optimal flow on a macroscopic, regional level. These relations are a characteristic of the network and independent of the vehicle demand [9]. Notably, well-defined MFDs with low scatter are present in cities [10], and snake clustering algorithms [11] aid in identifying the regions where these well-defined MFDs exist.

With the knowledge of MFDs, Model Predictive Control (MPC) has emerged as a tool to control the perimeter flow of traffic by regulating the traffic density to remain below the critical value arising from the MFD. The authors in [12] derive a linear model for the MFD dynamics. They show that linear MPC perimeter flow control is feasible and improves traffic flow. Similarly, [13] presents a nonlinear MPC scheme in which a regulation or economic optimization objective may be formulated. The MPC for the regulation optimization problem aims to steer the vehicle accumulation to an equilibrium point, while the MPC for the economic optimization problem tries to minimize the total time spent for all vehicles in the network. MFDs are used to model the plant and the prediction model for an MPC controller in [14], while the authors in [15] provide a robust proportional-integrator perimeter controller with a low computational complexity compared to MPC. Despite

its efficacy, MPC requires accurate models of city network dynamics and reliable predictions of external disturbances [16].

Data-driven approaches can be distinguished as indirect and direct methodologies [17]. Indirect methods use data to identify a parametric system model, and use this model to synthesize a control law. On the other hand, direct approaches are rooted in behavioral systems theory [18], and offer compelling prospects, exemplified by Data-Enabl**E**d Predictive Control (DeePC) [19]. A system is described in terms of past trajectories, the system behavior, instead of with a particular representation [20]. By harnessing persistently exciting data, DeePC furnishes non-parametric system models, enabling real-time control by solving an optimization problem akin to MPC. The algorithm utilizes a finite number of data samples to construct a Hankel matrix, representing the system's behavior [21]. This matrix is then embedded within a receding horizon optimization problem, where optimal control inputs are determined by minimizing a cost function while adhering to system constraints and constraining the future system trajectory to evolve as a linear combination of the Hankel matrix columns based on past system measurements. Some applications of DeePC include empty vehicle rebalancing for Mobility-on-Demand systems [22], optimal cruise control of connected and autonomous vehicles (CAV) in mixed traffic to stabilize traffic flow [23] or dissipate traffic waves [24], and, most recently, the control of dynamic traffic lights to control urban traffic [7].

One assumption in DeePC is that the data comes from a controllable, linear time-invariant (LTI) system, and, hence, any trajectory of the system can be constructed from a finite set of data samples generated from a persistently exciting input signal [19]. A traffic network, however, is nonlinear. Therefore, this assumption does not hold and we do not know how many data samples are required to describe the system behavior. One way to tackle this is with the use of regularization parameters in the cost function [17]. The authors in [7] successfully show that with the regularization parameters urban traffic can be controlled with DeePC for one specific demand. However, there are no guarantees that the controller also would function for different traffic demands.

To address the nonlinear nature of traffic networks and the variability in traffic demand, the concept of LPV systems can be used. In LPV systems, the system dynamics change according to a scheduling signal [25]. If the scheduling signal remains constant over discrete time intervals, then the LPV system can be described as a collection of LTI systems. The authors in [26], [27] present an approach to adapt the data-driven predictive control scheme to the LPV system setting. They do this by directly incorporating the scheduling signal in the Hankel matrix [28]. The idea is to excite the relevant dynamics of the system across all possible operating points during the data collection phase by requiring a persistently exciting scheduling signal on top of a persistently exciting input [29]. Another approach would be to use a large historical data set (for example, traffic data of the past year) to construct the Hankel matrix as this would contain most behaviors that the traffic network can exhibit. Since the optimization problem will grow significantly with both approaches, computational issues are bound to arise in practice [30]. Therefore, it would be computationally advantageous to have a collection of Hankel matrices where each Hankel matrix describes one LTI behavior and the Hankel matrices are switched whenever the LPV system switches LTI subsystems.

Even though traffic demands vary each day, there typically is a morning and an evening peak. Therefore, in our research, we model the traffic network as an LPV system with a morning and an evening subsystem. We propose to collect Hankel matrices from varying traffic demands for each subsystem. We suggest using k-means clustering [[31],[32]] to determine which behavior groups are relevant within the Hankel matrix collection. When running the controller online, we reselect a new Hankel matrix with linear least squares [[33], [34]] based on the current traffic measurements when the traffic network conditions change (e.g. when it transitions from morning to evening peak). We

run the DeePC urban traffic control algorithm from [7] with the selected Hankel matrix to control the system.

1.2 Contributions

The main contributions of our work are summarized in the following six points.

1. DeePC Control Pipelines for LPV Systems with Known and Unknown Switching Times: We explore the application of DeePC to LPV systems, addressing cases with both known and unknown switching times. The pipeline adapts to the system's varying dynamics by leveraging data-driven control principles.
2. Equivalent Closed-Loop Behavior between MPC and DeePC for LPV Systems: A formal proof is provided, establishing that under certain time intervals, the closed-loop behavior of LPV systems controlled by MPC and DeePC is equivalent. The theorem outlines conditions where the two methods yield the same control performance.
3. Feasibility of DeePC with Adaptive Hankel Matrix Switching for Nonlinear Urban Traffic Networks Modeled as LPV Systems: We investigate the feasibility of applying DeePC with adaptive Hankel matrix switching to control nonlinear urban traffic networks, modeled as LPV systems. The study demonstrates that DeePC can effectively manage traffic flows, by switching the Hankel matrix as the system switches dynamics.
4. Identification of Behavior Groups in Traffic Data via k-means Clustering on Hankel Matrices: We identify behavior patterns in traffic data by clustering Hankel matrices with k-means. The resulting clusters represent distinct traffic behaviors, enabling data-driven control strategies to adapt to varying traffic conditions.
5. Selection of Representative Hankel Matrix Using Linear Least Squares: We propose a method for selecting the most representative Hankel matrix based on real-time system measurements. The selection is performed using linear least squares, ensuring that the chosen Hankel matrix accurately reflects the current system state.
6. Increased Efficiency and Robustness in DeePC for Urban Traffic Control via Adaptive Hankel Matrix Switching: To enhance computational efficiency and tracking robustness, we introduce an adaptive approach that switches the Hankel matrix based on current system conditions rather than using a fixed matrix. This adaptive strategy reduces computational load while maintaining control performance in dynamic urban traffic scenarios.

The report is organized as follows. Chapter 2 introduces the basics in behavioral systems theory needed to formulate DeePC. Further, Chapter 2 explains how to describe LPV systems in the behavioral setting and introduces DeePC for urban traffic control. Finally, Chapter 2 explains k-means clustering and linear least squares regression. Chapter 3 details the proposed control pipelines for DeePC in LPV systems with known and unknown switching times, and combines the two approaches to formulate DeePC for urban traffic control with switching Hankel matrices. Chapter 4 presents the results that the proposed DeePC methods for LPV systems and the urban traffic network achieves, while Chapter 5 states the main takeaways of our research and gives ideas for future work.

Chapter 2

Preliminaries

2.1 Notation

We refer to the set of positive integers as $\mathbb{N}$ and the set of non-negative integers as $\mathbb{Z}_+$. The set of real numbers is denoted by $\mathbb{R}$, the set of all non-negative real numbers by $\mathbb{R}_+$. $\mathbb{R}^n$ and $\mathbb{R}^{m\times p}$ denote the set of n-dimensional vectors with real entries and $m \times p$-dimensional matrices with real entries, respectively. The transpose and image of the matrix $A \in \mathbb{R}^{m\times p}$ are denoted by A^T and im A. The pseudo-inverse of A is given by $A^\dagger = (A^T A)^{-1} A^T$. The zero and identity matrix are expressed as $\mathbb{O}$ respectively $\mathbb{I}$. Lastly, a map f from X to Y is denoted by $f : X \to Y$.

2.2 Behavioral representations of dynamical systems

2.2.1 LTI definitions

A dynamical system describes the dynamics of a number of given variables as functions of time, and can be defined as a mathematical model for a phenomenon that evolves over time [35]. The system is described as a triple $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$, where $\mathbb{T} \subseteq \mathbb{R}$ is the time set, $\mathbb{W}$ is the signal space in which the time trajectories take on their values, and $\mathcal{B} \subseteq \mathbb{W}^{\mathbb{T}}$ is the behavior [36]. $\mathbb{W}^{\mathbb{T}}$ consists of all trajectories $\mathbb{W} : \mathbb{T} \to \mathbb{W}$ that can occur according to the model [35]. We assume discrete-time LTI systems where $\mathbb{T} = \mathbb{Z}_+$ and $\mathbb{W} = \mathbb{R}^q$. Linearity and time-invariance of a system are described in terms of the system behavior.

Definition 2.1 (Linearity)**.** A static system $(\mathcal{U}, \mathcal{B})$ is linear if $\mathcal{U}$ is a vector space and $\mathcal{B}$ is a linear subspace. Analogously, a dynamical system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$ is linear when the signal space $\mathbb{W}$ is a vector space and $\mathcal{B}$ is a linear subspace of $\mathbb{W}^{\mathbb{T}}$.

Definition 2.2 (Time-invariance)**.** A system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$ is time-invariant if all the forward shifts $\sigma^t w$, $t > 0$, of a trajectory w of a time-invariant system is also a trajectory of the system. σ is the forward shift operator $(\sigma w)(t) \coloneqq w(t+1)$. Therefore $\mathcal{B} \subseteq \sigma\mathcal{B}$ must hold where $\sigma\mathcal{B} \coloneqq \{\sigma w \,|\, w \in \mathcal{B}\}$.

Definition 2.3 (Restricted behavior)**.** Let $\mathcal{B} \in \mathcal{L}^q$ and $T \in \mathbb{N}$. The restricted behavior is the set $\mathcal{B}|_{[1,T]} = \{w = \text{col}(w_1, \ldots, w_T) \in \mathbb{R}^{qT} \,|\, \exists v \in \mathcal{B} \,:\, w_t = v_t, \forall t \in T\}$.

A vector w belonging to the restricted behavior $\mathcal{B}|_{[1,T]}$ is a T-length trajectory [37]. This allows defining when a system is complete.

Definition 2.4 (Completeness)**.** A system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$ is complete if and only if $w|_{[t_0,t_1]} \in \mathcal{B}|_{[t_0,t_1]}$ for all $t_0, t_1 \in \mathbb{T}, t_0 \leqslant t_1 \Rightarrow w \in \mathcal{B}$.

A system is l-complete if a window with fixed width $t_1 - t_0 = l$ exists, where it is possible to decide if a time series w of length l belongs to a behavior $\mathcal{B}$. This is the case if and only if the behavior is closed in the topology of pointwise convergence [20], i.e., if $w_i \in \mathcal{B}$ for $i \in \mathbb{N}$ and $w_i(t) \to w(t)$, for all $t \in \mathbb{T}$, implies $w \in \mathcal{B}$. This translates to an LTI system having a finite dimensionality [20]. For a finite-dimensional signal space $\mathbb{W} = \mathbb{R}^q$ the class of all complete LTI systems is denoted by $\mathcal{L}^q$.

In the behavioral context a system is controllable if any two trajectories can be patched together in finite time.

Definition 2.5 (Controllability)**.** A system $\mathcal{B}$ is controllable if for any two trajectories $w_1, w_2 \in \mathcal{B}$, there is a third trajectory $w \in \mathcal{B}$, such that $w_1(t) = w(t)$, for all $t < 0$, and $w_2(t) = w(t)$, for all $t \geqslant 0$.

An LTI system $\mathcal{B} \in \mathcal{L}^q$ is characterized by a set of integer variants, which are intrinsic system properties that do not depend on the system representation [36]. A system is namely characterized by its number of inputs $m(\mathcal{B})$, its number of outputs $p(\mathcal{B})$, its order $n(\mathcal{B})$, and ist lag $\ell(\mathcal{B})$. $n(\mathcal{B})$ is also referred to as the number of system states. It holds that the system lag is smaller or equal to the system order $\ell \leqslant n$ [21].

To be able to describe any L-length trajectory of the behavior $\mathcal{B}$, the inputs $\tilde{u}$ in the collected trajectories must fulfill a persistency of excitation of order $L + n(\mathcal{B})$, also known as the Fundamental Lemma [21]. Intuitively, a persistently exciting input $\tilde{u}$ means that all of the relevant system dynamics of the system are excited. Mathematically defined in [21], it means that $\tilde{u}$ is persistently exciting of order $L + n(\mathcal{B})$ if $\tilde{u}$ does not exhibit any non-trivial linear relations of order $L + n(\mathcal{B})$.

State-space representation of LTI systems
Any discrete-time LTI system admits a parametric state-space representation of the form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \\ y(k) &= Cx(k) + Du(k), \end{aligned} \tag{2.1}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$, $y(k) \in \mathbb{R}^p$ are the system state, input, and output at the k'th time step, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$ are (constant) system matrices of the system, and $k \in \mathbb{T} = \mathbb{Z}_+ \subseteq \mathbb{R}$ is the discrete, positive integer time index.

This system representation leads to an alternative formulation of system controllability and leads to the notion of system observability. Informally, an LTI system is controllable if every element of the state vector may be influenced, and hence controlled, by the input vector. Therefore, it is possible to bring the system from any initial state $x(0)$ to any other admissible state $x(k)$ in a finite amount of steps as detailed by [38]. The system matrices A, B can be used to determine the controllability of the system.

Definition 2.6 (Controllability)**.** The LTI system (2.1) is controllable if and only if the controllability matrix $\mathcal{R}_n(A, B) \coloneqq [B, AB, A^2B, ..., A^{n-1}B]$ has full rank n.

For an LTI system to be observable, the initial system state $x(0)$ must be retrievable by observing the system output $y(k)$ for a finite amount of steps k as detailed by [38]. The system matrices A, C can be used to determine the observability of the system.

Definition 2.7 (Observability)**.** The LTI system (2.1) is observable if and only if the observability matrix $\mathcal{O}_n(A, C) \coloneqq [C, CA, CA^2, ..., CA^{n-1}]^T$ has full rank n.

The system lag ℓ is the smallest integer $\ell \in \mathbb{N}$ such that the observability matrix $\mathcal{O}_\ell(A, C) := [C, CA, CA^2, ..., CA^{\ell-1}]^T$ has full rank n.

An equivalent way of representing a behavior $\mathcal{B} \in \mathcal{L}^q$ is with the parametric state-space representation (2.1) denoted by $\mathcal{B}(A, B, C, D) = \{\text{col}(u, y) \in (\mathbb{R}^q)^{\mathbb{N}} \,|\, \exists x \in (\mathbb{R}^n)^{\mathbb{N}}$ such that $x(k+1) = Ax(k) + Bu(k)$, $y(k) = Cx(k) + Du(k)\}$. The minimal representation of the state-space representation is the representation with the smallest order $n(\mathcal{B})$.

The lower Toeplitz matrix $\mathcal{T}_N(A, B, C, D)$ is defined below.

$$\mathcal{T}_N(A, B, C, D) := \begin{pmatrix} D & \mathbb{0} & \cdots & \mathbb{0} \\ CB & D & \mathbb{0} & \vdots \\ \vdots & \ddots & \ddots & \mathbb{0} \\ CA^{N-2}B & \cdots & CB & D \end{pmatrix}$$

Lemma 2.1. *Let $\mathcal{B} \in \mathcal{L}^q$ and $\mathcal{B}(A, B, C, D)$ be a minimal state-space representation. Let $T_{ini}, N \in \mathbb{N}$ with $T_{ini} \geqslant l(\mathcal{B})$ and col$(u_{ini}, u, y_{ini}, y) \in \mathcal{B}|_{[T_{ini}, N]}$. Then there exists a unique $x_{ini} \in \mathbb{R}^{n(\mathcal{B})}$ such that*

$$y = \mathcal{O}_N(A, C)x_{ini} + \mathcal{T}_N(A, B, C, D)u. \tag{2.2}$$

Lemma 2.1 is stated in [19], and implies that the state to which the system is driven by the sequence of inputs u_{ini} is unique if the window of initial system data $\text{col}(u_{ini}, y_{ini})$ is sufficiently long.

2.2.2 LPV definitions

The authors in [25] introduce how to describe dynamical parameter-varying (PV) systems in the behavioral setting. Namely, a dynamical PV system is a 4-tuple $\Sigma = (\mathbb{T}, \mathbb{S}, \mathbb{W}, \mathcal{B})$, where $\mathbb{T} \subseteq \mathbb{R}$ is the time axis, $\mathbb{W}$ is the signal space, $\mathbb{S} \subseteq \mathbb{R}^s$ is the scheduling space, and $\mathcal{B} \subseteq (\mathbb{W} \times \mathbb{S})^{\mathbb{T}}$ is the behavior. Therefore, the behavior describes the space of all signal and scheduling trajectories that are compatible with the system. We assume linear PV (LPV) systems, and hence $\mathbb{T} = \mathbb{Z}_+$, $\mathbb{W} = \mathbb{R}^q$ throughout this work. The set of admissible scheduling trajectories is defined as the *projected scheduling behavior* $\mathcal{B}_\mathbb{S}$ [25]:

$$\mathcal{B}_\mathbb{S} = \pi_s\mathcal{B} = \Big\{ s \in \mathbb{S}^{\mathbb{T}} \mid \exists w \in \mathbb{W}^{\mathbb{T}} \text{ s.t. } (w, s) \in \mathcal{B} \Big\}, \tag{2.3}$$

where π_s is the projection onto $\mathbb{S}^{\mathbb{T}}$. Hence, $\mathcal{B}_\mathbb{S}$ describes all possible scheduling trajectories of $\mathbb{S}$. If the scheduling trajectory $s \in \mathcal{B}_\mathbb{S}$ is fixed, then the *projected behavior* $\mathcal{B}_s$ defines all the signal trajectories $w \in \mathbb{W}^{\mathbb{T}}$ that are compatible with the fixed scheduling trajectory s [25]:

$$\mathcal{B}_s = \Big\{ w \in \mathbb{W}^{\mathbb{T}} \mid (w, s) \in \mathcal{B} \Big\}. \tag{2.4}$$

Furthermore, if the scheduling trajectory $s \in \mathcal{B}_s$ remains constant, $s(t) = \bar{s}$ for all $t \in \mathbb{T}$ where $\bar{s} \in \mathbb{S}$, then the projected behavior $\mathcal{B}_s$ is called a *frozen behavior* $\mathcal{B}_{fz}$ [25]:

$$\mathcal{B}_{fz} = \Big\{ w \in \mathbb{W}^{\mathbb{T}} \mid (w, s) \in \mathcal{B} \text{ with } s(t) = \bar{s}, \forall t \in \mathbb{T} \Big\}. \tag{2.5}$$

A *frozen behavior* is equivalent to the behavior $\mathcal{B}$ of a single, LTI dynamical system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$ since the scheduling signal remains constant and does not alter the system.

Definition 2.8 (Frozen System, [25])**.** Let $\Sigma = (\mathbb{T}, \mathbb{S}, \mathbb{W}, \mathcal{B})$ be a PV system and consider $\mathcal{B}_{fz}$ for a given, constant $\bar{s} \in \mathbb{S}$. The dynamical system

$$\mathcal{F}_{fz} = (\mathbb{T}, \mathbb{W}, \mathcal{B}_{fz}) \tag{2.6}$$

is called a frozen system of Σ.

This allows us to define an LPV system in the behavioral context as follows.

Definition 2.9 (LPV system, [25])**.** The PV system Σ is called LPV, if the following conditions are satisfied:

- Linearity: $\mathbb{W}$ is a vector-space and $\mathcal{B}_s$ is a linear subspace of $\mathbb{W}^{\mathbb{T}}$ for all $s \in \mathcal{B}_{\mathbb{S}}$.
- $\mathbb{T}$ is closed under addition.
- Time-invariance: For any $(w, s) \in \mathcal{B}$ (a signal trajectory associated with a scheduling trajectory) and any $\tau \in \mathbb{T}$, all backward shifts σ^{-1} of a signal trajectory w, defined as $(\sigma^{-1}w)(t) \coloneqq w(t-1)$, and scheduling trajectory s, $(\sigma^{-1}s)(t) \coloneqq s(t-1)$, are part of the system behavior $\mathcal{B}$. Therefore, $\big((\sigma^{-1})^{\tau}w,\, (\sigma^{-1})^{\tau}s\big) \in \mathcal{B}$, and $(\sigma^{-1})^{\tau}\mathcal{B} \subseteq \mathcal{B}$.

Based on this definition, it holds that a frozen system $\mathcal{F}_{fz}$ is an LTI system. Therefore, the projected behaviors of an LPV system Σ with respect to constant scheduling trajectories define a set of LTI systems, called frozen system set.

Definition 2.10 (Frozen system set, [25])**.** Let $\Sigma = (\mathbb{T}, \mathbb{S}, \mathbb{W}, \mathcal{B})$ be an LPV system. The set of LTI systems

$$\mathcal{F}_{\Sigma} = \Big\{ \mathcal{F} = (\mathbb{T}, \mathbb{W}, \mathcal{B}') \,|\, \exists \bar{s} \in \mathbb{S} \text{ s.t. } B' = \mathcal{B}_{fz} \Big\} \tag{2.7}$$

is called the frozen system set of Σ.

Definition 2.11 (Completeness)**.** A system $\Sigma = (\mathbb{T}, \mathbb{S}, \mathbb{W}, \mathcal{B})$ is complete if $(w, s)|_{[t_0,t_1]} \in \mathcal{B}|_{[t_0,t_1]}$ for all $t_0, t_1 \in \mathbb{T}, t_0 \leqslant t_1 \Rightarrow (w, s) \in \mathcal{B}$.

We consider $\mathbb{T} = \mathbb{Z}_+$, $\mathbb{W} = \mathbb{R}^{q_w}$, and a closed scheduling space $\mathbb{S} = \mathbb{R}^{q_s}$ throughout this research. We denote the model class of all complete LPV systems that can be described as a frozen system set $\mathcal{F}_{\Sigma}$ by $\mathcal{L}^q_{LPV}$.

State-space representation of LPV systems

A common approach to model nonlinear systems is to linearize the system around an operating point. If the system state remains near the operating point, then the linearized LTI system model can be used to control the system. To be able to control the system in many operating points, a collection of local LTI system descriptions is necessary.

Linear Parameter-Varying (LPV) systems provide a system theoretical approach to describe a nonlinear system as a collection of local LTI systems, as outlayed in [25]. The central idea is that the nonlinear system is linearized around an operating point. If the system state remains near the operating point, then the linearized LTI system model can be used to control the system. To be able to control the system in many operating points, a collection of local LTI system descriptions is necessary. A state-space representation of an LPV system is given by:

$$\begin{aligned} x(k+1) &= A\big(s(k)\big)x(k) + B\big(s(k)\big)u(k), \\ y(k) &= C\big(s(k)\big)x(k) + D\big(s(k)\big)u(k), \end{aligned} \tag{2.8}$$

where the external variable $s(k) \in \mathbb{R}^s$, the so-called scheduling signal, dictates which linear dynamical relation between the control input $u(k)$ and system output $y(k)$ is valid at the k'th time step. The

dynamical relation is linear, as the system matrices $A(s), B(s), C(s), D(s)$ are constant for a given scheduling signal $s(k)$.

2.2.3 Hankel matrix

A Hankel matrix $\mathcal{H}$ of depth $L \in T$ for a T-length trajectory $w \in \mathbb{R}^{qT}$ is constructed by

$$\mathcal{H}_L(w) = \begin{bmatrix} w(1) & w(2) & w(3) & \dots & w(T-L+1) \\ w(2) & w(3) & w(4) & \dots & w(T-L+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w(L) & w(L+1) & w(L+2) & \dots & w(T) \end{bmatrix}. \tag{2.9}$$

The Hankel matrix in (2.9) is of dimensions $\mathcal{H}_L \in \mathbb{R}^{(qL)\times(T-L+1)}$.

2.2.4 Generalized fundamental lemma

Conditions under which the image of the Hankel matrix $\mathcal{H}_L(w)$ represents the restricted behavior $\mathcal{B}|_{[1,L]}$ are given in [18, Corollary 5].

Corollary 2.1. *If the data-generating system $\mathcal{B}$ is linear time-invariant, $\mathcal{H}_L(w_T) \subseteq \mathcal{B}|_{[1,L]}$, for all $L \in \{l(\mathcal{B})+1, \dots, L_{max}\}$. Moreover, for $L > l(\mathcal{B})$, im $\mathcal{H}_L(w_T) = \mathcal{B}|_{[1,L]}$ if and only if rank $\mathcal{H}_L(w_T) = m(\mathcal{B})L + n(\mathcal{B})$.*

Corollary 2.1 implies that there exists a linear combination of Hankel matrix columns to represent any L-length trajectory of the restricted behavior $\mathcal{B}|_{[1,L]}$ if the Hankel matrix is of rank $m(\mathcal{B})L+n(\mathcal{B})$ which requires $\tilde{u}$ to be persistently exciting of order $m(\mathcal{B})L + n(\mathcal{B})$ during the Hankel matrix data collection. This is formulated in [19, Lemma 4.2].

Lemma 2.2. *Consider a controllable system $\mathcal{B} \in \mathcal{L}^q$. Let $T, L \in \mathbb{N}$, and $w = col(u, y) \in \mathcal{B}|_{[1,T]}$. Assume $\tilde{u}$ to be persistently exciting of order $m(\mathcal{B})L + n(\mathcal{B})$. Then im $\big(\mathcal{H}_L(w)\big) = \mathcal{B}|_{[1,L]}$.*

2.3 DeePC formulation

Given a controllable, observable LTI system with unknown system matrices A, B, C, D, the reference tracking control problem can be tackled with direct data-driven control such as DeePC. The system dynamics, typically given as a state space system representation such as in (2.1), are represented non-parametrically by a Hankel matrix in DeePC, and are embedded in a finite horizon optimization problem. The authors in [19] formulate the DeePC optimization problem:

$$\begin{aligned} \min_{g,\,u,\,y} \quad & \sum_{k=0}^{N-1} \Big(\|y(k) - \hat{y}(t+k)\|_Q^2 + \|u(k) - \hat{u}(t+k)\|_R^2 \Big) \\ \text{s.t.} \quad & \begin{pmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{pmatrix} g = \begin{pmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \\ y \end{pmatrix} \quad , \\ & u(k) \in U, \quad \forall k \in \{0, \dots, N-1\}, \\ & y(k) \in Y, \quad \forall k \in \{0, \dots, N-1\}. \end{aligned} \tag{2.10}$$

The matrix $\big(U_p, Y_p, U_f, Y_f\big)^T$ in (2.10) contains offline collected system trajectories with persistently exciting inputs and gives a non-parametric system representation. The data is partitioned into past p and future f parts. The past section of the Hankel matrix containing U_p, Y_p is used to match the initial conditions u_{ini}, y_{ini} (the initial conditions are the T_{ini} most recent measurements of the system). The future section of the Hankel matrix containing U_f, Y_f is then used to calculate the optimal control inputs that minimize the reference tracking error $y - \hat{y}$ by conditioning the future trajectory y to be a valid continuation of the initial trajectory of the system y_{ini} ((u, y) must lie in the image of the Hankel matrix). The user can define the initial trajectory length by setting T_{ini}, and can do the same for the future trajectory length by choosing the prediction horizon N. The DeePC algorithm is formulated by the authors in [19]:

Algorithm 2.1 DeePC

Given: Data trajectories $w_d = \text{col}(u_d, y_d) \in \mathbb{R}^{(m+p)T}$, most recent input/output measurements $w_{ini} = \text{col}(u_{ini}, y_{ini}) \in \mathbb{R}^{(m+p)T_{ini}}$, reference trajectories $\hat{y} = (\hat{y}_0, \hat{y}_1, \ldots) \in (\mathbb{R}^p)^N$, $\hat{u} = (\hat{u}_0, \hat{u}_1, \ldots) \in (\mathbb{R}^m)^N$, input constraint set $U \subseteq \mathbb{R}^{mT_f}$, output constraint set $Y \subseteq \mathbb{R}^{pT_f}$, output cost matrix $Q \in \mathbb{R}^{p\times p}$, input cost matrix $R \in \mathbb{R}^{m\times m}$.

1: Solve (2.10) for $g^\star$
2: Compute optimal input sequence $u^\star = U_f g^\star$
3: Apply optimal input sequence $(u_t, \ldots, u_{t+j-1}) = (u_1^\star, \ldots, u_j^\star)$ for some $j \leqslant T_f$
4: Set t to $t + j$ and update u_{ini} and y_{ini} to the T_{ini} most recent input/output measurements
5: Return to Step 1

The authors in [19] prove that MPC and DeePC have an equivalent closed loop behavior if the input data u_d of the Hankel matrix is persistently exciting of order $T_{ini} + N + n(\mathcal{B})$, and the total data collection length T fulfills the inequality $T \geqslant \big(m(\mathcal{B}) + 1\big)\big(T_{ini} + N + n(\mathcal{B})\big) - 1$.

2.4 Urban traffic control

A traffic network is a collection of roads and intersections. The traffic density $\rho(t)$ and flow $\phi(t)$ describe the state of the system. The former tells how many cars are on the road ($\rho(t) = \frac{veh}{km}$), while the latter describes how many cars traverse the road in a given time span ($\phi(t) = \frac{veh}{min}$). Incidentally, the average speed of each vehicle is given by the fraction $v(t) = \frac{\rho(t)}{\phi(t)}$. Underground induction loop sensors provide measurements that can be used to estimate the traffic density and traffic flow, and to derive the average speed of the vehicles on the road. On one hand, the arrangement of roads and intersections of the traffic network influence the evolution of the state systems. On the other hand, the exogenous traffic demand $d(t)$ changes the dynamics of the traffic network as well. The traffic demand $d(t)$ says how many vehicles would like to go from one point in the traffic network to another point in the traffic network at time t.

2.4.1 Macroscopic fundamental diagram

Facilitating control on a road-to-road basis becomes infeasible as the city complexity increases. Therefore, the sensor measurements of all roads in different regions of the traffic network are aggregated to a regional network density and flow. Macroscopic fundamental diagrams (MFD) relate the space-mean flow, density, and speed of a traffic network on an aggregated level [9]. The peak of the density-flow MFD plot is the critical regional traffic density at which the traffic flow is maximal in the traffic network. If the traffic density increases further, the traffic flow will decrease (leading to congestion in the traffic network). The idea of urban traffic control is to keep the traffic density below the critical value given by the MFD, as then the traffic flow in the aggregated area will not

collapse. What makes MFDs particularly useful is that they are independent of the exogenous traffic demand when the region being aggregated is homogeneously congested as shown in [8]. This means that the critical traffic density and optimal traffic flow are characteristics of the traffic network (e.g. the road topology) and not the day-to-day varying traffic demand. Field experiments in [9] further suggest that well-defined MFDs exist in large urban areas. Finally, the MFDs will exhibit low scatter if the traffic congestion is evenly distributed across the network and the traffic demand changes gradually as outlayed in [10].

2.4.2 Snake Clustering

The larger a traffic network gets, the less likely it is for it to be homogeneously congested, and hence the concept of MFD is no longer applicable. To resolve this issue, a heterogeneous city can be clustered into connected, homogeneous regions. Each clustered region should exhibit a well-defined MFD again. The authors in [11] propose a three-step 'Snake' clustering algorithm that captures clusters which are spatially connected and homogeneously congested.

1. Create road link 'snakes' where the neighboring road link with the closest density value to the average density of the snake's links is appended.
2. Compute a measure of similarity between each pair of road links in the network.
3. Use graph theory to capture final clusters from the similarity matrix.

2.4.3 DeePC formulation for urban traffic control

We recall the framework presented in [7], where the authors use DeePC to dynamically control traffic lights in an urban traffic setting. To facilitate this, the city traffic network is clustered into p homogeneous regions with snake clustering, and an MFD for each region $i \in \{1, \dots, p\}$ is constructed to identify the critical regional traffic density $\rho_{cr,i}$ at which the regional traffic flow $\phi_i(t)$ peaks. Certain streets in each region contain underground induction loop sensors which measure the amount of vehicles that cross the sensor in a fixed period and the fraction of time that a vehicle spent above the sensor. Each regional density $\rho_i(t)$ and regional flow $\phi_i(t)$ are obtained by aggregating the sensor measurements of the i'th region. Since the traffic flow $\phi_i(t)$ is coupled with the traffic density $\rho_i(t)$ through the MFD, it suffices to only measure the traffic density or traffic flow to deduce the system's current traffic conditions.

The control input, $u(t) = (\lambda(t), d(t))$, consists of the m green-red traffic light ratios $\lambda_i(t) \in [0, 1], i \in \{1, 2, ..., m\}$ of the m dynamic traffic lights (e.g. if a traffic light cycle contains sixty seconds and $\lambda_i(t) = 0.75$, then the i'th traffic light will be green for 45s and red for 15s), and the exogenous demand $d(t)$ which tells how many cars want to go from one street to another street at time t. The exogenous traffic demand $d(t)$ is assumed to be known in the work of [7]. These two control inputs, of which only $\lambda(t)$ can be controlled, dictate the evolution of the traffic network density output $y(t) = \rho(t)$.

$$\rho(k+1) = f(\rho(t), \lambda(t), d(t)) \tag{2.11}$$

To circumvent the necessity to model the traffic dynamics in (2.11), the problem is cast into the DeePC framework (2.10). First, data trajectories $w_d = col(\lambda_d, \rho_d)$ of the system are collected and placed in a Hankel matrix as in (2.9). The data trajectories w_d contain persistently exciting traffic light ratio inputs for a known but not controllable traffic demand scenario, and the measured traffic densities of each region. The reference trajectory $\hat{\rho}$ for the system output is varied according to the current traffic density.

$$\begin{aligned} \hat{\rho}(t) &= 0, \quad && \rho(t) < \rho_{cr} \\ \hat{\rho}(t) &= \rho_{cr}, \quad && \rho(t) \geqslant \rho_{cr} \end{aligned} \tag{2.12}$$

To deal with the nonlinear dynamics of the traffic network, two regularizers on the optimization variable g are introduced into the cost function as proposed in [17].

$$\psi(g) = \alpha_1 \|g\|_1 + \alpha_2 \|(\mathbb{I} - \Pi)g\|_2^2 \tag{2.13}$$

The DeePC optimization problem for urban traffic control reads as follows:

$$\begin{aligned} \min_{g,\lambda,\rho} \quad & \sum_{k=0}^{N-1} \Big(\|\rho(k) - \hat{\rho}(t+k)\|_Q^2 + \|\lambda(k) - \hat{\lambda}(t+k)\|_R^2 + \psi(g) \Big) \\ \text{s.t.} \quad & \begin{pmatrix} \lambda_p \\ \rho_p \\ \lambda_f \\ \rho_f \end{pmatrix} g = \begin{pmatrix} \lambda_{\text{ini}} \\ \rho_{\text{ini}} \\ \lambda \\ \rho \end{pmatrix} \quad , \\ & \lambda(k) \in [0,1]^{N-1}, \\ & \rho(k) \in [0, \rho_{max}]^{N-1}. \end{aligned} \tag{2.14}$$

2.5 k-means clustering

First developed by the authors in [32] and [31], k-means partitions a set of n data points $\mathcal{X} \subset \mathbb{R}^d$ into $k \leqslant n$ clusters. After randomly initializing k cluster centers, the algorithm assigns each set of data points to the closest cluster center and updates the cluster centers based on the given cluster partitions, until the clusters remain the same (see Algorithm 2.2).

Algorithm 2.2 k-means clustering

Given: Set of n data points $\mathcal{X} \subset \mathbb{R}^d$ to be partitioned into k clusters.

1: **procedure** K-Means Clustering
2: **Initialization:** Randomly select k initial cluster centers $\mathcal{C} = \{c_1, ..., c_k\}$ from $\mathcal{X}$
3: **while** cluster centers $\mathcal{C}$ are changing **do**
4: Assign each set of data points in $\mathcal{X}$ to the closest cluster center c_i
5: Recalculate cluster centers c_i based on the cluster assignments $\mathcal{X} \to c_i$ for $i \in \{1, ..., k\}$
6: **end while**
7: **end procedure**

One issue with k-means clustering is that the clustering results strongly depend on the selection of the initial cluster centers. To combat this, the authors in [39] introduced an initialization method that selects the initial cluster centers based on the distance to the already selected initial cluster centers, instead of uniformly at random. They called their modified approach k-means++. Furthermore, the authors in [40] provide lower and upper performance bounds for greedy k-means++. In greedy k-means++, the first initial cluster center is selected at uniform. Then, the next cluster center is iteratively chosen by first choosing l center candidates with a probability related to the distance from the already selected initial centers, and then choosing the center candidate that maximizes the distance between the already chosen cluster centers. Algorithm 2.3 replaces the initialization step in Algorithm 2.2 to give the greedy k-means++ algorithm.

2.6 Linear least squares

Consider the linear equation $Ax = b$, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ are given, and $x \in \mathbb{R}^n$ is the variable to be computed. This equation generally does not have a solution x when $m > n$. The

Algorithm 2.3 Greedy k-means++ initialization

Given: Set of n data points $\mathcal{X} \subset \mathbb{R}^d$ to be partitioned into k initial clusters, l potential initial candidates, cost function $\varphi(x, \mathcal{C}) \coloneqq \min_{c \in \mathcal{C}} \|x - c\|_2^2$, $\Phi(\mathcal{X}, \mathcal{C}) \coloneqq \sum_{x \in \mathcal{X}} \varphi(x, \mathcal{C})$

procedure GREEDY K-MEANS++ INITIALIZATION
  Uniformly independently sample $c_1^1, ..., c_1^l \in \mathcal{X}$
  Let $c_1 = \arg\min_{c \in \{c_1^1, ..., c_1^l\}} \Phi(\mathcal{X}, c)$ and set $\mathcal{C}_1 = \{c_1\}$.
  for $i \leftarrow 1, 2, 3, ..., k-1$ **do**
    Sample $c_{i+1}^1, ..., c_{i+1}^l \in \mathcal{X}$ independently, sampling x with probability $\frac{\varphi(x, \mathcal{C}_i)}{\Phi(\mathcal{X}, \mathcal{C}_i)}$
    Let $c_{i+1} = \arg\min_{c \in \{c_i^1, ..., c_i^l\}} \Phi(\mathcal{X}, \mathcal{C}_i \cup \{c\})$ and set $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{c_{i+1}\}$.
  end forreturn $\mathcal{C} \coloneqq \mathcal{C}_k$
end procedure

linear least squares problem first formulated by the authors [[33], [34]] aims to fit the observed data b to the collected data A by finding an x that minimizes the squared residual:

$$\min_x \quad \|Ax - b\|_2^2 = 0. \tag{2.15}$$

The formulation in (2.15) makes the problem continuously differentiable. A unique minimal 2-norm solution x^* exists [41] by solving (2.15) for x^*:

$$x^* = (A^T A)^{-1} A^T b = A^\dagger b.$$

In other words, the minimal 2-norm least squares residual to the least squares problem is given by:

$$r_{min} = \|AA^\dagger b - b\|_2. \tag{2.16}$$

2.7 MPC formulation

If the system in (2.1) is controllable and observable, and the system matrices A, B, C, D are known, then the reference tracking control problem can be tackled with MPC. MPC calculates the optimal control inputs that minimize a cost function on output and input while respecting the constraints of the system dynamics in a receding-horizon optimization problem as follows:

$$\begin{aligned}
\min_{u,\, x,\, y} \quad & \sum_{k=0}^{N-1} \Big(\|y(k) - \hat{y}(t+k)\|_Q^2 + \|u(k) - \hat{u}(t+k)\|_R^2 \Big) \\
\text{s.t.} \quad & x(k+1) = Ax(k) + Bu(k), \quad \forall k \in \{0, \dots, N-1\}, \\
& y(k) = Cx(k) + Du(k), \quad \forall k \in \{0, \dots, N-1\}, \\
& x(0) = \hat{x}(t), \\
& u(k) \in \mathcal{U}, \quad \forall k \in \{0, \dots, N-1\}, \\
& y(k) \in \mathcal{Y}, \quad \forall k \in \{0, \dots, N-1\}.
\end{aligned} \tag{2.17}$$

In (2.17), the problem is solved for a time horizon N where $y \in \mathbb{R}^p$ is the optimization variable of the future output, $u \in \mathbb{R}^m$ is the optimization variable of the future input, and $\hat{y}_{t+k} \in \mathbb{R}^p$, $\hat{u}_{t+k} \in \mathbb{R}^m$ are the reference output and input to track. $Q \in \mathbb{R}^{p \times p}$ and $R \in \mathbb{R}^{m \times m}$ are semi-positive definite matrices on output and input in the cost function. The constraints on the input and output are given by the sets $\mathcal{U}$ and $\mathcal{Y}$. The state optimization variable $x \in \mathbb{R}^n$ is constrained to adhere to the system dynamics and initially start at the current state of the system $\hat{x(t)} \in \mathbb{R}^n$.

Chapter 3

Methodology

The DeePC approach detailed in Section 2.3 is designed for LTI systems. DeePC can deal with nonlinear systems to some extent by using additional regularizers in the cost function as shown in Section 2.4.3, where DeePC controls nonlinear traffic networks. However, there are no guarantees that the DeePC controller will perform well when the traffic demand is strongly different to the demand that was present during the data collection phase. Although different every day, the traffic demand has some structure (e.g. a peak in the morning and evening). We aim to robustify the DeePC approach in Section 2.4.3 and to increase the computational efficiency by making use of the structure that the traffic demand exhibits. First, we split the day into behavior times, mimicking an LPV system. For each behavior time, we collect many instances of data for different traffic demands and use k-means clustering to identify different behavior groups that exist in each behavior time. Next, when we run the DeePC controller, we switch the Hankel matrix whenever the system conditions change. Linear least squares tells us which Hankel matrix to select under the new system conditions. As a result, the DeePC urban traffic control with switching Hankel matrices promises to be more robust to changing traffic demands and computationally more efficient than using a fixed Hankel matrix for the entire LPV system. In the following, we introduce how we formulate DeePC for LPV systems when the switching times of the system are known and unknown, and show how to extend these pipelines to nonlinear urban traffic networks. Throughout this research, whenever we speak of LPV systems, we assume that they can be defined as a frozen system set $\mathcal{F}_\Sigma$ as defined in Definition 2.10. As a result, we assume that our LPV systems consist of sets of LTI subsystems. Furthermore, we assume that our scheduling signal takes discrete values $s \in \{1, \dots, S\}$, therefore setting the active LTI subsystem $A(s), B(s), C(s), D(s)$ among the S LTI subsystems. Finally, the LPV system switches at discrete points in time k^i_{switch} as illustrated in Figure 3.1.

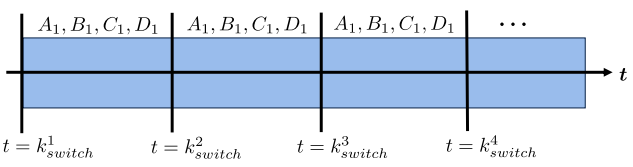


Figure 3.1: LPV system as a set of LTI subsystems A_i, B_i, C_i, D_i. At discrete points $t = k^i_{switch}$ the LTI subsystem matrices switch and remain constant until $t = k^{i+1}_{switch}$.

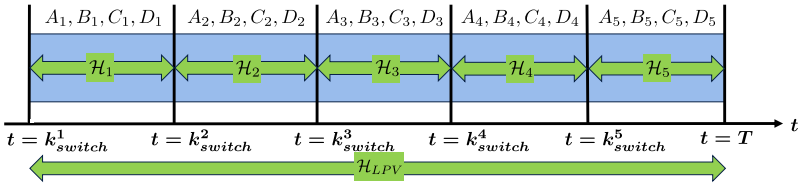


Figure 3.2: LPV system consisting of five LTI subsystems A_i, B_i, C_i, D_i. The concatenated Hankel matrix $\mathcal{H}_{LPV}$ represents the entire LPV system. The subsystem-specific Hankel matrix $\mathcal{H}_i$ represents the i'th LTI subsystem. LTI subsystems switch at discrete points $k_{switch} = \{k^1_{switch}, \dots, k^5_{switch}\}$.

3.1 DeePC for LPV systems

To control an LPV system with the assumptions we set, two approaches are possible as shown in Figure 3.2. One approach would be to construct a Hankel matrix $\mathcal{H}_{LPV}$ of the entire LPV system by collecting data from the entire LPV system. Another approach would be to construct subsystem-specific Hankel matrices $\mathcal{H}_i$ where only data from one LTI subsystem A_i, B_i, C_i, D_i is collected to construct the Hankel matrix $\mathcal{H}_i$, and to switch Hankel matrices when the LPV system switches subsystems at $t = k^{i+1}_{switch}$. The following example motivates why we choose the latter approach, that is, using subsystem-specific Hankel matrices $\mathcal{H}_i$ to control each LTI subsystem of the LPV system, and switching Hankel matrices when the LPV system switches LTI subsystems. The example shows that using subsystem-specific Hankel matrices is more powerful than using concatenated Hankel matrices, where the definition of a model and powerful are taken from [42].

Definition 3.1 (Model)**.** A model M for a phenomenon is a law which says that the phenomenon will only produce outcomes in M. The model M is a subset of the elements of the set S of the attributes of a phenomenon, $M \subset S$.

Definition 3.2 (Model Power)**.** The model M_1 of a given phenomenon is more powerful than the model M_2 if $M_1 \subset M_2$. M_1 allows fewer possibilities than M_2, and hence has more predictive power. The most powerful model explains a given set of observations and as little else as possible. To conclude, M_1 is the better model for a given phenomenon than M_2 because it predicts and explains less than M_2.

3.1.1 Example - Subsystem-specific behavior representation

Goal: Find $\mathcal{B}_1$ and $\mathcal{B}_2$ such that $\exists \bar{w}_1 \in \mathcal{B}_1$ where $\bar{w}_1 \notin \mathcal{B}_2$. Show that a concatenated system representation is less powerful than a specific system representation. As per Definition 3.2, the concatenated representation must be able to predict trajectories that cannot be produced by the current system behavior.

Steps: Let us assume we have two LTI systems as defined in (2.1), each with $m = 2$ inputs, $p = 2$ outputs, and $n \in \mathbb{N}$ states. Further, let us assume that each system remains static ($A_i, B_i, C_i = \mathbb{O}, i \in \{1, 2\}$). Hence, the state is not affected by the input and does not affect the output. This gives the two state-space system representations $y = D_1 u$, and $y = D_2 u$. The behaviors of the two static systems are mathematically expressed as:

$$\mathcal{B}_i = \{(u, y) \in \mathbb{R}^{m+p} \,|\, y = D_i u\}\,,\ i \in \{1, 2\}.$$

Let us define the L-depth matrix $\mathcal{M}_L$ as:

$$\mathcal{M}_L = \left[\begin{array}{c|c} \mathbb{O} & \mathbb{I}_{mL} \\ \hline \mathcal{O}_L & \mathcal{T}_L \end{array}\right] = \left[\begin{array}{c|cccc} \mathbb{O} & \multicolumn{4}{c}{\mathbb{I}_{mL}} \\ \hline C & D & \mathbb{O} & \cdots & \mathbb{O} \\ CA & CB & D & \mathbb{O} & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbb{O} \\ CA^{L-1} & CA^{L-2}B & \cdots & CB & D \end{array}\right].$$

If we set $L = 1$, and knowing that our systems are static, we are left with the vertical concatenation of the $m \times m$-dimensional identity matrix $\mathbb{I}_m$ and the system matrix D, $\mathcal{M}_1 = [\mathbb{I}_m, D]^T$. This leaves us with the matrices $\mathcal{M}_1^1 = [\mathbb{I}_m, D_1]^T$ respectively $\mathcal{M}_1^2 = [\mathbb{I}_m, D_2]^T$ for the two static systems. We can use $\mathcal{M}_1^i$ to describe the restricted behavior $\mathcal{B}_i|_{[1,1]}$ of the i'th static system with the image of $\mathcal{M}_1^i$ as $\mathcal{B}_i|_{[1,1]} = \text{im } \mathcal{M}_1^i$. In other words, im $\mathcal{M}_1^i$ gives a non-parametric representation of the i'th behavior B_i, when trajectories of length 1 are considered. Recall that we would like to show that $\exists \bar{w}_1 \in \mathcal{B}_1$ where $\bar{w}_1 \notin \mathcal{B}_2$. For this, let us choose D_1, D_2, and express the restricted behaviors in terms of the chosen system matrices.

- $D_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \Rightarrow \mathcal{B}_1|_{[1,1]} = \text{im } \mathcal{M}_1^1 = \text{im } \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$

- $D_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \Rightarrow \mathcal{B}_2|_{[1,1]} = \text{im } \mathcal{M}_1^2 = \text{im } \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$

If we compare the column vectors of im $\mathcal{M}_1^1$ and im $\mathcal{M}_1^2$, we can see that they are linearly independent. E.g. the first column vector $v = [1, 0, 0, 1]^T$ of im $\mathcal{M}_1^2$ cannot be expressed as a linear combination of the column vectors of im $\mathcal{M}_1^1$. Therefore, $\nexists g = [g_1, g_2]^T$ such that im $\mathcal{M}_1^1 g \stackrel{!}{=} v$. The same holds for the second column vector of im $\mathcal{M}_1^2$. This means that there exist trajectories belonging to $\mathcal{B}_1$ that do not also belong to $\mathcal{B}_2$, and vice versa. Hence, $\exists \bar{w}_1 \in \mathcal{B}_1$ where $\bar{w}_1 \notin \mathcal{B}_2$.

We can now consider representing the restricted behavior of a system by concatenating $\mathcal{M}_1^1$ and $\mathcal{M}_1^2$, giving $\mathcal{B}_i|_{[1,1]} = \text{im } [\mathcal{M}_1^1, \mathcal{M}_1^2]$. We see that the rank of the concatenated matrix is 3, while the rank of just one specific im $\mathcal{M}_1^i$ is 2. Since the concatenated matrix has a higher rank, it means that $\mathcal{B}_i|_{[1,1]} = \text{im } [\mathcal{M}_1^1, \mathcal{M}_1^2]$ is capable of predicting trajectories that cannot be predicted by $\mathcal{B}_i|_{[1,1]} = \text{im } \mathcal{M}_1^i$. In other words, $\mathcal{B}_i|_{[1,1]} = \text{im } [\mathcal{M}_1^1, \mathcal{M}_1^2]$ is capable of predicting trajectories that actually cannot be produced by $\mathcal{B}_i$. This means that the concatenated representation $\mathcal{B}_i|_{[1,1]} = \text{im } [\mathcal{M}_1^1, \mathcal{M}_1^2]$ is less powerful than the specific representation of $\mathcal{B}_i|_{[1,1]} = \text{im } \mathcal{M}_1^i$. If we increase L, we would arrive at the same result, because the column vectors of $\mathcal{M}_1^i, i \in \{1, 2\}$ would remain linearly independent.

Summary: The finite horizon representation $\mathcal{B}_i|_{[1,L]} = \text{im } [\mathcal{M}_L^1, \mathcal{M}_L^2]$ is less powerful than considering $\mathcal{B}_i|_{[1,L]} = \text{im } \mathcal{M}_L^i$ in each mode $i \in \{1, 2\}$. It is better to control a system with a system-specific behavior representation $\mathcal{B}_i|_{[1,L]} = \text{image } \mathcal{M}_L^i$ than a concatenated representation $\mathcal{B}_i|_{[1,L]} = \text{im } [\mathcal{M}_L^1, \mathcal{M}_L^2]$.

3.1.2 Problem setup

The example above illustrates why it is better to control a frozen system set LPV system with LTI subsystem-specific Hankel matrices $\mathcal{H}_i$. This is further motivated by the generalized fundamental lemma given in Corollary 2.1, as the rank of the non-parametric system representation should be $Lm(\mathcal{B}) + n(\mathcal{B})$, which is 2 in the example, and not 3 as for the concatenated representation. Therefore, we propose to:

- Collect data from each LTI subsystem and construct a representative Hankel matrix of this subsystem.
- Switch among Hankel matrices when the operating mode of the LPV system has switched.

Let us assume that the LPV switches at discrete points $t = k^i_{switch}$ as shown in Figure 3.1, and that the time in between two switches k^i_{switch}, k^{i+1}_{switch} is long enough for the system to stabilize and the transient to settle down. Let us also assume that each LTI subsystem A_i, B_i, C_i, D_i is controllable and observable.

These assumptions make it possible to collect a Hankel matrix for each LTI subsystem A_i, B_i, C_i, D_i and to control each subsystem with this subsystem-specific Hankel matrix. When a switch happens, it remains unclear which LTI subsystem the LPV system is switching to. Therefore, a Hankel matrix selection based on the current measured data is required after a switch has taken place to determine which Hankel matrix to use to control the new LTI subsystem.

3.1.3 Mode selection criterion

Given a collection of $N \in \mathbb{N}$ Hankel matrices $\mathcal{H}_{lib} = \{\mathcal{H}_1, \mathcal{H}_2 \ldots, \mathcal{H}_N\}$, and initial system measurements $w_{ini} = (u_{ini}, y_{ini}) \in \mathbb{R}^{(m+p)T_{ini}}$, linear least squares can be used to find the Hankel matrix which best represents the initial data w_{ini} by comparing the least squares residuals of all candidate Hankel matrices $\mathcal{H}_i$, $i \in \{1, \ldots, N\}$. Since only the initial data is measured, only the past data section of the Hankel matrix $\hat{\mathcal{H}}_i = \mathcal{H}_i(U_p, Y_p)^T$ is used to fit to the measurements w_{ini}. The Hankel matrix selection chooses the Hankel matrix with the minimal least squares residual $r\big(\mathcal{H}_i(U_p, Y_p)^T, w_{ini}\big)$ as written below.

$$\begin{aligned} \mathcal{H} &= \underset{\substack{\mathcal{H}_i \in \mathcal{H}_{lib} \\ i \in \{1,\ldots,N\}}}{\arg\min}\ r\Big(\mathcal{H}_i(U_p, Y_p)^T, w_{ini}\Big) \\ &= \underset{\substack{\mathcal{H}_i \in \mathcal{H}_{lib} \\ i \in \{1,\ldots,N\}}}{\arg\min}\ \big\|\hat{\mathcal{H}}_i \hat{\mathcal{H}}_i^{\dagger} w_{ini} - w_{ini}\big\|_2 \end{aligned} \tag{3.1}$$

To guarantee that we select the correct Hankel matrix, we need a w_{ini} vector which uniquely belongs to one LTI subsystem. Since we assume enough time in between two switches $k^i_{switch}, k^{i+1}_{switch}$, there are time instances for each LTI subsystem where we can measure a w_{ini} vector which uniquely belongs to its corresponding frozen behavior $\mathcal{B}_{fz}$. Knowing when this is the case depends on whether the switching times $t = k^i_{switch}$ of the LPV system are known or unknown. In the following, both settings are discussed. The resulting control pipelines are then used as a foundation to control the nonlinear urban traffic network.

3.1.4 LPV systems control for known switching times

If the switching times $t = k^i_{switch}$ are known, then a behavior-specific Hankel matrix $\mathcal{H}_i$ can be created for each subsystem. We can do this by collecting data with persistently exciting inputs from each subsystem as illustrated by the green arrows in Figure 3.3. The collection of $N \in \mathbb{N}$ Hankel

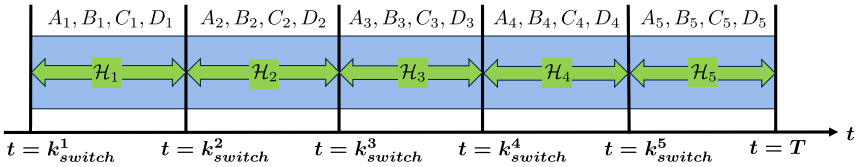


Figure 3.3: The switching times $k_{switch} = \{k^1_{switch}, \dots, k^5_{switch}\}$ are known. Behavior-specific Hankel matrices $\mathcal{H}_i$ can be created by collecting data from each LTI subsystem A_i, B_i, C_i, D_i.

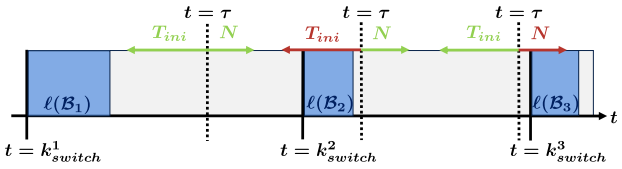


Figure 3.4: At any given time $t = \tau$, T_{ini} time steps before and N time steps after $t = \tau$ must all lie within one LTI subsystem. The green arrows respect this condition, the red arrows do not. T_{ini} must be greater than the lags $\ell(\mathcal{B}_i)$, shaded in blue, of all the frozen behaviors $\mathcal{B}_i$ shaded in blue.

matrices $\mathcal{H}_i, i \in \{1, \dots, N\}$ are packed into a Hankel library $\mathcal{H}_{lib} = \{\mathcal{H}_1, \dots, \mathcal{H}_N\}$. Whenever the system switches subsystems, a new Hankel matrix is chosen that fits the last T_{ini} measurements $w_{ini} = (u_{ini}, y_{ini})$ best in the least squares sense with (3.1). During the switching phase, w_{ini} consists of data from the previous and the current behavior. Therefore, the Hankel matrix selection might select the 'wrong' Hankel matrix during the transient switching phase. The latest after $k^i_{switch} + T_{ini}$ time steps, the Hankel matrix selection will select the correct Hankel matrix as w_{ini} only contains data coming from the new behavior at this point. To verify that the Hankel matrix being used for control actually represents the current behavior, we compare the predicted output of the controller y_{pred} with the measured output y_{meas} of the system. If there is a large error, then we are not controlling with the appropriate Hankel matrix, and must restart the selection process. The control pipeline for a known switching time is given as pseudocode in Algorithm 3.1. Furthermore, since we know the switching times $t = k^i_{switch}$, and we can identify which Hankel matrix to use to control the LTI subsystem with (3.1) (motivated by the example that $\exists \bar{w}_1 \in \mathcal{B}_1$ where $\bar{w}_1 \notin \mathcal{B}_2$, and, hence, the correct subsystem-specific Hankel matrix is retrievable with w_{ini}), [19, Theorem 5.1, Corollary 5.1] can be extended to LPV systems with the following assumptions.

(A1) We know the sequence of switching times $\{k^1_{switch}, k^2_{switch}, ..., k^S_{switch}\}$ for $S \in \mathbb{N}$.

(A2) Every frozen behavior $\mathcal{B}_{fz}$ of the LPV system is minimal. In other words, each LTI subsystem as defined in (2.1) is controllable and observable.

(A3) The integer $T_{ini} \in \mathbb{N}$ is greater than the maximum lag ℓ of all the frozen behaviors: $T_{ini} > \max_{\mathcal{B}_{fz} \in \mathcal{F}_\Sigma} \ell(\mathcal{B}_{fz})$.

(A4) While the i'th LTI subsystem is active at time step t, all time steps in the interval $[t - T_{ini}, t + N]$ must belong to the i'th LTI subsystem. This requires $T_{ini} < t - k^i_{switch}$ and $N < k^{i+1}_{switch} - t$ and is illustrated by the green arrows in Figure 3.4.

(A5) The data $\mathrm{col}(u^d, y^d) \in \mathcal{B}_{fz}|_{[1,T]}$ in $\mathrm{col}(U_p, Y_p, U_f, Y_f)$ of every frozen behavior $\mathcal{B}_{fz} \in \mathcal{F}_\Sigma$ of the LPV system is such that u^d of each frozen behavior $\mathcal{B}_{fz}$ is persistently exciting of order $T_{ini} + N + n(\mathcal{B}_{fz})$.

(A6) The length $T \in \mathbb{N}$ of the collected trajectories of the data $\mathrm{col}(u^d, y^d) \in \mathcal{B}_{fz}|_{[1,T]}$ fulfills $T \geqslant \big(m(\mathcal{B}_{fz}) + 1\big)\big(T_{ini} + N + n(\mathcal{B}_{fz})\big) - 1$ for each frozen behavior $\mathcal{B}_{fz} \in \mathcal{F}_\Sigma$.

(A7) The cost matrix on the output is positive-semidefinite, $Q \succeq 0$. The cost matrix on the input is positive-definite $R \succ 0$.

(A8) The constraint sets for the input and output, $\mathcal{U}$ and $\mathcal{Y}$, are convex and non-empty.

Theorem 3.1 (Equivalent Closed Loop Behavior)**.** *Consider system $\mathcal{B} \in \mathcal{L}^q_{LPV}$. Further, consider the solution of the MPC optimization problem (2.17) $z^\star_{MPC} = (u^\star_{MPC}, y^\star_{MPC}, x^\star_{MPC}) =$ argmin(2.17), and the solution of the DeePC problem (2.10) $z^\star_{DeePC} = (u^\star_{DeePC}, y^\star_{DeePC}, g^\star_{DeePC}) =$ argmin(2.10). Suppose (A1)-(A8) hold. Then, if $z^\star_{MPC} \in$ argmin(2.17) and $z^\star_{DeePC} \in$ argmin(2.10), $(u^\star_{MPC}, y^\star_{MPC}) = (u^\star_{DeePC}, y^\star_{DeePC})$ for all $t \in [k^i_{switch} + T_{ini}, k^{i+1}_{switch} - N]$. Hence, under the time interval $t \in [k^i_{switch} + T_{ini}, k^{i+1}_{switch} - N]$, the closed loop optimal control sequence $u^\star$ and output $y^\star$ of the MPC and DeePC optimization problem are identical.*

Proof. By assumption (A5), Lemma 2.2 gives that $\text{image}\big(\text{col}(U_p, Y_p, U_f, Y_f)\big) = \mathcal{B}_{fz}|_{[1,T_{ini}+N]}$ for each frozen behavior. Hence, the feasible set of (2.10) for each frozen behavior is $\{(u, y) \in \mathcal{U}^N \times \mathcal{Y}^N \,|\, \text{col}(u_{ini}, u, y_{ini}, y) \in \mathcal{B}_{fz}|_{[1,T_{ini}+N]}\}$, where $\mathcal{U}^N$ is the cartesian product of $\mathcal{U}$ with itself N-times (similarly for $\mathcal{Y}^N$). According to Lemma 2.1, the feasible set of (2.10) can be written as the set of pairs $(u, y) \in \mathcal{U}^N \times \mathcal{Y}^N$ satisfying:

$$y = \mathcal{O}_N(A, C)x_{ini} + \mathcal{T}_N(A, B, C, D)u.$$

Further, coming from Lemma 2.1, a unique x_{ini} can be determined given a sufficiently long window of initial system data $\text{col}(u_{ini}, y_{ini})$ [19]. We now look at the feasible set of (2.17). By rewriting the constraints in (2.17) we obtain:

$$y = \mathcal{O}_N(A, C)\hat{x}(t) + \mathcal{T}_N(A, B, C, D)u, \quad u \in \mathcal{U}^N,\, y \in \mathcal{Y}^N,$$

where $\hat{x}(t)$ is the estimation of the state $x(t)$ at time t. Setting the state estimate $\hat{x}(t) = x_{ini}$ yields the equal feasible sets since the state-space coordinates of $\mathcal{B}_{fz}(A, B, C, D)$ and the system in (2.17) are identical. Since $R \succ 0$, the cost function in (2.17) is strictly convex in the decision variable u. Thus, since the constraints are convex and non-empty, a solution $(u^\star_{MPC}, x^\star_{MPC}, y^\star_{MPC})$ to (2.17) exists, and $u^\star_{MPC}$ is unique [43]. Similarly, the cost function in (2.10) is strictly convex in the decision variable u and the constraints are convex and non-empty. Hence, a solution $(g^\star_{DeePC}, u^\star_{DeePC}, y^\star_{DeePC})$ to (2.10) exists, and $u^\star_{DeePC}$ is unique. Since the cost function in (2.17) and (2.10) coincide, and the feasible sets of (2.17) and (2.10) are equal, then $u^\star_{MPC} = u^\star_{DeePC}$ for each frozen behavior $\mathcal{B}_{fz}$ in the interval $t \in [k^i_{switch} + T_{ini}, k^{i+1}_{switch} - N]$. Applying control inputs $\big(u(t), ..., u(t+k)\big) = (u^\star_0, u^\star_k)$ for some $k \leqslant N - 1$ to the frozen LTI subsystem (2.1) yields corresponding output sequences $\big(y(t), ..., y(t+k)\big) = (y^\star_0, ..., y^\star_k)$. Updating $\text{col}(u_{ini}, y_{ini})$ to the most recent input/output measurements while remaining in the interval $t \in [k^i_{switch} + T_{ini}, k^{i+1}_{switch} - N]$ and setting the state estimate in (2.17) to x_{ini} yields equal feasible sets. Repeating the above argument, both algorithms compute an identical optimal control sequence. This argument can be repeated for all iterations of the algorithms and for all frozen behaviors, proving the result. □

Theorem 3.1 implies that the closed loop behavior of using DeePC on an LPV system will be equivalent to the closed loop behavior of using MPC in the interval $t \in [k^i_{switch} + T_{ini}, k^{i+1}_{switch} - N]$, assuming that after each switch DeePC switches to the appropriate Hankel matrix and MPC changes its model to the according state-space representation. In other words, if the T_{ini}-length initial trajectory preceding the current time step t and the N-length future trajectory following the current time step t all belong to the same frozen behavior, then DeePC and MPC behave the same in closed loop. See Algorithm 3.1 for the pseudocode of the proposed control pipeline for LPV systems with known switching times.

Algorithm 3.1 LPV Systems Control for Known Switching Times - Online Hankel Selection

Given: Hankel library $\mathcal{H}_{lib} = \{\mathcal{H}_1, \ldots, \mathcal{H}_N\}$, $\mathcal{H}_i$ represents i'th LTI subsystem, error tolerance ϵ for error between predicted and measured output, known switching times $k_{switch} = \{k^0_{switch}, ..., k^n_{switch}\}$, initial data length T_{ini}, initial input u_{ini}, initial output y_{ini}, simulation length K.

procedure DeePC with Known k_{switch}
  for $k = 1$ to K **do**
    Calculate $u^\star$, y_{pred} with DeePC$(u_{ini}, y_{ini}, \mathcal{H})$
    Measure y_{meas} after applying $u^\star$ to the system
    if $k_{switch} \leqslant k \leqslant k_{switch} + T_{ini}$ **then** ▷ Morning to evening peak switch
      if $|y_{meas} - y_{pred}| > \epsilon$ **then**
        Find $\mathcal{H}^\star$ in $\mathcal{H}_{lib}$ that minimizes $\|\mathcal{H} \cdot \mathcal{H}^\dagger \cdot (u_{ini}, y_{ini}) - (u_{ini}, y_{ini})\|_2$
        Switch $\mathcal{H}$ in DeePC to $\mathcal{H}^\star$
      end if
    end if
  end for
end procedure

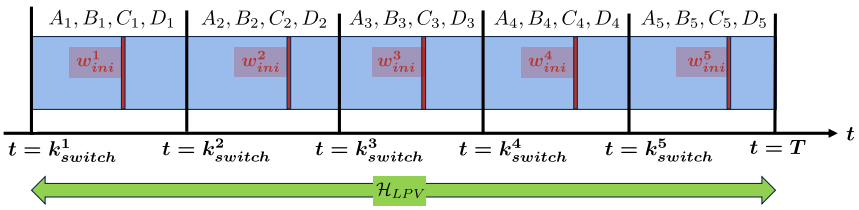


Figure 3.5: The switching times $k_{switch} = \{k^1_{switch}, \ldots, k^5_{switch}\}$ are unknown. A Hankel matrix $\mathcal{H}_{LPV}$, shown by the green arrow, is collected from the entire LPV system. T_{ini}-length data measurements w^i_{ini} of each LTI subsystem A_i, B_i, C_i, D_i are available.

3.1.5 LPV systems control for unknown switching times

If the switching times k^i_{switch} are unknown, then it is not possible to directly collect behavior-specific Hankel matrices $\mathcal{H}_i$ as shown in Figure 3.3 because we do not know when we are in which LTI subsystem. We can only collect a Hankel matrix $\mathcal{H}_{LPV}$ from the entire LPV system as shown in Figure 3.5. As the example in Section 3.1.1 illustrates, the (concatenated) Hankel matrix $\mathcal{H}_{LPV}$ of the entire LPV system is not the most powerful Hankel matrix to control each LTI subsystem A_i, B_i, C_i, D_i. Instead, we would like to have separate Hankel matrices $\mathcal{H}_i$ for each LTI subsystem as shown in Figure 3.3 to control each LTI subsystem A_i, B_i, C_i, D_i. Let us assume we have an initial trajectory w^i_{ini} from each subsystem A_i, B_i, C_i, D_i as illustrated in Figure 3.5. In particular, let us assume that w^i_{ini} uniquely belongs to just the i'th restricted frozen behavior $\mathcal{B}^i_{fz}|_{[1,T_{ini}]}$. Hence, $w^i_{ini} \in \mathcal{B}^i_{fz}|_{[1,T_{ini}]}$ and $w^i_{ini} \in \text{im } \mathcal{H}_i$. Since we assume that w^i_{ini} uniquely belongs to just the i'th restricted frozen behavior, then w^i_{ini} must not belong to any other restricted behavior $\mathcal{B}^j_{fz}|_{[1,T_{ini}]}$ with $j \neq i$. Hence, $w^i_{ini} \notin \mathcal{B}^{j\neq i}_{fz}|_{[1,T_{ini}]}$ and $w^i_{ini} \notin \text{im } \mathcal{H}_{j\neq i}$.

Therefore, w^i_{ini} only belongs to the subspace spanned by $\mathcal{H}_i$. If we now project w^i_{ini} onto $\mathcal{H}_i$ with linear least squares, then $\mathcal{H}_i$ should reconstruct w^i_{ini} without an error as w^i_{ini} lies in the subspace of $\mathcal{H}_i$. Hence, the minimizing linear least squares residual as calculated in (2.16) should zero. If, however, we project w^i_{ini} onto $\mathcal{H}_{j\neq i}$ with linear least squares, then the projection of w^i_{ini} onto $\mathcal{H}_{j\neq i}$ will not fully reconstruct w^i_{ini} because w^i_{ini} does not lie in the subspace of $\mathcal{H}_{j\neq i}$ [44]. Therefore, the minimizing linear least squares residual as calculated in (2.16) will be strongly greater than zero.

Mathematically this is expressed as follows:

$$r_{min}(\mathcal{H}_i, w^i_{ini}) = \|\mathcal{H}_i\mathcal{H}_i^\dagger w^i_{ini} - w^i_{ini}\|_2 \approx 0,$$
$$r_{min}(\mathcal{H}_j, w^i_{ini}) = \|\mathcal{H}_j\mathcal{H}_j^\dagger w^i_{ini} - w^i_{ini}\|_2 \gg 0 \quad \text{for } j \neq i.$$

We can use this fact to adapt the column ranges in $\mathcal{H}_{LPV}$ until we have filtered out the subsystem-specific Hankel matrices $\mathcal{H}_i$.

Offline Hankel matrix preprocessing

The idea of the offline Hankel matrix preprocessing step is to remove all column ranges $[i, j]$ in $\mathcal{H}_{LPV}$ that produce a large residual $\text{res}(\mathcal{H}_{LPV}[i, j], w^i_{ini})$, where the residual is calculated as in (2.16). In doing so, the remaining column ranges give a Hankel matrix $\mathcal{H}_i$ that only represents the i'th LTI subsystem from which w^i_{ini} had been collected. As a result, we have a Hankel matrix $\mathcal{H}_i$ that is more powerful to control the i'th LTI subsystem than with the concatenated $\mathcal{H}_{LPV}$. Figure 3.6 illustrates this idea, while Algorithm 3.2 details the specifics of the column range adaptation.

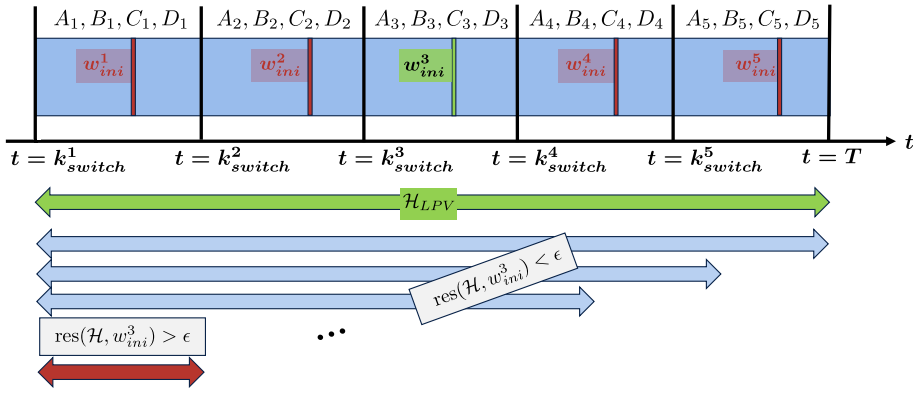
Algorithm 3.2 Offline Hankel Matrix Preprocessing

Given: Concatenated Hankel matrix $\mathcal{H}_{LPV}$ of LPV system, T_{ini}-length measured data $w_{ini} = (u_{ini}, y_{ini})$ of one LTI subsystem, minimum column range w_{min}, range of starting columns *range*, residual tolerance ϵ, total columns *cols* of $\mathcal{H}_{LPV}$.

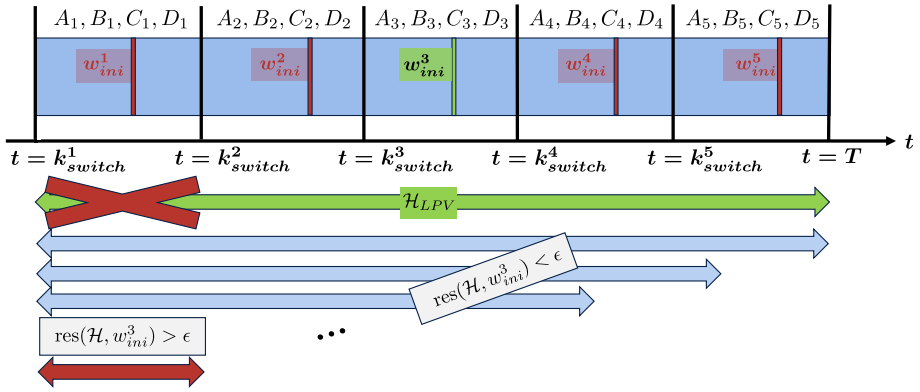
procedure Offline Hankel Matrix Preprocessing
  Empty list L where columns to be removed are saved
  for *start* in *range* **do**
    $end = cols$
    Consider $\mathcal{H}_{ini}$ containing the column range $[start, end]$ of $\mathcal{H}_{LPV}$
    while $[start, end] > w_{min}$ **do**
      $res = \|\mathcal{H}_{ini} \cdot \mathcal{H}_{ini}^\dagger \cdot w_{ini} - w_{ini}\|_2$
      if $res < \epsilon$ **then** ▷ Small residual: Window of columns too large
        Decrease *end*
        $\mathcal{H}_{ini}$ contains smaller column range $[start, end]$ of $\mathcal{H}_{LPV}$
      else ▷ Big residual: Remove these columns
        Add column range $[start, end]$ to L
      end if
    end while
  end for
  Remove all columns of $\mathcal{H}_{LPV}$ that are in L
  Remaining $\mathcal{H}$ represents w_{ini}
end procedure

LPV systems control with unknown switching times

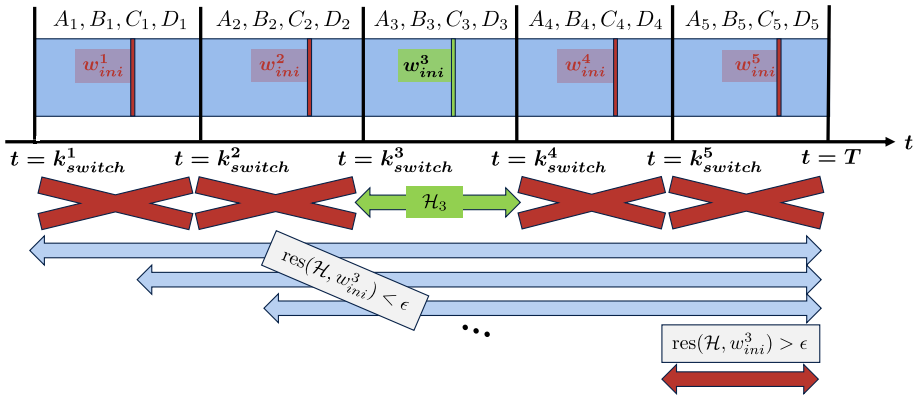
The first step in the control pipeline with unknown switching times is to create a Hankel library $\mathcal{H}_{lib} = \{\mathcal{H}_1, \ldots, \mathcal{H}_N\}$ with Algorithm 3.2, by filtering out the $N \in \mathbb{N}$ Hankel matrices $\mathcal{H}_i$ from $\mathcal{H}_{LPV}$ with the initial data measurements w^i_{ini} from the N LTI subsystems. One problem we still have, is that we do not know the switching times of the LPV system, and so we do not know a-priori when to switch Hankel matrices. What we do know is that DeePC as shown in (2.10) calculates control inputs that are compatible with the Hankel matrix to track a reference trajectory. Therefore, if the Hankel matrix is a bad representation of the current LTI subsystem, then applying the control input to the system will lead to very different outputs than what the model predicts. We can use this to find the switching times of the system online. Whenever the measured output



(a) Adapt the column range of $\mathcal{H}_{LPV}$, shown by blue arrows, until linear least square residual is greater than error tolerance, $\text{res}(\mathcal{H}_{LPV}, w^3_{ini}) > \epsilon$.



(b) Remove the columns where $\text{res}(\mathcal{H}_{LPV}, w^3_{ini}) > \epsilon$ from $\mathcal{H}_{LPV}$ as shown by the red cross, as they do not fit w^3_{ini} well in a least squares sense.



(c) Repeat this process for varying starting columns, shown by blue arrows, until the only column range left in $\mathcal{H}_{LPV}$ is the column range of $\mathcal{H}_3$ that represents the LTI subsystem as shown by the green arrow.

Figure 3.6: Offline Hankel matrix preprocessing used to find a subsystem-specific Hankel matrix $\mathcal{H}_3$ from initial LTI subsystem measurements w^3_{ini}.

differs too greatly from the predicted output, the LPV system must have switched subsystems and we need to switch our Hankel matrices. We can detect the switch online if the absolute value of the output error is greater than a user-defined threshold ϵ. Therefore, a switch is detected online if $|y_{meas} - y_{pred}| > \epsilon$.

Algorithm 3.3 shows pseudocode of the control pipeline for an LPV system with unknown switching times where the Hankel library $\mathcal{H}_{lib} = \{\mathcal{H}_1 \dots \mathcal{H}_N\}$ has been created offline with Algorithm 3.2 and the switching times are detected online with the metric $|y_{meas} - y_{pred}| > \epsilon$.

Algorithm 3.3 LPV Systems Control for Unknown Switching Times - Online Hankel Selection

Given: Hankel library $\mathcal{H}_{lib} = \{\mathcal{H}_1, \dots, \mathcal{H}_N\}$, $\mathcal{H}_i$ represents i'th LTI subsystem, error tolerance ϵ for error between predicted and measured output, initial data length T_{ini}, initial input u_{ini}, initial output y_{ini}, simulation length K.

procedure DeePC with Unknown k_{switch}
  for $k = 1$ to K **do**
    Calculate $u^\star$, y_{pred} with DeePC$(u_{ini}, y_{ini}, \mathcal{H})$
    Measure y_{meas} after applying $u^\star$ to the system
    if $|y_{meas} - y_{pred}| > \epsilon$ **then** ▷ Switch detected online
      Find $\mathcal{H}^\star$ in $\mathcal{H}_{lib}$ that minimizes $\|\mathcal{H} \cdot \mathcal{H}^\dagger \cdot (u_{ini}, y_{ini}) - (u_{ini}, y_{ini})\|_2$
      Switch $\mathcal{H}$ in DeePC to $\mathcal{H}^\star$
    end if
  end for
end procedure

3.2 DeePC for traffic network as an LPV system

We propose building on the foundations of the control pipelines in Algorithm 3.1 and Algorithm 3.3 to control the nonlinear traffic network as an LPV system.

3.2.1 Problem setup

In the context of urban traffic, we assume that the exogenous input variable $d(t)$ of the traffic demand leads to different behaviors of the nonlinear traffic network. Further, we assume that a day has a morning and an evening peak, and we can split the day into two behavior times I, II as shown in Figure 3.7. Although the traffic demand varies each day, we assume that the general timings of the morning and evening peak remain similar. Therefore we assume that we know when the switch between morning and evening peak happen $\big($(behavior times I, II) from Figure 3.7$\big)$. Lastly, in each behavior time there exist frozen behaviors (hence, LTI systems), allowing to describe a day of traffic as an LPV system.

Since the traffic network is nonlinear, we do not know how many frozen behaviors are necessary to describe the nonlinear system. Therefore, we propose to collect many Hankel matrices for various traffic demands $d(t)$. Then, we cluster the Hankel matrices collected from the various traffic demands into behavior groups, to search for the main frozen behaviors that describe the nonlinear system. Whenever the behavior times switch (which we assume to know), we switch the Hankel matrices. Additionally to this expected switch, we check for any unexpected behavior switches by monitoring the absolute value of the output error ($|y_{pred} - y_{meas}|$). If the absolute value exceeds a user defined threshold ϵ, so if $|y_{pred} - y_{meas}| > \epsilon$, then we also propose switching Hankel matrices. Such an unexpected switch may happen because we are approximating the nonlinear system with an LTI system. If the predictions do not align with the measurements, there must be a different LTI system that approximates the current conditions better.

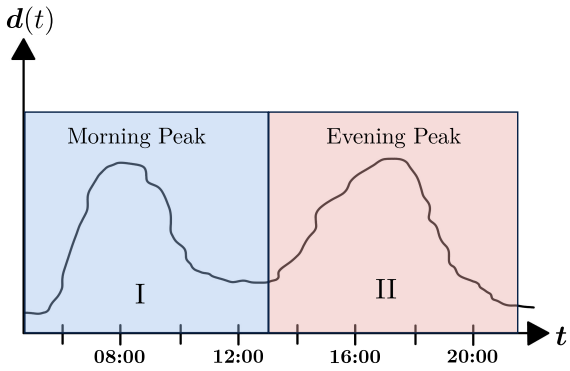


Figure 3.7: The traffic demand $d(t)$ has a morning and an evening peak. The shape of the peak varies depending on the traffic demand. The switch from morning to evening peak remains similar, allowing to split the day into two behavior times I, II.

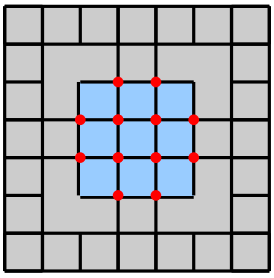


Figure 3.8: Traffic grid consisting of outer region 0 shaded in grey and inner region 1 shaded in blue. The traffic lights at the 12 intersections marked in red can by dynamically controlled.

3.2.2 Traffic demands

We use the traffic grid shown in Figure 3.8 to simulate the traffic dynamics of a city. The inner region 1 shaded in blue represents the commercial city center, whereas the outer region 0 shaded in grey represents the residential areas. The twelve intersections marked in red have traffic lights that can be dynamically controlled.

To reduce the computational complexity and duration of our simulations, we model a traffic day as a two hour period where the first hour represents the morning and the second hour represents the evening. The simulation acts as a digital twin of reality. This brings the advantage to simulate arbitrary traffic demands, and to be able to apply arbitrary control inputs to the system without affecting the real traffic network. This is important, since we need to apply persistently exciting control inputs during the data collection phase. Figure 3.9 and Figure 3.10 show the synthetic demand shapes we consider for the morning and evening peaks.

During the morning peak, the traffic flows from the outer region 0 to the city center region 1. This represents people commuting to work. As a result, only the car flow from region 0 to region 1 plotted in orange in Figure 3.9 ramps up or down, while the traffic demand from the other region pairs remain constant. The amount of cars that are spawned every 60 seconds is given by the demand value from the demand plot. For example, if the value of the orange graph is 20, that means that every three seconds a car will be spawned with the intention of driving from region 0 to region 1. Across 60 seconds, this will then amount to 20 cars. To add variability to the demand modules, each graph is corrupted by normally-distributed noise which is why the graphs in Figure 3.9 are not straight. Additionally, the exact routing a car takes to get from region i to region j (where

$i, j \in \{0, 1\}$) is random. This means that a car can start a trip on any edge in region i and end on any edge in region j.

The traffic demands are defined in the same way for the evening peak, except that now the traffic flows from the inner region 1 to the outer region 0. This represents people commuting back to the residential areas. As a result, only the car flow from region 1 to region 0 plotted in green in Figure 3.10 ramps up or down, while the traffic demand from the other region pairs remain constant. The traffic demand for each region pair $i, j \in \{0, 1\}$ in Figure 3.9 (e.g. traffic demand going from region 0 to region 1) is of the form below.

$$d_{ij}(t) = \begin{cases} a_{ij}t + b_{ij} + \mathcal{N}(0, n_{ij}), & \text{if } 0 \leqslant t < \frac{T_{end}}{2}, \\ c_{ij} + \mathcal{N}(0, n_{ij}), & \text{if } \frac{T_{end}}{2} \leqslant t < T_{end} \end{cases}$$

In the equation above, $a_{ij} \in \mathbb{R}$ is a real number, $b_{ij}, c_{ij} \in \mathbb{Z}_+$ are non-negative integers, and $\mathcal{N}(0, n_{ij})$ is a normal Gaussian distribution with zero mean and non-negative standard deviation $n_{ij} \in \mathbb{R}_+$. A random number is drawn from $\mathcal{N}(0, n_{ij})$ at each discrete time instance $t \in \mathbb{Z}_+$. Each discrete time instance t represents a minute, and therefore $d(t)_{ij}$ represents how many vehicles will start their journey from region i to region j during the t'th minute.

Weekday

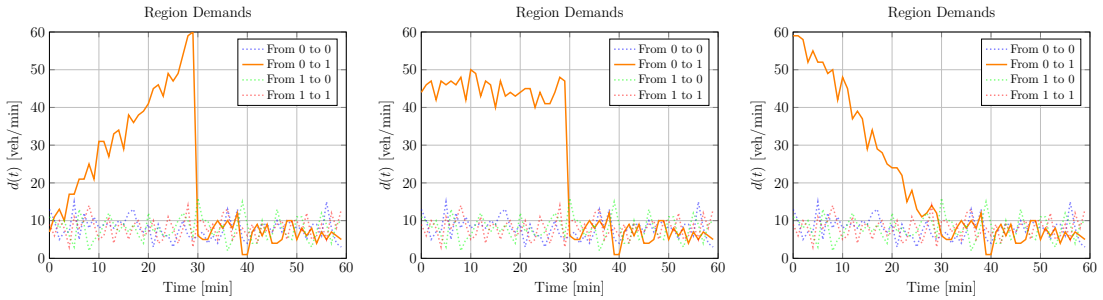
On a weekday, we assume that the traffic in region 1 congests early in the morning when people commute into the commercial city center to work, and in the evening when people commute back to the residential city areas. Therefore, we assume that the morning peak must either stay constant or ramp down as plotted in Figure 3.9b respectively Figure 3.9c. The evening peak may have any shape plotted in Figure 3.10. Combining the demand modules leads to the combinations plotted in Figure 3.11. The combinations provide weekdays where people arrive to work early, late or throughout the morning, and people return home early, late or throughout the evening.

Weekend

On a weekend day, we assume that the traffic starts slower into the day. Hence, cars driving to the commercial center congest in the later morning, while cars driving back to the residential areas might congest at different times of the afternoon, depending on how long people stay in the city center. Therefore, we assume that the morning peak must ramp up as plotted in Figure 3.9a. The evening peak may have any shape plotted in Figure 3.10. Combining the demand modules leads to the combinations plotted in Figure 3.12. The combinations provide weekend days where people gradually arrive in the city center, and people return home early, late or throughout the evening.

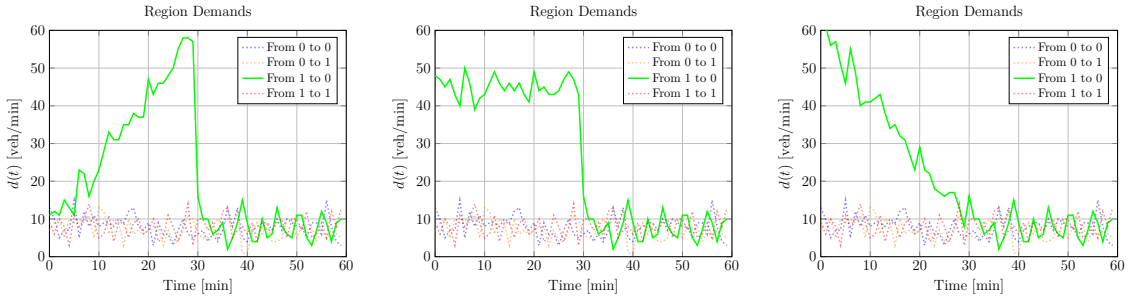
3.2.3 Data clustering

Since the traffic network is a nonlinear system, there is no guarantee that Lemma 2.2 holds. In other words, if we collect data $w = (u, y)$ for a specific traffic demand $d(t)$ where u is persistently exciting, we cannot guarantee that the resulting Hankel matrix is a good representation of the traffic network with the traffic demand $d(t)$. Because of potential nonlinearities, the traffic light cycle input may excite different dynamics of the traffic network depending on which traffic light cycle was chosen. Additionally, we collect multiple Hankel matrices for each demand module plotted in Figure 3.9 and Figure 3.10. We then cluster the collected Hankel matrices with k-means as in Algorithm 2.2 where the initialization is done with Algorithm 2.3. The clustering gives us behavior groups among the collected Hankel matrices.



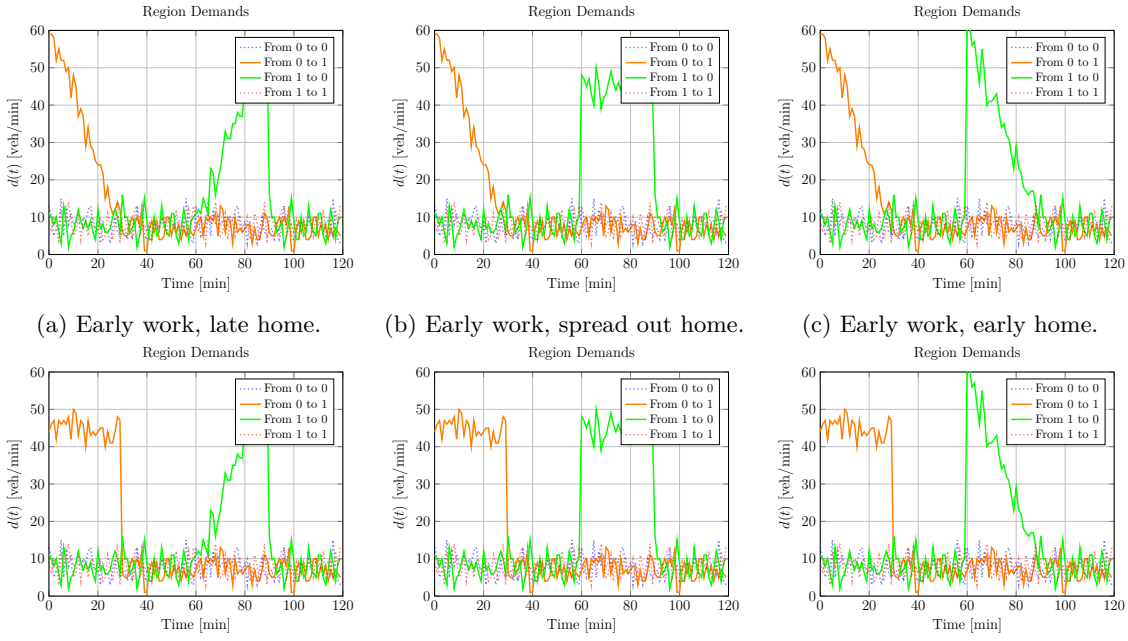
(a) Traffic peak ramps up. (b) Constant traffic peak. (c) Traffic peak ramps down.

Figure 3.9: Demand modules for the morning peak where most traffic flows from the outer region 0 to the inner region 1.



(a) Traffic peak ramps up. (b) Constant traffic peak. (c) Traffic peak ramps down.

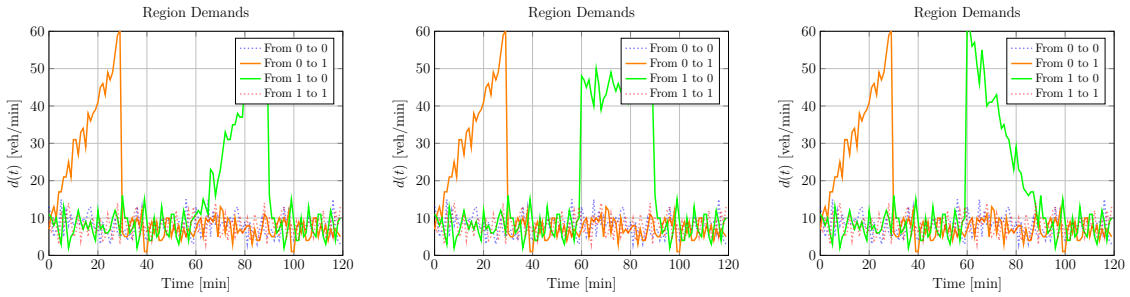
Figure 3.10: Demand modules for the evening peak where most traffic flows from the inner region 1 to the outer region 0.



(a) Early work, late home. (b) Early work, spread out home. (c) Early work, early home.

(d) Spread out work, late home. (e) Spread out work, spread out home. (f) Spread out work, early home.

Figure 3.11: Weekday traffic demand combinations.



(a) Late work, late home. (b) Late work, spread out home. (c) Late work, early home.

Figure 3.12: Weekend traffic demand combinations.

3.2.4 DeePC for traffic network as an LPV system

The control approach we propose for nonlinear traffic networks as an LPV system builds on the control pipelines presented for LPV systems with known and unknown switching times, and uses k-means clustering to find behavior groups.

First, we collect multiple Hankel matrices $\mathcal{H}_i$ for each traffic demand module. All of the collected Hankel matrices are stored in a Hankel matrix library $\mathcal{H}_{lib} = \{\mathcal{H}_1, \ldots, \mathcal{H}_N\}$. Due to potential nonlinearities in the urban traffic network, it may be that Hankel matrices collected from the same traffic demand do not represent the same system behavior. Therefore, we apply k-means clustering to the N Hankel matrices in $\mathcal{H}_{lib}$ to look for k behavior groups. The k cluster centers $\mathcal{C}_i$, which are the average values of all the Hankel matrices belonging to the i'th cluster, are stored in a second Hankel matrix library, the behavior group library $\mathcal{G}_{lib} = \{\mathcal{G}_1, \ldots, \mathcal{G}_k\}$. Each $\mathcal{G}_i$ contains the cluster center $\mathcal{C}_i$ and all the Hankel matrices belonging to the i'th cluster.

When we run the DeePC controller for the urban traffic network as an LPV system, we switch the Hankel matrix of the controller whenever the urban traffic network transitions from the morning to the evening peak, or when the predicted output differs too much from the measured output.

Whenever the urban traffic network transitions from the morning to the evening peak, we assume that the system switches behavior groups. Therefore, among the behavior groups in $\mathcal{G}_{lib}$, we look for the cluster center $\mathcal{C}_i$ which fits the current traffic conditions (u_{ini}, y_{ini}) best in the least squares sense. From the Hankel matrices belonging to the behavior group of the cluster center $\mathcal{C}_i$, we select the Hankel matrix $\mathcal{H}_i$ that represents y_{ini} best in the least squares sense for the DeePC controller.

Furthermore, if we observe an absolute error between the predicted output y_{pred} and the measured output y_{meas} that exceeds a user-defined threshold ϵ, so $|y_{pred} - y_{meas}| > \epsilon$, then we also switch the Hankel matrix of the DeePC controller. In this case, we select the Hankel matrix from all possible Hankel matrices in $\mathcal{H}_{lib}$ that represents y_{ini} best in the least squares sense. The procedure is described in pseudocode in Algorithm 3.4.

Algorithm 3.4 DeePC for Traffic Network as an LPV System - Online Hankel Selection

Given: Hankel library $\mathcal{H}_{lib} = \{\mathcal{H}_1, \ldots, \mathcal{H}_N\}$ with N Hankel matrices, behavior group library $\mathcal{G}_{lib} = \{\mathcal{G}_1, \ldots, \mathcal{G}_k\}$ with k behavior groups, $\mathcal{G}_i$ contains cluster center $\mathcal{C}_i$ and all Hankel matrices of i'th cluster, error tolerance ϵ for error between predicted and measured output, known switching times k_{switch}, initial data length $Tini$, initial input u_{ini}, initial output y_{ini}, simulation length K.

procedure URBAN TRAFFIC DEEPC
  for $k = 1$ to K **do**
    Calculate $u^\star$, y_{pred} with DeePC$(u_{ini}, y_{ini}, \mathcal{H})$
    Measure y_{meas} after applying $u^\star$ to the system
    if $|y_{meas} - y_{pred}| > \epsilon$ **then** ▷ Switch detected online
      Find $\mathcal{H}^\star$ in $\mathcal{H}_{lib}$ that minimizes $\|\mathcal{H} \cdot \mathcal{H}^\dagger \cdot y_{ini} - y_{ini}\|_2$
      Switch $\mathcal{H}$ in DeePC to $\mathcal{H}^\star$
    else
      if $k_{switch} \leqslant k \leqslant k_{switch} + T_{ini}$ **then** ▷ Morning to evening peak switch
        Find $\mathcal{C}^\star$ in $\mathcal{G}_{lib}$ that minimizes $\|\mathcal{C} \cdot \mathcal{C}^\dagger \cdot (u_{ini}, y_{ini}) - (u_{ini}, y_{ini})\|_2$
        Find $\mathcal{H}^\star$ in $\mathcal{C}^\star$ that minimizes $\|\mathcal{H} \cdot \mathcal{H}^\dagger \cdot y_{ini} - y_{ini}\|_2$
        Switch $\mathcal{H}$ in DeePC to $\mathcal{H}^\star$
      end if
    end if
  end for
end procedure

Chapter 4

Results

All simulations were conducted on a 2.4 GHz Intel® Core™ i9-9980HK processor, with 32 GB of RAM, running a 64-bit Windows 11 Home operating system.

4.1 LPV systems

We simulated an LPV system that consists of five different controllable and observable LTI systems (frozen behaviors). Each LTI subsystem has 16 inputs, 2 outputs and 4 states, and was randomly generated with the Python Control Systems Library (control.drss). During the data collection phase, the system switches subsystems every 200 steps. We chose a prediction horizon $N = 4$ and initial trajectory length $T_{ini} = 4$, giving a Hankel matrix of depth $L = N + T_{ini} = 8$. The concatenated Hankel matrix $\mathcal{H}_{LPV}$ was generated by applying a persistently exciting input to one LPV simulation run of 1000 steps (where every 200 steps the LTI subsystem switched), and arranging the data generated from this simulation run according to (2.9).

4.1.1 LPV systems control results for known switching times

Since we know the switching times, we directly construct subsystem-specific Hankel matrices $\mathcal{H}_i$, $i \in \{1, \ldots, 5\}$ during the data collection phase where each $\mathcal{H}_i$ receives the data from the 200 simulation steps of that specific subsystem. We control the LPV system with switching $\mathcal{H}_i$ as detailed in Algorithm 3.1 and compare this to controlling the LPV system with a fixed, concatenated $\mathcal{H}_{LPV}$. To select the correct $\mathcal{H}_i$, we need at most T_{ini} simulation steps, since then the T_{ini} most recent measurements will all be generated from just one subsystem. During the switching transition, the T_{ini} most recent measurements are a mixture of two LTI subsystems. After this switching transition phase, we keep $\mathcal{H}_i$ fixed until the next switch.

For our comparison of Algorithm 3.1 and $\mathcal{H}_{LPV}$ we care about the steady-state output reference tracking error and the controller computation time. Therefore, for our comparison we simulate each subsystem for 10 steps (instead of 200 steps as in the collection phase), as this suffices to illustrate the control performance differences of the two approaches during the steady-state phase. The comparison simulation was conducted with five different LPV systems. Figure 4.1 shows the output reference tracking of one of the simulations (the other plots can be found in Appendix A.1.1).

Plotted in blue is the control approach detailed in Algorithm 3.1 where after each subsystem switch we switch to the best subsystem-specific Hankel matrix $\mathcal{H}_i$ in the linear least squares sense. We observe that after each switch, there is a transient phase where the output initially diverges away

Control Approach	Average Steady-State Output Error [-]	Average computation time [s]
$\mathcal{H}_{LPV}$	$[1.61 \pm 1.17,\ 1.51 \pm 0.64]$	505.1 ± 104.2
$\mathcal{H}$ as in Algorithm 3.1	$[\mathbf{0.19} \pm 0.12,\ \mathbf{0.15} \pm 0.12]$	$\mathbf{8.5} \pm 1.2$
Improvement Factor [-]	Alg. 3.1 {8.5, 10.1}x more accurate	Alg. 3.1 59x more efficient

Table 4.1: Comparison of controlling LPV system with Algorithm 3.1 and $\mathcal{H}_{LPV}$. Average steady-state output errors with standard deviation and controller computation times with standard deviation of simulation runs with five different LPV systems. Improvement factor quantifying gain with Algorithm 3.1 over $\mathcal{H}_{LPV}$. Best metric in bold.

from the reference value. This is because during this phase the T_{ini} most recent measurements belong to two LTI subsystems, whereas the image of each $\mathcal{H}_i$ represents only one LTI subsystem. Hence, the controller cannot accurately match the initial conditions to the image of the Hankel matrix. The control actions that the controller calculates are therefore not optimal and lead to this temporary divergence. After the transition phase ends, the correct $\mathcal{H}_i$ is uniquely identified and the T_{ini} most recent measurements can be matched to the image of the Hankel matrix. Therefore, the controller can stabilize the system and track the reference output.

The red graph shows the reference tracking output of the concatenated Hankel matrix $\mathcal{H}_{LPV}$. We can see that during the switching transition phase the output diverges much less than with Algorithm 3.1. This is because the concatenated Hankel matrix captures the switching dynamics of the LPV system. However, once the transition phase ends, the reference tracking does not significantly improve because $\mathcal{H}_{LPV}$ is less powerful than $\mathcal{H}_i$ at describing one specific LTI subsystem as illustrated in Section 3.1.1. As a result, the average reference output tracking error is larger with $\mathcal{H}_{LPV}$ than switching $\mathcal{H}_i$ with Algorithm 3.1.

Table 4.1 summarizes the average output error after the switching transition phase and the average controller computation times for the five simulation runs. As the numbers show, the average reference output tracking error is reduced by an order of magnitude, namely over 8 times for the first output and over 10 times for the second output, when controlling the LPV system with Algorithm 3.1 instead of $\mathcal{H}_{LPV}$. Furthermore, the control computation time is drastically reduced by nearly 60 times with the use of Algorithm 3.1 instead of $\mathcal{H}_{LPV}$. The smaller standard deviation with Algorithm 3.1 than $\mathcal{H}_{LPV}$ suggests that Algorithm 3.1 is more robust to different system compositions than $\mathcal{H}_{LPV}$.

4.1.2 LPV systems control results for unknown switching times

We construct subsystem-specific Hankel matrices $\mathcal{H}_i$ with Algorithm 3.2 by adapting the column range of $\mathcal{H}_{LPV}$ to the initial measurements w^i_{ini} of each specific subsystem $i \in \{1, \dots, 5\}$. We control the LPV system with switching $\mathcal{H}_i$ as detailed in Algorithm 3.3 and compare this to controlling the LPV system with a fixed, concatenated $\mathcal{H}_{LPV}$. Whenever the prediction error is above the error tolerance (we chose $\epsilon = 0.5$), we look for a new $\mathcal{H}_i$ to match the w_{ini} most recent measurements.

As in the previous section, we care about comparing the steady-state reference output tracking error and controller computation time of Algorithm 3.3 and $\mathcal{H}_{LPV}$. We again do this by simulating each subsystem for 10 steps. Figure 4.2 shows the reference output tracking of one of the simulations (the other four simulation plots are shown in Appendix A.1.2).

The blue graph shows the reference output tracking when controlling the system with Algorithm 3.3, where we switch the subsystem-specific Hankel matrix $\mathcal{H}_i$ whenever the prediction error is larger than $\epsilon = 0.5$. During the switching transition phase, the output of the controller diverges from the desired output (because we are not using the Hankel matrix that represents the new subsystem,

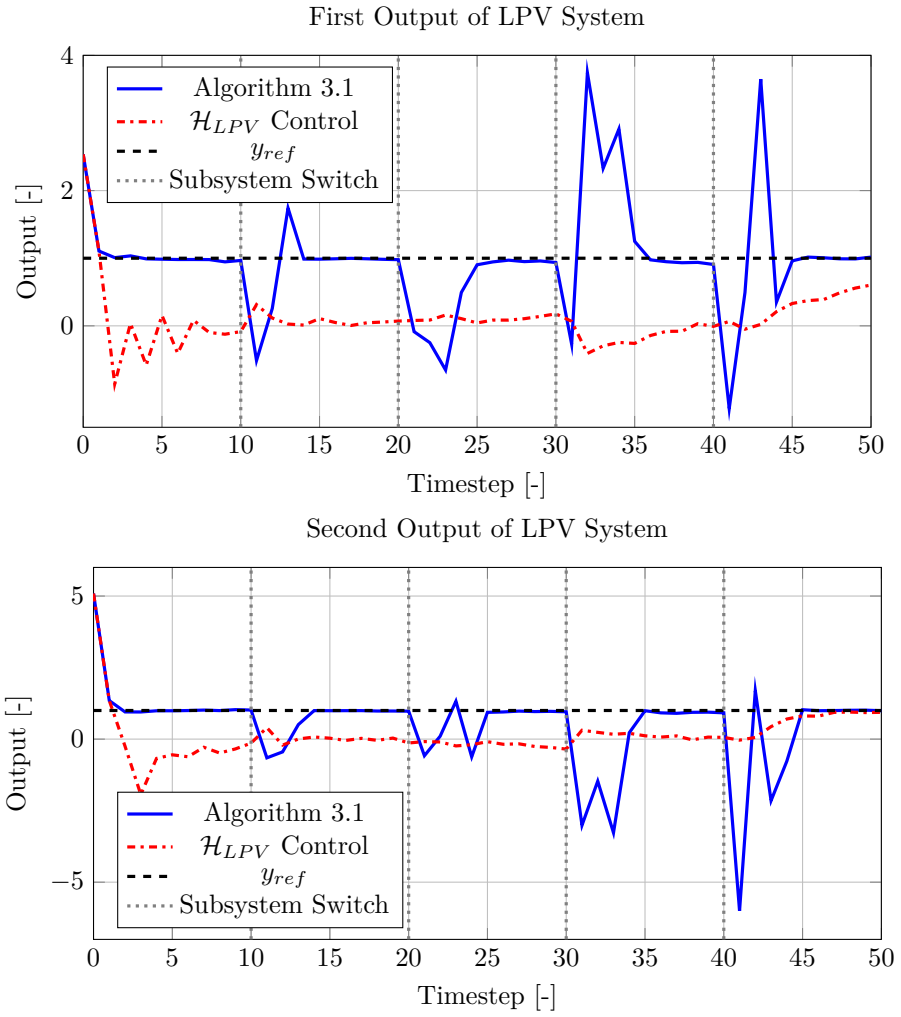


Figure 4.1: Reference output tracking of LPV system with known switching times. Switching subsystem-specific Hankel matrices with Algorithm 3.1 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (red line). Algorithm 3.1 produced an average error of $[0.04, 0.029]$ for the first respectively second output, and took 8.8s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.91, 0.94]$, and took 337.8s to compute.

and the last T_{ini} measurements of our system belong to two LTI subsystems and therefore cannot accurately be matched to the image of any subsystem-specific Hankel matrix $\mathcal{H}_i$). The algorithm detects this because the predicted output does not match the measured output. As the transition phase settles down (this takes at most $T_{ini} = 4$ time steps), the algorithm finds the subsystem-representative Hankel matrix, making it possible to accurately match the initial conditions to the image of the Hankel matrix, and stabilizes the system. As before, we observe that the red graph coming from controlling the system with a concatenated Hankel matrix $\mathcal{H}_{LPV}$ diverges less from the reference trajectory during the switching transition phase than Algorithm 3.3 because the switching dynamics are included in $\mathcal{H}_{LPV}$. However, it fails to track the reference output as accurately as Algorithm 3.3 because the concatenated Hankel matrix $\mathcal{H}_{LPV}$ is less powerful than the subsystem-specific Hankel matrix $\mathcal{H}_i$.

Table 4.2 summarizes the average reference output tracking errors of the five simulation runs when controlling the LPV system with Algorithm 3.3 and the concatenated Hankel matrix $\mathcal{H}_{LPV}$. For the former, the steady-state error is defined as the output error when the prediction error is below the error threshold of 0.5, as then the controller has found a subsystem-specific Hankel matrix that represents the current LTI subsystem well. For the concatenated approach $\mathcal{H}_{LPV}$ the output error that remains T_{ini} or more steps after a switch is used to calculate the steady-state error. As we can see, Algorithm 3.3 reduces the average output error by over 4 times for both the first and second output. The controller computation time is reduced by about 55 times. The smaller standard deviation with Algorithm 3.3 than $\mathcal{H}_{LPV}$ suggests that Algorithm 3.3 is more robust to different system compositions than $\mathcal{H}_{LPV}$.

4.1.3 LPV systems control results discussion

The simulations for LPV systems with known and unknown switching systems confirmed that a subsystem-specific Hankel matrix $\mathcal{H}_i$ is more powerful at representing an LTI subsystem than a concatenated Hankel matrix $\mathcal{H}_{LPV}$. This was shown by the increased reference output tracking accuracy of controlling the LPV system with both Algorithm 3.1 and Algorithm 3.3 versus $\mathcal{H}_{LPV}$. Additionally, the lower standard deviations in the reference output tracking errors when using Algorithm 3.1 and Algorithm 3.3 instead of $\mathcal{H}_{LPV}$ suggest that the proposed algorithms are more robust to changing LPV compositions. In other words, if the controller must guarantee to stay within an error bound in steady-state operations, then a more conservative, higher upper bound must be chosen for $\mathcal{H}_{LPV}$ because the results vary more.

Another key advantage of controlling the LPV system with switching Hankel matrices instead of a fixed, concatenated Hankel matrix was seen by the reduced computation times for each simulation run.

Since $\mathcal{H}_{LPV}$ is a concatenation of all LTI subsystems of the LPV system, the controller generally is less prone to diverging from the reference output during the transient, subsystem switching phase. Nevertheless, as seen by e.g. Figure A.8, there were some simulation runs where controlling the LPV system with $\mathcal{H}_{LPV}$ also led to a strong output divergence from the reference output during the switching phase. Further analysis would be required to draw conclusions of the reference output tracking evolution during the switching phase when controlling the LPV system with switching Hankel matrices or a fixed, concatenated Hankel matrix.

Control Approach	Average Steady-State Output Error [-]	Average computation time [s]
$\mathcal{H}_{LPV}$	$[1.21 \pm 0.43,\ 1.32 \pm 0.75]$	483.0 ± 139.7
$\mathcal{H}$ as in Algorithm 3.3	$[\mathbf{0.25} \pm 0.05,\ \mathbf{0.31} \pm 0.12]$	$\mathbf{8.7} \pm 0.9$
Improvement Factor [-]	Alg. 3.3 {4.8, 4.3}x more accurate	Alg. 3.3 55.5x more efficient

Table 4.2: Comparison of controlling LPV system with Algorithm 3.3 and $\mathcal{H}_{LPV}$. Average steady-state output errors with standard deviation and controller computation times with standard deviation of simulation runs with five different LPV systems. Best metric in bold.

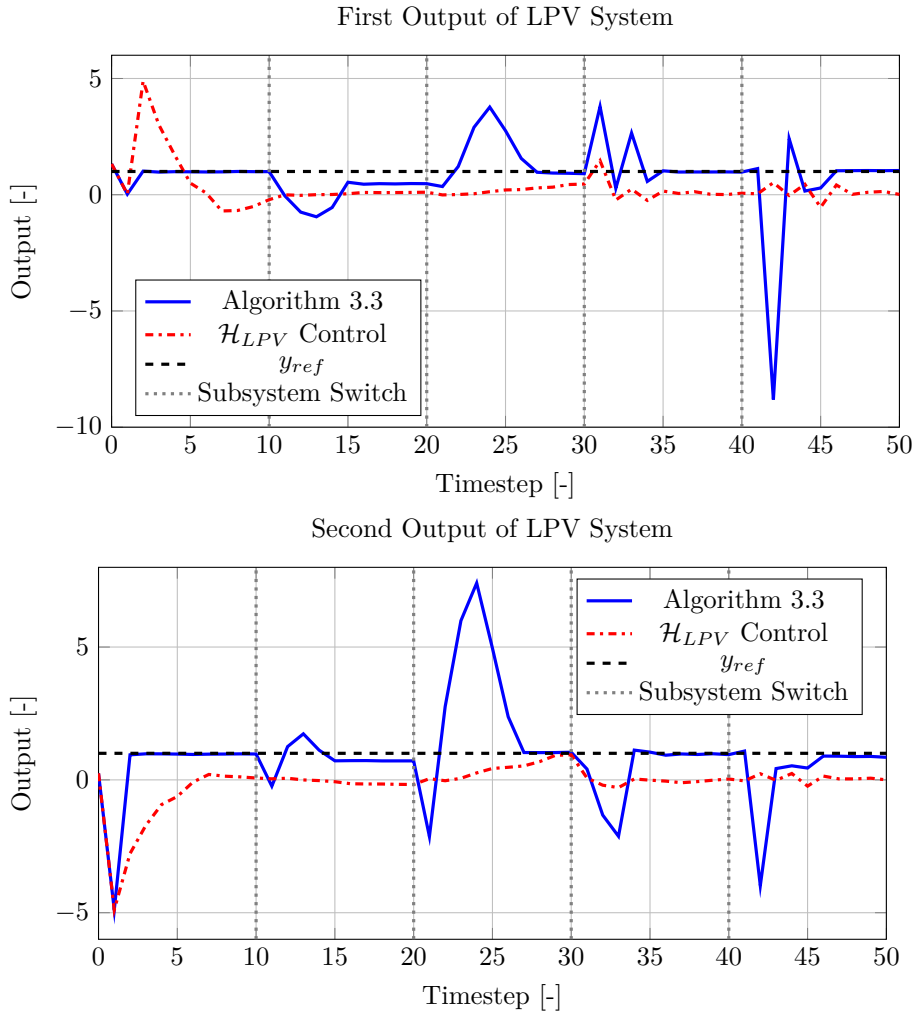


Figure 4.2: Reference output tracking of LPV system with unknown switching times. Switching subsystem-specific Hankel matrices with Algorithm 3.3 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (red line). Algorithm 3.3 produced an average error of $[0.28, 0.39]$ for the first respectively second output, and took 7.8s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.95, 0.91]$, and took 564.9s to compute.

4.2 DeePC for urban traffic network as an LPV system

4.2.1 Traffic grid

All the traffic network simulations were carried out on the traffic grid pictured in Figure 3.8. The grid consists of two regions 0, 1, and contains twelve intersections in the inner region 1 where the traffic lights can be dynamically controlled. The simulations were carried out with the Simulation of Urban Mobility (SUMO) package, an open source, highly portable, microscopic and continuous traffic simulation package [45]. We simulated the nine traffic demands plotted in Figure 3.11 and Figure 3.12. We simulated a *No Control* scenario where all the green-red traffic light cycles remained fixed, a scenario where the twelve controllable traffic lights were controlled with a fixed, concatenated Hankel matrix $\mathcal{H}_{LPV}$, and a scenario where the twelve controllable traffic lights were controlled with switching Hankel matrices as described in Algorithm 3.4. The simulation was carried out ten times for each scenario and each traffic demand.

4.2.2 Clustering results

For each traffic demand module plotted in Figure 3.9 and Figure 3.10 we collected three Hankel matrices. We can observe that the traffic demand profile influences the clustering. One behavior group cluster contains the traffic demand profiles where the traffic demand from region 0 to region 1 (or region 1 to region 0) ramps up (corresponding to late congestion). The other behavior group clusters contain a mix of demand profiles where the traffic demand is either ramping down or spread out, corresponding to early or constant congestion. It makes sense that the early congestion may lead to a similar behavior as a spread out congestion because the network gets congested right away while late congestion leads to a different behavior in the traffic network as the congestion grows gradually. Furthermore, the directionality of the traffic demand (e.g. morning commute versus evening commute) did not influence the k-means clustering.

4.2.3 DeePC urban traffic control results

Weekday results

The average weekday results for the demand profiles plotted in Figure 3.11 are captured in Table 4.3. As mentioned before, the *No Control* results column simulated the traffic demand with a fixed, identical traffic light cycle for each traffic light in the grid network, while the second results column (DeePC with fixed $\mathcal{H}_{LPV}$) dynamically controlled the twelve traffic lights marked in red in Figure 3.8 with a fixed Hankel matrix $\mathcal{H}_{LPV}$ that was collected for the entire traffic demand (no distinction between morning and evening peak), and the third results column controlled the traffic lights with Algorithm 3.4. The simulation was repeated ten times for each control setting and traffic demand profile.

As seen in Table 4.4, when the controller switches Hankel matrices according to Algorithm 3.4, the total control input computation time is reduced by over 3 times than when the controller uses a fixed Hankel matrix $\mathcal{H}_{LPV}$. Further, we see that the travel time per vehicle with Algorithm 3.4 is reduced by over two times compared to the *No Control* scenario, and by nearly two times compared to using a fixed $\mathcal{H}_{LPV}$. In particular, the standard deviation is much smaller using Algorithm 3.4 than for *No Control* or $\mathcal{H}_{LPV}$. This means that Algorithm 3.4 is more robust at preventing large congestion and/or traffic gridlock for various traffic demands. In other words, we have no a-priori guarantee that $\mathcal{H}_{LPV}$ will control the urban traffic network well. On the contrary, the control inputs calculated from the controller with $\mathcal{H}_{LPV}$ might lead to a higher traffic congestion than using no controller at all if $\mathcal{H}_{LPV}$ strongly misrepresents the system behavior (see e.g. Table A.4

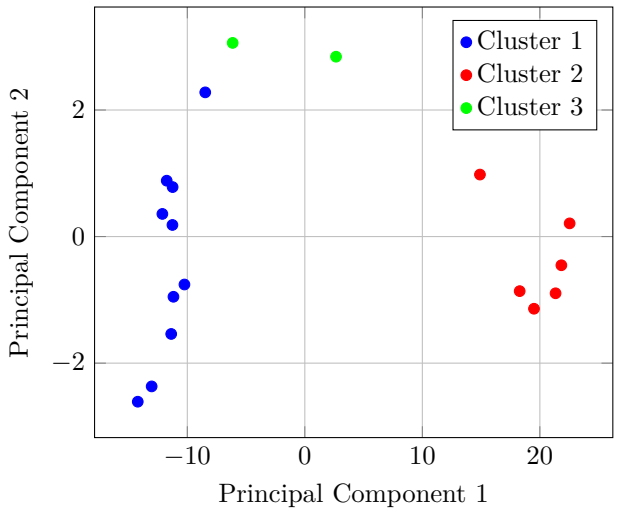


Figure 4.3: Clustered behavior groups. Cluster 2 contains traffic demand where the peak ramps up as plotted in Figure 3.9a, Figure 3.10a. Cluster 1, Cluster 3 contain traffic demands where the peak ramps down or is spread out as plotted in Figure 3.9b, Figure 3.9c, Figure 3.10b, Figure 3.10c.

and Figure A.18). Although we also do not have any a-priori guarantee that Algorithm 3.4 will do a better job at controlling the traffic network, because we have a large collection of Hankel matrices at our disposal and we adapt the Hankel matrix to the current system conditions, there is a much larger likelihood that at least one Hankel matrix in our Hankel matrix library will represent the current system behavior well, and control the system well. Since we allow the Hankel matrix to be switched when the predicted output does not match the measured output, and also switch the Hankel matrix when the system transitions from the morning to the evening peak, we correct for bad system representations online, by switching to the Hankel matrix at our disposal that best represents the observed traffic conditions.

To conclude, Algorithm 3.4 prevents traffic congestion robustly as underlined by the average waiting time per vehicle which is over two times lower than for *No Control* or $\mathcal{H}_{LPV}$. The computational time is also heavily reduced when using Algorithm 3.4 instead of a fixed $\mathcal{H}_{LPV}$. Overall, using Algorithm 3.4 to control the traffic network leads to a large decrease in emitted emissions.

Figure 4.4 shows the traffic density and traffic flow evolution in region 1 for the traffic demand profile in Figure 3.11a. Plotted in blue are the average simulation density and traffic flow results when the urban traffic network was controlled as described in Algorithm 3.4, while the red graph shows the average results of controlling the system with a fixed, concatenated Hankel matrix $\mathcal{H}_{LPV}$, and the black graph shows the results when no controller is used where all the green-red traffic light cycles remain fixed. In this figure we can see from the red graph that control with $\mathcal{H}_{LPV}$ does not prevent the traffic congestion of the second peak, whereas, shown by the blue graph, control with Algorithm 3.4 does, hence being more robust. This is because it may be that $\mathcal{H}_{LPV}$ does not represent the system behavior well across the entire simulation. Algorithm 3.4, however, realizes online if a Hankel matrix does not represent the system behavior well and then switches the Hankel matrix. This leads to using a Hankel matrix that best represents the current traffic conditions and leads to preventing traffic congestion more robustly than with $\mathcal{H}_{LPV}$.

Figure 4.5 shows the traffic density and traffic flow evolution in region 1 for the traffic demand profile in Figure 3.11b. We observe that both the controller with a fixed, concatenated Hankel matrix $\mathcal{H}_{LPV}$ and the controller with switching Hankel matrices as described in Algorithm 3.4

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Computation time [s]	**14.35**± 5.23	90.67 ± 6.30	27.36 ± 0.60
Travel time per vehicle [min]	25.35 ±15.26	18.66 ± 10.04	**10.01** ± 2.09
Waiting time per vehicle [min]	23.69 ± 15.51	16.77 ± 10.16	**8.04** ± 2.12

Table 4.3: Average weekday results with standard deviation. Best metric of each row in bold.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Computation time [s]	**9.16** ± 0.88	93.31 ± 7.82	28.09 ± 0.11
Travel time per vehicle [min]	10.52 ± 1.53	9.58 ± 2.06	**9.06** ± 1.93
Waiting time per vehicle [min]	8.73 ± 1.55	7.77 ± 2.10	**7.24** ± 1.96

Table 4.4: Average weekend results with standard deviation. Best metric of each row in bold.

manage to prevent the traffic gridlock that happens when no controller is used. Therefore, both controllers fulfill the objective of reducing traffic congestion. However, the controller with $\mathcal{H}_{LPV}$ uses nearly three times as much time to calculate the control inputs across the entire simulation, highlighting the increased computational efficiency of switching Hankel matrices.

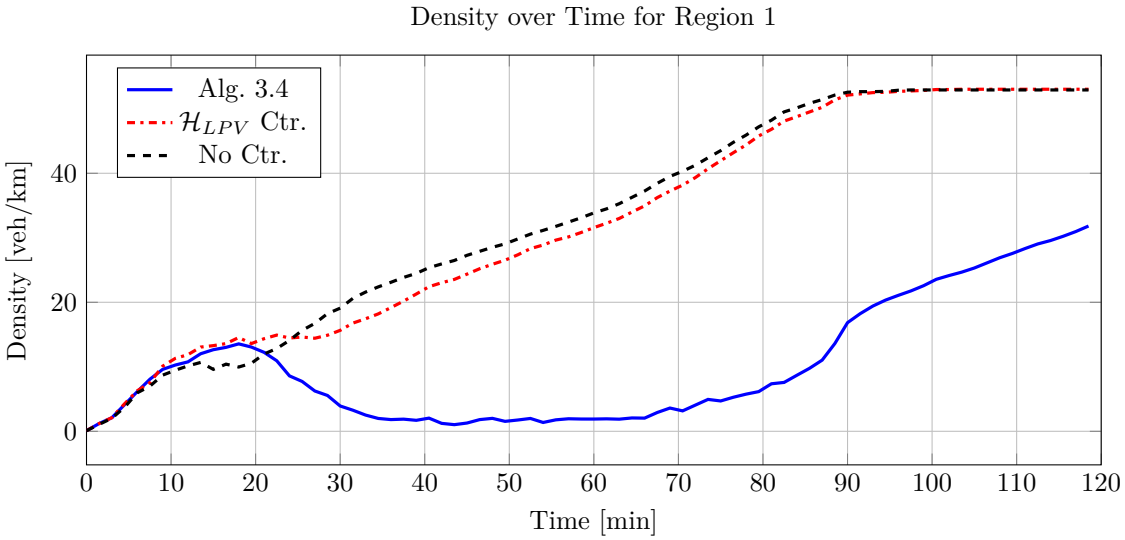
The results of the other weekday traffic demand profiles are plotted and tabulated in Appendix A.2.1.

Weekend results

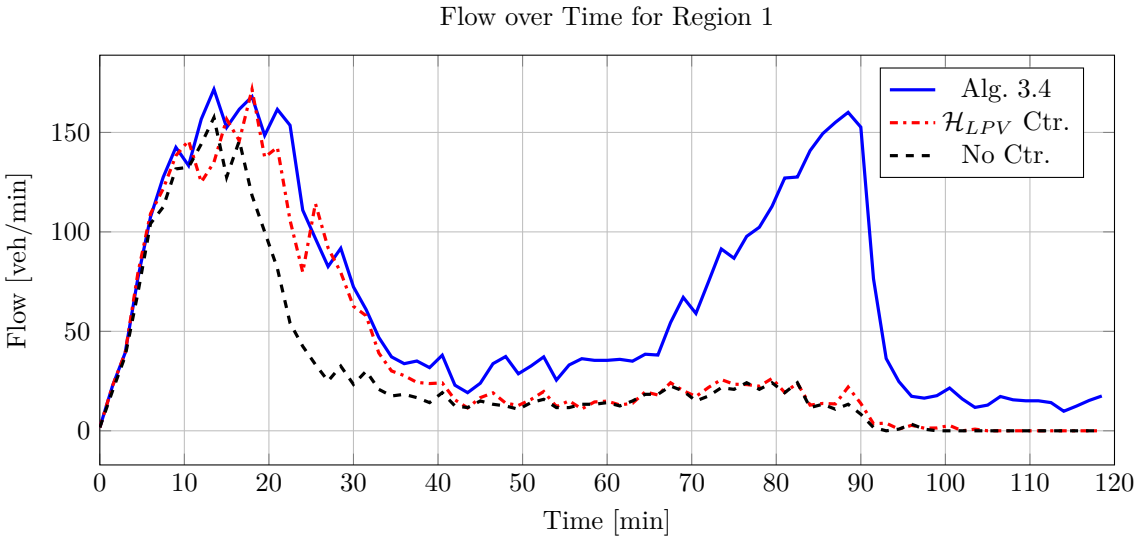
The average weekend results for the demand profiles plotted in Figure 3.12 are captured in Table 4.4.

We observe that Algorithm 3.4 reduces the computational time by over 3 times compared to using a fixed Hankel matrix $\mathcal{H}_{LPV}$. Further, we observe a smaller standard deviation and smaller average value in the travel time per vehicle and waiting time per vehicle metric for Algorithm 3.4 versus $\mathcal{H}_{LPV}$. This indicates that Algorithm 3.4 is computationally more efficient and more robust at preventing traffic congestion than $\mathcal{H}_{LPV}$. The average travel time per vehicle is reduced by nearly 1.5 minutes when using Algorithm 3.4 instead of *No Control*.

To conclude, Algorithm 3.4 prevents traffic congestion robustly and reduces the computational time. The results of all the weekend traffic demand profiles are plotted and tabulated in Appendix A.2.2.

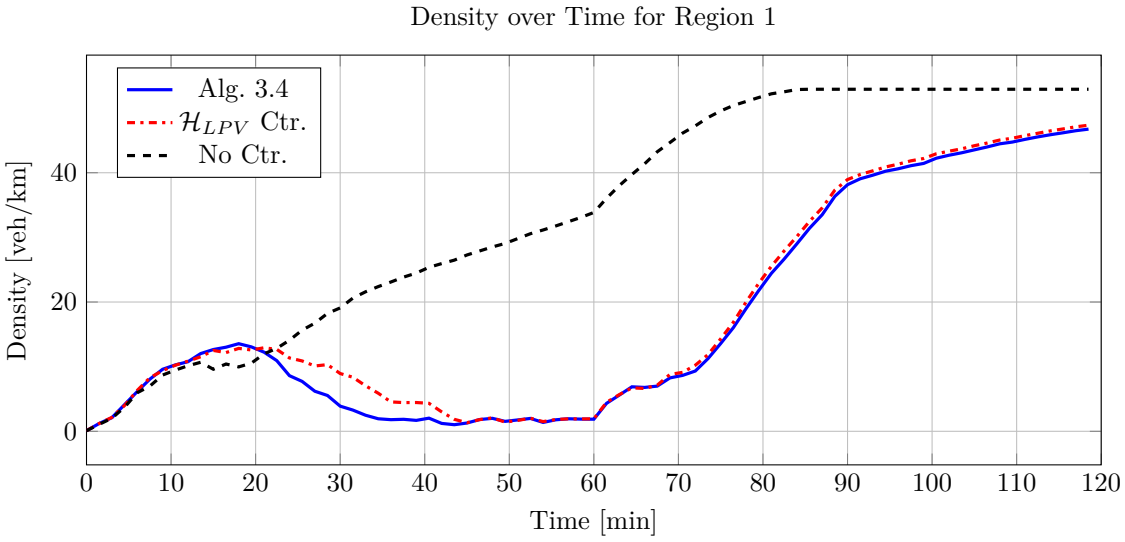


(a) Density for Region 1

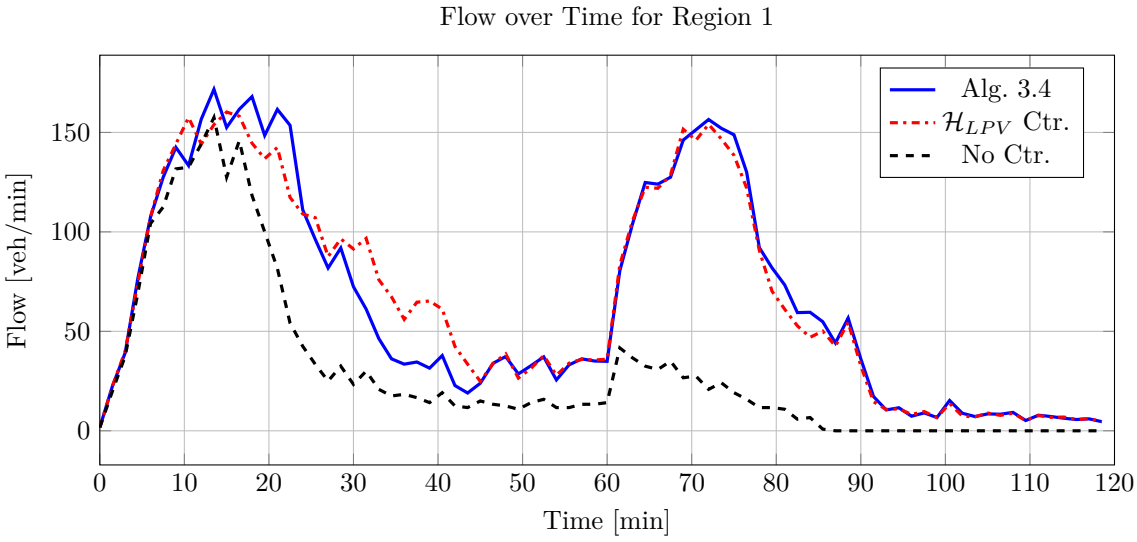


(b) Flow for Region 1

Figure 4.4: Traffic density and flow results for region 1 for a traffic demand that is ramps down in the morning and ramps up in the evening as plotted in Figure 3.11a.



(a) Density for Region 1



(b) Flow for Region 1

Figure 4.5: Traffic density and flow results for region 1 for a traffic demand that ramps down in the morning and is spread out in the evening as plotted in Figure 3.11b.

4.2.4 DeePC urban traffic control results discussion

The simulation results for the weekday and weekend traffic demands show that the average travel time per vehicle can be reduced when dynamically controlling traffic lights with DeePC. Furthermore, the results highlight the advantages of switching the Hankel matrix of the DeePC controller according to Algorithm 3.4 instead of keeping a fixed, concatenated Hankel matrix $\mathcal{H}_{LPV}$. Namely, switching Hankel matrices as explained in Algorithm 3.4 helps to robustify the controller's capability of reducing traffic congestion when the traffic demand varies and strongly reduces the computational time it takes for the controller to calculate control inputs. Due to potential nonlinearities in the urban traffic network, there are neither guarantees that DeePC with $\mathcal{H}_{LPV}$ nor DeePC with switching Hankel matrices as explained in Algorithm 3.4 will perform better at controlling urban traffic networks than using no controller at all. This is because we do not know a-priori whether the Hankel matrix DeePC is using represents our system behavior well. The simulation results show that, on one hand, if we have a Hankel matrix that represents the system behavior well, we can strongly reduce traffic congestion and save emissions. On the other hand, the simulation results suggest that using Algorithm 3.4 strongly robustifies reducing traffic congestion for varying traffic demands. This is because Algorithm 3.4 switches the Hankel matrix when it detects that the Hankel matrix does not represent our system behavior well anymore, leading to operating the DeePC controller with the Hankel matrix at our disposal that best represents the current traffic conditions. Hence, Algorithm 3.4 does not blindly trust a Hankel matrix that is fixed a-priori but instead adapts to the traffic conditions that are measured online.

Chapter 5

Conclusion

As urbanization continues to accelerate, traffic congestion in cities is becoming an increasingly urgent issue. To tackle this problem, data-driven control methodologies such as DeePC can be used to dynamically adjust the green-red traffic light cycle depending on the current traffic conditions.

DeePC is formulated for LTI systems which does not align with urban traffic network systems which are generally nonlinear. To increase the computational efficiency and performance robustness of DeePC in urban traffic control for varying traffic demands, we modeled the urban traffic network as an LPV system and extended DeePC to this class of systems.

We showed that controlling an LPV system with DeePC yields better results when the controller switches among subsystem-specific Hankel matrices instead of keeping a fixed, concatenated Hankel matrix. This is due to the fact that a subsystem-specific Hankel matrix is more powerful at modeling an LTI subsystem than a fixed, LPV-specific Hankel matrix, as it only predicts trajectories that can be produced by subsystem behavior. Hence, the steady-state reference output tracking error is smaller while additionally reducing the computational efforts.

We showed that when the switching times k_{switch} of the LPV system are known, then DeePC and MPC exhibit the identical closed loop behavior for any $t \in [k^i_{switch} + T_{ini}, k^{i+1}_{switch} - N]$. When the switching times are unknown, we proposed using the error between the predicted output of the controller and the measured output of the system as a metric to detect when the LPV system switches subsystems online.

Our control pipelines for LPV systems with known and unknown switching times were combined to control urban traffic as an LPV system. We assumed that we can model the morning and evening peak of the traffic demand with two subsystems, or behavior times, as an LPV system.

Because the traffic network may exhibit nonlinearities, we can not guarantee that the same traffic demand excites the same system behavior, because the system behavior also depends on the control actions of the traffic lights. Therefore, we collected multiple Hankel matrices for each traffic demand, and clustered them into behavior groups with k-means.

Whenever the system switched behavior times, we selected a new Hankel matrix with linear least squares based on the current traffic conditions. Again, because the system is nonlinear, there was no guarantee that we were selecting the best Hankel matrix. Therefore, we additionally compared the predicted system output with the measured output, and switched Hankel matrices if the error grew too strongly.

The simulation results showed that the proposed control approach with switching Hankel matrices was more robust at preventing traffic congestion with varying traffic demands than DeePC with a fixed, concatenated Hankel matrix, and was computationally more efficient. The computational efforts were reduced by over three times, while the average travel time per vehicle was nearly halved for the weekday traffic demand scenarios and reduced by seven percent for the weekend traffic demand scenarios. The main reason for the improved performance and robustness was that the switching Hankel matrix approach did not blindly trust a fixed Hankel matrix to represent our system behavior well, but instead chose a Hankel matrix from a large collection of matrices which best matched the current traffic conditions, and switched the Hankel matrix online when a mismatch between the predicted and measured output was detected.

Compared to keeping the traffic light cycle fixed (hence, not using any controller), our switching Hankel matrix approach more than halved the average travel time per vehicle for the weekday scenario while for the weekend scenario it was reduced by over ten percent in the traffic grid simulations.

To conclude, our DeePC controller with switching Hankel matrices as we proposed, robustly reduced the traffic congestion in urban traffic with varying traffic demands, and was computationally more efficient than when DeePC used a fixed Hankel matrix.

5.1 Future work

Further research should be done on how to control the LPV system during the transient phase when two subsystems switch. It would be interesting to investigate if the tracking during the transient switching phase becomes more robust if we temporarily switch to an LPV-specific Hankel matrix. Another avenue of research could focus on finding a strategy that finds the optimal amount of behavior times, or subsystems, to describe a full day of traffic. One idea could be to create behavior times based on the rate of change of the traffic density instead of the traffic demand shape. A distinction between constant, increasing, or decreasing traffic density may be used to split a day into behavior times. It would also be interesting to define traffic demand based on historical traffic data. This could help determine how many behavior times we expect a day to have and when these expected switches happen. Furthermore, it would be useful to find a method to determine how many behavior cluster groups are good to use for given Hankel matrices. Seeing whether the directionality of traffic flow could be integrated in the clustering step would be interesting too. Different clustering methods, such as subspace clustering algorithms, should be implemented and compared when clustering Hankel matrices. Additionally, other ways of detecting system switches online could be investigated. Perhaps past data could be added to the newly selected Hankel matrix after a switch happens to dampen the transient effects. It would also be interesting to test different Hankel selection methods, and playing with the trajectory length of data that is being fit to the Hankel matrices. Finally, it would be interesting to test the proposed algorithm on a more complex, realistic traffic grid network.

Appendix A

Appendix

A.1 LPV Systems Results

A.1.1 Known Switching Times

Here are the result plots of the five simulation runs controlling an LPV system with known switching times with Algorithm 3.1 versus a concatenated Hankel matrix $\mathcal{H}_{LPV}$.

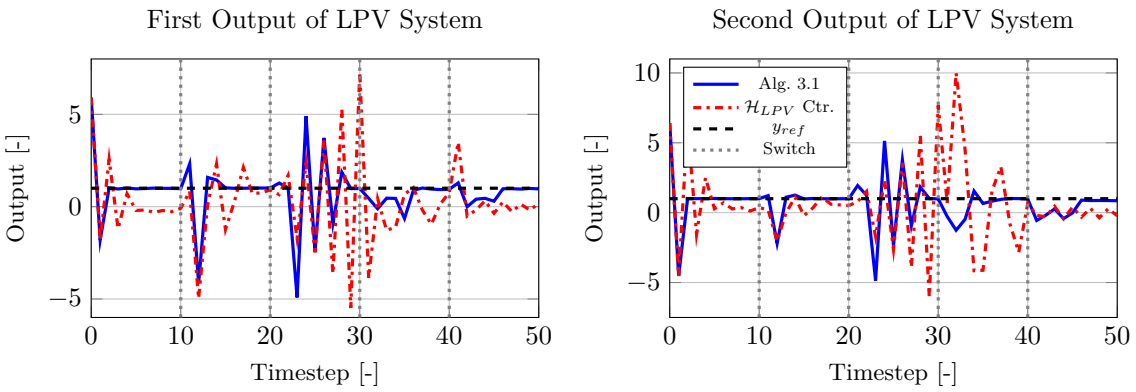


Figure A.1: Reference output tracking of LPV system with known switching times. Switching Hankel matrices with Algorithm 3.1 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.1 produced an average error of $[0.40, 0.38]$ for the first respectively second output, and took 9.9s to compute. The average error for $\mathcal{H}_{LPV}$ was $[1.68, 1.88]$, and took 454.8s to compute.

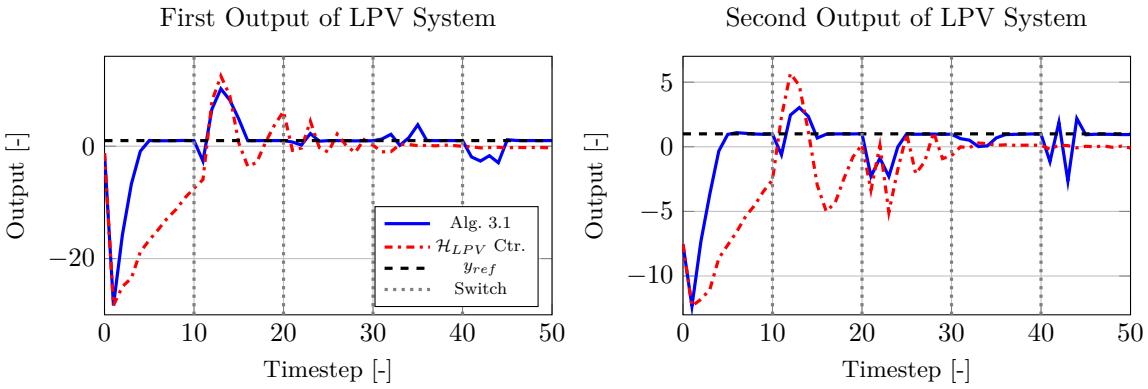


Figure A.2: Reference output tracking of LPV system with known switching times. Switching Hankel matrices with Algorithm 3.1 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.1 produced an average error of $[0.24, 0.05]$ for the first respectively second output, and took 9.4s to compute. The average error for $\mathcal{H}_{LPV}$ was $[3.95, 2.59]$, and took 648.1s to compute.

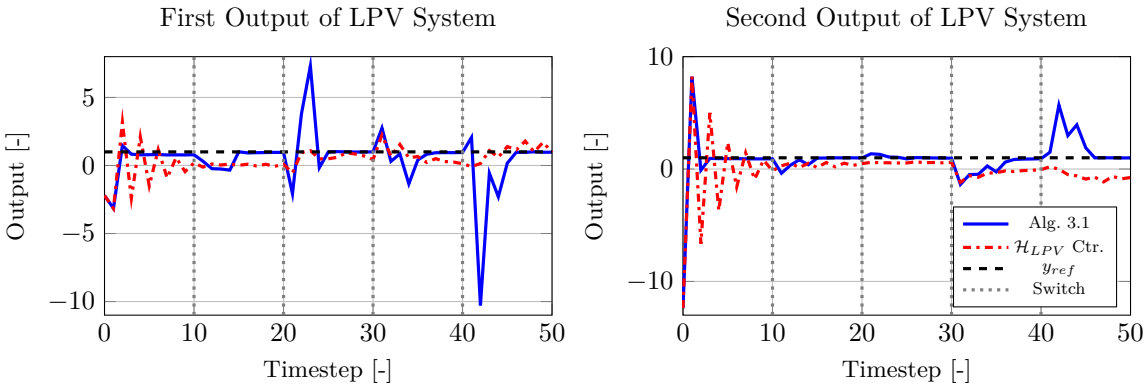


Figure A.3: Reference output tracking of LPV system with known switching times. Switching Hankel matrices with Algorithm 3.1 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.1 produced an average error of $[0.12, 0.13]$ for the first respectively second output, and took 6.7s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.70, 1.07]$, and took 526.4s to compute.

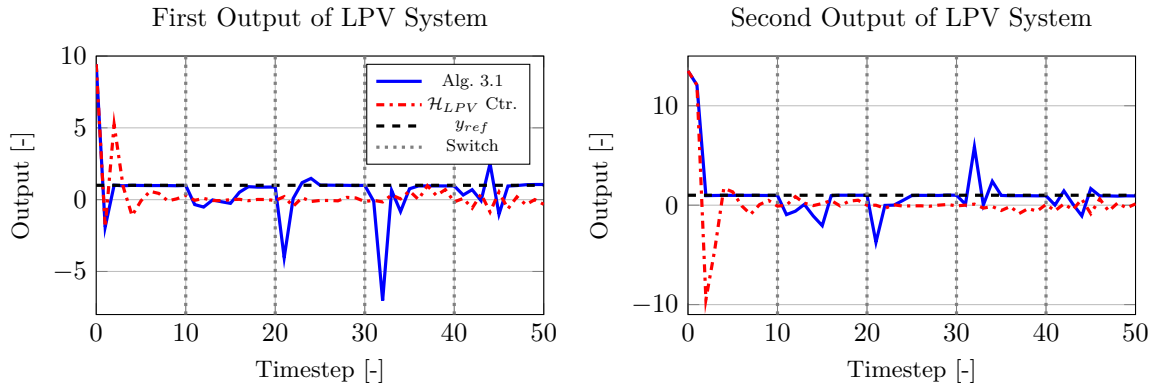


Figure A.4: Reference output tracking of LPV system with known switching times. Switching Hankel matrices with Algorithm 3.1 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.3 produced an average error of $[0.17, 0.15]$ for the first respectively second output, and took 7.5s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.90, 1.06]$, and took 558.5s to compute.

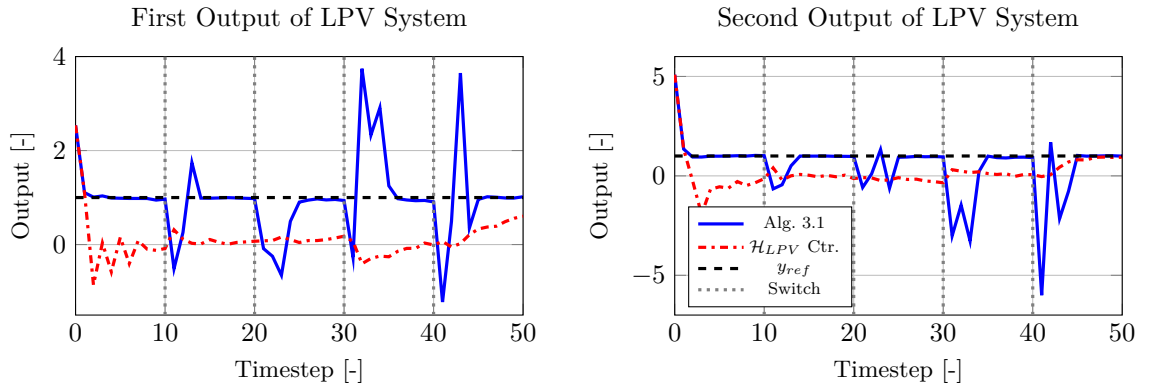


Figure A.5: Reference output tracking of LPV system with known switching times. Switching Hankel matrices with Algorithm 3.1 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.1 produced an average error of $[0.04, 0.03]$ for the first respectively second output, and took 8.8s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.91, 0.94]$, and took 337.8s to compute.

A.1.2 Unknown Switching Times

Here are the result plots of the five simulation runs controlling an LPV system with unknown switching times with Algorithm 3.3 versus a concatenated Hankel matrix $\mathcal{H}_{LPV}$.

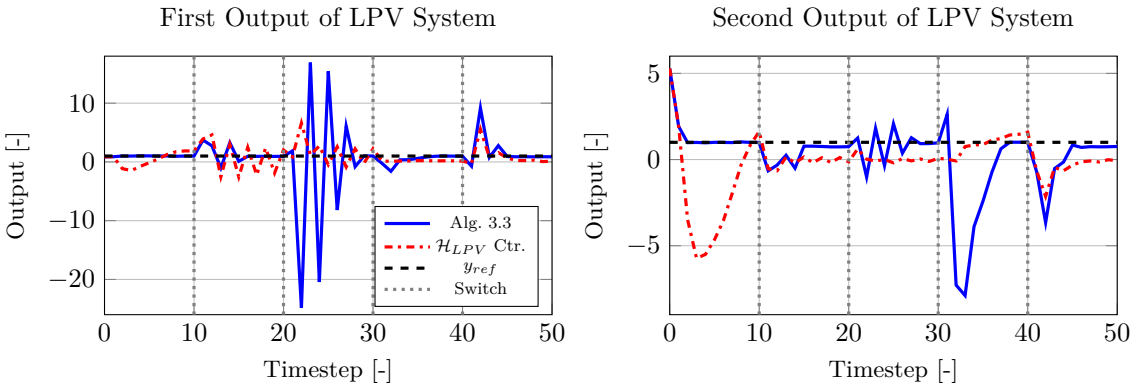


Figure A.6: Reference output tracking of LPV system with unknown switching times. Switching Hankel matrices with Algorithm 3.3 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.3 produced an average error of $[0.26, 0.32]$ for the first respectively second output, and took 8.1s to compute. The average error for $\mathcal{H}_{LPV}$ was $[1.19, 1.21]$, and took 708.3s to compute.

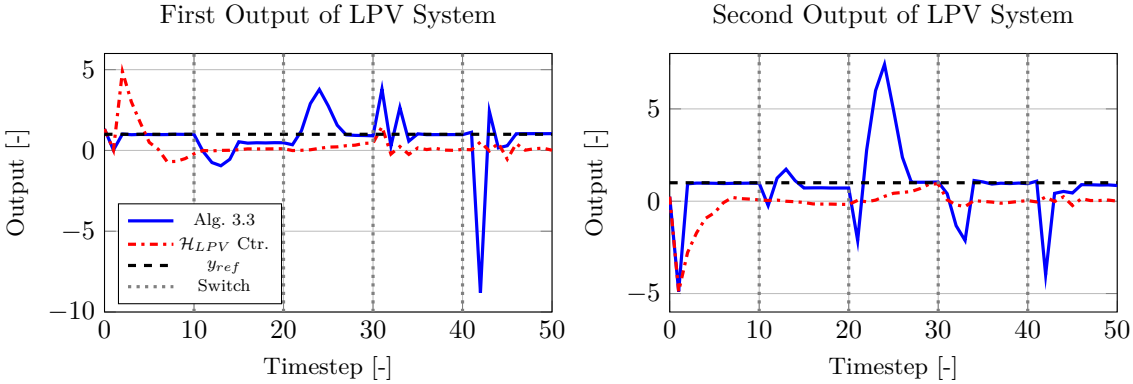


Figure A.7: Reference output tracking of LPV system with unknown switching times. Switching Hankel matrices with Algorithm 3.3 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.3 produced an average error of $[0.28, 0.39]$ for the first respectively second output, and took 7.8s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.95, 0.91]$, and took 564.9s to compute.

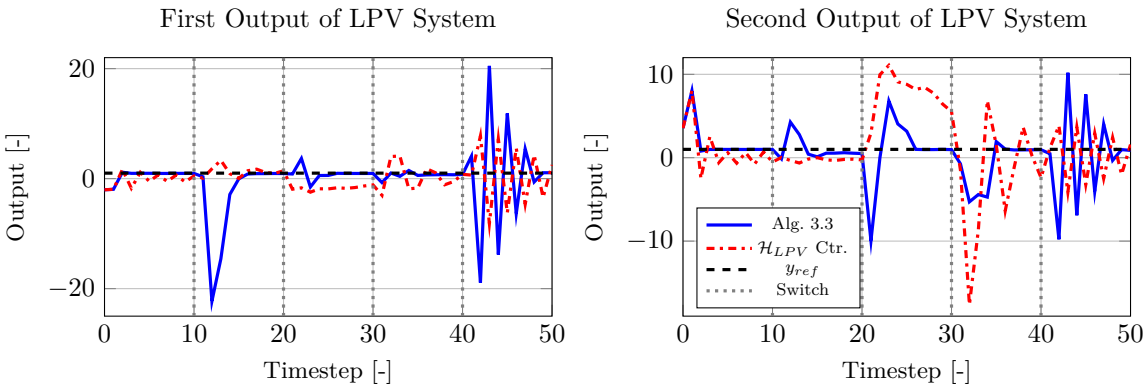


Figure A.8: Reference output tracking of LPV system with unknown switching times. Switching Hankel matrices with Algorithm 3.3 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.3 produced an average error of $[0.30, 0.50]$ for the first respectively second output, and took 10.0s to compute. The average error for $\mathcal{H}_{LPV}$ was $[1.93, 2.80]$, and took 456.4s to compute.

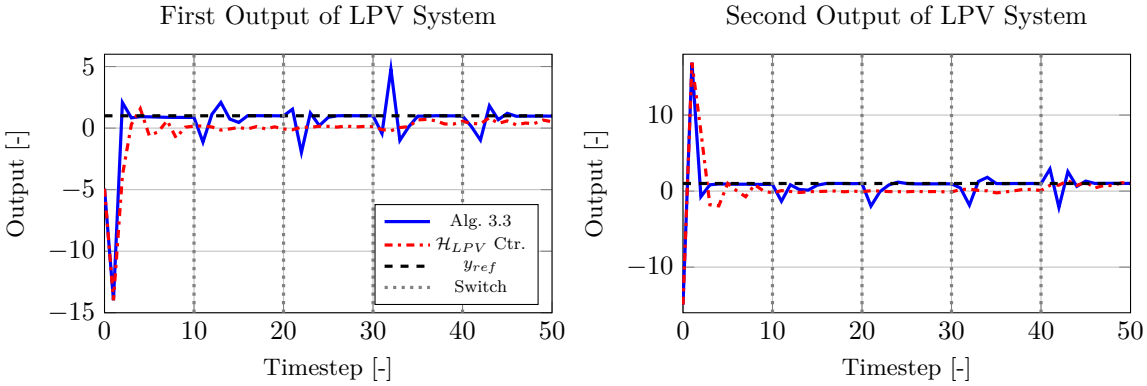


Figure A.9: Reference output tracking of LPV system with unknown switching times. Switching Hankel matrices with Algorithm 3.3 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.3 produced an average error of $[0.16, 0.18]$ for the first respectively second output, and took 8.2s to compute. The average error for $\mathcal{H}_{LPV}$ was $[0.80, 0.83]$, and took 342.3s to compute.

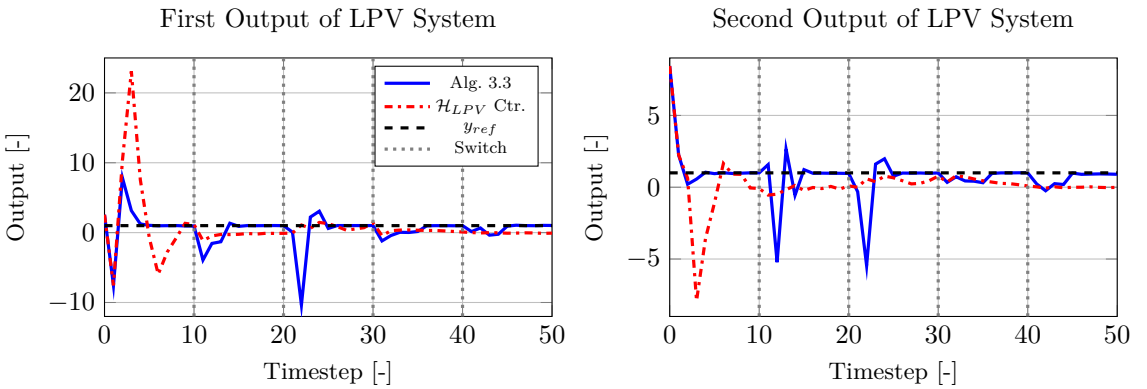


Figure A.10: Reference output tracking of LPV system with unknown switching times. Switching Hankel matrices with Algorithm 3.3 (blue line) versus concatenated Hankel matrix $\mathcal{H}_{LPV}$ (orange line). Algorithm 3.3 produced an average error of $[0.23, 0.20]$ for the first respectively second output, and took 9.5s to compute. The average error for $\mathcal{H}_{LPV}$ was $[1.16, 0.83]$, and took 343.0s to compute.

A.2 DeePC for Traffic Network as an LPV System

A.2.1 Weekday Results

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**20.53**	99.92	27.04
Travel time per vehicle [min]	42.93	38.81	**7.53**
Waiting time per vehicle [min]	41.56	37.16	**5.52**

Table A.1: Traffic demand Figure 3.11a weekday results. Best metric in bold.

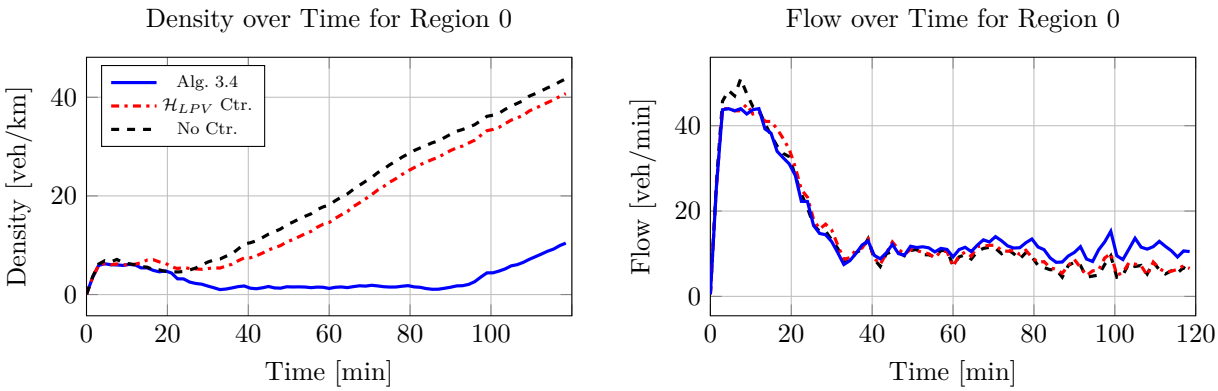


Figure A.11: Traffic density and flow results for region 0 for a traffic demand that ramps down in the morning and ramps up in the evening as plotted in Figure 3.11a.

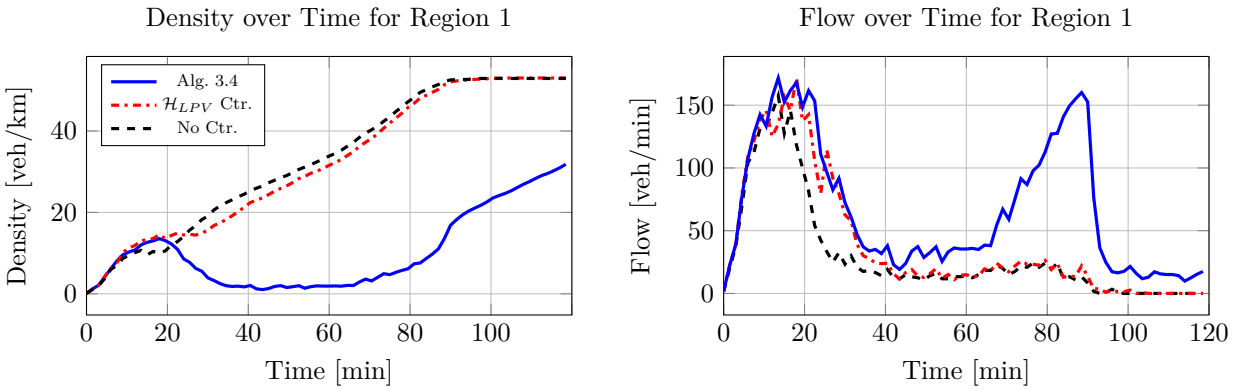


Figure A.12: Traffic density and flow results for region 1 for a traffic demand that ramps down in the morning and ramps up in the evening as plotted in Figure 3.11a.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**20.63**	84.67	28.35
Travel time per vehicle [min]	43.63	12.74	**12.04**
Waiting time per vehicle [min]	42.27	10.81	**10.16**

Table A.2: Traffic demand Figure 3.11b weekday results. Best metric in bold.

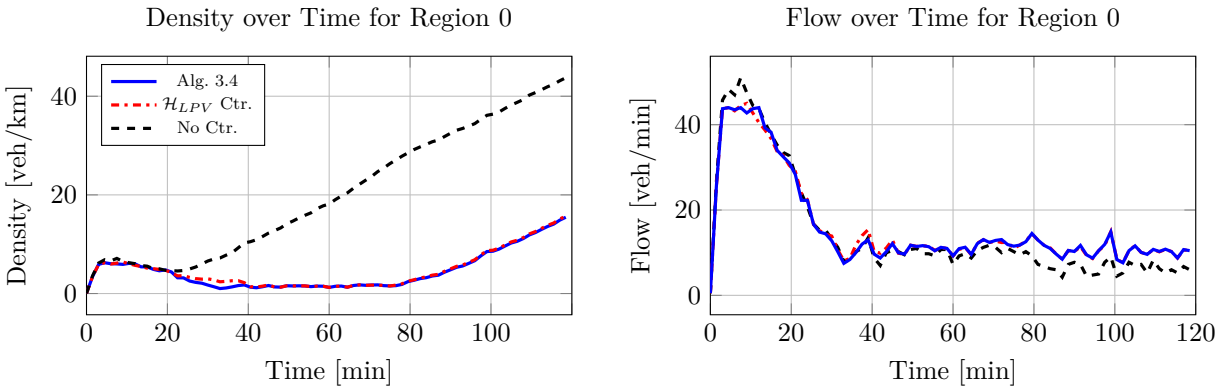


Figure A.13: Traffic density and flow results for region 0 for a traffic demand that ramps down in the morning and is spread out in the evening as plotted in Figure 3.11b.

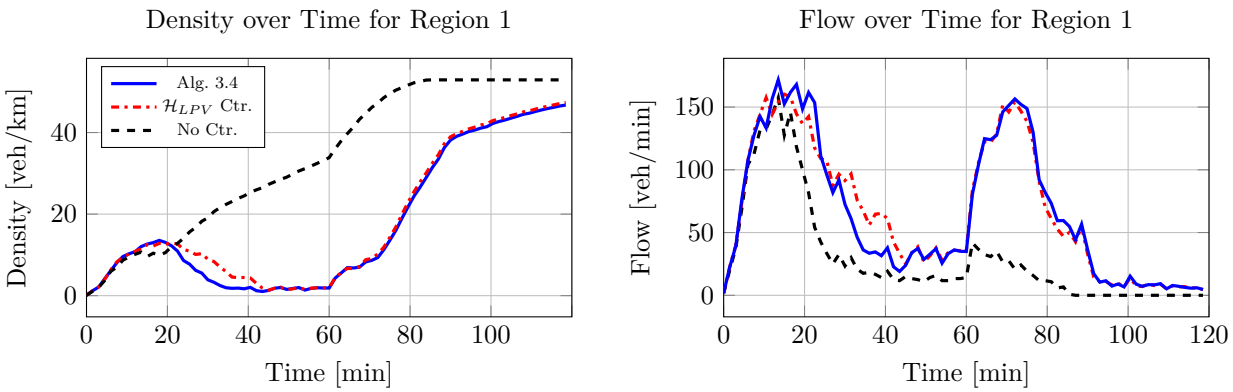


Figure A.14: Traffic density and flow results for region 1 for a traffic demand that ramps down in the morning and is spread out in the evening as plotted in Figure 3.11b.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**17.14**	82.24	27.50
Travel time per vehicle [min]	34.25	**11.44**	12.50
Waiting time per vehicle [min]	32.75	**9.53**	10.54

Table A.3: Traffic demand Figure 3.11c weekday results. Best metric in bold.

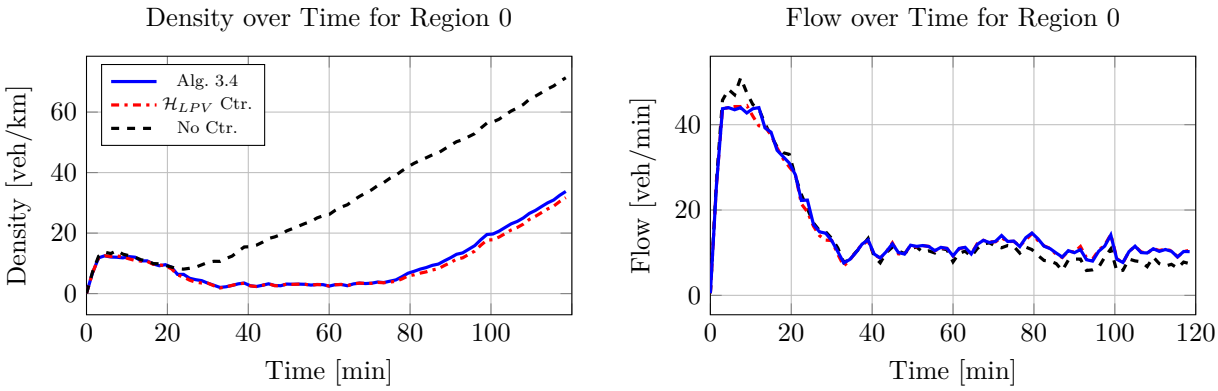


Figure A.15: Traffic density and flow results for region 0 for a traffic demand that ramps down in the morning and ramps down in the evening as plotted in Figure 3.11c.

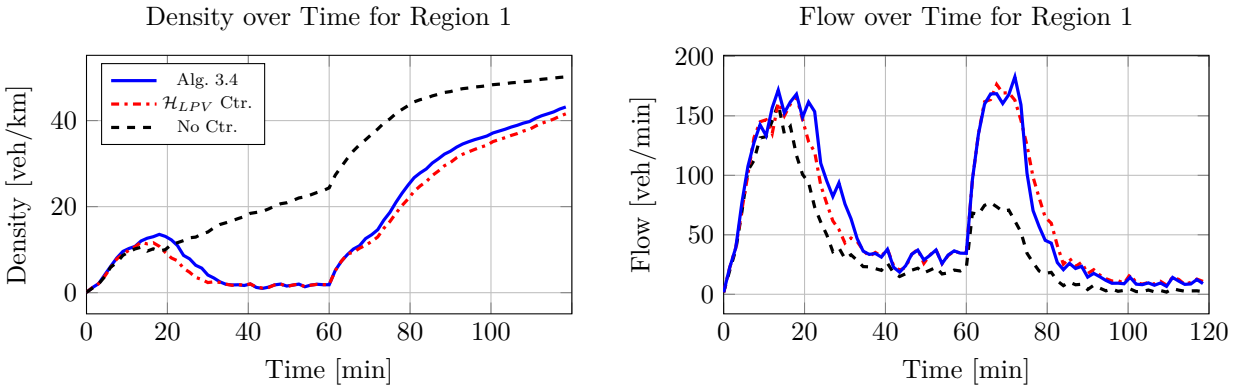


Figure A.16: Traffic density and flow results for region 1 for a traffic demand that ramps down in the morning and ramps down in the evening as plotted in Figure 3.11c.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**8.43**	93.71	26.36
Travel time per vehicle [min]	8.47	24.18	**6.98**
Waiting time per vehicle [min]	6.56	22.39	**4.96**

Table A.4: Traffic demand Figure 3.11d weekday results. Best metric in bold.

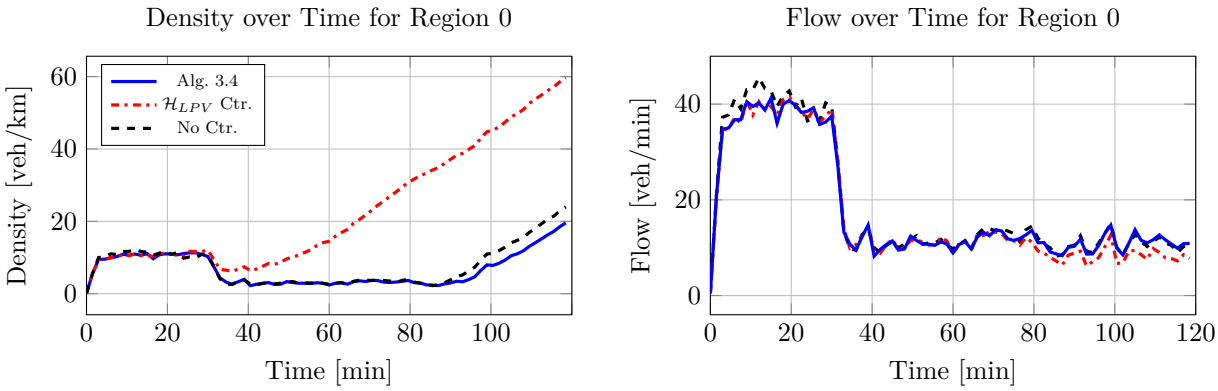


Figure A.17: Traffic density and flow results for region 0 for a traffic demand that is spread out in the morning and ramps up in the evening as plotted in Figure 3.11d.

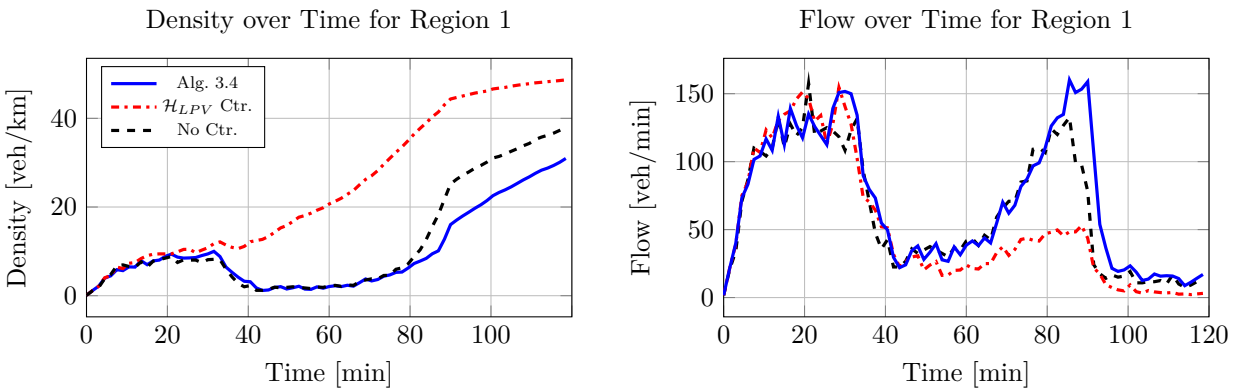


Figure A.18: Traffic density and flow results for region 1 for a traffic demand that is spread out in the morning and ramps up in the evening as plotted in Figure 3.11d.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**9.62**	95.88	27.28
Travel time per vehicle [min]	11.27	11.31	**10.47**
Waiting time per vehicle [min]	9.41	9.40	**8.50**

Table A.5: Traffic demand Figure 3.11e weekday results. Best metric in bold.

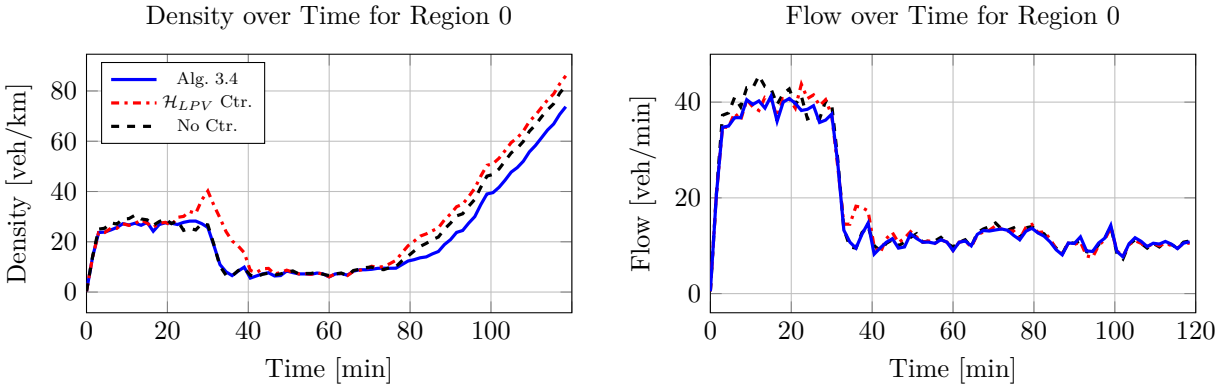


Figure A.19: Traffic density and flow results for region 0 for a traffic demand that is spread out in the morning and is spread out in the evening as plotted in Figure 3.11e.

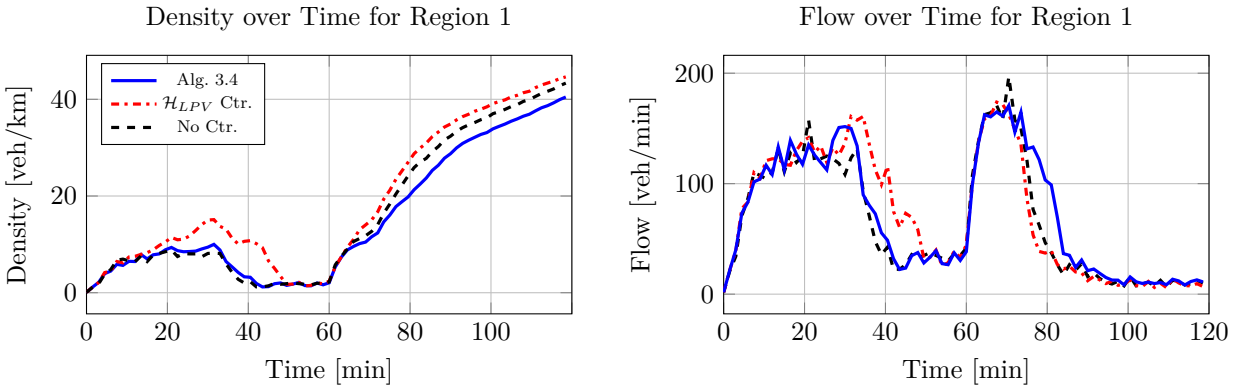


Figure A.20: Traffic density and flow results for region 1 for a traffic demand that is spread out in the morning and is spread out in the evening as plotted in Figure 3.11e.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**9.72**	87.61	27.62
Travel time per vehicle [min]	11.53	13.46	**10.55**
Waiting time per vehicle [min]	9.60	11.34	**8.55**

Table A.6: Traffic demand Figure 3.11f weekday results. Best metric in bold.

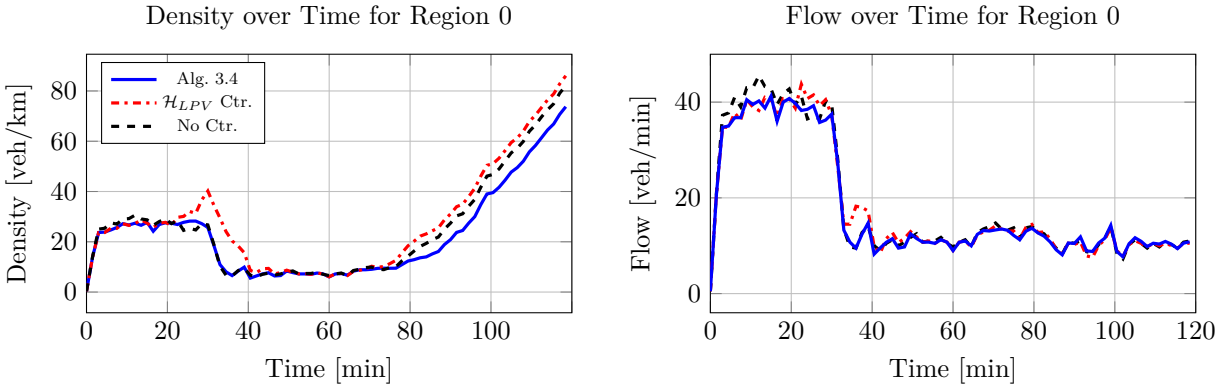


Figure A.21: Traffic density and flow results for region 0 for a traffic demand that is spread out in the morning and ramps down in the evening as plotted in Figure 3.11f.

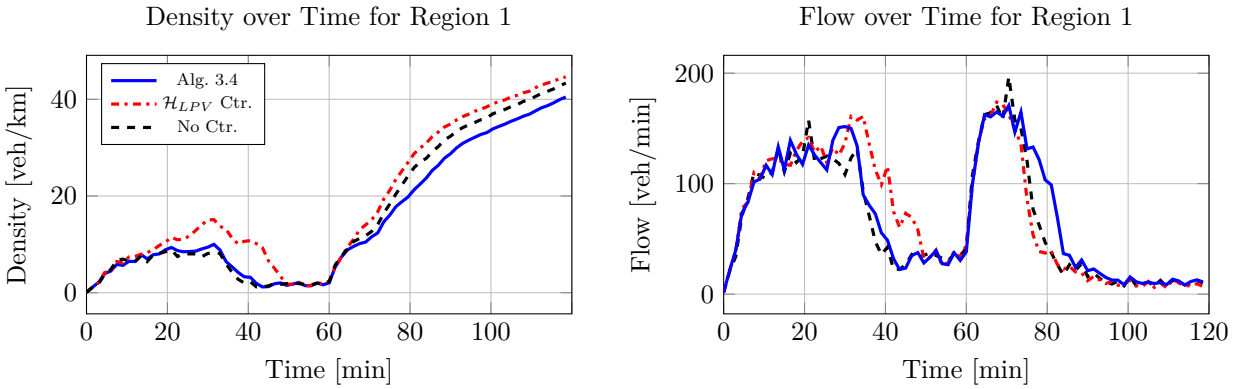


Figure A.22: Traffic density and flow results for region 1 for a traffic demand that is spread out in the morning and ramps down in the evening as plotted in Figure 3.11f.

A.2.2 Weekend Results

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**7.91**	94.62	27.96
Travel time per vehicle [min]	8.35	6.81	**6.33**
Waiting time per vehicle [min]	6.54	4.97	**4.47**

Table A.7: Traffic demand Figure 3.12a weekend results. Best metric in bold.

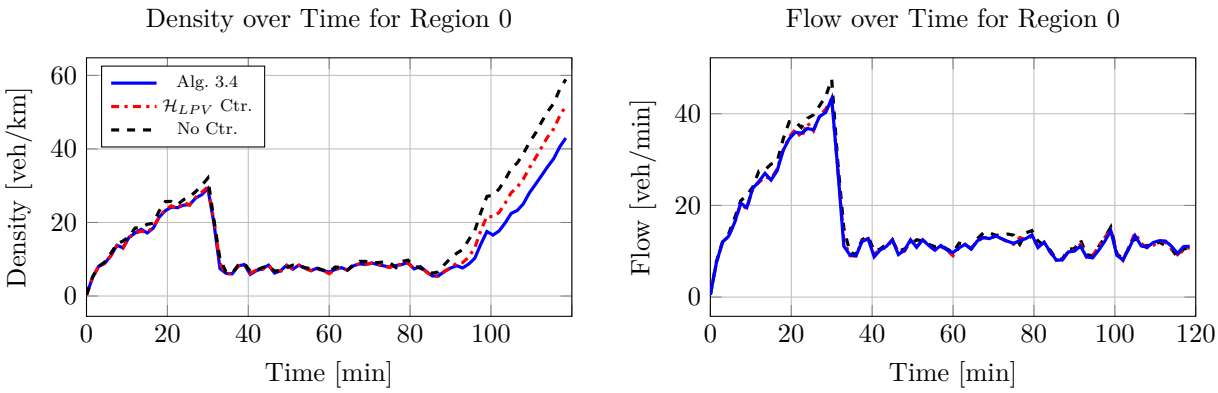


Figure A.23: Traffic density and flow results for region 0 for a traffic demand that ramps up in the morning and evening as plotted in Figure 3.12a.

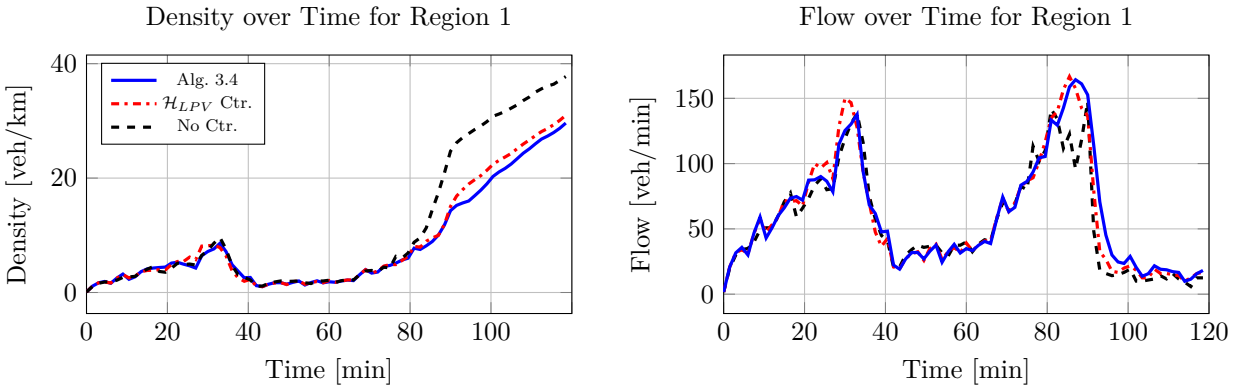


Figure A.24: Traffic density and flow results for region 1 for a traffic demand that ramps up in the morning and evening as plotted in Figure 3.12a.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**9.80**	83.14	28.09
Travel time per vehicle [min]	11.64	11.76	**10.57**
Waiting time per vehicle [min]	9.87	10.02	**8.77**

Table A.8: Traffic demand Figure 3.12b weekend results. Best metric in bold.

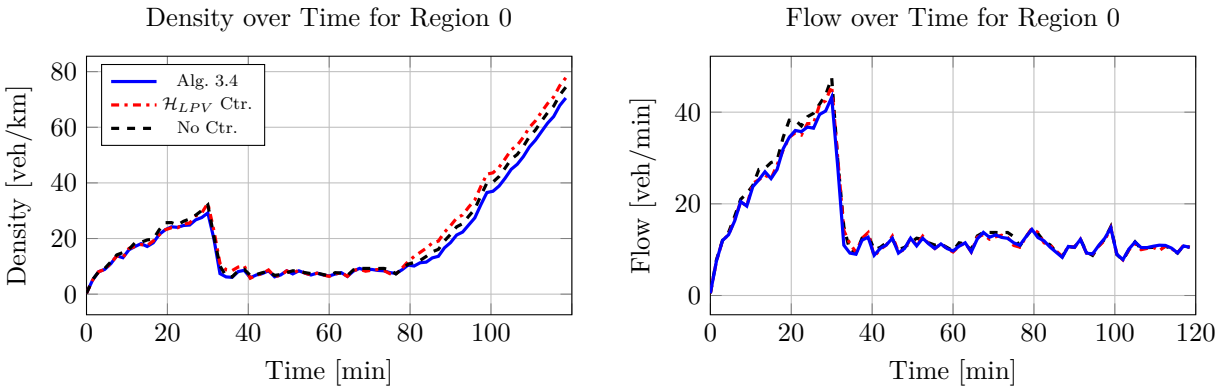


Figure A.25: Traffic density and flow results for region 0 for a traffic demand that ramps up in the morning and is spread out in the evening as plotted in Figure 3.12b.

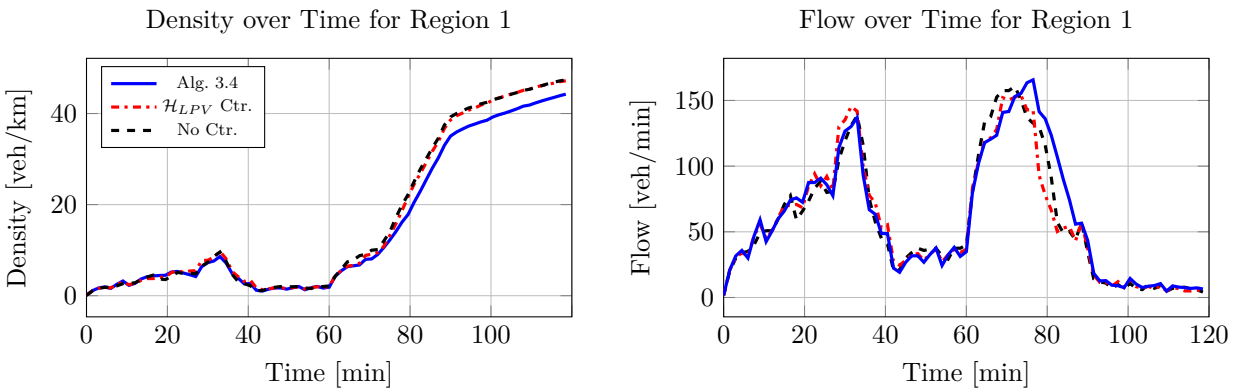


Figure A.26: Traffic density and flow results for region 1 for a traffic demand that ramps up in the morning and is spread out in the evening as plotted in Figure 3.12b.

Metric	No Control	DeePC with fixed $\mathcal{H}_{LPV}$	Algorithm 3.4
Calculation time [s]	**9.77**	102.16	28.22
Travel time per vehicle [min]	11.56	**10.18**	10.28
Waiting time per vehicle [min]	9.79	**8.32**	8.48

Table A.9: Traffic demand Figure 3.12c weekend results. Best metric in bold.

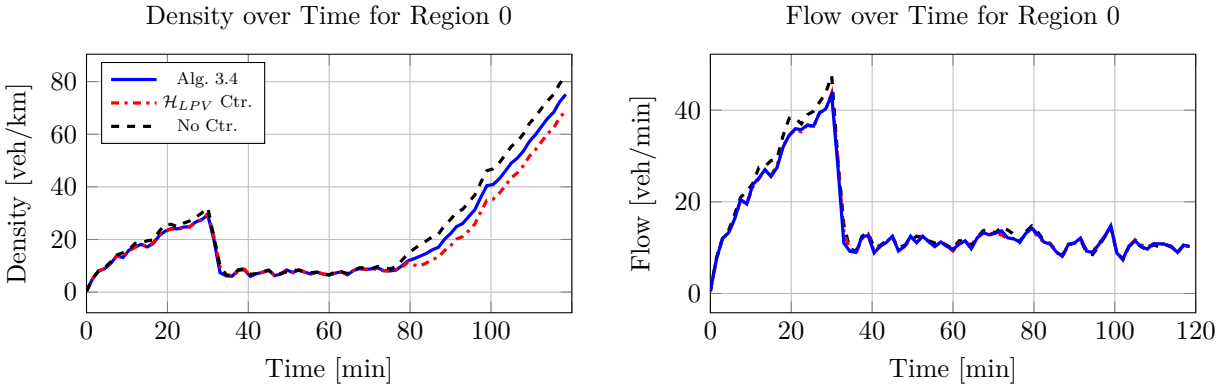


Figure A.27: Traffic density and flow results for region 0 for a traffic demand that ramps up in the morning and ramps down in the evening as plotted in Figure 3.12c.

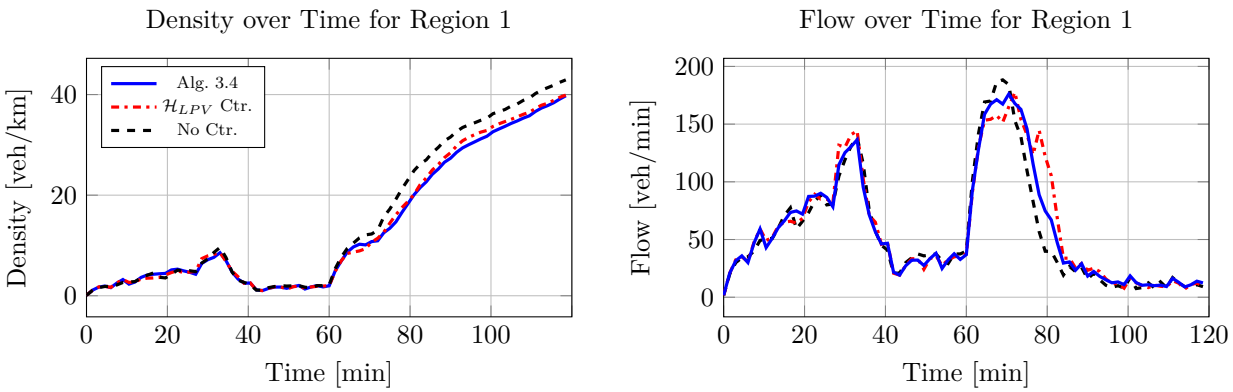


Figure A.28: Traffic density and flow results for region 1 for a traffic demand that ramps up in the morning and ramps down in the evening as plotted in Figure 3.12c.

Bibliography

[1] "World Urbanization Prospects The 2018 Revision."

[2] F. S. Office, "Road vehicles - Stock, level of motorisation." [Online]. Available: https://www.bfs.admin.ch/bfs/en/home/statistiken/mobilitaet-verkehr/verkehrsinfrastruktur-fahrzeuge/fahrzeuge/strassenfahrzeuge-bestand-motorisierungsgrad.html

[3] D. Schrank, L. Albert, B. Eisele, and T. Lomax, "2021 Urban Mobility Report," Tech. Rep., Jun. 2021. [Online]. Available: https://static.tti.tamu.edu/tti.tamu.edu/documents/mobility-report-2021.pdf

[4] K. Zhang and S. Batterman, "Air pollution and health risks due to vehicle traffic," *Science of The Total Environment*, vol. 450-451, pp. 307–316, Apr. 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0048969713001290

[5] "Impact of driving conditions and driving behaviour - ULEV - EC Public Wiki." [Online]. Available: https://wikis.ec.europa.eu/display/ULEV/Impact+of+driving+conditions+and+driving+behaviour

[6] A. Kontses, G. Triantafyllopoulos, L. Ntziachristos, and Z. Samaras, "Particle number (PN) emissions from gasoline, diesel, LPG, CNG and hybrid-electric light-duty vehicles under real-world driving conditions," *Atmospheric Environment*, vol. 222, p. 117126, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1352231019307654

[7] A. Rimoldi, C. Cenedese, A. Padoan, F. Dörfler, and J. Lygeros, "Urban traffic congestion control: a DeePC change," Nov. 2023, arXiv:2311.09851 [cs, eess, math, stat]. [Online]. Available: http://arxiv.org/abs/2311.09851

[8] N. Geroliminis, "Macroscopic modeling of traffic in cities," *January 2007*, Jan. 2007.

[9] N. Geroliminis and C. F. Daganzo, "Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings," *Transportation Research Part B: Methodological*, vol. 42, no. 9, pp. 759–770, Nov. 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261508000180

[10] N. Geroliminis and J. Sun, "Properties of a well-defined macroscopic fundamental diagram for urban traffic," *Transportation Research Part B: Methodological*, vol. 45, no. 3, pp. 605–617, Mar. 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261510001372

[11] M. Saeedmanesh and N. Geroliminis, "Clustering of heterogeneous networks with directional flows based on "Snake" similarities," *Transportation Research Part B: Methodological*, vol. 91, pp. 250–269, Sep. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261515302605

[12] A. Kouvelas, M. Saeedmanesh, and N. Geroliminis, "A Linear Formulation for Model Predictive Perimeter Traffic Control in Cities," *July 2017*, Jul. 2017.

[13] I. I. Sirmatel and N. Geroliminis, "Stabilization of city-scale road traffic networks via macroscopic fundamental diagram-based model predictive perimeter control," *Control Engineering Practice*, vol. 109, p. 104750, Apr. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066121000277

[14] J. Haddad, M. Ramezani, and N. Geroliminis, "Model predictive perimeter control for urban areas with macroscopic fundamental diagrams," in *2012 American Control Conference (ACC)*, Jun. 2012, pp. 5757–5762, iSSN: 2378-5861. [Online]. Available: https://ieeexplore.ieee.org/document/6314693

[15] J. Haddad and A. Shraiber, "Robust perimeter control design for an urban region," *Transportation Research Part B: Methodological*, vol. 68, pp. 315–332, Oct. 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261514001179

[16] M. Keyvan-Ekbatani, A. Kouvelas, I. Papamichail, and M. Papageorgiou, "Exploiting the fundamental diagram of urban networks for feedback-based gating," *Transportation Research Part B: Methodological*, vol. 46, no. 10, pp. 1393–1403, Dec. 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261512000926

[17] F. Dorfler, J. Coulson, and I. Markovsky, "Bridging Direct and Indirect Data-Driven Control Formulations via Regularizations and Relaxations," *IEEE Transactions on Automatic Control*, vol. 68, no. 2, pp. 883–897, Feb. 2023. [Online]. Available: https://ieeexplore.ieee.org/document/9705109/

[18] I. Markovsky and F. Dörfler, "Behavioral systems theory in data-driven analysis, signal processing, and control," *Annual Reviews in Control*, vol. 52, pp. 42–64, Jan. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1367578821000754

[19] J. Coulson, J. Lygeros, and F. Dörfler, "Data-Enabled Predictive Control: In the Shallows of the DeePC," Mar. 2019, arXiv:1811.05890 [math]. [Online]. Available: http://arxiv.org/abs/1811.05890

[20] I. Markovsky, J. C. Willems, S. Van Huffel, and B. De Moor, *Exact and Approximate Modeling of Linear Systems*. Society for Industrial and Applied Mathematics, 2006. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9780898718263

[21] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. M. De Moor, "A note on persistency of excitation," *Systems & Control Letters*, vol. 54, no. 4, pp. 325–329, Apr. 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167691104001434

[22] P. Zhu, G. Ferrari-Trecate, and N. Geroliminis, "Data-enabled Predictive Control for Empty Vehicle Rebalancing," in *2023 European Control Conference (ECC)*, Jun. 2023, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/10178140

[23] J. Wang, Y. Zheng, K. Li, and Q. Xu, "DeeP-LCC: Data-EnablEd Predictive Leading Cruise Control in Mixed Traffic Flow," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 6, pp. 2760–2776, Nov. 2023, arXiv:2203.10639 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2203.10639

[24] J. Wang, Y. Zheng, J. Dong, C. Chen, M. Cai, K. Li, and Q. Xu, "Implementation and Experimental Validation of Data-Driven Predictive Control for Dissipating Stop-and-Go Waves in Mixed Traffic," Nov. 2022, arXiv:2204.03747 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2204.03747

[25] R. Tóth, *Modeling and Identification of Linear Parameter-Varying Systems*, ser. Lecture Notes in Control and Information Sciences, M. Morari and M. Thoma, Eds. Berlin, Heidelberg: Springer, 2010, vol. 403. [Online]. Available: http://link.springer.com/10.1007/978-3-642-13812-6

[26] C. Verhoek, R. Tóth, and H. S. Abbas, "Direct Data-Driven State-Feedback Control of Linear Parameter-Varying Systems," Nov. 2023, arXiv:2211.17182 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2211.17182

[27] C. Verhoek, J. Berberich, S. Haesaert, R. Tóth, and H. S. Abbas, "A Linear Parameter-Varying Approach to Data Predictive Control," Nov. 2023, arXiv:2311.07140 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2311.07140

[28] C. Verhoek, H. S. Abbas, R. Tóth, and S. Haesaert, "Data-Driven Predictive Control for Linear Parameter-Varying Systems," *IFAC-PapersOnLine*, vol. 54, no. 8, pp. 101–108, 2021, arXiv:2103.16160 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2103.16160

[29] C. Verhoek, R. Tóth, S. Haesaert, and A. Koch, "Fundamental Lemma for Data-Driven Analysis of Linear Parameter-Varying Systems," in *2021 60th IEEE Conference on Decision and Control (CDC)*, Dec. 2021, pp. 5040–5046, arXiv:2103.16171 [cs, eess, math]. [Online]. Available: http://arxiv.org/abs/2103.16171

[30] L. Schmitt, J. Beerwerth, and D. Abel, "Data selection and data-enabled predictive control for a fuel cell system," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 4436–4441, Jan. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896323022449

[31] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. University of California Press, Jan. 1967, vol. 5.1, pp. 281–298. [Online]. Available: https://projecteuclid.org/ebooks/berkeley-symposium-on-mathematical-statistics-and-probability/Proceedings-of-the-Fifth-Berkeley-Symposium-on-Mathematical-Statistics-and/chapter/Some-methods-for-classification-and-analysis-of-multivariate-observations/bsmsp/1200512992

[32] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982, conference Name: IEEE Transactions on Information Theory. [Online]. Available: https://ieeexplore.ieee.org/document/1056489

[33] A.-M. Legendre, *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805.

[34] C. F. Gauss, *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*. Friedrich Perthes and I.H. Besser, 1809.

[35] J. W. Polderman and J. C. Willems, *Introduction to Mathematical Systems Theory*, ser. Texts in Applied Mathematics, J. E. Marsden, L. Sirovich, M. Golubitsky, W. Jäger, and F. John, Eds. New York, NY: Springer, 1998, vol. 26. [Online]. Available: http://link.springer.com/10.1007/978-1-4757-2953-5

[36] A. Padoan, J. Coulson, and F. Dörfler, "Controller Implementability: A Data-Driven Approach," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, Dec. 2023, pp. 6098–6103, iSSN: 2576-2370. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10384203

[37] A. Padoan, J. Coulson, H. J. van Waarde, J. Lygeros, and F. Dörfler, "Behavioral uncertainty quantification for data-driven control," Apr. 2022, arXiv:2204.02671 [cs, eess, math]. [Online]. Available: http://arxiv.org/abs/2204.02671

[38] K. Ogata, *Modern Control Engineering*, 4th ed. USA: Prentice Hall PTR, 2001.

[39] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, ser. SODA '07. USA: Society for Industrial and Applied Mathematics, Jan. 2007, pp. 1027–1035.

[40] C. Grunau, A. A. Özüdoğru, V. Rozhoň, and J. Tětek, "A Nearly Tight Analysis of Greedy k-means++," Jul. 2022, arXiv:2207.07949 [cs]. [Online]. Available: http://arxiv.org/abs/2207.07949

[41] G. H. Golub and C. F. V. Loan, "An analysis of the total least squares problem," *SIAM Journal on Numerical Analysis*, vol. 17, no. 6, pp. 883–893, 1980. [Online]. Available: http://www.jstor.org/stable/2156807

[42] J. C. Willems, "From time series to linear system—Part II. Exact modelling," *Automatica*, vol. 22, no. 6, pp. 675–694, Nov. 1986. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0005109886900051

[43] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[44] "Linear Regression Models," in *Statistical Models and Methods for Financial Markets*, T. L. Lai and H. Xing, Eds. New York, NY: Springer, 2008, pp. 3–35. [Online]. Available: https://doi.org/10.1007/978-0-387-77827-3_1

[45] P. Alvarez Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, November 2018, pp. 2575–2582. [Online]. Available: https://elib.dlr.de/127994/

ETH zürich

Automatic Control Laboratory

Prof. Dr. J. Lygeros, Prof. Dr. F. Dörfler, Prof. Dr. R. S. Smith

Title of work:

Data preconditioning for data-driven predictive control in urban traffic control

Thesis type and date:

Master's Thesis, 7. 10. 2024

Supervision:

Alessio Rimoldi
Dr. Carlo Cenedese
Dr. Alberto Padoan
Prof. Dr. John Lygeros

Student:

Name:	Philippe Brigger
E-mail:	briggerp@student.ethz.ch
Legi-Nr.:	18-927-822
Semester:	HS 2024

Statement regarding plagiarism:

By signing this statement, I affirm that I have read and signed the Declaration of Originality, independently produced this paper, and adhered to the general practice of source citation in this subject-area.

Declaration of Originality:

`https://www.ethz.ch/content/dam/ethz/main/education/rechtliches-abschluesse/leistungskontrollen/declaration-originality.pdf`

Zurich, 7.10.2024: P. Brigger