Fixing traffic in Zurich: a data-driven control approach

Fixing traffic in Zürich: a data-driven control approach

A. Rimoldi, TBD [*], ...

October 6, 2023

Abstract

Perimeter control based on the Macroscopic Fundamental Diagram is a well established paradigm in the field of urban traffic control. The optimization problem aims at maximizing the flow in the city regions by bringing the densities near the critical point of the MFD. Several algorithms have been developed to solve this problem, most of them relying on model based predictive control. These approaches work well both in congested and uncongested settings, however they come at the cost of expensive modeling efforts especially when dealing with large scale systems such as a city. In order to solve this problem we apply a novel data driven algorithm (DeePC) to a real scale simulation of the city of Zürich, we test this algorithm against linear model predictive control (LMPC) obtaining better performance with no modeling effort.

1 Introduction

Traffic congestion in cities is a major problem for many aspects of society, as cities continue to grow infrastructural solutions become more and more unfeasible. High traffic densities lead to great losses due to time and fuel inefficiencies, which in turn causes economical, environmental and health related damages.
As the climate crisis worsen, a call is made upon the cities to become greener and limit the unnecessary time spent in congested situations to a minimum. A solution to this problem is represented by traffic control strategies, these approaches can influence in very different ways the city for example by using traffic lights, dynamical speed limits or connected and automated vehicles (CAVs) as actuators.
In particular perimeter flow control has established itself as a valuable control strategy to lower urban congestion. In this framework a city is divided in several regions and the problem becomes to find the optimal flows between these areas in order to achieve maximum flow.
The state of the art algorithm used to solve this problem is Liner Model Predictive Control (LMPC) [1], the algorithm successfully solves the problem and its capable of lowering travel time and emissions both in uncongested and congested conditions. One limitations of model based control approaches is that the

[*]A. Rimoldi is with the Automatic Control Laboratory, ETH Zürich, Zürich, Switzerland (Email:alessio.rimoldi.work@gmail.com).

estimation of the model can be very expensive, both economically and time wise, when dealing with large dynamical systems such as a city.
In this regard we apply a novel data-driven control algorithm, Data-Enabled Predictive Control (DeePC) and in the process get rid of the model estimation process and achieve new state of the art performances. The two algorithms are evaluated against a No Control policy on a real-scale simulation of the city of Zürich and compared with respect to: average time spent in the network, average waiting time and various emissions metrics including CO2.
Contributions: The main contributions of the paper are four:

(i) Achieve a high precision road-to-road clustering on the city of Zürich;

(ii) Test perimeter control strategies on the largest simulation in the literature to the best of our knowledge;

(iii) Test perimeter control strategies on existing simulated infrastructure;

(iv) Achieve new state of the art performance using DeePC while lowering modelling costs.

Related work: Data-driven predictive control for LTI systems has been studied in [2]. The regularizations terms used in the DeePC cost function have been studied in [3]. Multiple approaches have been explored in the context of urban traffic optimization, the classical approach would be to use model based controllers, some examples of this can be seen in [1] [4] [5] [6]. On the side of data driven solutions we have several different methodologies which can been seen in [7] [8], most notably a distributed implementation of DeePC using CAVs to control urban traffic [9].There also exists a data-driven MPC formulation [10]. Another example of end-to-end data-driven solution is represented by reinforcement learning which has attracted attention in the recent past in urban traffic control applications [11] [12], these however lack interpretability and especially with deep reinforcement learning can lead to a black box type approaches, DeePC improves on this being a gray-box algorithm. The Macroscopic Fundamental Diagram and its properties can be viewed in [13] [14] [15]. The clustering algorithm used to obtain the regions in this paper is explained in detail in [16].

Organization: The remainder of the paper is organised as follows. Section 2 lays the general view of a city as a framework for the rest of the paper, the non linear dynamics of traffic are also presented here. Later this section provides basic definitions regarding behavioral systems, Willems fundamental lemma, the DeePC algorithm, the Macroscopic Fundamental Diagram and the general form of the optimization problem at hand. Section 3 is concerned with explaining the Simulation setup we used to test the algorithm, this includes the clustering procedure used to retrieve the regions, the estimated Macroscopic Fundamental Diagrams extracted from the data, the choice of the actuators and the linear model predictive control formulation used to as a comparison for DeePC. Section 4 we provide the results of the algortihms against a no control law baseline, we confront the approaches on several metrics including average travel time, waiting time, CO2 emissions amongst others. Section 5 provides a summary of the main results and an outlook to future research directions. Appendix A provides additional material such as the comparison of the clustering obtained at different times of the simulations.

Notation: Standard mathematical notation is used. $\mathbb{N}$ and $\mathbb{Z}_+$ denote the set of positive integer numbers and the set of non-negative integer numbers, respectively. $\mathbb{R}$, $\mathbb{R}^n$ and $\mathbb{R}^{p\times m}$ denote the set of real numbers, the set of n-dimensional vectors with real entries, and the set of $p \times m$-dimensional matrices with real entries, respectively. M^T, $\operatorname{image} M$ and $\ker M$ denote the transpose, the image and the kernel of the matrix $M \in \mathbb{R}^{p\times m}$, respectively. Map, function, and operator are used synonymously. A map f from X to Y is denoted by $f : X \to Y$; $(Y)^X$ denotes the collection of all such maps. The restriction of $f : X \to Y$ to a subset $X' \subset X$ is denoted by $f|_{X'}$ and is defined by $f|_{X'}(x)$ for $x \in X'$. If $\mathscr{F} \subset (Y)^X$, then $\mathscr{F}|_{X'}$ denotes $\{f|_{X'} : f \in \mathscr{F}\}$.

2 Methodology

Let us consider a sequence of state observations x from a dynamical non-linear system and a sequence of outputs y corresponding to the outputs, assume that we have also a sequence of inputs u given to the system. In the past this variables have been used to create models of the system such as the one later used in 3.5. The classical state/space representation has proved effective and incredibly useful in control applications, however it comes at the cost of having to perform expensive system identification procedure to be used. For this reason we now present a novel effective approach which allows us to control highly non-linear systems in a data-driven fashion. Our dynamical system of choice are the traffic flows in a city, as described in the following section.

2.1 The general view of a city

A city is nothing more then an aggregation of infrastructure and the only ones that bear interests for us in this discussion are the roads and the intersections between them. A city can often be partitioned in several areas based on similarities in traffic density and overall population, we refer to these areas as the *regions* throughout this work.
Each road belonging to a city region can be described by two quantities, the traffic density ρ_{road} and the traffic flow ϕ_{road}.
By averaging the densities and outputs of the roads over the regions, we obtain the regional densities $\boldsymbol{\rho}$ and flows $\boldsymbol{\phi}$, which can be identified as the output $\boldsymbol{y}$ and underlying state $\boldsymbol{x}$ vectors in our optimization problem. The intersections can act as the actuators of our system, to which we pass our input $\boldsymbol{u}$ in an attempt to reach a desirable state.

More formally a city can be formalized of as a graph $\mathscr{G} = (V, E)$, where E is a set of edges (roads) and V is a set of nodes (intersections), we assume it is observed for a time interval $[0, \dots, T]$. To each edge $i \in E$ are associated two local discrete functions, a *density function* $\rho_i(t)$ and a *flow function* $\phi_i(t)$ defined for every $t \in [0, \dots, T]$, note that not all these functions are necessarily known, *i.e*, not all the streets in the network are necessarily monitored.
In the same way, not all the intersections of the network can be used as actuators in our control problem as they might not be traffic lights (ex. priority intersection or may simply lack the infrastructure for the control). Therefore we select the subset $M \subseteq V$ with cardinality m of intersections suitable for control, for each

element of M we specify a discrete input function $u_i(t)$, this function represents the percentage of green time assigned to i-th element of M at time t, we define the vector of the inputs as $\boldsymbol{u}(t) = (u_1(t), u_2(t), \dots, u_m(t))$. We partition the network into a set of regions R with cardinality n each composed by a subset of edges $E_i \subset E$.
The partitioning should satisfy the following conditions:

- each regions $r_i \in R$ in the network should be *homogeneous*, *i.e.*, the variance of the means of the functions $\rho_i(t)$ over the observation interval $[0, \dots, T]$ for the edges in r_i should be small [16];
- each region $r_i \in R$ should be *locally connected*, *i.e.*, r_i should have a unique connected component.

By satisfying these conditions we obtain a partitioning in which each region r_i satisfies the conditions necessary to produce a low scatter Macroscopic Fundamental Diagram (MFD), therefore we can attach to each region r_i its MFD G_i. We now fix an instant $t \in [0, \dots, T]$ and consider the i-th region r_i, we can define the density function of the region $\rho_i(t)$ as the mean of the local densities $\rho_r oad(t)$ on E_i, the flow function of the region $\phi_i(t)$ can be defined in the same way. We denote by $\boldsymbol{\rho}(t)$ the vector of the regional densities at time t, *i.e.*, $\boldsymbol{\rho}(t) = (\rho_1(t), \rho_2(t), \dots, \rho_n(t))$ and in the same way $\boldsymbol{\phi}(t) = (\phi_1(t), \phi_2(t), \dots, \phi_n(t))$.
The optimal state of the network $\mathbf{y}_{ref}$ is defined as the state of highest possible flow, that is, the point of maximum of the MFD, since we have that for every region r_i the regional flow $\phi_i(t)$ can be obtained as $\phi_i(t) = G_i(\rho_i(t))$ we can define the optimal state of the network in terms of the densities $\rho_i(t)$, that is

$$y^i_{ref} = \arg \max_{\rho_i \in [0,1]} G_i(\rho_i) \tag{1}$$

and $\mathbf{y}_{ref} = (y^1_{ref}, \dots, y^n_{ref})$.

2.2 Aggregated traffic dynamics for a partitioned city

Let us consider a city partitioned in $\mathscr{R} = \{1, \dots, N\}$ different regions, the details on how to obtain such a clustering will be discussed later on. Let us now fix a time t, for each region we consider the regional density ρ_i, this quantity can be easily estimated in reality through the use of *induction loop detectors*. Density is a well behaved function inside a homogeneous region and can act as a proxy for the regional flow ϕ_i through the use of the MFD.
Consider also a control law $\boldsymbol{u} \in \mathbb{R}^m$, the choice of the input can be very different, ranging from dynamical speed limits for roads, green times for traffic lights or routings for CAVs. The entrances of this vector are not necessarily tied to a specific region. One can decide to consider subvectors $\boldsymbol{u}_i$ as inputs to a specific region, this can be interesting to explore and leads to a distributed control approach, however this line of thought is beyond the scope of this work and we do not consider it here. We have that each input $u(t) \in [\underline{u}, \bar{u}]$ where the lower limit $\underline{u}$ and upper limit $\bar{u}$ depend on the nature of the input chosen.
The state of our dynamic system is then the column vector of the regional densities $\boldsymbol{\rho}(t) := col(\{\rho_i(t)\}_{i \in \mathscr{R}}) \in \mathbb{R}^N$. At each time t a demand $q_{ij}(t)$ is

generated. It represents the number of vehicles that enter the traffic in region i with destination in j normalized by the network length, this is an exogenous variable of our problem as we have no control over it. Traffic demand is typically represented by a non-linear function characterized by to peaks, it reaches a peak during the early morning and in the evening as most people commute from or to work. As it's behaviour is known it can be forecasted fairly accurately (some references here). We denote by $q_i(t) := col(\{q_{ij}(t)\}_{j\in\mathscr{R}}) \in \mathbb{R}^N$ the column vector of demands originated in i and by $\boldsymbol{q}(t) := col(\{q_i(t)\}_{i\in\mathscr{R}}) \in \mathbb{R}^{N^2}$ the total demand of the city.
The dynamics of the density $\boldsymbol{\rho}$ can be formalized by the following non linear equation

$$\boldsymbol{\rho}(t+1) = f(\boldsymbol{\rho}(t), \boldsymbol{q}(t), \boldsymbol{u}(t)) \tag{2}$$

where $f : \mathscr{R}^N \to \mathscr{R}^N$.
This trajectories can be approximated through modelling as the one later used in 3.5, the approach of DeePC is however different, the trajectories are linearly approximated starting from the data contained in the Hankel matrix. In order to work the DeePC algorithm needs a reference trajectory, this trajectory represents a desirable state towards which we wish to conduct our system. As we mentioned before, one way to obtain this trajectory is to use the MFD, we now give a brief explanation of this tool.

2.2.1 Macroscopic Fundamental Diagram

The need of an aggregate description for traffic dynamics is prompted by the fact that existing models which use unaggregated information to predict valuable information about the traffic work well in uncongested settings but turn out to be quite unprecise in congested settings. Moreover these models often use Origin-Destination (O-D) matrices which quickly become large for sufficiently big populations this leads to a problem of scalability in these approaches.
The MFD quickly rose to popularity in the Traffic Engineering field after its first conceptualization [13], it is based on two postulates: (i) The homogeneously congested regions of a network exhibit a MFD relating *production* defined as the product of the average flow and the network length, and *accumulation* defined as product of the average density and the network length; (ii)The trip completion rate is is proportional to the production.
It was later proved the existence of empirical MFDs [14] for various traffic networks in the world, in particular Yokohama (Japan) and Zürich (Switzerland), and show interesting statistical properties, for example its distribution seem to follow a Chi-Square probability distribution.
The functional form of an MFD can be represented as a third degree polynomial 1, it has a first phase of rather linear growth of the flow, this linear behaviour is due to due low interactions between vehicles ,until a *critical density* ρ_{max} is reached. The critical value density is the point at which the flow reaches its maximum point, which is also know as *capacity* ϕ_{max}, all the density values greater than this represent congested stated and have to be considered undesirable. In our work the role of the MFD is to give us an empirical method to identify the optimal state for each region, this state is uniquely identified by the average density that leads to the maximum possible flow for each region.

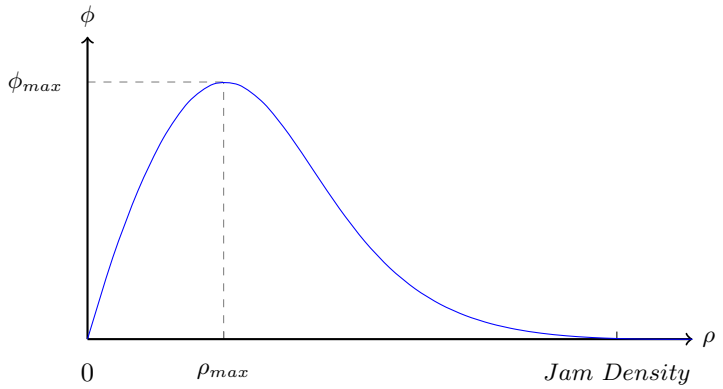


Figure 1: An Example of Macroscopic Fundamental Diagram

2.3 DeePC Formulation

2.3.1 Behavioural System Theory

Behavioural system theory is a powerful tool to describe a dynamical system whenever one is not concerned with a particular system representation. It was first introduced in [17] and it describes linear systems in terms of the subspace of the signal space in which the trajectories of the system live. As opposed with classical system theory, where a particular parametric representation, such as for example the *state-space model*, is used to describe the input/output behaviour.

2.4 Discrete-time LTI dynamical systems

A *dynamical system* Σ is a triple $\Sigma = (\mathbb{T}, \mathbb{W}, \mathscr{B})$, where $\mathbb{T}$ is the *time set*, $\mathbb{W}$ is the *signal space*, and $\mathscr{B} \subseteq (\mathbb{W})^{\mathbb{T}}$ is the *behavior* of the system. In this spirit, a *model class* $\mathscr{M}$ is defined as a family of subsets of $\mathbb{W}^{\mathbb{T}}$, i.e. $\mathscr{M} \subseteq 2^{\mathbb{W}^{\mathbb{T}}}$, so that a behaviors is a simply *model* of a dynamical system. By a convenient abuse of notation, we also use interchangeably the terms "system" and "dynamical system". We also routinely identify systems with their behaviors. We exclusively focus on *discrete-time* systems, with $\mathbb{T} = \mathbb{N}$ and $\mathbb{W} = \mathbb{R}^q$.

2.4.1 Finite-dimensional LTI systems

A system $\mathscr{B}$ is *linear* if $\mathscr{B}$ is a linear subspace, *time-invariant* if $\mathscr{B}$ is shift-invariant, *i.e.*, $\sigma^{\tau-1}(\mathscr{B}) \subseteq \mathscr{B}$ for all $\tau \in \mathbb{N}$, and *complete* if $\mathscr{B}$ is closed in the topology of pointwise convergence [18, Proposition 4]. The result states, $w \in \mathscr{B}$ if and only $\sigma^\tau w|_{L-\tau+1} \in \sigma^\tau \mathscr{B}|_{L-\tau+1}$ for every $\tau \in \mathbb{Z}_+$ and $L \in \mathbb{N}$ such that $\tau \leq L$. The model class of all complete LTI systems is denoted by $\mathscr{L}^q$. By a convenient abuse of notation, we write $\mathscr{B} \in \mathscr{L}^q$.

2.4.2 Kernel representations

Completeness of a discrete-time linear system $\mathscr{B}$ is equivalent to $\mathscr{B}$ being closed in the topology of pointwise convergence [18, Proposition 4]. Every

finite-dimensional LTI system $\mathscr{B} \in \mathscr{L}^q$ admits a *kernel representation* of the form

$$\mathscr{B} = \ker R(\sigma), \tag{3}$$

where the operator $R(\sigma)$ is defined by the polynomial matrix $R(z) = R_0 + R_1 z + \ldots + R_\ell z^\ell$, with $R_i \in \mathbb{R}^{p\times q}$ for $i \in \boldsymbol{\ell}$, and the set $\ker R(\sigma)$ is defined as $\{w : R(\sigma)w = 0\}$. Without loss of generality, we assume that $\ker R(\sigma)$ is a *minimal representation* of $\mathscr{B}$, *i.e.*, p is as small as possible over all kernel representations of $\mathscr{B}$.

2.4.3 Integer invariants and complexity of an LTI system

The structure of a system $\mathscr{B} \in \mathscr{L}^q$ is characterized by a set of integer invariants [18, Section 7], defined as

- the *number of inputs* $m = q - \text{row dim} R$,
- the *number of outputs* $p = \text{row dim} R$,
- the *structure indices* $\ell_i = \deg \text{row}_i R$ for $i \in \mathbf{p}$,
- the *lag* $\ell = \max_{i\in\mathbf{p}}\{\deg \text{row}_i R\}$, and
- the *order* $n = \sum_{i\in\mathbf{p}} \deg \text{row}_i R$,

where row dim$R$ and $\deg \text{row}_i R$ are the number of rows and the degree of the i-th row of $R(z)$, respectively. Without loss of generality, we assume that the structure indices are always ordered as $\ell_1 \leq \ldots \leq \ell_p$ and we define $\ell_0 = 0$ and $\ell_{p+1} = \infty$. With this convention, the time set $\mathbb{N}$ can be partitioned as $\mathbb{N} = (\ell_0, \ell_1] \cup \cdots \cup (\ell_p, \ell_{p+1}]$, so that, for any $L \in \mathbb{N}$, there is $k_L \in \mathbf{p} + \mathbf{1}$ such that $L \in (\ell_{k_L-1}, \ell_{k_L}]$. The integer invariants are intrinsic properties of a system, as they do not depend on its representation [19, Proposition X.3]. Furthermore, the complexity of a system $\mathscr{B} \in \mathscr{L}^q$ is uniquely specified by the number of inputs m, the order n, and the lag of ℓ [20, Theorem 25]. The number of inputs m and the structure indices $\ell_1, \ldots, \ell_p$ they are in one-to-one correspondence with the *complexity* $c : \mathscr{L}^q \to [0,1]^{\mathbb{N}}$ of a system $\mathscr{B} \in \mathscr{L}^q$ [20, Theorem 25], defined as $c(L) = \dim / qL$ for $L \in \mathbb{N}$. Following [21], we identify the complexity of system $\mathscr{B} \in \mathscr{L}^q$ with the triple $c = (m, \ell, n)$. The class of all systems $\mathscr{B} \in \mathscr{L}^q$ with complexity k is denoted by $\mathscr{L}^{q,k}$. By a convenient abuse of notation, we also write $\mathscr{B} \in \mathscr{L}^{q,k}$.

2.4.4 Partitions

Given a permutation matrix $\Pi \in \mathbb{R}^{q\times q}$ and an integer $0 < m < q$, the map

$$(u, y) = \Pi^{-1} w \tag{4}$$

defines a *partition* of $w(t) \in \mathbb{R}^q$ into the variables $u(t) \in \mathbb{R}^m$ and $y(t) \in \mathbb{R}^{q-m}$. Any partition (4) induces the natural projections $\pi_u : w \mapsto u$ and $\pi_y : w \mapsto y$. The partition (4) is an *input-output partition* of $\mathscr{B} \in \mathscr{L}^q$ if (i) u is a *free variable*, *i.e.*, $\pi_u(\mathscr{B}) = (\mathbb{R}^m)^{\mathbb{N}}$, (ii) m is the number of inputs of $\mathscr{B}$, and (iii) y does not anticipate u, *i.e.*, for all $L \in \mathbb{N}$ and $w, \bar{w} \in \mathscr{B}$, the condition $\{\pi_u w(t) = \pi_u \bar{w}(t) \text{ for } t \in \mathbf{L}\}$ implies $\{\pi_y w(t) = \pi_y \bar{w}(t) \text{ for } t \in \mathbf{L}\}$.

2.4.5 State-space representation

Given an input-output partition (4), any finite-dimensional LTI system $\mathscr{B} \in \mathscr{L}^q$ can be described by the equations

$$\sigma x = Ax + Bu, \quad y = Cx + Du, \tag{5}$$

and admits a (*minimal*) *input/state/output representation*

$$\mathscr{B} = \left\{(u, y) \in (\mathbb{R}^q)^{\mathbb{N}} \,:\, \exists\, x \in (\mathbb{R}^n)^{\mathbb{N}} \text{ s.t. (5) holds}\right\}, \tag{6}$$

where $\left[\begin{smallmatrix} A & B \\ C & D \end{smallmatrix}\right] \in \mathbb{R}^{(n+p)\times(n+m)}$ and n is the order of $\mathscr{B}$.

2.4.6 Data-Driven behaviors Hankel, PE

The Hankel matrix is at the heart of the DeePC algorithm, it used as a non-parametric model to describe the relationship between the inputs and the outputs of the system and its standard form is defined as

Definition 1 (Hankel Matrix)**.** Given a vector of signals $\boldsymbol{\eta} = (\eta(1), \ldots, \eta(T))$ spanning the interval $[1, T]$ the Hankel Matrix of order L, $\mathscr{H}_L(\boldsymbol{\eta}) \in \mathbb{R}^{L\times T-L+1}$ is defined as follows

$$\mathscr{H}_L(\boldsymbol{\eta}) = \begin{pmatrix} \eta(1) & \ldots & \eta(T-L+1) \\ \vdots & \ddots & \vdots \\ \eta(L) & \ldots & \eta(T) \end{pmatrix} \tag{7}$$

⌟

2.5 Time series and Hankel matrices

We use the terms *time series* and *trajectory* interchangeably. The set of time series $w = (w(1), \ldots, w(T))$ of length $T \in \mathbb{N}$, with $w(t) \in \mathbb{R}^q$ for $t \in \mathbf{T}$, is denoted by $(\mathbb{R}^q)^{\mathbf{T}}$. The set of infinite-length time series $w = (w(1), w(2), \ldots)$, with $w(t) \in \mathbb{R}^q$ for $t \in \mathbb{N}$, is denoted by $(\mathbb{R}^q)^{\mathbb{N}}$.

2.5.1 The cut operator

Restricting time series over subintervals gives rise to the cut operator. Formally, given a finite-length time series $w \in (\mathbb{R}^q)^{\mathbf{T}}$ and $L \in \mathbf{T}$, the *cut operator* is defined as

$$w|_L = (w(1), \ldots, w(L)) \in (\mathbb{R}^q)^{\mathbf{L}} \tag{8}$$

For infinite-length time series, the definition holds verbatim with $w \in (\mathbb{R}^q)^{\mathbb{N}}$ and $L \in \mathbb{N}$. Applied to a set of time series $\mathscr{W} \subseteq (\mathbb{R}^q)^{\mathbf{T}}$ or $\mathscr{W} \subseteq (\mathbb{R}^q)^{\mathbb{N}}$, the cut operator acts on all time series, defining the *restricted* set $\mathscr{W}|_L = \{w|_L \,:\, w \in \mathscr{W}\}$. By a convenient abuse of notation, we identify the trajectory (8) with the corresponding vector $(w(1), \ldots, w(L)) \in \mathbb{R}^{qL}$.

2.5.2 The shift operator

Shifting elements of time series gives rise to the shift operator. Formally, given $w \in \mathbb{R}^{qT}$ and $\tau \in \mathbf{T}$, the *shift operator* is defined as

$$\sigma^{\tau-1} w = (w(\tau), \dots, w(T)) \in \mathbb{R}^{q(T-\tau+1)}. \tag{9}$$

For infinite-length time series, the shift operator is defined as $w \mapsto \sigma^{\tau-1} w$, with $\sigma^{\tau-1} w(t) = w(t+\tau-1)$, for any $\tau \in \mathbb{N}$. Applied to a set of time series $\mathscr{W} \subseteq \mathbb{R}^{qT}$ or $\mathscr{W} \subseteq (\mathbb{R}^q)^{\mathbb{N}}$, the shift operator acts on all time series in the set giving rise to the *shifted* set $\sigma^\tau \mathscr{W} = \{\sigma^\tau w : w \in \mathscr{W}\}$.

2.5.3 Hankel matrices

The *Hankel matrix* of depth $L \in \mathbf{T}$ associated with the time series $w \in \mathbb{R}^{qT}$ is defined as

$$H_L(w) = \begin{bmatrix} w(1) & w(2) & \cdots & w(T-L+1) \\ w(2) & w(3) & \cdots & w(T-L+2) \\ \vdots & \vdots & \ddots & \vdots \\ w(L) & w(L+1) & \cdots & w(T) \end{bmatrix}.$$

In the DeePC algorithm however, a slightly different version of this matrix is used, namely instead of $\eta(t)$ being a scalar for all $t \in [0, T]$ we have a vector of dimension r at each time point, $\boldsymbol{\eta}(t) \in \mathbb{R}^r$, this means that our signal now belongs to $\mathbb{R}^{T \times r}$ and the structure of the resulting Hankel matrix can now be more accurately described as being a *Block Hankel matrix* belonging to $\mathscr{H}_L(\eta) \in \mathbb{R}^{Lr \times T-L+1}$.

$$\mathscr{H}_L(\boldsymbol{\eta}) = \begin{pmatrix} \eta_1(1) & \dots & \eta_1(T-L+1) \\ \vdots & \ddots & \vdots \\ \eta_r(1) & \dots & \eta_r(T-L+1) \\ \vdots & \ddots & \vdots \\ \eta_1(L) & \dots & \eta_1(T-L+1) \\ \vdots & \ddots & \vdots \\ \eta_r(L) & \dots & \eta_r(T) \end{pmatrix} \tag{10}$$

We now introduce two other important concepts, *controllability* and *persistency of excitation*.

Definition 2 (Controllability)**.** A system $\mathscr{B} \in \mathscr{L}^{m+p}$ is said to be *controllable* if $\forall T \in \mathbb{Z}_{\geq 0}, w^1 \in \mathscr{B}_T, w^2 \in \mathscr{B}$ there exists $w \in \mathscr{B}$ and $T' \in \mathbb{Z}_{\geq 0}$ such that $w_t = w_t^1$ for $1 \leq t \leq T$ and $w_t = w_{t-T-T'}^2$ for $t > T + T'$. The set of controllable systems is denoted as $\mathscr{L}_c^{m+p} \subseteq \mathscr{L}^{m+p}$ ⌟

In other words a system is controllable if any two trajectory can be patched together in finite time.

Definition 3 (Persistently Exciting)**.** Let $L, T \in \mathbb{Z}_{\geq 0}$ such that $T \geq L$. The signal $\boldsymbol{u} = col(u_1, \dots, u_T) \in \mathbb{R}^{Tm}$ is persistently exciting of order L if the Hankel matrix

$$\mathscr{H}_L(u) = \begin{pmatrix} u_1 & \dots & u_{T-L+1} \\ \vdots & \ddots & \vdots \\ u_L & \dots & u_T \end{pmatrix} \tag{11}$$

has full rank, in other words when

$$rank(\mathscr{H}_L(\boldsymbol{u})) = LTm \tag{12}$$

⌟

The notion of persistency of excitation is crucial importance in system identification and adaptive control, it describes an input signal which is sufficiently rich and long as to excite the system yielding an output sequence which is representative of the system behaviour.
The condition 12 stated in the above definition can be thought of in the following way, the signal $\boldsymbol{u}$ is persistently exciting if there exist no non-trivial linear relations of order L among the components $u(t)$,*i.e.*, if they are linearly independent, let us put in more formal terms this description. Assume that a response $\boldsymbol{\eta} = (\eta(1), \eta(2), \ldots, \eta(T)) \in \mathscr{B}_T$ of a linear time-invariant system is observed, for some L, $1 \leq L \leq T$, consider the 'windows' of length L given by

$$\begin{aligned}
&[\eta(1), \ldots, \eta(L)] \\
&[\eta(2), \ldots, \eta(L+1)] \\
&\vdots \\
&[\eta(T-L+1), \ldots, \eta(T)]
\end{aligned}$$

The question that we are trying to answer, that is, how can we be sure that an observed signal contains enough information to give us good model of the observed system, can be reformulated as
Under which conditions do these windows span the whole space of all possible windows of length L which the system is able to produce?
It is clear that these windows correspond to the columns in the Hankel matrix $\mathscr{H}_L(\boldsymbol{\eta})$, let $\mathscr{R}_{\mathscr{B}} := \{n \in \mathbb{R}^{m+p} \mid n^\top(\sigma)\mathscr{B} = 0\}$ be the module of *annihilators* of $\mathscr{B}$ and define $\mathscr{R}^{\lambda}_{\mathscr{B}} := \{n \in \mathscr{R}_{\mathscr{B}} \mid \text{each element of n is of degree less than } \lambda\}$.
Since $\boldsymbol{\eta} \in \mathscr{B}_T$ we have that any $n \in \mathscr{R}^{L-1}_{\mathscr{B}}$ is such that $n^\top \mathscr{H}_L(\boldsymbol{\eta}) = 0$, therefore the left kernel of $\mathscr{H}_L(\boldsymbol{\eta})$ contains the vectors generated by $\mathscr{R}^{L-1}_{\mathscr{B}}$.
The question now is *When does the left kernel contain all the annihilators?*
That is, when do we have

$$leftkernel(\mathscr{H}_L(\boldsymbol{\eta})) = \mathscr{R}^{L-1}_{\mathscr{B}}$$

or equivalently,

$$colspan(\mathscr{H}_L(\boldsymbol{\eta})) = \mathscr{B}_L$$

We now present a result known in behavioural system theory as *The Fundamental Lemma*, it has first been proved in [22] and it states

Lemma 1 (Fundamental Lemma of BT (Theorem 1, [22])**.** Let us consider a controllable system $\mathscr{B} \in \mathscr{L}^{m+p}_c$. Let $T, t \in \mathbb{Z}_{>0}$, and $\boldsymbol{w} = col(\boldsymbol{u}, \boldsymbol{y}) \in \mathscr{B}_T$. Then if $\boldsymbol{u}$ is persistently exciting of order $t + \boldsymbol{n}(\mathscr{B})$ we have

$$colspan(\mathscr{H}(\boldsymbol{w})) = \mathscr{B}_t$$

For the proof of this result see the A.

2.5.4 The Optimization Problem

Let us start by denoting the behaviour of the traffic system at hand as $\mathscr{B} \in \mathscr{L}^{m+p}$ where m and p are respectively the number of inputs and the number of outputs, let $\mathbf{n}(\mathscr{B})$ be the *minimal representation* of the system, which in this case will be equal to the number of regions in our network.
The cost function c in this framework will then be a discrete function, dependent on the regional density $\boldsymbol{\rho}$, the input to our system $\boldsymbol{u}$ and the optimal trajectory $\boldsymbol{y}_{ref}$.

$$c(t) = c(\boldsymbol{\rho}(t), \boldsymbol{u}(t), \boldsymbol{y}_{ref}), \tag{13}$$

given a time horizon N we can now formalize the optimization problem as

$$\begin{aligned} &\arg\min_{\boldsymbol{u}} \sum_{k=0}^{N-1} c(\boldsymbol{\rho}(k), \boldsymbol{u}(k), \boldsymbol{y}_{ref}) \\ \textit{subject to} \quad &\boldsymbol{u}(k) \in \mathbb{U} \quad k \in \{0, \dots, N-1\} \qquad (14) \\ &\boldsymbol{d}(k) \in \mathbb{D} \quad k \in \{0, \dots, N-1\} \end{aligned}$$

$$(15)$$

where $\mathbb{U}, \mathbb{D}$ are two subsets respectively of $\mathbb{R}^m$ and $\mathbb{R}^p$.
One can now choose to solve this problem with various tools, if one would like to use DeePC to solve this general optimization problem the formulation would be the following.
Let $T, T_{ini}, N \in \mathbb{Z}_{>0}$ be such that $T \geq (m+1)(T_{ini} + N + \mathbf{n}(\mathscr{B})) - 1$. Let $u^d = col(u(1), \dots, u(T)) \in \mathbb{R}^{Tm}$ be a sequence of T random input ratios applied to $\mathscr{B}$, and $y^d = col(\rho(1), \dots, \rho(T)) \in \mathbb{R}^{Tp}$ be the corresponding output, by taking u^d random at each step we ensure that the input to be *persistently exciting.*
Next we partition the data in *past data* and *future data*, formally

$$\begin{pmatrix} U_p \\ U_f \end{pmatrix} := \mathscr{H}_{T_{ini}+N}(u^d), \quad \begin{pmatrix} Y_p \\ Y_f \end{pmatrix} := \mathscr{H}_{T_{ini}+N}(y^d), \tag{16}$$

where $U_p$and U_f consists respectively of the first T_ini and the last N block rows of $\mathscr{H}_{T_{ini}+N}(u^d)$ (similarly for Y_p and Y_f).
Once we have the Hankel matrix we can start controlling the system directly, define the cost function as

$$c(y(t), u(t)) = \sum_{k=0}^{N-1} ||y(k) - \bar{y}||^2 + ||u(k) - \bar{u}||^2 \tag{17}$$

where $\bar{y} = \boldsymbol{y}_{ref}$ is the optimal trajectory given by the maximum point of the MFD of each region, and $\bar{u}$ is the desired value in input set $\mathbb{U}$.
Let the system $\mathscr{B}$ progress of T_{ini} steps in order to collect the initial trajectory

$w_{ini} = col(u_{ini}, y_{ini})$, then at each decision time t solve the quadratic problem

$$\begin{aligned} \arg\min_{\boldsymbol{u}} &= \sum_{k=0}^{N-1} ||y(k) - y_{ref}||_Q^2 + ||u(k) - \bar{u}||_R^2 + \lambda_g ||g||_1 + \lambda_1 ||(I - \Pi)g||_1 \\ \textit{subject to} \quad & \begin{pmatrix} U_p \\ U_f \\ Y_p \\ Y_f \end{pmatrix} g = \begin{pmatrix} u_{ini} \\ y_{ini} \\ u \\ y \end{pmatrix} \\ & u(k) \in \mathbb{U} \quad k \in \{0, \dots, N-1\} \\ & y(k) \in \mathbb{D} \quad k \in \{0, \dots, N-1\} \end{aligned} \tag{18}$$

where we have $\mathbb{U} = [u_{min}, u_{max}]$ and $\mathbb{D} = [d_{min}, d_{max}] \subseteq [0, 1]$, and Π is the matrix given by $\Pi = \begin{bmatrix} U_p \\ Y_p \\ U_f \end{bmatrix}^\dagger \begin{bmatrix} U_p \\ Y_p \\ U_f \end{bmatrix}$.

3 Simulations Framework

3.1 Problem specific to the city of Zurich

The city of Zürich is a highly functional social, economical and financial center which attracts thousands of people, the metropolitan area now counts more than one million and half people and it is the area with the highest Human Development Index (HDI) in Switzerland, closely followed by Geneva.
The city is embedded in a web of highways, the Autobahn A3 and A4 being two of the most prominent, which bring traffic flows from the surrounding area in and out the city.
This high social and economical development comes at the cost of having high traffic densities and congestion, especially during the morning and evening peaks when most of the people commute from home to work and viceversa. According to [23] Zürich has been the 51-th most congested city in the world, with an average time of 21 minutes to cover 10 kilometers at an average speed of 24 km/h, this impacts the city in various ways, the most significant health concerns being the air pollution and the increase probability of accidents happening.
Amid the climate crisis its also important to consider that congestion leads to unnecessary CO2 emissions which could be avoided by having a higher traffic flow. Since major structural changes would be extremely expensive and impact historical architecture especially near the city center, the problem calls for a different solution, this solution can come in the form of control algorithms that use traffic lights, dynamic speed limits or other actuators to influence the city traffic flow and ensure that it stays near the maximum possible achievable.

3.2 Mesoscopic Simulation Setup

To show that DeePC is a feasible option when controlling large traffic networks we backtested the algorithm against the state of the art LMPC. We carried out

our tests on an highly detailed SUMO [24] simulation of the city of Zürich, the simulation has been developed by Lukas Ambühl at Transcality [25], the network itself counts 6917 nodes, of which more than 800 are traffic lights, connected by 14566 edges.

Figure 2: The Simulation Network

SUMO is a microscopic simulation framework, however for a network of this scale a microscopic simulation would be too computationally expensive, therefore we chose to use the mesoscopic model provided by SUMO, the model used by SUMO is known as the Eissfeldt model and it is described in detail in [26]. The trips and the routing in the simulation are produced from a probability distribution which has been chosen to reflect as closely as possible the distribution of the traffic in Zürich for the modelled periods.
The types of vehicles modeled are cars, delivery vans, trucks and busses. The traffic lights are realistically modeled following a cycle composed of various phases which depends on the topological structure of the traffic light, for example in 3 we can observed one of the intersections used as actuator in the simulation, the intersection represented is the one between Rämistrasse and the QuaiBrücke. The period of the cycle of this traffic light is of 48 seconds, it has nine phases which are listed below in tuples (phase, duration).
For each of this actuators we pass as an input u the ratio between green time and the total cyscle duration $u = \frac{green\ time}{cycle\ period}$.

The letters follow a specific code where g is used for a green lane light, G is a green lane light with priority, y is an orange light and r is used for a red light. We can notice that each cycle is formed by 3 subcycles where the main green phase has duration 11, the transition phase has duration 3 and the totally red phase has duration 2; this structure is common with slight variations to all the traffic lights used for the control and will be important later on.

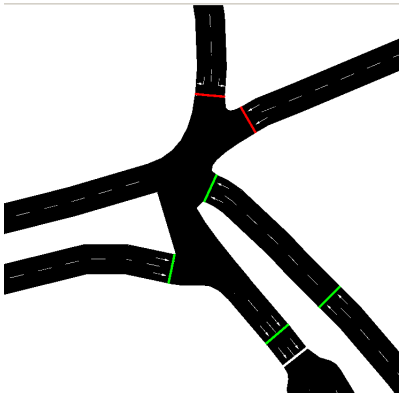


Figure 3: One of the Controlled Intersections

$$
\begin{aligned}
&1 : (rrrrrrGGGG, 11.0)\\
&2 : (rrrrrryyyy, 3.0)\\
&3 : (rrrrrrrrrr, 2.0)\\
&4 : (GGGGrrrrrr, 11.0)\\
&5 : (yyyyrrrrrr, 3.0)\\
&6 : (rrrrrrrrrr, 2.0)\\
&7 : (rrrrGGrrGG, 11.0)\\
&8 : (rrrryyrryy, 3.0)\\
&9 : (rrrrrrrrrr, 2.0)
\end{aligned}
$$

We now look deeper into the choice of the actuators used in the simulation.

3.2.1 Actuators selection

One of the drawbacks of using DeePC as controller is the fact that the dimension of the Hankel matrix $\mathscr{H}$ scales both with the dimension of the input m and the one of the output p. In our case the dimension of the output does not represent a problem, as the partitioning of the city decreases the number of outputs from the number of roads in the network to the number of regions considered. One could chose to do the same for the actuators, clustering the traffic lights by region of belonging, however in doing so one would looses of the great advantages of DeePC which is the granularity of the control.
There is also one other more practical aspect that must be considered when choosing the actuators, in a real city the infrastructure needed to a control a traffic light often takes substantial space, therefore it happens that this red lights are often several hundred meters inside a region, instead than on the border, where there is enough space to house them.
In order to have a fair comparison with the city current control, which is explored in [27], we chose to use as actuators the intersections that the city of Zürich is using. These intersections come with the benefit that are positions on high

volume traffic flows which determine a big portion of the traffic demand during the peaks.

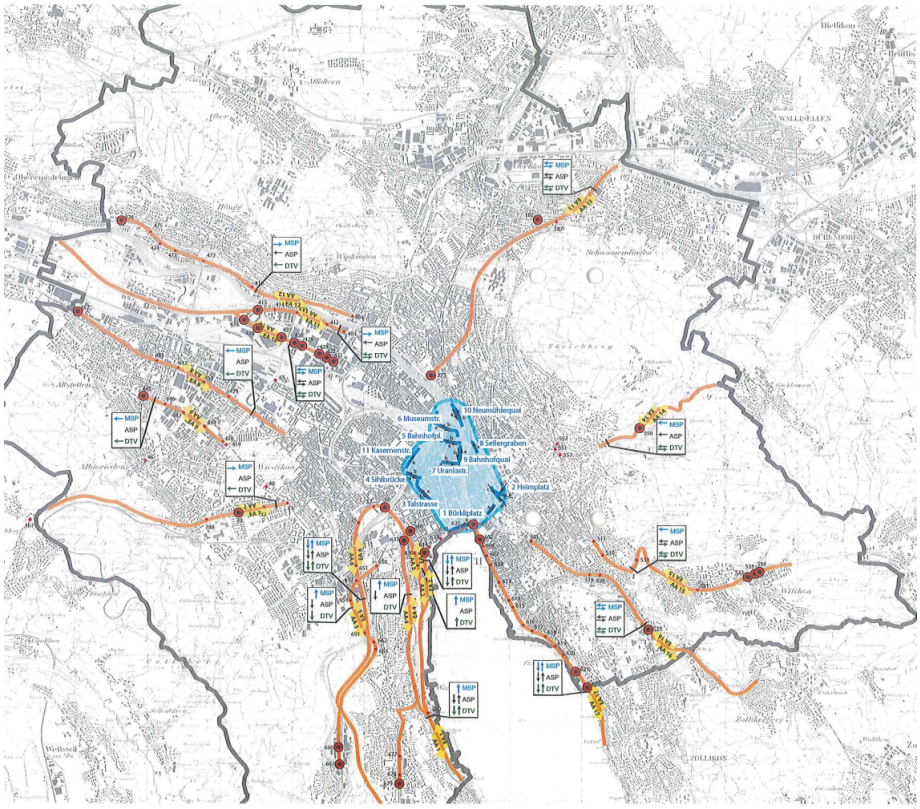


Figure 4: Positions of the actuators

3.3 Clustering and MFD

3.3.1 Region Partitioning Retrieval

The algorithm used in this work has been devised by N. Geroliminis and M. Saeedmanesh in [16], the main idea is to construct a similarity measure between the edges of the network by constructing a "Snake" for each edge and then confronting the snakes one by one. The Snakes are constructed in the following way, to each edge is associated it's average density over taken over the chosen period of observation, a snake is started at each edge as an ordered list initially containing only the first edge. In a recursion loop, the neighbouring edges of the snake are checked to find the one with the average density closest to the average density of the whole Snake, this edge it's then added to the snake, this ensures that the overall variance of the densities in the snake is at each step minimal, the neighbours of the snake are defined as the edges of the network, currently not in the snake, that share one node with at least one edge of the snake.
Once all the snakes have been computed, a similarity matrix W can be constructed by using the following metric, for each pair of edges $i, j \in E$ we consider

the snake starting at i and j which we will call respectively S_i and S_j. Next starting at length $k = 2$, we count the number of common edges in the subsnakes of length k S_i^k, S_j^k until the full length of the snakes l is reached, in this way the edges that appear sooner in the snake will be counted more times and will therefore have a greater impact on the similarity of the two snakes.
This metric leads to clusters which have low variance in their average densities, meaning that their are homogeneous and so satisfy one of the conditions to have a low scatter MFD, moreover it encourages clusters to have an high connectivity so that disconnected edges are less likely and it is able to detect the direction of the flows in traffic.
Note that the choice of the snake length l is an important factor especially when dealing with large networks, fixing a number l which is less then the cardinality of E has two main benefits: in the first place it lowers greatly the computational cost of constructing the snakes and later of the computation of the similarity metric; most importantly it leads indirectly to more compact and connected clusters, this is due to how the metric is constructed, fixing l means that all the edges that are far enough from the starting edge will have automatically similarity equal to 0 as the snakes will not have common parts.
This algorithm is a very powerful tool to discover homogeneous regions in networks, however it comes at the cost of being computationally demanding when dealing with large networks, the main source of this cost is the computation of the successive intersections between the snakes S_i, S_j.
To deal with this problem we made a slight modification to the way in which the metric is being computed, instead of computing the intersection between S_i^k and S_j^k for each k until l is reached, we fixed a number of steps $s \leq \lfloor l/s \rfloor$ and compute the intersection between the sublists $S_i^{k \times s}, S_j^{k \times s}$.
This has the effect of reducing the computational cost of computing one entrance of W, we also took advantage of the fact that the similarity matrix is a symmetric matrix, leading us to compute only the upper triangular part of W.
This modification comes at the cost of having less connected clusters, especially near the borders between two regions, and the loss of this information is greater as s grows. To compensate for this, we devised an automatic post-processing procedure which helps to restore the network connectivity iteratively.
In the last step of the paper [16] the clustering is retrieved by using *Symmetric Non-negative Matrix Factorization* (SNMF), we chose to use *Spectral Clustering* instead as we tested it different clustering algorithms such as: SNMF, OPTICS, DBSCAN, Agglomerative Clustering; and it yielded the best results in terms of cluster connectivity and accuracy.
In figure 5 we ran the algorithm on small SUMO simulation, with a 5x5 grid structure on which the vehicles trips have been defined only on the 3x5 right side, in this way we obtained two clearly defined regions, the left one (purple) characterized by an average traffic density of 0 and the right one (yellow) with a traffic density greater than 0.

3.3.2 The Post-processing Procedure

The goal of the procedure is to restore the connectivity of the regions, so that we don't have multiple connected components for a given cluster, it is based on a simple assumption : since the traffic density is locally spatially correlated, the probability of an edge i to belong to cluster k is dependent on how many of its

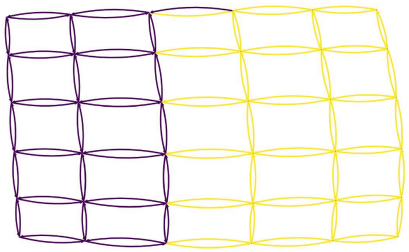


Figure 5: The Clustering obtained on a grid 5x5 with two clearly divided regions.

neighbouring edges belong to k.
This leads to the following algorithm:

Algorithm 1 Post-Processing Procedure

```
1: procedure PROCEDURE
2:     maxiter ∈ ℕ
3:     k ← 0
4:     while k < maxiter do
5:         for i ∈ E do
6:             label(i) ← most frequent label in Neighbours(i)
7:         k ← k + 1
```

If *maxiter* is large enough the effect of the procedure is to shrink the isolated clusters of edges until they merge smoothly with the surrounding area, and clearly defines the boarders in the areas where the two regions encounter.

3.4 The Center of Zürich

The center of Zürich is a small diamond shape area 6 which is covers the Old city of Zürich (Die Altstadt) and it is an area of critical importance for the municipality which means to protect it from extreme congestion situations. Given Our partitioning however this area falls in two distinct homogeneous region and therefore doesn't satisfy the first postulate to exhibit a well behaved low-scatter MFD.
To address this problem we decided to split this area into two additional regions of the network defined by the intersection between the area defined as "the center" by the municipality and the respective region of belonging in our clustering.

Figure 6: The City Center

3.4.1 Macroscopic Fundamental Diagram Retrieval

The Data used to estimate the empirical Macroscopic Diagram of the regions in the simulation has been collected in a similar fashion as the ones used during the clustering, that is the evening peak model. There are however some differences, the first one being that in order to have a well defined curve that reaches zero flow at maximum possible density, the scaling factor of the traffic must be appropriate, meaning that there must be enough vehicles to send the system into a grid-lock state or close to it, in our case we chose a scale factor of 2, indicating twice the default traffic of the simulation.
We chose to collect the edge-wise density $\rho_i(t)$ and flow $\phi_i(t)$ data for each $i \in V$ every 3 seconds giving us a total of 7200 data points, such granularity is not needed in order to have a well defined MFD, however since it is a one time process we chose to have highly precise data.
After this collection the data has been averaged based on the clustering obtained at the previous step, that is we averaged the data over the regions and built the vectors of regional densities and flow $\boldsymbol{\rho}(t), \boldsymbol{\phi}(t)$. Then we used a 4-th degree polynomial to estimate the functional form of the MFD by least square, we then constructed the piecewise approximation needed by the Model Predictive Control linear model. In the toy example of the result section we produced synthetical MFDs as downpointing parabolas by randomly choosing the vertices, the MFDs used in the example can be seen in 7.
As can bee seen in 8 some of the regions were not completely saturated, these are the external regions of the city on which the distribution of the traffic is comparatively low, however this does not impact the position of the critical density point.

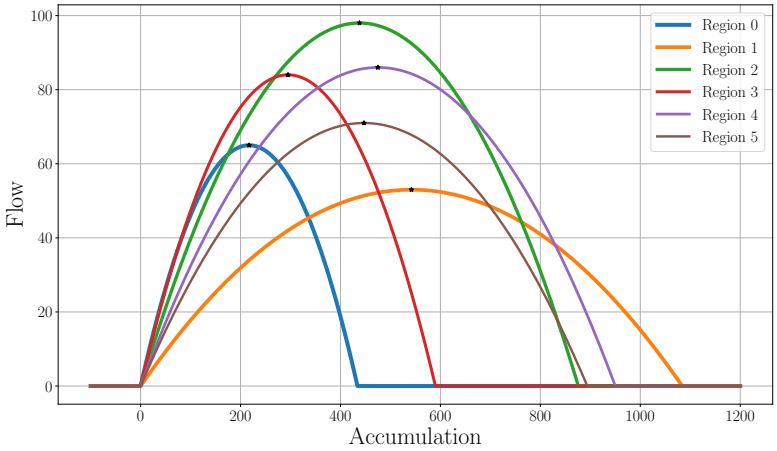


Figure 7: The MFDs used in the Toy Example

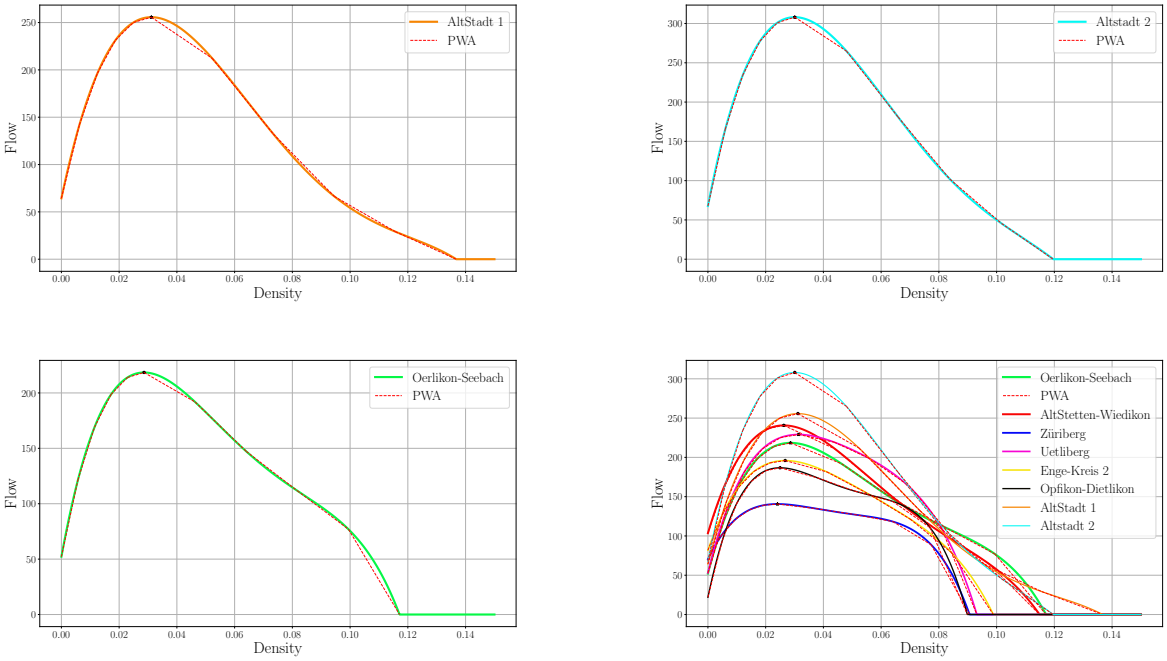


Figure 8: MFDs recovered from the Zürich SUMO Simulation

3.5 MPC formulation for comparison

To have a comparison of the results of DeePC we also implement a Model Predictive Control approach, there exist several MPC formulations to solve the traffic optimization problem both linear and non-linear, however in practice

non-linear formulations are avoided due to their higher computational cost. The state-of-the art in practice is represented by the Linear Model Predictive Control (LMPC) first devised in [1].

Consider a urban network partitioned in a set R of homogeneous regions with well defined MFDs. The index $i \in R = \{1, \dots, N\}$ denotes one region in the system with MFD G_i, $n_i(t)$ denotes the total accumulation (the number of vehicles) in the region i at time t, and $n_{ij}(t)$ denotes the number of vehicles currently in i and with destination $j \in R$ at time t.
Let $R_i \subseteq R$ denote the subset of neighbouring region of i, meaning the regions that can be directly reached from i. The discrete relation between flow and accumulation can be described by the following first order difference equations

$$n_{ij}(k+1) = n_{ij}(k) + T_p\big(q_{ij}(k) - \sum_{h \in N_i} M_{ij}^h(k) + \sum_{h \in N_i} M_{hj}^i\big) \tag{19}$$

$$n_{ii}(k+1) = n_{ij}(k) + T_p\big(q_{ij}(k) - M_{ii}(k) - \sum_{h \in N_i} M_{ij}^h(k) + \sum_{h \in N_i} M_{hj}^i\big) \tag{20}$$

where $i, j \in R, i \neq j; k = 0, \dots, T$ is the discrete time index. The exogenous variables $q_{ij}(k)$ denote the uncontrollable traffic demand generated inside region i with destination j.
The variables $M_{ij}^h(k)$ denote the transfer flows from region i to region j which pass through region h while $M_{ii}(k)$ is the internal trip completion rate.We assume that for each region there exists a production MFD between the accumulation $n_i(k)$ and the total production $P_i(n_i(k))$, which describes the performance of a system in an aggregated way. The transfer flows $M_{ij}^h(k)$ and internal completion rate $M_{ii}(k)$ are calculated according to the corresponding production MFD of the region and the accumulation $n_{ij}(k)$ as follows

$$M_{ii}^h(k) = \theta_{ii}^h \frac{n_{ii}(k)}{n_i(k)} \frac{P_i(n_i(k))}{L_i} \tag{21}$$

$$M_{ij}^h(k) = u_{ih}(h)\theta_{ij}^h \frac{n_{ij}(k)}{n_i(k)} \frac{P_i(n_i(k))}{L_i} \tag{22}$$

where L_i is the average trip length in region i and the parameters $\theta_{ii}(k), \theta_{ij}^h(k)$ represent the route choice of the driver and are assumed to be exogenous variables. The control variables $u_{ih}(k), i \in R, h \in R_i$ denote the fraction of the flow that is allowed to transfer from region i to region h, therefore are physically constrained between 0 and 1,

$$0 \leq u_{ih}(k) \leq 1, \quad i \in R, h \in R_i \tag{23}$$

Note that these equations allow drivers to choose any arbitrary sequence of regions as their route and their path can cross region boundaries multiple times. The dynamics presented int his section lead however to a non-linear model. We now take some steps to linearize the model.

3.5.1 Linearizing the model

We start by denoting the accumulation proportion as $\alpha_{ii} := n_{ii}(k)/n_i(k)$ and $\alpha_{ij}(k) = n_{ij}(k)/n_i(k), i \in R,\ j \in R_i$, this quantities can be estimated in real

time by using the extended Kalman filter or maximum likelihood estimation, in our setting however this is not needed. We introduce the decision variables

$$f_{ii}(k) = u_{ii}(k)G_i(n_i(k))\theta_{ii}(k)\alpha_{ii}(k) \tag{24}$$

$$f_{ih}(k) = u_{ih}(k)G_i(n_i(k))\sum_{j\in R}\theta^h_{ij}(k)\alpha_{ij}(k) \tag{25}$$

$$\forall\ i,j \in R,\ h \in \mathbb{R}_i$$

In 24 the variables $\theta_{ii}(k), \theta^h_{ij}(k), \alpha_{ii}(k), \alpha_{ij}(k)$ are considered time varying exogenous signals and as result the non-linearity of the problem comes from the product of the decision variables $u_{ih}(k)$ with the MFD functions.
To overcome this problem, the MFDs of the regions are approximated with a piecewise affine (PWA) functions $G_i(n_i(k))$ that form a convex set. Each MFD can be approximated with $l = 1, \ldots, N_i$ affine functions, and we denote by $G^l_i(n_i(k))$ each affine function l.
In conclusion, the control variable are bounded $u_{ih}(k) \in [0,1], \forall i \in R$ and the MFDs can be approximated by PWA functions, as a result the constraint define a convex set.

3.5.2 LMPC Optimization Problem

By adding all the states n_{ij} and n_{ii} for each region i, we get a linear model that does not consider OD data,but only aggregated demands at the regional level. In this case the derived linear problem can be solved online as follows, given an optimization horizion N_p

$$\arg\max_{f_{ii}(k), f_{ih}(k)} \sum_{k=k_p}^{kp+Np-1}\sum_{i\in R} L_i[f_i ik + f_{ih}(k)]$$

$$\begin{aligned}
\textit{subject to}\quad & n_i(k+1) = n_i(k) + T_p\big(q_i(k) - f_i i(k) - \sum_{h\in R_i} f_{ih}(k) + \sum_{h\in\mathbb{R}_i} f_{hi}(k)\big)\\
& 0 \le f_{ii}(k) \le \theta_{ii}(k)\alpha_{ii}(k)G^l_i(n_i(k))\\
& 0 \le f_{ih}(k) \le G^l_i(n_i(k))\sum_{j\in R}\theta^h_{ii}(k)\alpha_{ij}(k)\\
& 0 \le n_i(k) \le n_{i,max}\\
& k = k_p, k_p+1, \ldots, k_p+N_p-1\\
& \forall i,j \in R,\ h \in R_i,\ l = 1,\ldots,N_i
\end{aligned}$$

where the objective function aims at maximizing the total production of the system. All constraint of this problem are linear, and as a consequence, the computational requirements are rather low even for networks with a great number of regions and a long prediction horizon.

3.6 Mapping to the actuators

In order to use the LMPC in the simulation setting we need to make a further step in modelling. The formulation presented above uses as input variables u_{ij} the fraction of the flow allowed to transfer from region i to region j. Since the

u_{ij} are constrained physically to the interval $[0, 1]$ we can also interpret them as the green time to cycle duration ratio used as input to the simulation. Caution must be taken however, by interpreting this way we must be sure that each variable is mapped to the correct actuator, that is, to the traffic lights which control the flow direction between i and j.
The mapping between the LMPC inputs and the actuators has been constructed by confronting the routing matrix retrieved by the simulation with the flow studies conducted by the traffic engineers of Zürich.

4 Results

4.1 Numerical Simulation

Our initial conjecture was that DeePC could work well in highly non linear setting such as a traffic system, this is an advantage of using the Hankel matrix as a model which, when constructed correctly, can approximate well the non-linear trajectories of the system. To test this conjecture we first compared DeePC and LMPC on a simple numerical simulation similar to the one in [1]. This numerical simulation is not supported by a framework such as SUMO behind it, instead the data are produced by the following model

$$
\begin{aligned}
n_{ij}(k+1) &= n_{ij}(k) + T_p\big(q_{ij}(k) - \sum_{h \in N_i} M^h_{ij}(k) + \sum_{h \in N_i} M^i_{hj}\big) \\
M^h_{ii}(k) &= \theta^h_{ii} \frac{n_{ii}(k)}{n_i(k)} \frac{P_i(n_i(k))}{L_i} \\
M^h_{ij}(k) &= u_{ih}(k) \theta^h_{ij} \frac{n_{ij}(k)}{n_i(k)} \frac{P_i(n_i(k))}{L_i} \\
n_{ii}(k+1) &= n_{ij}(k) + T_p\big(q_{ij}(k) - M_{ii}(k) - \sum_{h \in N_i} M^h_{ij}(k) + \sum_{h \in N_i} M^i_{hj}\big)
\end{aligned}
$$

where all the quantities are defined in the same way as in 2.2.

We consider a network composed of 6 regions and fully connected, meaning that the graph representing the network is a fully connected graph and each region can be reached by all others. We further assume that the drivers follow the shortest route policy, this assumption combined with a fully connected network lead to a simple structure for the routing tensor $\Theta = \{\theta^h_{ij}\}_{ijh}$, the routing tensor is structured in the following way: each matrix $\theta_{i..}$ contains the number of vehicles currently in region i, its rows j contain the fraction of vehicles with destination j and the columns h contain the fraction of these vehicles passing through region h. Thus our assumption leads to a routing vector in which every matrix $\theta_{i..}$ is an identity matrix, as each destination region can be reached from all others and by taking the shortest route we don't need to cross a third region h. The simulation is a "day" composed on 480 timesteps. The MFDs are randomly generated and represented in 7. The demand is similarly random generated, we chose to randomly select a peak time for each region inside an interval $[T_{start}, T_{end}]$ so that all the regions peaked around the same time but not exactly overlap with each other, the demand has been modeled as a simple piecewise continuous

functions of this form,

$$q_{ij}(t) = \begin{cases} c_i & t \leq T^i_{start} \\ a_i t + b_i & T^i_{start} < t \leq T^i_{peak} \\ -a_i t + (c_i - a_i T^i_{end}) & T^i_{peak} < t \leq T^i_{end} \\ c_i & t > T^i_{end} \end{cases}$$

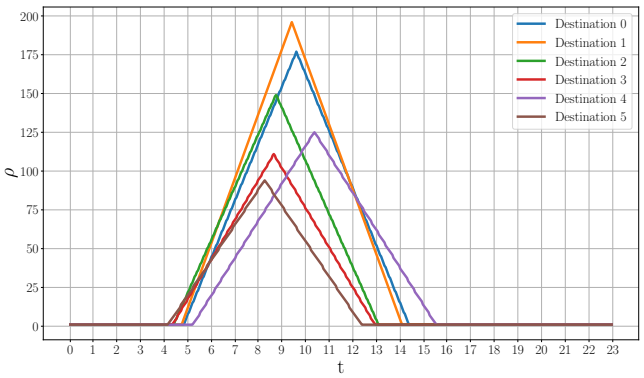
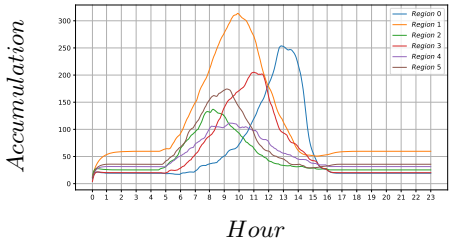


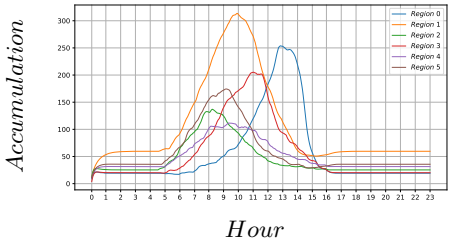
Figure 9: The Demands Originated in Region 1

4.1.1 Results of The Numerical Example

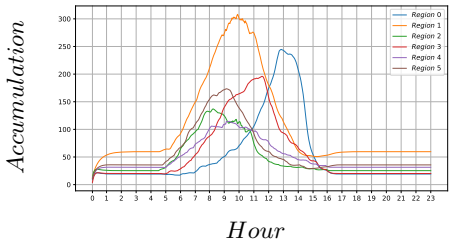
Below we show the results of the DeePC and LMPC algorithm applied in the numerical simulation setting, in two different conditions, an uncongested setting and a congested one in which region 0 experiences a grid lock situation if not managed. We consider in both cases an horizon $N_p = 3$ and for DeePC we select the two parameters λ_{ini} and λ_g based on a one dimensional error study conducted on both parameters



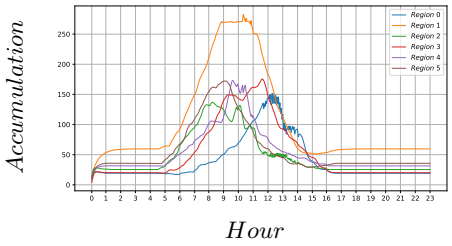
(a) No Control Uncongested



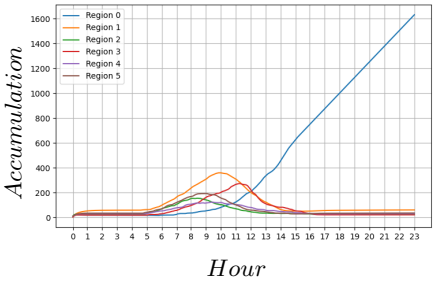
(b) LMPC Uncongested $N_p = 3$



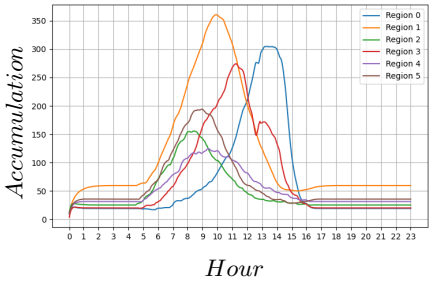
(c) LMPC Uncongested $N_p = 10$



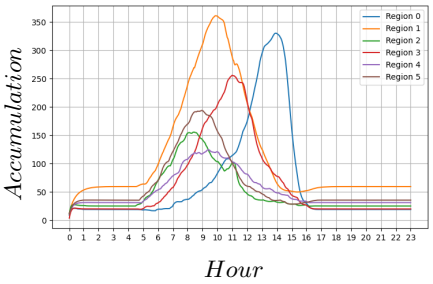
(d) LMPC Uncongested $Np = 20$



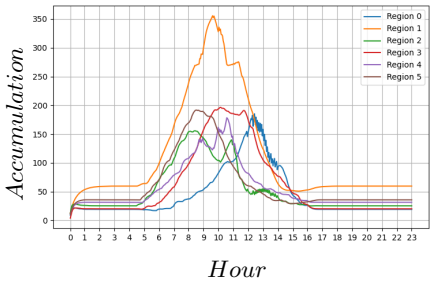
(e) No Control Congested



(f) LMPC Congested $N_p = 3$



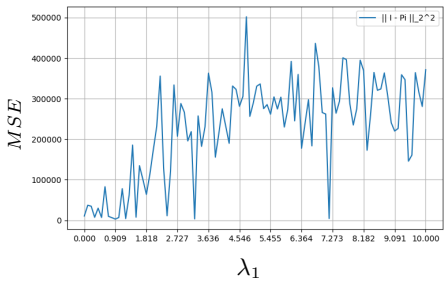
(g) LMPC Congested $N_p = 10$



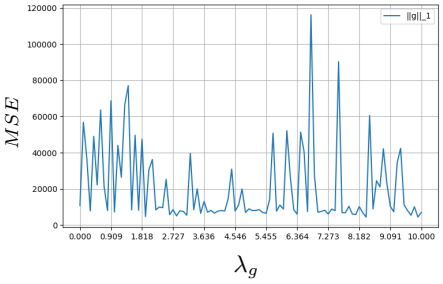
(h) LMPC Congested $Np = 20$

As can be seen, in both cases the performance of LMPC improve with a greater time horizon. Now we show the results of the error study that lead to the choice of the parameters for DeePC, the error measure used is the Mean Square

Error between the accumulation and the critical accumulation point defined by the MFD.

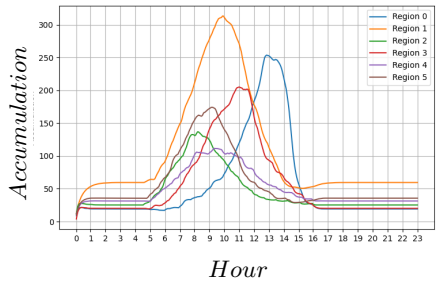


(a) MSE with repect to λ_1

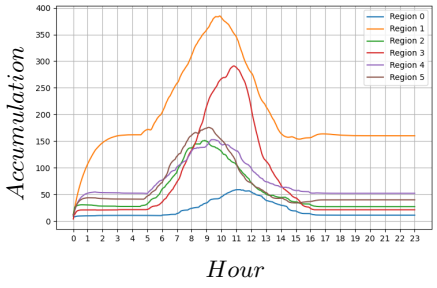


(b) MSE with respect to λ_g

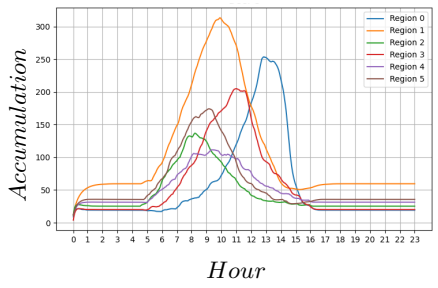
Based on this study we can see that the Error has minimum points at $\lambda_g = 0.1013$ and $\lambda_1 = 7.178$, to understand the effects of the parameters we fix the length of the initial trajectory $T_{ini} = 2$ and the time horizon $N = 1$, then we apply this two parameters to the DeePC algorithm we present three cases $(\lambda_g, \lambda_1) = \{(0.1013, 0), (0, 7.128), (0.1013, 7.128)\}$



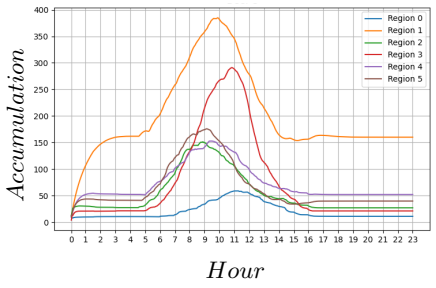
(a) No Control Uncongested



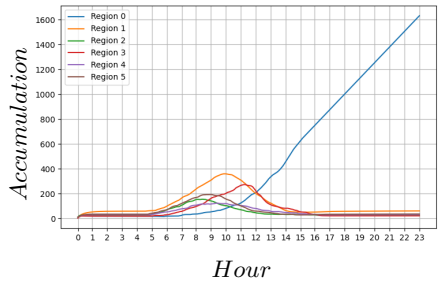
(b) DeePC Uncongested $(0.1013, 0)$



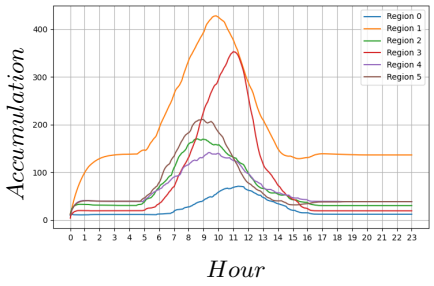
(c) DeePC Uncongested $(0, 7.128)$



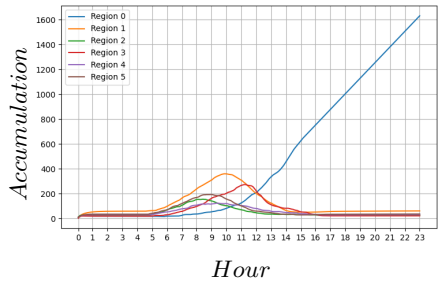
(d) DeePC Uncongested $(0.1013, 7.128)$



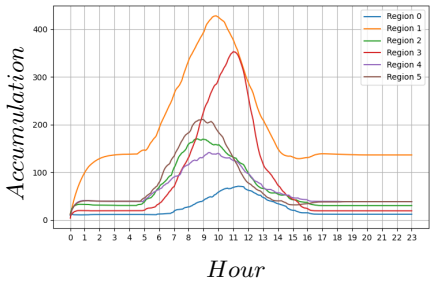
(a) No Control Congested



(b) DeePC Congested $(0.1013, 0)$



(c) DeePC Congested $(0, 7.128)$



(d) DeePC Congested $(0.1013, 7.128)$

This shows that the selection of the tuning of the parameters, together with the Hankel matrix construction, can affect the results significantly, however once the parameters have been found the algorithm can handle the non-linearity in our system in both cases with small time horizon $N = 1$.

4.2 Zürich Simulation

4.2.1 Results of the Clustering

Below we show the clustering obtained on the city of Zürich using the Snake Algorithm and the Post-Processing procedure.

4.2.2 The Regions of Zürich

The following partitioning of the city of Zürich has been obtained on the evening traffic demand peak modeled in [25] and on which later we will deploy our control algorithms, the observation period is 6 hours long, starting at 15:00 PM ending at 21:00 PM, various number of clusters have been tried, we choose 6 as final number of cluster as it shows clearly the main areas recognized by the municipality of city of Zürich.
The difference between the clustering taken on the simulation of the morning peak can be seen in A. In Figure 14 we can see the clustering without any post-processing, the areas of Wiedikon (red), Üetliberg (purple),Enge (yellow),Universität - Züriberg (blue), Oerlikon-Affoltern-Seebach (green), Dietikon-Opfikon (black).

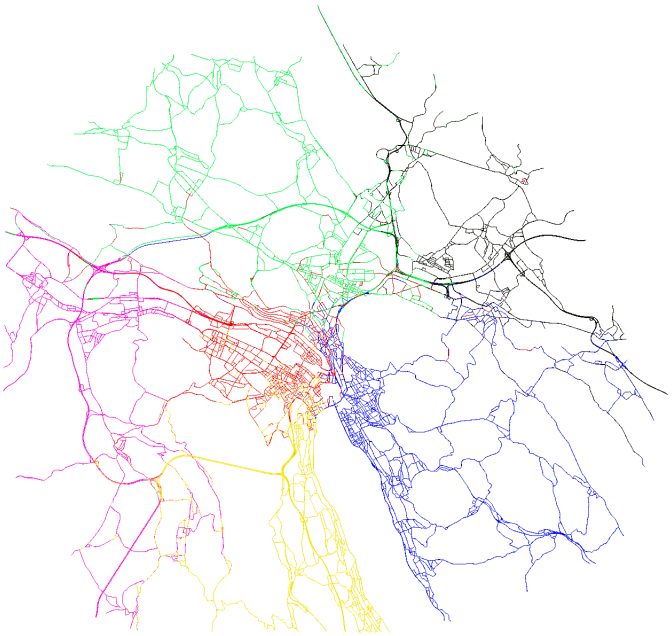


Figure 14: The Clustering of Zürich during the evening peak, no Post-Processing

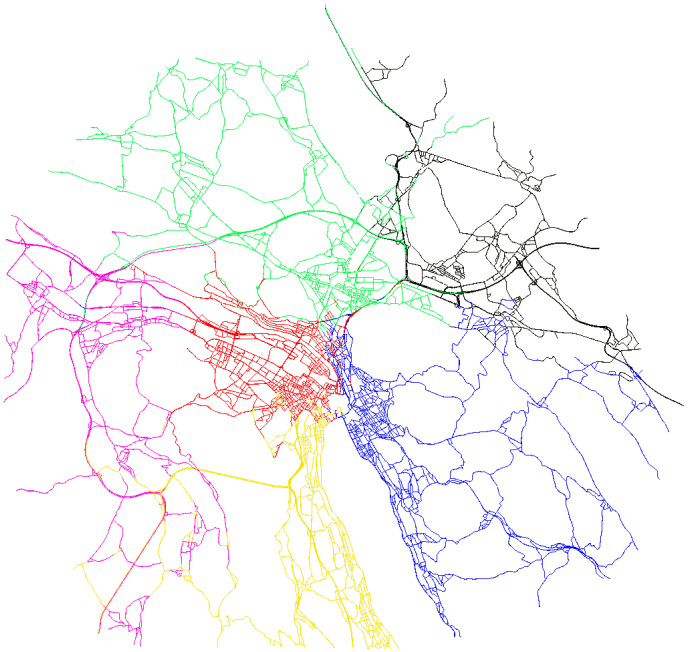


Figure 15: The Clustering of Zürich after 20 iterations of the Procedure

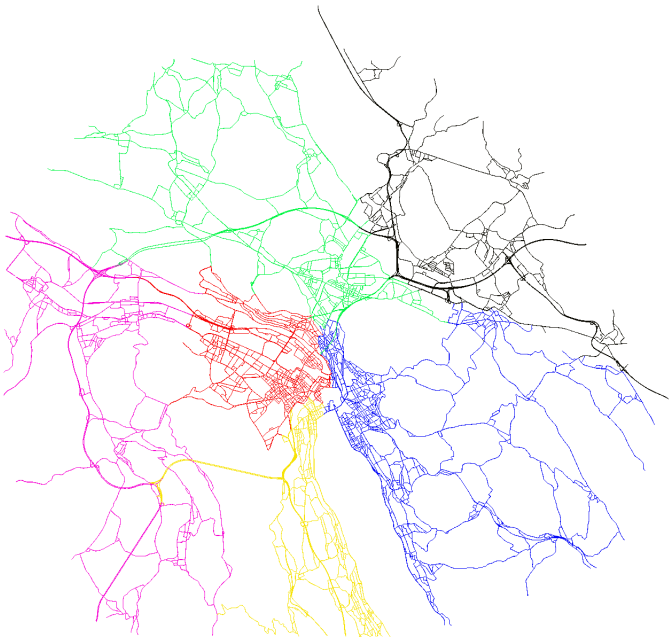


Figure 16: The Clustering of Zürich after 50 iterations of the Procedure

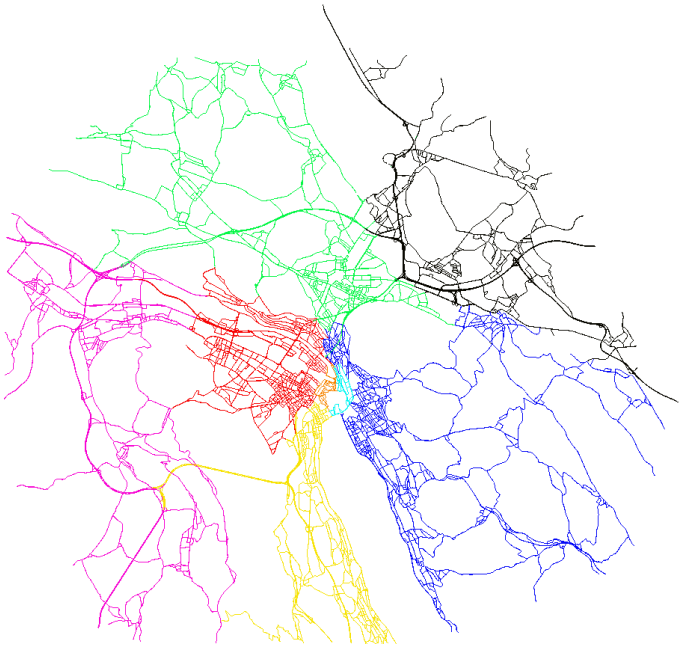


Figure 17: The Clustering of Zürich with the center added.

4.2.3 Results of the Control

Thanks to the encouraging results obtained on the numerical simulation, we now proceed at comparing the two algorithms on the real scale simulation of Zürich. We compare the algortihms using various metrics metrics obtained as an average over all the vehicles that appeared in the simulation. The travel time metric is useful as it summarises the performance of the algorithm as it is highly correlated with all the other metrics, in particular the fuel consumption and the waiting time.
Both the algorithms have a time horizon of $N = 4$ which corresponds to 288 seconds of simulation time, the parameters used for the DeePC algorithm are $(\lambda_g, \lambda_1) = (1, 1)$ without additional tuning performed. We again consider two cases, an uncongested situation in which the density of the regions remains quite low and a congested situation in which the central regions of the city are near grid lock.

Model Comparison Uncongested Setting

	DeePC	LMPC	NoControl
Travel Time (min)	**26.5**	28.1	29.1
Waiting Time (min)	**13.9**	15.3	16.1
CO Emitted (kg)	**0.179**	0.194	0.203
CO2 Emitted (kg)	**4.348**	4.599	4.745
HC Emitted (kg)	**0.00091**	0.00099	0.001035
PMx Emitted (kg)	**0.000093**	0.000096	0.0001032
NOx Emitted (kg)	**0.00185**	0,00197	0.00203

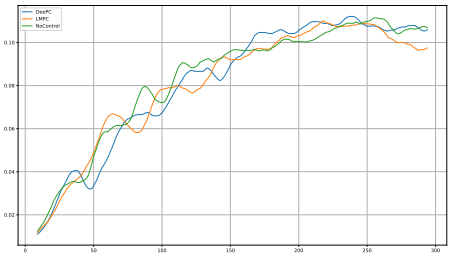
Model Comparison Congested Setting

	DeePC	LMPC	NoControl
Travel Time (min)	**84.0**	85.1	86.8
Waiting Time (min)	**59.2**	59.7	60.7
CO Emitted (kg)	**0.698**	0.705	0.719
CO2 Emitted (kg)	**13.239**	13.414	13.680
HC Emitted (kg)	**0.003487**	0.003526	0.003596
PMx Emitted (kg)	**0.000310**	0.000314	0.000320
NOx Emitted (kg)	**0.00589**	0.00597	0.00609

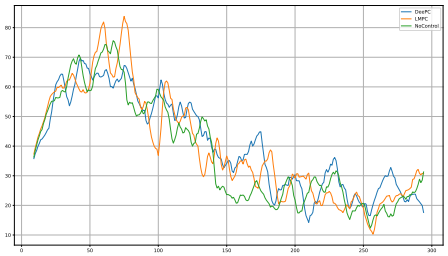
In particular in the uncongested simulation 167703 vehicles were produced resulting in a total of 44,989 hours spent waiting and 795,904 kilograms of CO2 emitted. Using LMPC the hours spent waiting were 42,894 and the CO2 emitted has been 772,456 kilograms resulting in 2,095 hours and 23,448 kilograms of CO2 saved, respectively an improvement of 4,6% and 2,9% with respect to NoControl. DeePC beated both NoControl and LMPC resulting in 39,103 hours spent waiting and 730,182 kilograms emitted, saving 5,886 hours and $\mathbf{65,722}$ kilograms of CO2, respectively an improvement of $\mathbf{13,0\%}$ and $\mathbf{8,2\%}$ with respect to NoControl.
In the congested setting the total of vehicles produced has been 207,178 a 25% increase with respect the uncongested case, it resulted in 209,701 hours spent waiting and 2,834,343 kilograms of CO2 emitted. Under the LMPC control the

simulation resulted in 206,920 hours spent waiting and 2,785,689 kilograms of CO2, that is 2,781 hours and 48,654 kilograms of CO2 saved, respectively an improvement of 1,3% and 1,7% improvement with respect to NoControl.
DeePC beated both NoControl and LMPC resulting in 205,278 hours spent waiting and 2,750,937 kilograms of CO2 emitted, that is 4,423 hours and **83, 406** kilograms of CO2 saved, respectively an improvement of **2, 1%** and **2, 9%** with respect to NoControl.

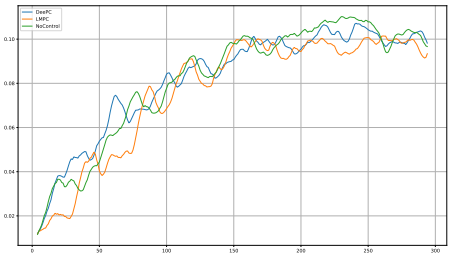
Below the graphs representing the time series of the flow and density for some of the regions under the optimization algorithms can be seen, the time series presented have been smoothed by using a Moving Average filter of order 10 .



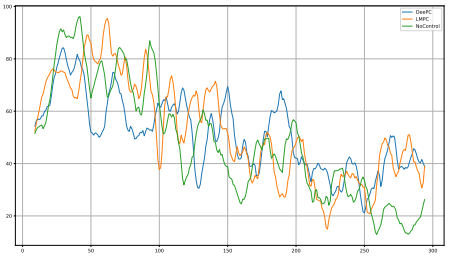
(a) Density City Center 1 Altstadt-Wiedikon (orange)



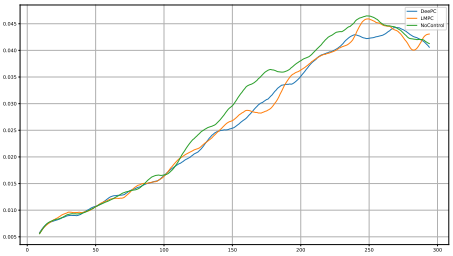
(b) Flow City Center 1 Altstadt-Wiedikon (orange)



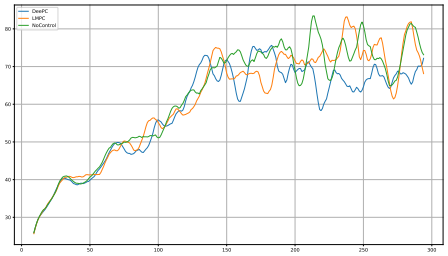
(c) Density City Center 2 Altstadt-Universität (cyan)



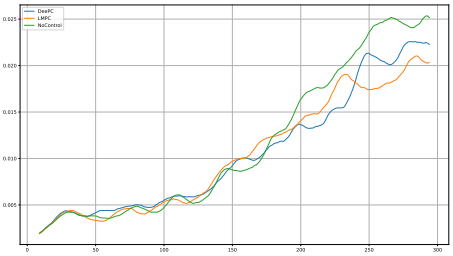
(d) Flow City Center 2 Altstadt-Universität (cyan)



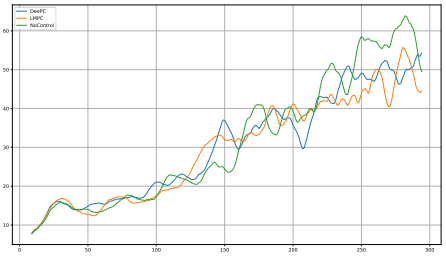
(e) Density Oerlikon-Seebach (green)



(f) Flow Oerlikon-Seebach (green)



(a) Density Dietikon-Opfikon (black)



(b) Flow Dietikon-Opfikon (black)

5 Conclusion

Data-Driven Predictive Control is a suitable approach for optimizing traffic flows in cities, the algorithm beats the state of the art Linear Model Predictive Control in terms of average travel time and emissions.
The better performance of DeePC are explained by the absence of the intermediate mapping between the outputs of the algorithm and the actuators of the simulation.
This mapping is a phase of modelling needed for LMPC and results in multiple actuators being assigned the same input as they control the same flow, for DeePC this step is not needed, resulting in a personalized input for each actuator and a more granular control. The reduction of the improvement in the congested case is due to the fact that when approaching grid-lock both the algorithms have less effective choices to better the situation.
Moreover the implementation and deployment of DeePC is much cheaper and less time expensive as most of the modelling steps are not required, yielding better results even in highly non-linear settings as already confirmed empirically by the relevant literature. We've also presented the first attempt at controlling a simulation of the entire city of Zürich resulting in a highly detailed road-by-road clustering which closely resembles the areas used by the municipality. Further research directions include attempting to ease the computational cost of DeePC which for large input spaces becomes a bottleneck of the procedure, a way to do this would be to devise a distributed version of the algorithm or iteratively approximate the inputs. Finally another research direction remain to be discussed, how to retrieve persistently exciting data from a city without impacting the lives of its inhabitants. A possibly highly effective way to tackle this would be to retrieve the data from a realistic simulation, as the one used in this setting and the fine tune the algorithm in the real case. Another possible avenue would be to sacrify the randomness of the input used in the Hankel matrix construction and instead using sinusoidal inputs which shouldn't affect the traffic dynamics too radically, more work is needed in order decide which is the best approach.

A Appendix

A.1 Fundamental lemma of behavioural system theory

the Fundamental lemma sits at the heart of behavioural system theory, it was first proved by J.C. Willems in [22], it replaces the need of a model or system identification process and allowas any trajectory of a controllable LTI system to be constructed using a finite number of data samples generated by a persistently exciting input signal. In a sense, the Hankel matrix itself is itself a non-parametric predictive model based on raw data. it allows on to implicitly estimated the state of an LTI system, predict its future behaviour and design optimal feedforward control inputs. Below we give the proof of this cardinal result

Fundamental Lemma of BT. We only need to prove that

$$leftkernel(\mathscr{H}_L(\boldsymbol{\eta})) = \mathscr{R}_{\mathscr{B}}^{L-1}$$

The inclusion $\mathscr{R}_{\mathscr{B}}^{L} \subseteq leftkernel(\mathscr{H}_L(\boldsymbol{\eta})$ is obvious. Consider the reverse inclusion $leftkernel(\mathscr{H}_L(\boldsymbol{\eta}) \subseteq \mathscr{R}_{\mathscr{B}}^{L}$. Assume by contrary, that

$$0 \neq r^\top = [r_0^\top, r_1^\top, \dots, r_{L-1}^\top] \in leftkernel(\mathscr{H}_L(boldsymbol\eta))$$

but $r(\xi) = r_0 + r_1 + \cdots + r_{L-1}\xi^{L-1} \notin \mathscr{R}_{\mathscr{B}}^{L-1}$. Consider now $\mathscr{H}_{L+\boldsymbol{n}(\mathscr{B})}(\boldsymbol{\eta})$, obviously we have that $leftkernel(\mathscr{H}_{L+\boldsymbol{n}(\mathscr{B})}(\boldsymbol{\eta}))$ contains $\mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})-1} + R$, with $R \subset \mathbb{R}^{m+p}[\xi]$ being the linear span of

$$R = span\{r^\top(\xi), \xi r^\top(\xi), \dots, \xi^{\boldsymbol{n}(\mathscr{B})} r^\top(\xi)\}$$

Now recall that

$$dim(\mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})-1}) = (L + \boldsymbol{n}(\mathscr{B}))\boldsymbol{p}(\mathscr{B}) - \boldsymbol{n}(\mathscr{B})$$

Clearly $dim(R) = \boldsymbol{n}(\mathscr{B}) + 1$, we now shoe that the persistency of excitation assumption implies $R \cap \mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})} \neq \{0\}$, then

$$dim(\mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})-1} + R) = (L + \boldsymbol{n}(\mathscr{B}))\boldsymbol{p}(\mathscr{B}) + 1$$

But the persistency of excitation implies

$$rank(\mathscr{H}_{L+n(\mathscr{B})}(\boldsymbol{\eta}) \geq (L + \boldsymbol{n}(\mathscr{B}))\boldsymbol{m}(\mathscr{B})$$

Hence

$$\begin{aligned} dim(\mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})-1} + R) &= (L + \boldsymbol{n}(\mathscr{B}))\boldsymbol{p}(\mathscr{B}) + 1 \\ &\leq dim(leftkernel(\mathscr{H}_{L+\boldsymbol{n}(\mathscr{B})}(\boldsymbol{\eta}))) \\ &\leq (L + \boldsymbol{n}(\mathscr{B}))\boldsymbol{p}(\mathscr{B}) \end{aligned}$$

Therefore $R \cap \mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})} \neq \{0\}$ Consequently there exists a linear combination of

$$r^\top(\xi), \xi r^\top(\xi), \dots, \xi^{\boldsymbol{n}(\mathscr{B})} r^\top(\xi)$$

that is contained in $\mathscr{R}_{\mathscr{B}}^{L+\boldsymbol{n}(\mathscr{B})}$. In terms of the minimal kernel representation $K(\frac{d}{dt})w = 0$ of $\mathscr{B}$, this means that there is $0 \neq f \in \mathbb{R}[\xi]$, such that $fr = FK$,

for some $0 \neq F \in \mathbb{R}^{1\times rowdim(K)}[\xi]$. if $deg(f) \geq 1$, then ther is $\lambda' \in \mathbb{C}$, such that $f(\lambda') = 0$, hence $F(\lambda')K(\lambda') = 0$. Now we use the fact that $K(\sigma)\eta = 0$ of $\mathscr{B}$ is a minimal kernel representation of a controllable behaviour if and only if $K(\lambda)$ has full row rank for all $\lambda \in \mathbb{C}$. Hence controllability implies $F(\lambda') = 0$. This implies that each element of F have a common root λ'. Cancel this common factor. Proceed until $deg(f) = 0$. But then $r = FK$. This contradicts the initial assumption $r^\top \notin \mathscr{R}_{\mathscr{B}}^{L-1}$. Hence $leftkernel(\mathscr{H}_L(\eta)) \subseteq \mathscr{R}_{\mathscr{B}}^{L-1}$. □

A.2 More details on the obtained clustering

The snake clustering algorithm has the nice property of being able to identify the direction of the flows in a traffic system, this is important as MFD based approaches are sensible to this directionality. To understand why consider a simple intersection between two roads 20 in which only one side of the intersection is experiencing congestion.

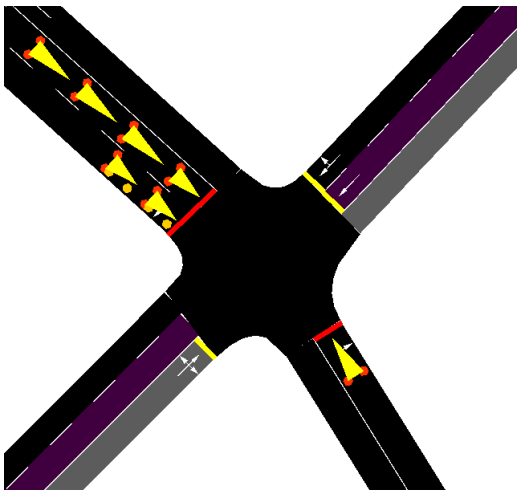


Figure 20: One sided congested

If we consider the MFD of the whole intersection, we would find that the effect of the average density of the congested an uncongested side balance out, giving us a wrongly positive leaning view of the situation by looking at the resulting MFD. This can be accounted for in different ways, one is using a dynamical MFD which modifies it's shape as the day developes. Another is to use different partitioning based the time of the day, which accounts for the different directionalities of these flows, the snake clustering algortihm offers the possibilities of finding such clusters. Below we compare the clusters obtained during the morning and evening simulated peaks in demand to appreciated the differences between them

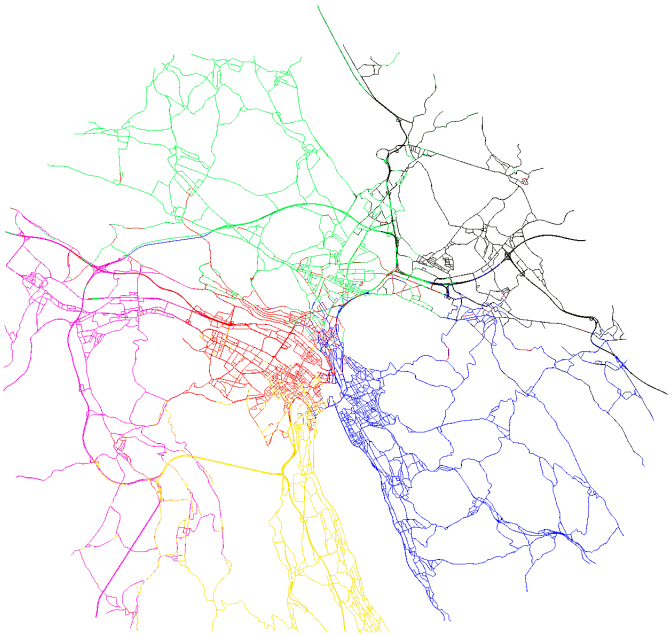


Figure 21: The Clustering of Zürich during the evening peak

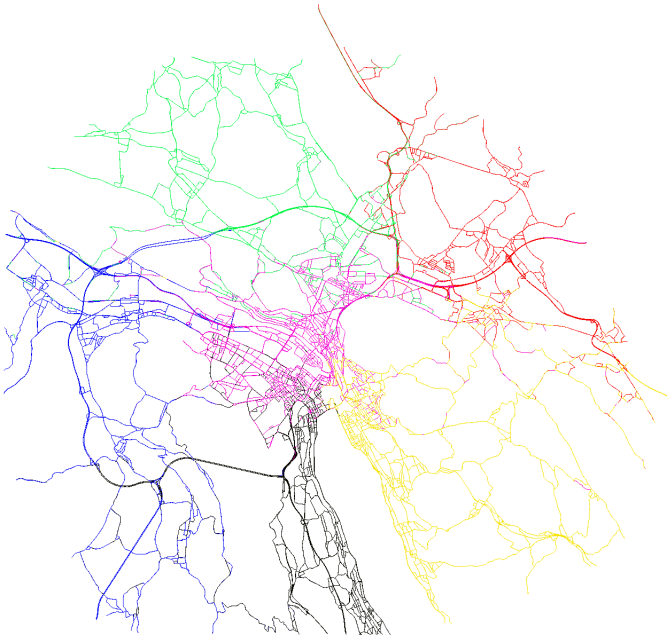


Figure 22: The Clustering of Zürich during the morning peak

As we can see while most of the outer regions remain the same the areas of Oerlikon and Wiedikon tend to change substantially, it appears that in the morning peak a consistent part of the traffic flows is coming from the Oerlikon and converging in the northern part of Zürich before entering the city. This

has a clear reflection in the real case as during the morning many commuters move from the outern parts of Zürich towards the cities working areas such as Wiedikon.

References

[1] N. Geroliminis, M. Saeedmanesh, and K. A., "A linear formulation for model predictive perimeter traffic control in cities," 2017.

[2] J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control: In the shallows of the deepc," in *Proc. Eur. Control Conf.*, Naples, Italy, 2019, pp. 307–312.

[3] J. Coulson, F. Dörfler, and I. Markovsky, "Bridging direct indirect data-driven control formulations via regularizations and relaxations," 2021.

[4] N. Geroliminis and I. Sirmatel, "Economic model predictive control of large-scale urban road networks via perimeter control and regional route guidance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 4, 2018.

[5] N. Geroliminis, J. Haddad, and M. Ramezani, "Optimal perimeter control for two urban regions with macroscopic fundamental diagrams: A model predictive approach," *IEEE Transactions on Intelligent Transportation Systems*, 2012.

[6] N. Geroliminis and S. I., "Stabilization of city-scale road traffic networks via macroscopic fundamental diagram-based model predictive perimeter control," *Control Engineering Practice*, vol. 109.

[7] Y. Ren, Z. Hou, and G. N. Sirmatel, I.I., "Data driven model free adaptive iterative learning perimeter control for large scale urban road networks," *Transportation Research Part C*, 2020.

[8] T. Lei, Z. Hou, and Y. Ren, "Data-driven model free adaptive perimeter control for multi-region urban traffic networks with route choice," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, 2020.

[9] J. Wang, Y. Lian, Y. Jiang, Q. Xu, K. Li, and C. Jones, "Distributed data-driven predictive control for cooperatively smoothing mixed traffic flow," *Transportation Research Part C*, vol. 155, 2023.

[10] M. Hirche, J. Berberich, M. Müller, and F. Allgöwer, "Data-driven distributed mpc of dynamically coupled linear systems," 2022.

[11] C. Chen, Y. Huang, W. Lam, T. Pan, S. Hsu, A. Sumalee, and R. Zhong, "Data efficient reinforcement learning and adaptive optimal perimeter control of network traffic dynamics," *Transportation Research Part C*, vol. 142, 2022.

[12] J. Yoon, S. Kim, Y. Byon, and H. Yeo, "Design of reinforcement learning for perimeter control using network transmission model based macroscopic traffic simulation," *Plos One*, 2020.

[13] N. Geroliminis and D. C. F., "Macroscopic modelling of traffic in cities," *86th Annual Meeting Transportation Research Board*, vol. 42, 2007.

[14] ——, "Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings," *Transportation Research Part B*, 2008.

[15] N. Geroliminis and J. Sun, "Properties of a well-defined macroscopic fundamental diagram for urban traffic," *Transportation Research Part B*, vol. 45, pp. 605–617, 2011.

[16] M. Saeedmanesh and N. Geroliminis, "Clustering of heterogeneous networks with directional flows based on "snake" similarities," *Transportation Research Part B*, vol. 91, pp. 250–269, 2016.

[17] W. J.C., "The behavioural approach to open and interconnected systems," 2007.

[18] J. C. Willems, "From time series to linear system—Part I. Finite dimensional linear time invariant systems," *Automatica*, vol. 22, no. 5, pp. 561–580, 1986.

[19] ——, "Paradigms and puzzels in the theory of dynamical systems," *IEEE Transactions on Automatic Control*, vol. 36, no. 3, 1991.

[20] ——, "From time series to linear system—Part III. Approximate Modelling," *Automatica*, vol. 23, no. 1, pp. 87–115, 1987.

[21] I. Markovsky and F. Dörfler, "Identifiability in the behavioral setting," *IEEE Transactions on Automatic Control*, vol. 68, no. 3, 2023.

[22] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. M. De Moor, "A note on persistency of excitation," *Syst. Control Lett.*, vol. 54, no. 4, pp. 325–329, 2005.

[23] "Tomtom traffic index," https://www.tomtom.com/traffic-index/ranking/, accessed: 2023-09-10.

[24] "Simulation of urban mobility : Sumo," https://sumo.dlr.de/docs/index.html, accessed: 2023-09-11.

[25] L. Ambühl, "Zürich highly detailed simulation," *Transcality : https://transcality.com/*.

[26] N. G. Eissfeldt, "Vehicle-based modelling of traffic : Theory and application to environmental impact modelling," Ph.D. dissertation, Universität zu Köln, 2004.

[27] L. Ambühl, A. Loder, M. Menendez, and K. W. Axhausen, "A case study of zurich's two- layered perimeter control," in , Wien, Österreich, 2018.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

FIXING THE TRAFFIC IN ZÜRICH:
A DATA-DRIVEN CONTROL APPROACH

Authored by (in block letters): RIMOLDI ALESSIO

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
21/09/2023	Rimoldi Alessio

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.