Universidad de Sevilla

Departamento de Ingeniería de Sistemas y Automática

Doctoral Thesis

Data based predictive control: Application to water distribution networks

Author:

José Ramón Salvador Ortiz

Advisors:

Dr. Daniel Rodríguez Ramírez

Dr. David Muñoz de la Peña Sequedo

Acknowledgements

Four years ago a different stage in my academic life began and this thesis is the result of the hard work carried out during this period. First of all, I would like to thank my advisors, Daniel Rodríguez Ramírez, David Muñoz de la Peña Sequedo and Teodoro Álamo Cantarero. Their teachings and dedication have made me a better researcher. They are wonderful teachers, marvellous mentors and better people.

Besides, I have to thank my research group GEPOC in the Departamento de Ingeniería de Sistemas y Automática of the University of Sevilla. I would like to mention specially Daniel Limón, Ignacio Alvarado and Mario Pereira for their help and advices with several issues in the development of my thesis and, in particular, with the quadruple-tank plant. In addition, I would like to thank Jose María Manzano and Pablo Krupa for their help with all the small issues and concerns of the PhD student life.

It is important to thank Marcello Farina and Enrico Terzi for their hospitality and their collaboration during my stay in the Politecnico di Milano. Thanks also to Ye Wang for the work done together and his help with EPANET during his several visits to Sevilla.

Finally, I would really like to thank my family for their support and encouragement, because everything I am, I owe to my parents. Thanks to my brother for being always there when I need him. Also I would like to dedicate this thesis to my wife for her patience, dedication and love. I hope that all of them will be very glad about this achievement.

Jose Ramón Salvador Ortiz
Sevilla, 2019

Contents

Notation

$v^\top$	Transposed of v.
$A^\top$	Transposed of A.
$\|v\|$	Euclidean norm of v.
$\|v\|_\infty$	Infinity norm of v.
$\|v\|_A$	A-norm of v, i.e. $\|v\|_A = \sqrt{v^\top Av}$.
$\hat{v}(t+k)$	Estimation of vector $v(t+k)$ in time t, i.e. $\hat{v}(t+k) = v(t+k\|t)$.
I_n	Identity matrix of order n.
$0_{a,b}$	Matrix of dimensions $a \times b$ with all entries equal to 0.
$\mathbf{1}_n$	Vector with n entries equal to 1.
$A \otimes v$	Kronecker product.
$\mathcal{A} \oplus \mathcal{B}$	Minkowski sum.
$\mathcal{A} \ominus \mathcal{B}$	Pontryagin difference.
$A_{i\bullet}$	i-th row of A.
$A_{\bullet i}$	i-th column of A.

Chapter 1

Introduction

This chapter presents the motivation, objectives and outline of this thesis. Furthermore, a literature review of the advanced control methods related with the algorithms developed in the following chapters is provided. Finally, a list of the contributions published in journals and conference proceedings from the work carried out during this thesis is also given.

1.1 Motivation

Control techniques research has always been driven by the necessities of industry. From classical control methods to more advanced control techniques, companies have chased to operate their plants trying to improve different objectives, sometimes conflicting, such as safety, efficiency, reliability and economic optimality. Tackling all these objectives simultaneously in large scale complex systems such as water distribution networks, chemical processes or renewable energy plants, provide a difficult challenge from the control point of view. Thus, it is desirable to develop control strategies to address these issues.

Traditionally, most of the controllers employed in industry are classical PIDs because of their simplicity and reliability. However, advance control techniques have the potential to take into account multiple objectives and operate a system in a more efficient and safe way. Most advanced control techniques, like model predictive control (MPC) which is widely extended in industry, are based on a model of the system which is used to make predictions and estimations of its closed-loop behaviour. High quality models can be obtained from first principles or input/output analysis techniques, however model identification is, generally speaking, a hard problem to cope with when dealing with large scale nonlinear systems. In general, a simplified model of a complex system can provide poor predictions of its behaviour while a high quality complex models may not be appropriate for controller design tasks.

Nowadays, the use of large amounts of data is present in most scopes of our lives. Internet and new technologies have allowed us to manage a huge amount of information while new devices with environment inter-actuation, interconnection capabilities and bigger storage capacity have helped us to produce and store information data of almost everything in our daily life. In industry, although the incorporation of this class of technologies has taken some time to be considered, the use of data can provide a paradigm shift. At the present time, it is common to find numerous different sensors, reading every measurable

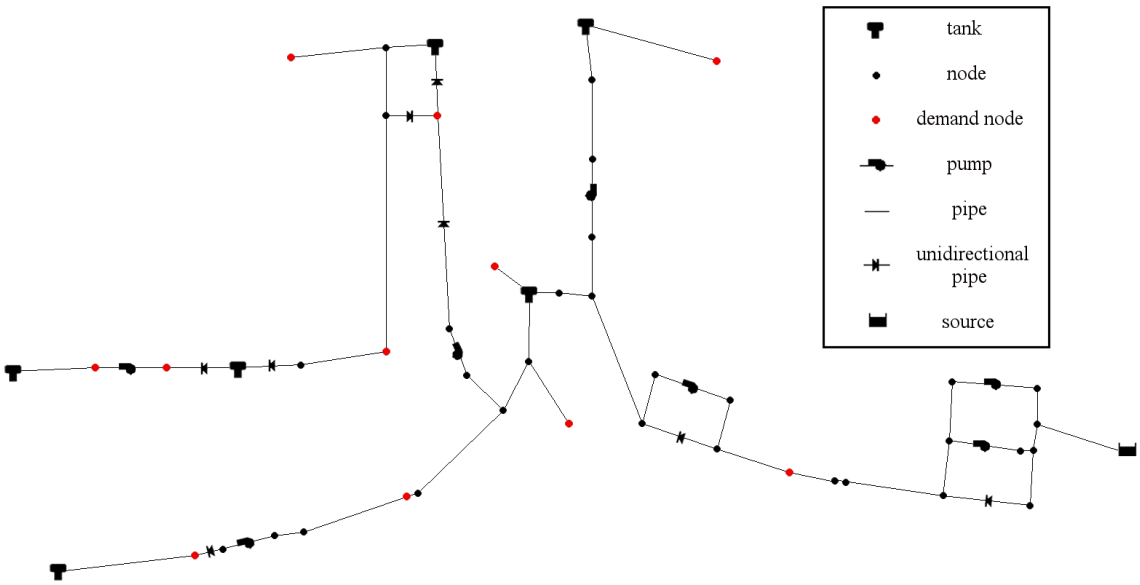


Figure 1.1.1: Richmond water distribution network diagram.

aspect of an industrial process, interconnected with other devices. The large amount of available historian data has the potential to provide new methods to improve performance and other objectives. In recent years, the research community has been interested in using the information that can be obtained from this data. Even if the plant is controlled by a classical and simple technique, the stored data can be very useful for understanding the system dynamics behaviour, for estimating disturbances signals and for identification proposes. The literature collects all these strategies under the broad concepts of *data-driven/based* and *machine learning* methods.

Water distribution networks (WDN) pose control challenges due to their size, diverse nature of the process and manipulated variables and disturbance rich operating conditions. These networks are an example of large scale complex systems for which data can be obtained and used to improve its operation. As we can see in figure 1.1.1, where a simplified model of Richmond's WDN [111] is shown, this kind of system is a complex network with a high number of tanks, pipes, nodes and pumps that can be found in every city of the world. Usually, the dynamics that define the pipe flows and pressures are nonlinear and their actuators, in most cases, are binary. The large number of sensors typically found in these systems, can provide us a huge amount of process data. Furthermore, client demands, generally considered as disturbances, and sensors measurement noises, are uncertainties hard to model but of which historic data is in general available.

Motivated by these issues, the development and application of data based predictive control algorithms for complex systems, such as water distribution networks, in which historic data is used instead of standard modelling and identification approaches, has been carried out during this thesis. The objective of these algorithms is to improve the performance of model-based methods like MPC using historical data, tackling the identification problem from different point of view.

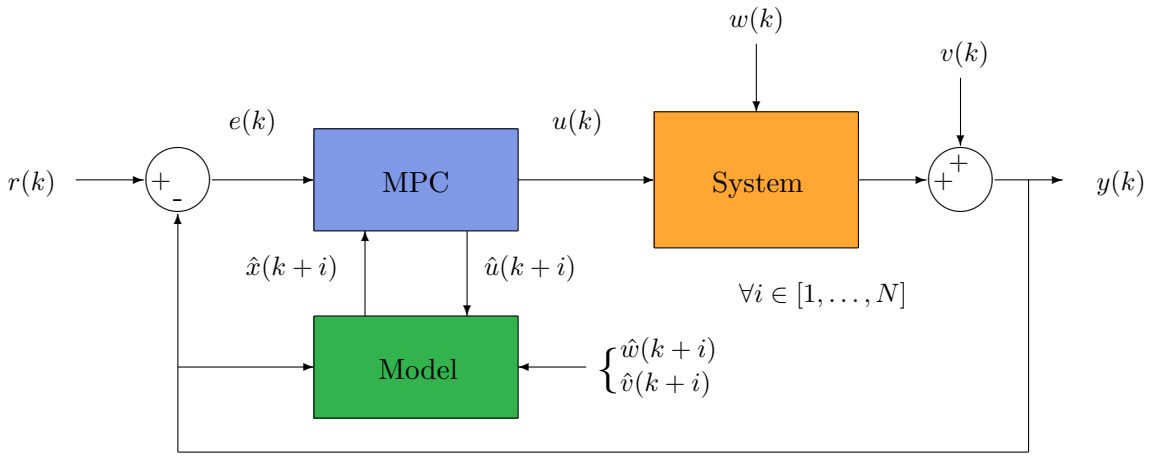


Figure 1.2.1: Standard MPC scheme.

1.2 Model predictive control

Model predictive control (MPC) [25, 62] is one of the most successful forms of feedback control due to its ability to control almost any kind of process while considering operating constraints explicitly. MPC techniques rely on the use of a model to predict the evolution of the system state over a prediction horizon and thus compute the values of the control signals that minimize a performance cost. Therefore, the use of a model is a key concept in these techniques. The theoretical bases have been well established for both linear and nonlinear systems [88] and, in many cases, real time implementation is possible on a wide range of applications, from process to automotive industries. These are some of the reasons that explain why MPC is more spread in the industry than any other form of modern control. Figure 1.2.1 shows the scheme of a MPC controller in closed-loop with a system, in which the control action $u(k)$ is applied in receding horizon manner. In this diagram a reference signal $r(k)$ for the output $y(k)$ is considered. The system has state disturbances $w(k)$ and output noise $d(k)$. Note the importance of the model in order to make predictions.

In a general case without disturbances, MPC considers discrete models that estimate the next state value with the current state and input as follows:

$$x(k+1) = f(x(k), u(k)),$$

where k is the sample time, $x(k) \in \mathbb{R}^{n_x}$ is the state vector, $u(k) \in \mathbb{R}^{n_u}$ is the control action vector and $f(\cdot, \cdot)$ represents a set of equations that model the dynamic behaviour of the system.

Another ingredient of MPC is the cost function V, which describes the criteria to optimize. This function is usually definite positive and can consider closed-loop operation costs, economic costs, model error minimization or a combination of them. General optimal control formulations are based on a minimization problem that takes into account an infinite number of steps in the future. In MPC however, only a finite amount of them are typically considered in order to define a optimization problem that can be solved

numerically. This is known as the prediction horizon N. It is very common in literature to divide the cost function in to terms: stage cost and terminal cost. Given an initial state $x(k)$, a general discrete cost function formulation with a finite prediction horizon is:

$$V(x(k), \mathbf{u}) = \sum_{i=0}^{N-1} \ell(\hat{x}(k+i), \hat{u}(k+i)) + \ell_N(\hat{x}(k+N)),$$

where $\hat{x}(k+i) = x(k+i|k)$ and $\hat{u}(k+i) = u(k+i|k)$ are the state and control action prediction (both made from instant k) of $x(k+i)$ and $u(k+i)$ respectively, $\mathbf{u}$ is the sequence of predicted control action vectors from $\hat{u}(k)$ to $\hat{u}(k+N-1)$, $\ell(\cdot,\cdot)$ is the stage cost, which depends on the i-th predicted state and control action vectors with $i \in \{0, \ldots, N-1\}$, and $\ell_N(\cdot)$ is the terminal cost, which depends on the N-th predicted state vector $\hat{x}(k+N)$.

Constraints are the last component considered in this sort of strategy. They are a set of mathematical equations and inequalities which represents physical limits of the system, i.e minimal and maximal flow of a pump or the lower and higher value of a tank level, or virtual limits imposed to preserve a safe control, to respect dynamic behaviour of the model, to maintain closed-loop stability, etc.

Taking into account a model of the system, a cost function in order to optimize some criteria and the appropriate constraints, MPC poses the following general optimization problem:

$$\begin{aligned} \min_{\mathbf{u}} \quad & V(x, \mathbf{u}) \\ s.t. \quad & h(x, \mathbf{u}) = 0 \\ & g(x, \mathbf{u}) \leq 0, \end{aligned}$$

where x is the current measured state and $h(x, \mathbf{u})$ and $g(x, \mathbf{u})$ are the equality and inequality constraints respectively. Predictions can satisfy the model including it in $h(x, \mathbf{u})$ as equality equations and physical limits, generally defined as sets $x \in \mathcal{X}$ and $u \in \mathcal{U}$, can be added to $g(x, \mathbf{u})$ as inequalities. The solution obtained from MPC optimization problem, denoted as $\mathbf{u}^*$ is usually applied in a receding horizon manner, that is

1. Solve the optimization problem.
2. Apply the first component of $\mathbf{u}^*$.
3. Wait for the next time step in order to read or estimate the state value x.
4. Repeat the procedure.

Assuming a system without disturbances and a perfect model, $\mathbf{u}^*$ will be the sequence of inputs that, starting from $x(k)$, carries the system to $x(k+N)$ in N steps. Nevertheless, in real complex systems with disturbances and noise, MPC approaches can be applied because the receding horizon strategy compensates the errors due to the uncertainties and disturbances. This translates into the necessity of solving an on-line optimization problem every time step and the importance of an accurate model as simple as possible.

As previously commented, there are many MPC approaches depending on the minimization criteria of the cost function or the constraints considered which provide guaranteed closed-loop properties for the system, such as, stability, recursive feasibility, robustness against uncertainties, etc. Some examples are set-point tracking MPC [88], robust MPC [69] or economic MPC [87]. The reviews [67, 68] show the latests advances and MPC state of the art.

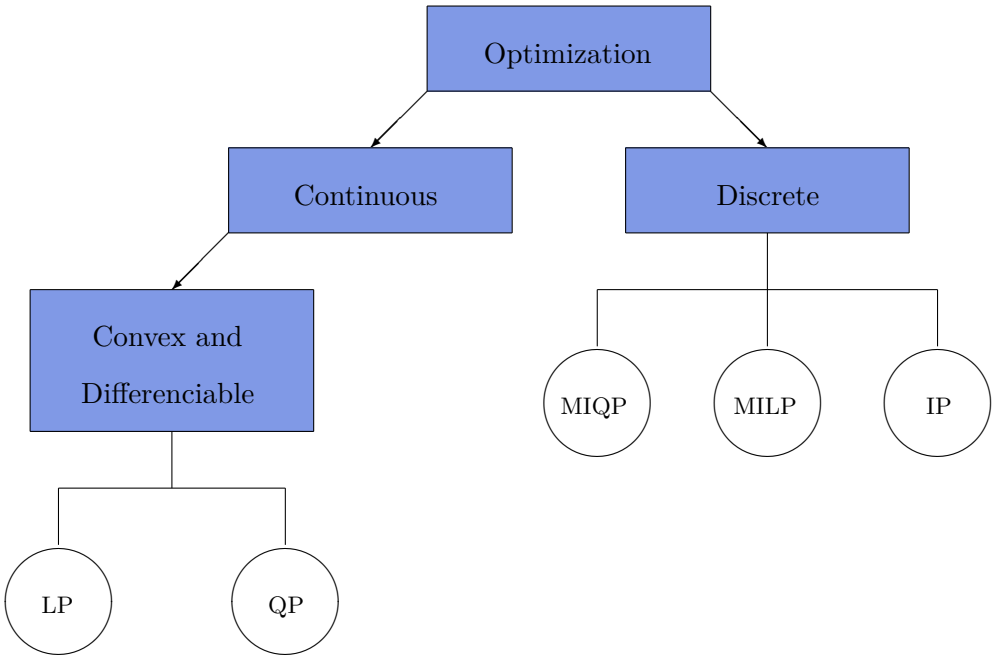


Figure 1.2.2: Classification diagram of optimization problems.

Optimization problems obtained applying MPC techniques can be classified depending on different criteria. We can see in figure 1.2.2 a classification regarding the nature of model variables, constraints properties and cost function qualities. Linear programming (LP) and quadratic programming (QP) problems are usually found in the context of MPC for linear systems with linear and quadratic cost functions respectively. However, sometimes, mixed integer programming (MIP) or integer programming (IP) can be found due to discrete sensors or actuators.

There are cases in which the real time application of MPC can be difficult due to the computational burden associated with the computation of the control law, specially when the model is nonlinear or based on a mix of integer and real variables [8, 26, 32]. For example, this is the situation when an on-off actuator is used. Such actuators appear in the form of pulse width modulation switches, power electronic devices or thrusters in spacecrafts, to name a few examples. Having the control action restricted to a set of discrete values makes the optimization problem associated to MPC much harder to solve.

In order to solve a binary optimization problem it is necessary, in general, to compute all the possible combinations of the control actions along the prediction horizon and select the one with the lowest cost. As the number of combinations grows exponentially with the length of the prediction horizon, this can only be done if the prediction horizon is small as in [41, 98], where MPC of switched power electronics is considered. When the prediction horizon is large, the number of candidate solutions must be reduced by some strategy [105].

This is the case in WDN if the decision variables are the switching of several on-off valves or pumps. An example can be observed in [112] where an operational optimization of the

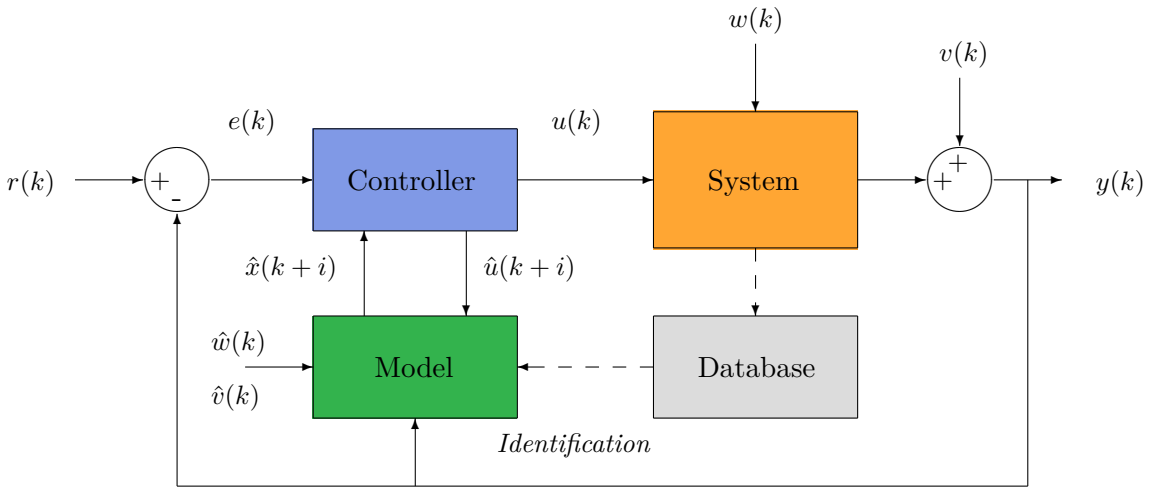


Figure 1.2.3: Data based predictive control diagram. Identifying a model from the data.

binary signals which control the pumps of Richmond's WDN [111] is made. In addition, water demands, which are considered disturbance signals in these networks, usually have a daily periodicity, so, when model predictive methods are used, it is common to control these systems with a large prediction horizon. In [78] we can observe an example of controlling the Barcelona's WDN applying a robust periodic MPC with a prediction horizons equal to a day. Thus, large prediction horizons are needed to obtain good performance with this class of systems.

There are many applications in which obtaining a prediction model is not a problem using standard identification techniques [61]. Figure 1.2.3 shows a general predictive control scheme in which model identification is used. However, there are also large and complex processes for which the task of identifying a good model can be very difficult. Moreover, in those cases, the resulting model (if identified) may be too complex to be used with most MPC techniques, as they may result in an optimization problem difficult to solve on line each sampling time. When first-principle models are not available, procedures to use the measured data, both online and off-line, are required for designing the predictive controller, avoiding the need for initially accurate dynamic models [82]. This situation appears in the control of large infrastructures such as water distribution networks in which simplified models are often used, see for example [22, 76, 81]. For this reason it is necessary to research alternative model identification strategies.

1.3 Data driven inference

This section gives an overview of the generic inference approaches that can be found in the literature. Nowadays, there has been an increasing interest of the MPC community on developing controllers based on machine learning techniques because of the recent theoretical and technological ground-breaking advances in this field.

There exists a wide extension of inference approaches, such as Gaussian process (GP), parametrized models, set membership or direct weight optimization (DWO). These meth-

ods have been applied in different data based control strategies.

The theory of Gaussian processes has been used to derive different learning methods [14, 86] designed to solve regression problems and probabilistic classification. These techniques can be used with identification proposes because they predict by means of kernels which interpolate probabilistically the available data. One of its main advantages is that they can compute confidence intervals with its predictions and make an online/adaptative adjustment of the kernels. Other recent approach derived from this field is Kinky inference [24] that has been applied to MPC in [65].

Parametrized modelling based on data is one of the main tendencies in control. In MPC, different data-driven strategies to obtain and adapt prediction models using ad-hoc identification and adaptive techniques have been proposed [44, 55, 113, 117] which need knowledge of the system to fit the structure of the process model. In [50], a data-driven subspace approach is introduced to design the predictive controller. More recently, reinforcement learning has been studied in [57, 97].

Set membership is another data driven inference method. In order to quantify the mismatch between the model and the real system, knowing the uncertainty of the model is crucial. Against the common assumption in literature that a bound on the parametric uncertainty is available [13, 19, 56, 69], set membership [70] is an identification method that derive this bound with noisy input-output data of the real system. In this approach, data is reorganized in regressors with the size given by the assumed order of the system, which establishes predictor dimension too. Then, several LP problems are posed and have to be solved in order to find the predictor that minimizes this uncertainty bound. In recent years, a number of contributions have addressed this problem exploiting set membership techniques [47, 71], that are promising in this context since they allow to quantify the model uncertainty from data [73].

Direct weight optimization methods provide low computational burden algorithms[1] [90, 91, 92] carried out using affine combinations of locally weighted past data. Other works related with this methodology are [21, 20], in which predictors based on bounding techniques are considered. These methods will be used in this thesis to define different strategies. A brief summary of the fundamental concepts of this approach is presented in the following.

Given data obtained from an unknown system $\{\phi(t), y(t)\}_{t=0}^{n_D}$, where

$$y(t) = f(\phi(t)) + e(t)$$

is the output vector, $f(\cdot)$ is not known, $\phi(t)$ is the regression vector and $e(t)$ is a disturbance signal, the goal is to obtain a predictor of the system $\hat{f}(\phi)$. To find it, one of the simplest ideas is to consider a linear predictor as

$$\hat{f}(\phi^*) = w_0 + \sum_{t=1}^{n_D} w_t y(t)$$

at a given point ϕ^*. The performance of predictors depends on the selection of w_0 and w_t weights. The problem of finding the value of these weights is called the direct weight optimization problem.

[1] Notice that the computational burden increases with the amount of data taken into account.

This class of algorithms have been used for example in lazy learning approaches [6, 7], in which training is deferred until a query is to be answered. Other works that have considered similar ideas in control applications include [40], in which local weighted projection regression is used together with partial least squares combined with a predictive controller, [45] where local learning is used in a data driven control method that performs a model free dynamic linearisation within the context of a an adaptive predictive control and [83] which tackles the problem of trajectory tracking with a hierarchical three level controller that relies on past memorized optimal input-output pairs that are adaptively merged using a similarity measurement.

One of the main ideas of this thesis is that it is possible to include DWO as part of the optimization carried out in a predictive control approach. In this thesis, we have used direct weight optimization methods in several problems with different objectives. One of them is to use past data to predict the future behaviour of the system and, with these predictions, to estimate some cost function value which can be optimized. Other problem is to recover the closed-loop control law implicit in the past historian process data of the system. Finally, to use data in order to identify an unknown system while optimizing the control action is another strategy considered.

1.4 Data driven/based control

In recent years the terms data driven/based control have been applied to different control strategies based on completely different paradigms. Since data driven is a broad concept, this section presents a literature review of different scopes in which researchers apply these techniques and the goals they try to reach with them.

Data driven approaches based on time domain models [46] and frequency domain methods [51] have been researched, as well as modern paradigms like behavioural [66] or big data based control [99]. Data driven controller tuning methods which directly synthesize a controller with an iterative procedure [27, 43] or which obtain desirable properties as to ensure closed-loop stability [106] have been developed. Data driven predictive control [79, 100] and iterative learning control [31] have also been proposed.

Using data based inference strategies to obtain prediction models is the most popular data based control approaches. In section 1.3 some of the generic inference strategies that are used in these approaches were presented. The use of data to identify a model, to build regressors and predictors, to train neural networks which simulate systems dynamics or to modify on-line a set of parameters from a parametrized model has been widely studied in the literature.

An approach widely used is to derive an explicit model from data to later use it in the controller as in [5, 3]. Regression trees and ensemble learning have been used by Jain *et al.* [48] to obtain a prediction model from data. A major breakthrough in system identification was the developing of subspace identification methods (see the book [110] and works [108, 109] of Van Overschee and De Moor) which have also been used in the context of data driven predictive control [50].

A different technique used by Canale *et al.* [28], based on nonlinear set membership [72], is used to obtain an approximate model with a bound on the worst-case model error that can be used to infer closed loop-stability properties. Prediction models are also

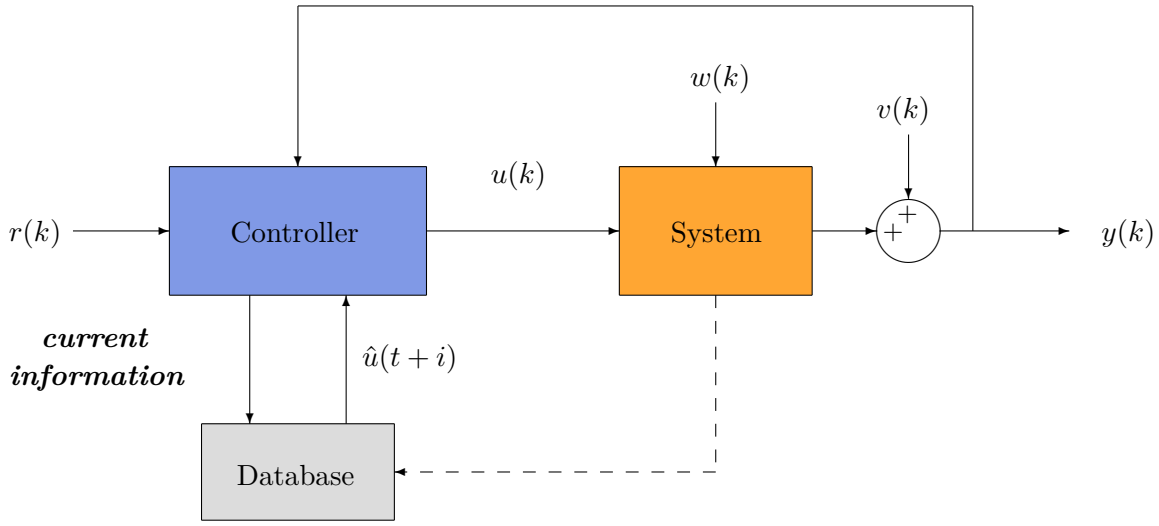


Figure 1.4.1: Model free data based predictive control diagram.

inferred from experimental data of inputs and outputs of the plant in the work of Limón *et al.* [59]. Prediction methods can also return an interval obtained from historic input-output measurements that bounds the system output as in the work of Bravo *et al.* [20].

A different point of view is to completely avoid the model estimation phase. Approaches framed in this scope are usually known as model-free data driven techniques. Their goal is to provide a control law through the data avoiding explicitly to derive a model.

All these strategies have a similar general scheme shown in figure 1.4.1. Notice that, in this case, the problem of model identification is avoided because the controller directly defines the control law using the database and the current information available (reference value, current output, estimated state, predicted future disturbance sequence, recent previous inputs and outputs, etc.). Favoreel *et al.* [36] used this approach to derive, by means of matrix decompositions, an LQG control law directly from data. Model free data driven MPC has been presented by Piga *et al.* [79] in which no model is explicitly obtained from data and a hierarchical structure with an inner linear controller is used. Other strategy without explicit model is presented by Tanaskovic *et al.* [100], where past data obtained from the system are used off-line to guarantee a predictable closed loop behaviour whereas on-line collected data is used to adapt the controller.

Another approach would be the use of some form of machine learning technique to learn the controller directly from data, like in the work of Fagiano *et al.* [35] where a l_1-norm regularized learning algorithm based on convex programming is presented; or to solve iteratively infinite horizon control problems by using approximate dynamic programming [11, 80] or approximate Q-learning techniques [58, 12] in which the direct computation of the performance function is avoided by using a suitable approximation [106] or applying distributed optimization algorithms to tackle adaptative dynamic programming problems [101]; or to identify the process dynamics [50].

In [79] data-driven direct controller synthesis are combined with MPC to control systems

under input and output constraints without the need of a model of the open-loop process. Using similar techniques, model-free optimal control has been proposed in [96]. In [5] some machine learning techniques are proposed to estimate the global uncertainty of the system, in order to improve predictions. In [28, 59] nonlinear predictive controllers are designed based on data using Lipschitz interpolation techniques.

A data-driven control design technique which is based on the on-line inversion of the model and copes with MIMO nonlinear system is presented in [75]. Other works, as in [33], focus on achieving high tracking performance through learning for unknown LTI systems subject to unknown disturbances. Nonlinear systems with output saturation are addressed in [23].

1.5 Objective of the thesis

In this thesis, the main goal is to propose novel data based predictive controllers to cope with complex industrial infrastructures such as water distribution networks. This sort of systems have several inputs and outputs, complicate nonlinear dynamics, binary actuators and they are usually perturbed by disturbances and noise and require real-time control implementation. The proposed controllers have to deal successfully with these issues while using the available information, such as past operation data of the process, or system properties as fading dynamics.

To this end, the control strategies presented in this work follow a predictive control approach. The control action computed by the proposed data-driven strategies are obtained as the solution of an optimization problem that is similar in essence to those used in MPC based on a cost function that determines the performance to be optimized. In the proposed approach however, the prediction model is substituted by an inference data based strategy, either to identify a model, an unknown control law or estimate the future cost of a given decision. As in MPC, the proposed strategies are based on a receding horizon implementation, which implies that the optimization problems considered have to be solved online.

In order to obtain problems that can be solved efficiently, most of the strategies proposed in this thesis are based on DWO for ease of implementation and computational complexity reasons. Linear convex combination is a simple and strong tool in continuous domain and computational load associated with the constrained optimization problems generated by linear convex combination are relatively soft. This fact makes the proposed data based predictive approaches suitable to be used in real time applications.

One drawback of using this approach is that the number of optimization variables and constraints depend directly on the size of the considered database. Nevertheless, this issue can be addressed by selecting the most adequate information (similar to the current situation according to output, state, input, disturbances,etc.), in particular, data which is close to the current state or situation of the system. Using local data can be interpreted as an implicit local linearisation of the system every time we solve the model-free data driven optimization problem. This implies that even though, model free data driven approaches presented in this thesis are based on linear theory, they can successfully deal with nonlinear systems because of the implicit information available in the database.

In the next section, an outline of this thesis is provided. In each chapter, a different control strategy has been developed.

1.6 Thesis outline

The outline of this thesis is organized as follows:

Chapter 2. *MPC for partially fading memory systems*

The first control problem tackled in this thesis was motivated by drinking water networks. In particular, the objective of the first research was to develop an MPC controller for a WDN with on-off actuator and valves able to consider long prediction horizons. To this end, the following chapter presents a heuristic algorithm to implement a model predictive controller for systems with binary inputs in which the effect of the control signal on the response partially vanishes before reaching steady state, for example systems that exhibit both fast and slow stable dynamics. The proposed algorithm is based on an iterative procedure that constructs a reduced set of suboptimal solutions. The size of this set can be set accordingly to the computing capabilities and the sample time. The iterative procedure rejects possible solutions profiting from the partial fading memory property of the system and an approximation of the optimal cost-to-go function.

Chapter 3. *Historian data based predictive control*

One of the main problems of applying the proposed MPC for partially fading memory systems to large scale WDN was how to approximate the optimal cost-to-go functions. Motivated by this issue, in this chapter we present a data-based strategy in which DWO is used to estimate the performance of a given future trajectory, as a previous step to evaluate optimal costs. This idea lead to the heuristic historian data based predictive control strategy presented in this chapter. The control actions are computed based on past historian data. The historian stores closed loop operation data of the process with different controllers used in the past which may not provide sufficient information for a precise system nor controller identification. The proposed predictive controller computes the current control actions as a weighted sum of past control actions so that an estimation of the performance cost over a prediction horizon is minimized. Only a subset of the past control actions in the historian close to the current state of the process are considered in the current control computations to carry out a local linearisation. This predictive strategy is well suited to control applications of large and complex processes for which it is difficult to carry out identification experiments such as water distribution systems. This strategy is used to control a water distribution system simulated using the EPANET software, in particular, the Richmond water distribution system. The trajectories of a set of relay controllers are used through the proposed approach to take into account pressure constraints and periodic references.

Chapter 4. *Offset free data driven control*

The proposed historian data based controller was applied not only to the Richmond case study, but also to a laboratory scaled four tank process. The results showed that even if the historian data was obtained from offset free closed-loop trajectories (in particular obtained using a PI controller), in order to obtain zero tracking error the proposed control scheme had to be modified. Motivated by this issue, the problem of learning an off-set free control law using DWO was tackled. This chapter presents a data driven control strategy able to track a set point without steady state error. The control sequence is computed as an affine combination of past control signals, which belong to a set of trajectories

stored in a process historian database. This affine combination is computed so that the variance of the tracking error is minimized. It is shown that offset free control, that is, zero mean tracking error, is achieved under the assumption that the state is measurable, the underlying dynamics are linear and the trajectories of the database share the same error dynamics and are in turn offset free. The proposed strategy learns the underlying controller stored in the database while maintaining its offset free tracking capability in spite of differences in the reference, disturbances and operating conditions. No training phase is required and newly obtained process data can be easily taken into account.

Chapter 5. *Data-based predictive control via direct weight optimization*

In the previous chapters, DWO has been used to estimate the cost of a future trajectory and to learn the underlying controller stored in a database. In this chapter we take a different approach and focus on using DWO to estimate the unknown system model and to decide the optimal control simultaneously. A novel data-based predictive control scheme in which the prediction model is obtained from a linear combination of past system trajectories is presented. The proposed controller optimizes the weights of this linear combination taking into account simultaneously performance and the variance of the estimation error. For unconstrained systems, dynamic programming is used to obtain an explicit linear solution of a finite or infinite horizon optimal control problem. When constraints are taken into account, the controller needs to solve online a quadratic optimization problem to obtain the optimal weights, possibly considering also local information to improve the performance and estimation.

Chapter 6. *Learning based predictive control for MIMO systems*

Up to here, the main idea has been to use DWO in different control problems. Nevertheless, the effect of the estimation and measuring errors or the effect of unknown perturbations have not been considered. In this chapter, we present a joint work with Enrico Terzi, Marcello Farina, Lorenzo Fagiano and Ricardo Scattolini from the *Politecnico di Milano* carried out during an academic stay, in which the issue of robustness in data based predictive control is considered. In particular, a learning-based approach for robust predictive control design for multi-input multi-output (MIMO) linear systems is presented in this chapter. The identification stage allows to obtain multi-step ahead prediction models and to derive tight uncertainty bounds. The identified models are then used by a robust model predictive controller, that is designed for the tracking problem with stabilizing properties. Numerical results show the effectiveness of the proposed approach on a benchmark example, a quadruple-tank process linearised around a working point. The proposed algorithm is later used to control the nonlinear model of the benchmark example using data gathered from it. The resulting controller, suitably modified to account for the nonlinear system gain matrix, results in remarkable tracking performances.

Chapter 7. *Conclusions and future lines*

An analysis of the results obtained in the research developed along this thesis is presented in the last chapter. Furthermore, future research lines which continue with the work carried out are also proposed.

1.7 Contributions

All the work carried out during this thesis has been performed joint with my advisors and under their supervision. In addition, some of the results are a joint work with researchers of several groups from other universities. As result of the referred collaborations, the publications that support this thesis are presented next.

1.7.1 Journal papers

- J.R. Salvador, T. Alamo, D.R. Ramirez and D. Muñoz de la Peña. **Model predictive control of partially fading memory systems with binary inputs**. *Journal of Process Control*, 64: 141-151, April 2018.
- Y. Wang, J.R. Salvador, D. Muñoz de la Peña, V. Puig and G. Cembrano. **Economic model predictive control based on a periodicity constraint**. *Journal of Process Control*, 68: 226-239, August 2018.
- J.R. Salvador, D.R. Ramirez, T. Alamo, and D. Muñoz de la Peña. **Offset free data driven control: Application to a process control trainer**. *IET Control Theory & Applications*, 2019. Accepted for publication.
- J.R. Salvador, D. Muñoz de la Peña, D.R. Ramirez and T. Alamo. **Predictive control of a water distribution system based on process historian data**. *Optimal Control, Applications and Methods*. Accepted subject to minor revision.

1.7.2 Conference papers

- J.R. Salvador, T. Alamo, D.R. Ramirez and D. Muñoz de la Peña. **Predictive control with on-off actuators of partially fading memory systems**. Conference communication. In proceedings of the *20th IFAC World Congress*, Toulouse, France, July 2017.
- Y. Wang, J.R. Salvador, D. Muñoz de la Peña, V. Puig and G. Cembrano. **Periodic nonlinear economic model predictive control with changing horizon for water distribution networks**. Conference communication. In proceedings of the *20th IFAC World Congress*, Toulouse, France, July 2017.
- J.R. Salvador, D. Muñoz de la Peña, D.R. Ramirez and T. Alamo. **Historian data based predictive control of a water distribution network**. Conference communication presented by the author. In proceedings of the *16th European Control Conference (ECC)*, Limassol, Cyprus, June 2018.
- J.R. Salvador, D. Muñoz de la Peña, T. Alamo and A. Bemborad. **Data-based predictive control via direct weight optimization**. Conference poster. In proceedings of the *6th IFAC Conference on Nonlinear Model Predictive Control*, Madison, Wisconsin(USA), August 2018.
- J.R. Salvador, D.R. Ramirez, T. Alamo, and D. Muñoz de la Peña. **Data driven control: an offset free approach**. Conference communication. In proceedings of the *17th European Control Conference (ECC)*, Naples, Italy, June 2019.
- J.R. Salvador, E. Terzi, M. Farina, D.R. Ramirez, L. Fagiano, D. Muñoz de la Peña and R. Scattolini. **Learning-based predictive control for MIMO systems**.

Conference communication. Accepted in the *58th Conference on Decision and Control (CDC)*, Nice, France, December 2019.

Chapter 2

Model predictive control of partially fading memory systems with binary inputs

Binary programming problems are a specific case of MILP/MIQP, or more generally speaking, of mixed integer programming (MIP) problems. Brute force algorithms that take into account all the possibles combinations in predictive control with large predictions horizons can not be implemented, specially in real time applications which have certain time slot to compute the next input.

On approach is to consider only a set of the possible combinations using some selection criteria. Some examples of these strategies are branch and bound, genetic algorithms or L-band algorithms ([1, 37]). In [29] two different approaches based on branch and bound techniques and genetic algorithms were applied to the control of a batch reactor with on-off valves. Genetic algorithms have also been used by Schmitz *et al.* [95] for the control of the aeration in a waste water treatment plant with the goal of lowering the operating costs. Branch and bound algorithms has been used together with an event driven sampling mechanism to apply an MPC strategy to the control of pH in photobiorreactors [10, 77]. Attitude control of a spacecraft using on-off thrusters with linear parameter varying models and branch and bound techniques has been considered in [4]. The solution of the MIQP problem using methods not based on branch and bound has also been considered in the context of optimal control problems of systems with discrete inputs. Sager *et al.* [93] used, in the optimal control of a subway train with discrete gears, a convexification on the control inputs and a relaxation of the MIQP problem in a proposed strategy based on the direct multiple shooting method ([16]). Finally in some rare cases, the optimal control policy can be identified a priori and online optimization can be avoided ([89]).

All these approaches provide suboptimal solutions, but they can be applied in real time processes because of their reduced computational timing which, in some cases as branch and bound, can be set to a fixed value. The efficiency provided by these suboptimal methods depends on the exploitation of some property of the class of systems considered.

This chapter presents a heuristic algorithm to obtain a suboptimal solution of the MPC optimization problem for processes with binary inputs in which a part of the dynamics

is not influenced by past values of the control signal beyond a certain time instant. We denote this class of systems as partial fading memory systems (PFMS). The algorithm constructs iteratively a series of sets of candidate solutions that are likely to contain the optimal solution, in a similar way to the L-Band algorithm, profiting from the partial fading memory property of the system and an approximation of the optimal cost-to-go function. The size of this set provides a trade-off between optimality and computational burden. These properties are illustrated by means of a simulated example.

2.1 Problem formulation

The system considered throughout the chapter will be represented by a discrete-time linear model:

$$x(t+1) = Ax(t) + Bu(t) \tag{2.1}$$

where $x(t) \in \mathbb{R}^{n_x}$ is the state of the process, $u(t) \in \{0, 1\}$ is the binary control input, $t \in \mathbb{Z}$ is discrete valued, $A \in \mathbb{R}^{n_x \times n_x}$ is the state transition matrix and $B \in \mathbb{R}^{n_x}$ is the input matrix.

Definition 1 (Partially fading memory system)**.** *A system is said to be a partially fading memory system if for any given state sequences $x_a(t)$, $x_b(t)$ with different initial states but driven by the same input sequence, the following holds:*

$$\|x_a(t) - x_b(t)\| \leq \|C_s(x_a(t) - x_b(t))\| + \sigma_f^t \|C_f(x_a(0) - x_b(0))\|, \ \forall t \geq L$$

where $L \in \mathbb{N}$ is the fading time of the fast dynamics, $\sigma_f \in [0, 1)$ is the fading parameter such that $\sigma_f^L \ll 1$ and C_s and C_f are projection matrices that satisfy $\|x_a(t) - x_b(t)\| \leq \|C_s(x_a(t) - x_b(t))\| + \|C_f(x_a(t) - x_b(t))\| \ \forall t$, where $\|\cdot\|$ is a given vector norm.

This definition implies that part of the state $x(t)$ for time instant t is only barely influenced by past values of $u(t)$ beyond $t - L$ (i.e, $u(t - L), u(t - L - 1), \dots$). This can be seen as the fast dynamics part of the state, denoted as $z^f(t)$ so that

$$z^f(t) = C_f x(t).$$

On the other hand, the remaining part of the state is influenced by past values of $u(t)$ beyond $u(t - L)$. This part of the state vector, the slow dynamics of system (2.1), will be denoted as $z^s(t)$, so that

$$z^s(t) = C_s x(t).$$

Systems that combine stable and integrating dynamics and systems with both fast and slow stable dynamics are examples of PFMS.

Assumption 1. *System (2.1) is a partially fading memory system (PFMS).*

Assuming that (2.1) is a partial fading memory system, it is possible to partition the eigenvalues of matrix A in two sets, ξ_s and ξ_f related to slow and fast dynamics respectively, such that

$$\min_{\lambda \in \xi_s} |\lambda| > \max_{\lambda \in \xi_f} |\lambda| = \sigma_f. \tag{2.2}$$

Since all the eigenvalues in ξ_s are different from the ones in ξ_f, there exists a matrix T such that

$$A = T \begin{bmatrix} H_s & 0 \\ 0 & H_f \end{bmatrix} T^{-1} \tag{2.3}$$

where H_s, H_f are Jordan blocks that represent the slow and fast dynamics of the system respectively. That is, the eigenvalues of H_s and H_f are contained in ξ_s and ξ_f respectively. The slow and fast projections, denoted z_s and z_f, are obtained from

$$z = \begin{bmatrix} z_s \\ z_f \end{bmatrix} = T^{-1}x \tag{2.4}$$

where $z \in Z \subset \mathbb{R}^{n_x}$. The following property shows that if $\sigma_f < 1$, then system (2.1) satisfies Assumption 1. Moreover, the proof that we present for this property provides a procedure to obtain matrices C_s and C_f.

Property 1 (Fading parameter in linear systems)**.** *Suppose that the eigenvalues of matrix A are partitioned in two set as in (2.2) with $\sigma_f < 1$, then system (2.1) is a partially fading memory system with fading parameter σ_f.*

PROOF: Suppose that the same first control input $u(0)$ is applied to different initial conditions $x_a(0)$, $x_b(0)$. We have,

$$\begin{aligned} x_a(1) - x_b(1) &= Ax_a(0) + Bu(0) - Ax_b(0) - Bu(0) \\ &= A(x_a(0) - x_b(0)). \end{aligned}$$

From (2.3) we obtain

$$\begin{aligned} x_a(1) - x_b(1) &= T \begin{bmatrix} H_s & 0 \\ 0 & H_f \end{bmatrix} T^{-1}(x_a(0) - x_b(0)) \\ T^{-1}\left(x_a(1) - x_b(1)\right) &= \begin{bmatrix} H_s & 0 \\ 0 & H_f \end{bmatrix} T^{-1}(x_a(0) - x_b(0)), \end{aligned}$$

and taking into account (2.4)

$$z_a(1) - z_b(1) = \begin{bmatrix} H_s & 0 \\ 0 & H_f \end{bmatrix} (z_a(0) - z_b(0)).$$

Proceeding in a recursive way, we obtain that if the same control sequence is applied to both projected initial conditions

$$z_a(t) - z_b(t) = \begin{bmatrix} H_s^t & 0 \\ 0 & H_f^t \end{bmatrix} (z_a(0) - z_b(0)). \tag{2.5}$$

There are two possible cases depending on whether the maximum singular value of matrix H_f, denoted as $\bar{\sigma}(H_f)$, is equal or greater than the fading parameter.

a) $\bar{\sigma}(H_f) = \sigma_f$.
This is the most general case and occurs, for example, when the eigenvalues of H_f are different.

We have,

$$\begin{aligned} z_a(t) - z_b(t) &= \begin{bmatrix} z_a^s(t) - z_b^s(t) \\ z_a^f(t) - z_b^f(t) \end{bmatrix} \\ &= \begin{bmatrix} z_a^s(t) - z_b^s(t) \\ H_f^t(z_a^f(0) - z_b^f(0)) \end{bmatrix} \\ &= \begin{bmatrix} z_a^s(t) - z_b^s(t) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ H_f^t\left(z_a^f(0) - z_b^f(0)\right) \end{bmatrix}. \end{aligned} \tag{2.6}$$

From the triangular inequality we now obtain, for the Euclidean norm $\|\cdot\|_2$, the following inequality

$$\begin{aligned} ||z_a(t) - z_b(t)||_2 &\leq \left\| \begin{bmatrix} z_a^s(t) - z_b^s(t) \\ 0 \end{bmatrix} \right\|_2 + \left\| \begin{bmatrix} 0 \\ H_f^t \left(z_a^f(0) - z_b^f(0) \right) \end{bmatrix} \right\|_2 \\ &= ||z_a^s(t) - z_b^s(t)||_2 + \left\| H_f^t \left(z_a^f(0) - z_b^f(0) \right) \right\|_2 \\ &\leq ||z_a^s(t) - z_b^s(t)||_2 + \left\| H_f^t \right\| \left\| z_a^f(0) - z_b^f(0) \right\|_2, \end{aligned}$$

where $\|H_f\|$ denotes the matrix norm induced with the Euclidean norm ($\|H_f\| = \bar{\sigma}(H_f) = \sigma_f$). Since the induced matrix norm is submultiplicative we have $\|H_f^t\| \leq \|H_f\|^t = \sigma_f^t$. Therefore,

$$||z_a(t) - z_b(t)||_2 \leq ||z_a^s(t) - z_b^s(t)||_2 + \sigma_f^t \left\| z_a^f(0) - z_b^f(0) \right\|_2. \tag{2.7}$$

If (2.4) is applied again,

$$\begin{aligned} \|x_a(t) - x_b(t)\|_2 &= \|T(z_a(t) - z_b(t))\|_2 \\ &\leq \|T\| \|z_a(t) - z_b(t)\|_2 \\ &= \bar{\sigma}(T) \|z_a(t) - z_b(t)\|_2. \end{aligned}$$

Then, from (2.7) we obtain

$$\begin{aligned} \|x_a(t) - x_b(t)\|_2 \leq\ & \bar{\sigma}(T) \left\| \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix} T^{-1} \left(x_a(t) - x_b(t) \right) \right\|_2 \\ & + \sigma_f^t\, \bar{\sigma}(T) \left\| \begin{bmatrix} 0 & \mathrm{I} \end{bmatrix} T^{-1} \left(x_a(0) - x_b(0) \right) \right\|_2. \end{aligned}$$

We conclude,

$$\|x_a(t) - x_b(t)\|_2 \leq \|C_s(x_a(t) - x_b(t))\|_2 + \sigma_f^t \left\| C_f \left(x_a(0) - x_b(0) \right) \right\|_2.$$

where $C_s = \bar{\sigma}(T) \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix} T^{-1}$ and $C_f = \bar{\sigma}(T) \begin{bmatrix} 0 & \mathrm{I} \end{bmatrix} T^{-1}$. Thus, the value of the fading parameter is

$$\sigma_f = \max_{\lambda \in \xi_f} |\lambda| = \bar{\sigma}(H_f) = ||H_f||.$$

b) $\bar{\sigma}(H_f) > \sigma_f$
This may occur only when H_f has repeated eigenvalues. Consider the following generalized eigenvalue minimization problem,

$$\begin{aligned} \min_{\alpha_f, P_f} \quad & \alpha_f \\ s.t. \quad & H_f^\top P_f H_f \leq \alpha_f P_f \\ & P_f \geq \mathrm{I}. \end{aligned} \tag{2.8}$$

Since H_f is strictly stable we have from quadratic Lyapunov theory ([17]) that this optimization problem is always feasible and the optimal solution α_f^* belongs to the interval $[0, 1)$. Suppose that (P_f, α_f), with $\alpha_f \in [0, 1)$ is a feasible solution to the previous optimization problem. We define $\sigma_f = \sqrt{\alpha_f} < 1$ and

$$P = \begin{bmatrix} \mathrm{I} & 0 \\ 0 & P_f \end{bmatrix}.$$

This matrix defines a weighted norm, that is

$$||z||_P = \sqrt{z^\top P z}.$$

Since the pair (P_f, α_f) is a feasible solution for (2.8), we have $H_f^\top P_f H_f \leq \alpha_f H_f = \sigma_f^2 H_f$. Denoting $\Delta z^f(t) = z_a^f(t) - z_b^f(t)$ we obtain

$$\begin{aligned}
\|\Delta z^f(t)\|_{P_f} &= \|z_a^f(t) - z_b^f(t)\|_{P_f} \\
&= \|H_f \left(z_a^f(t-1) - z_b^f(t-1) \right) \|_{P_f} \\
&= \|H_f \Delta z^f(t-1)\|_{P_f} \\
&= \sqrt{(\Delta z^f(t-1))^\top H_f^\top P_f H_f \Delta z^f(t-1)} \\
&\leq \sqrt{\sigma_f^2 (\Delta z^f(t-1))^\top P_f \Delta z^f(t-1)} \\
&= \sigma_f \|\Delta z^f(t-1)\|_{P_f}.
\end{aligned}$$

Proceeding in a recursive fashion we have

$$\|\Delta z^f(t)\|_{P_f} \leq \sigma_f^t \|\Delta z^f(0)\|_{P_f}.$$

Since $P_f > \mathrm{I}$ we have $P_f^{-1} < \mathrm{I}$. Therefore,

$$\begin{aligned}
\|\Delta z^f(t)\|_2 &= \sqrt{(\Delta z^f(t))^\top P_f^{\frac{1}{2}} P_f^{-1} P_f^{\frac{1}{2}} \Delta z^f(t)} \\
&\leq \sqrt{(\Delta z^f(t))^\top P_f^{\frac{1}{2}} P_f^{\frac{1}{2}} \Delta z^f(t)} \\
&= \|\Delta z^f(t)\|_{P_f} \\
&\leq \sigma_f^t \|\Delta z^f(0)\|_{P_f} \\
&= \sigma_f^t \|P_f^{\frac{1}{2}} \Delta z^f(0)\|_2.
\end{aligned} \tag{2.9}$$

Then

$$\begin{aligned}
\|x_a(t) - x_b(t)\|_2 &= \|T\, T^{-1} (x_a(t) - x_b(t)) \|_2 \\
&\leq \bar{\sigma}(T) \left\| T^{-1} (x_a(t) - x_b(t)) \right\|_2 \\
&= \bar{\sigma}(T) \|z_a(t) - z_b(t)\|_2 \\
&= \bar{\sigma}(T) \|\Delta z(t)\|_2 \\
&\leq \bar{\sigma}(T) \left(\|\Delta z^s(t)\|_2 + \left\| \Delta z^f(t) \right\|_2 \right).
\end{aligned}$$

From (2.9)

$$\begin{aligned}
\|x_a(t) - x_b(t)\|_2 &\leq \bar{\sigma}(T) \left(\|\Delta z^s(t)\|_2 + \sigma_f^t \left\| P_f^{\frac{1}{2}} \Delta z^f(0) \right\|_2 \right) \\
&= \bar{\sigma}(T) \left(\|z_a^s(t) - z_b^s(t)\|_2 + \sigma_f^t \left\| P_f^{\frac{1}{2}} \left(z_a^f(0) - z_b^f(0) \right) \right\|_2 \right) \\
&= \bar{\sigma}(T) \left\| \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix} T^{-1} (x_a(t) - x_b(t)) \right\|_2 \\
&\quad + \sigma_f^t \bar{\sigma}(T) \left\| P_f^{\frac{1}{2}} \begin{bmatrix} 0 & \mathrm{I} \end{bmatrix} T^{-1} (x_a(0) - x_b(0)) \right\|_2.
\end{aligned}$$

We conclude that

$$\|x_a(t) - x_b(t)\|_2 \leq \|C_s\left(x_a(t) - x_b(t)\right)\|_2 + \sigma_f^t \|C_f\left(x_a(0) - x_b(0)\right)\|_2$$

where $C_s = \bar{\sigma}(T) \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix} T^{-1}$ and $C_f = \bar{\sigma}(T) P_f^{\frac{1}{2}} \begin{bmatrix} 0 & \mathrm{I} \end{bmatrix} T^{-1}$. □

The control objective is to regulate the state of the system to the origin while minimizing a particular performance index. To this end we propose to use an MPC formulation based on a stage cost $\ell(x, u)$ and a terminal cost $\ell_N(x)$.

Assumption 2 (Lipschitz continuity)**.** *It is assumed that $\ell(x, u)$ and $\ell_N(x)$ are Lipschitz continuous with respect to x. That is,*

$$\begin{aligned} |\ell(x_a, u) - \ell(x_b, u)| &\leq \rho_\ell \|x_a - x_b\|, \ \forall (x_a, x_b, u) \\ |\ell_N(x_a) - \ell_N(x_b)| &\leq \rho_{\ell_N} \|x_a - x_b\|, \ \forall (x_a, x_b) \end{aligned}$$

for some ρ_ℓ and ρ_{ℓ_N} with $\rho_\ell, \rho_{\ell_N} \in \mathbb{R} > 0$.

We notice that no convexity assumption is required on $\ell(\cdot, \cdot)$ or $\ell_N(\cdot)$. Following the conventions in MPC, N time steps sequences into the future will be considered for the input, $\mathbf{u} \in U_N$, with $U_N \subseteq \{0, 1\}^N$. Given an initial state x and a candidate input sequence $\mathbf{u} \in U_N$, the performance cost $V(x, \mathbf{u})$ is defined as

$$V(x, \mathbf{u}) = \sum_{k=0}^{N-1} \ell(x(k), u(k)) + \ell_N(x(N)), \tag{2.10}$$

where $x(0) = x$, $x(k+1) = Ax(t) + Bu(t)$ and N is the prediction horizon.

The Lipschitz continuity assumption on $\ell(\cdot, u)$ and $\ell_N(\cdot)$ and the PFMS nature of the system imply that the performance cost $V(\cdot, \mathbf{u})$ is also Lipschitz continuous, thus

$$|V(x_a, \mathbf{u}) - V(x_b, \mathbf{u})| \leq \rho_V \|x_a - x_b\|, \ \forall x_a, x_b, \mathbf{u} \tag{2.11}$$

for some $\rho_V \in \mathbb{R} > 0$.

The MPC controller solves at each sampling time the following binary optimization problem:

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} V(x, \mathbf{u}) \tag{2.12}$$

in which only the first component of the optimal solution $\mathbf{u}^*$ is applied, i.e., $u(t) = u^*(0)$. The optimal cost function will be then defined as

$$V^*(x) = V(x, \mathbf{u}^*) \tag{2.13}$$

which, by definition, satisfies

$$V^*(x) \leq V(x, \mathbf{u}), \ \forall \mathbf{u} \in U_N.$$

Optimization problem (2.12) is a binary optimization problem which, in general, is NP-Hard.

2.2 Optimization algorithm

This section presents a heuristic algorithm to obtain a suboptimal solution of problem (2.12) in an efficient way. The algorithm is an iterative procedure that builds a series of candidate solution sets of increasingly longer binary input sequences until those sequences have a length equal to the prediction horizon N. Those candidate solution sets will have a size (cardinality) that can be set accordingly to the computing power and available storage capacity, and are likely to contain a near optimal solution to (2.12).

2.2.1 Criterion for candidate set reduction

The reduction of the size of the candidate sets will be based on a screening criterion which, given two input sequences with only the first $k-1$ components known, will screen which one will result in a higher performance cost and therefore could be discarded. The problem with this approach is that at iteration k the performance cost for a sequence is unknown, because the sequence is incomplete, so instead of using the performance cost for screening non optimal sequences, it is necessary to use an approximation of this value.

Given an initial state x, a candidate solution $\mathbf{u}$ and a time step k, the performance cost can be divided in two terms: the cost up to time step k and the cost from time step k to the end of the prediction horizon, i.e.

$$V(x,\mathbf{u}) = V_0^{k-1}(x,\mathbf{u}) + \ell_k^N(x_k,\mathbf{u}) \tag{2.14}$$

where, with a slight abuse of notation, x_k is the predicted state for time step k using model (2.1) and

$$V_0^{k-1}(x,\mathbf{u}) = \sum_{j=0}^{k-1} \ell(x(j),u(j))$$

where $x(0) = x$ and

$$\ell_k^N(x,\mathbf{u}) = \sum_{j=k}^{N-1} \ell(x(j),u(j)) + \ell_N(x(N))$$

where $x(k) = x$.

The optimal cost to go for an incomplete binary input sequence is defined as follows

$$\ell_{k,N}^*(x) = \min_{\mathbf{u}} \ell_k^N(x,\mathbf{u}). \tag{2.15}$$

Property 2 (Lipschitz continuity of $\ell_{k,N}^*(x)$)**.** *The optimal cost to go $\ell_{k,N}^*(x)$ is Lipschtiz continuous.*

PROOF: By definition $\ell_k^N(x,\mathbf{u})$ is Lipschitz continuous in x as it is the sum and composition of Lipschitz continuous functions. Consider $\mathbf{u}_a^*, \mathbf{u}_b^*$ the minimizers of (2.15) for $x = x_a$ and $x = x_b$ respectively. Then the following inequality holds:

$$\ell_{k,N}^*(x_a) = \ell_k^N(x_a,\mathbf{u}_a^*) \leq \ell_k^N(x_a,\mathbf{u}_b^*). \tag{2.16}$$

On the other hand, being $\ell_k^N(x(k),\mathbf{u})$ Lipschitz continuous in x, with Lipschitz constant γ_{l^*}, then

$$\ell_k^N(x_a,\mathbf{u}_b^*) \leq \ell_k^N(x_b,\mathbf{u}_b^*) + \gamma_{l^*}\|x_a - x_b\|. \tag{2.17}$$

Taking into account that $\ell_k^N(x_b, \mathbf{u}_b^*) = \ell_{k,N}^*(x_b)$ and (2.16):

$$\ell_{k,N}^*(x_a) - \ell_{k,N}^*(x_b) \leq \gamma_{l^*} \|x_a - x_b\|. \tag{2.18}$$

Exchanging $\mathbf{u}_a^*$ by $\mathbf{u}_b^*$ and x_a by x_b leads to

$$\ell_{k,N}^*(x_b) - \ell_{k,N}^*(x_a) \leq \gamma_{l^*} \|x_a - x_b\|. \tag{2.19}$$

Thus taking into account (2.18) and (2.19):

$$|\ell_{k,N}^*(x_a) - \ell_{k,N}^*(x_b)| \leq \gamma_{l^*} \|x_a - x_b\|, \tag{2.20}$$

which implies that the optimal cost to go is Lipschitz continuous. □

The optimal cost to go for a particular state is defined by (2.15), however this function is hard to obtain. We assume that there exists a function $\hat{\ell}_k^N(x)$ that approximates the optimal cost. As an approximation, $\hat{\ell}_k^N(x)$ will differ from the optimal cost to go by an error $\Phi_k(x)$, so that

$$\Phi_k(x) = \ell_{k,N}^*(x) - \hat{\ell}_k^N(x) \tag{2.21}$$

is the cost to go approximation error function.

Assumption 3. *It is assumed that there exists some constants γ_Φ and σ_Φ such that the error function $\Phi_k(x)$ satisfies:*

$$\Phi_k(x_a) - \Phi_k(x_b) < \gamma_\Phi \, ||x_a - x_b|| + \sigma_\Phi, \quad k = 1, \ldots, N \tag{2.22}$$

Note that if the cost to go approximation $\hat{\ell}_k^N(x)$ is Lipschitz continuous, then $\Phi_k(x)$ is also Lipschitz continuous, because it is the difference of two Lipschitz functions (see property 2). In this case this assumption is satisfied with $\sigma_\Phi = 0$. Nevertheless, the results presented in the following do not require the Lipschitz continuity of $\hat{\ell}_k^N(x)$.

Using this approximated cost to go function, the approximated cost of a given incomplete input sequence is defined as follows:

$$\hat{V}^k(x, \mathbf{u}) = V_0^{k-1}(x, \mathbf{u}) + \hat{\ell}_k^N(x_k) \tag{2.23}$$

where, with a slight abuse of notation, x_k is the predicted state for time step k using model (2.1). This cost function is used in the algorithm to reject possible candidates.

Theorem 1 (Screening criterion)**.** *Let $\boldsymbol{u}_a$ and $\boldsymbol{u}_b \in U_N$ be two different input binary sequences. If the approximate costs at time instant k satisfy*

$$\hat{V}^k(x_0, \boldsymbol{u}_b) - \hat{V}^k(x_0, \boldsymbol{u}_a) \geq \gamma_\Phi \, ||x_a - x_b|| + \sigma_\Phi, \tag{2.24}$$

where x_0 is the initial state that is equal for all possible input and x_a, x_b are the k-steps ahead predicted states from x_0 for $\boldsymbol{u}_a, \boldsymbol{u}_b$ respectively, then, $\boldsymbol{u}_b$ will not be the optimal solution to problem (2.12), therefore it can be discarded.

PROOF: Note that for any sequence $\mathbf{u}$:

$$V_0^{k-1}(x_0, \mathbf{u}) + \ell_{k,N}^*(x_k) = \hat{V}^k(x_0, \mathbf{u}) + \ell_{k,N}^*(x_k) - \hat{\ell}_k^N(x_k)$$

Thus, taking into account this and the definition of $\Phi_k(x)$ in (2.21):

$$\begin{aligned} V_0^{k-1}(x_0,\mathbf{u}_a)+\ell_{k,N}^*(x_a) & & \\ -V_0^{k-1}(x_0,\mathbf{u}_b)-\ell_{k,N}^*(x_b) &= \hat{V}^k(x_0,\mathbf{u}_a)-\hat{V}^k(x_0,\mathbf{u}_b) \\ &\quad +\ell_{k,N}^*(x_a)-\ell_{k,N}^*(x_b) \\ &\quad -\hat{\ell}_k^N(x_a)+\hat{\ell}_k^N(x_b) \\ &= \hat{V}^k(x_0,\mathbf{u}_a)-\hat{V}^k(x_0,\mathbf{u}_b) \\ &\quad +\Phi_k(x_a)-\Phi_k(x_b) \end{aligned} \tag{2.25}$$

Furthermore, taking into account (2.22) in the right hand side of equality (2.25) yields:

$$\begin{aligned} V_0^{k-1}(x_0,\mathbf{u}_a)+\ell_{k,N}^*(x_a) & & \\ -V_0^{k-1}(x_0,\mathbf{u}_b)-\ell_{k,N}^*(x_b) &\leq \hat{V}^k(x_0,\mathbf{u}_a)-\hat{V}^k(x_0,\mathbf{u}_b) \\ &\quad +\gamma_\Phi\,||x_a-x_b||+\sigma_\Phi \end{aligned} \tag{2.26}$$

Now it is easy to see that if

$$\hat{V}^k(x_0,\mathbf{u}_b)-\hat{V}^k(x_0,\mathbf{u}_a)\geq\gamma_\Phi\,||x_a-x_b||+\sigma_\Phi$$

then

$$\hat{V}^k(x_0,\mathbf{u}_a)-\hat{V}^k(x_0,\mathbf{u}_b)+\gamma_\Phi\,||x_a-x_b||+\sigma_\Phi\leq 0$$

which taking into account (2.26) leads to

$$V_0^{k-1}(x_0,\mathbf{u}_a)+\ell_{k,N}^*(x_a)-V_0^{k-1}(x_0,\mathbf{u}_b)-\ell_{k,N}^*(x_b)\leq 0$$

and finally to

$$V_0^{k-1}(x_0,\mathbf{u}_b)+\ell_{k,N}^*(x_b)\geq V_0^{k-1}(x_0,\mathbf{u}_a)+\ell_{k,N}^*(x_a) \tag{2.27}$$

□

This result can be interpreted as a guarantee that there exists a complete input sequence beginning with the first k components of $\mathbf{u}_a$ with a lower cost than the best (i.e., with lowest cost) input sequence beginning with the first k components of $\mathbf{u}_b$. Thus, the sequence $\mathbf{u}_b$ can be discarded. Note that this implies, that, if the bounds γ_Φ and σ_Φ are known, theorem 1 can be used to obtain the optimal solution of problem (2.12).

Notice that neglecting σ_Φ in the screening criterion amounts to a suboptimality bounded by σ_Φ. That is,

$$\hat{V}^k(x_0,\mathbf{u}_b)-\hat{V}^k(x_0,\mathbf{u}_a)\geq\gamma_\Phi\,||x_a-x_b||$$

implies that

$$V_0^{k-1}(x_0,\mathbf{u}_b)+\ell_{k,N}^*(x_b)\geq V_0^{k-1}(x_0,\mathbf{u}_a)+\ell_{k,N}^*(x_a)-\sigma_\Phi.$$

The screening criterion of theorem 1 could be applied for all possible candidate sequence pairs considered for solution of (2.12). However this is not a realistic approach, as the number of candidate sequences will be always high enough to pose a computational burden too heavy for a real time solution of (2.12). The approach considered here is to compare only pairs that result in $\|x_a-x_b\|$ small, as it will be easier to reject one of the sequences in the pair. To find pairs that result in $\|x_a-x_b\|$ small, consider $x_a(k)$, $x_b(k)$ whose

input sequences share the same most recent L input values. If the system is PFMS (see definition 1) then

$$\|x_a(k) - x_b(k)\| \leq \|z_a^s(k) - z_b^s(k)\| + \sigma_f^L \|z_a^f(k-L) - z_b^f(k-L)\|.$$

The best way to ensure that $\|x_a(k) - x_b(k)\|$ is small, is to compare pairs in which $z_a^s(k)$ and $z_b^s(k)$ are similar (which implies that $\|z_a^s(k) - z_b^s(k)\|$ is small), and then, in addition to that, compare pairs that also share the same or similar most recent L input values. The latter condition implies that $\sigma_f^L \|z_a^f(k-L) - z_b^f(k-L)\|$ is small, since $\sigma_f^L \ll 1$. This shows how to take advantage of the PFMS nature of the system to guarantee that $\|x_a(k) - x_b(k)\|$ is small.

These observations suggest that an algorithm that uses the screening criterion should rely on a grouping of the candidate input sequences, based on the slow state z^s they reach. Then, in each group, the screening criterion should be applied to pairs of input sequences that have similar most recent L input values.

2.2.2 Iterative algorithm for PFMS

We present next an algorithm based on the screening criterion presented in the previous subsection. The algorithm is suboptimal because γ_Φ and σ_Φ cannot be computed in general. Instead, σ_Φ will be supposed to be zero and γ_Φ will be chosen in a way such that the number of candidates is lower than a fixed value M. The algorithm starts with a set of 2^k different binary input sequences with the first k non-zero components with $k = \lfloor \log_2 M \rfloor$, where M is the maximum allowable size of the candidate solution set. This initial set contains all the possible beginnings, up to time instant k, of the binary input sequences of length N that are candidate solutions to problem (2.12).

At each iteration k, the algorithm builds a hypothesis set of binary input sequences (candidates) of k components. The hypothesis set doubles the number of candidates of the previous step candidates set. Each hypothesis sequence is then compared with two other close candidates and γ_Φ^k is chosen so that the resulting candidates set has less than or equal to M components, repeating the procedure until a full size candidate set is obtained, choosing the solution from this set.

In order to determine the comparison pairs that provide small deviations in the k-steps ahead predictions of the state of the corresponding hypotheses sequences, and hence maximize the possibilities of rejection, the algorithm profits from the partial fading property of the system. Instead of classifying the hypothesis by their full state predictions, the algorithm first classifies all the hypothesis sequences in a predefined N_G number of groups on behalf of the lower dimension slow state predicted k-steps ahead from the initial state. Then, each hypothesis is compared with the two other hypothesis from the group that have the most similar last sequence of inputs. If the last input values before a given time step k of each sequence of a comparison pair are similar, the part of the predicted state that fades faster (see definition 1) will be approximately equal after step k. These two steps are motivated by the necessity of comparing sequences that have a similar k-step predicted state, which in turn will make the gap needed to apply the screening criterion in (2.24) small. With such a small gap it will be much easier to reject sequences applying the screening criterium of theorem 1.

Algorithm 1 (MPC for PFMS)**.** *The input parameters are: M (the maximum of the candidate set), N (the prediction horizon) and implicitly the initial state and the process model. The output parameter is $\boldsymbol{u}^*$ (the suboptimal solution to problem (2.12)). The algorithm steps can be stated as:*

1. *Let $k = \lfloor \log_2 M \rfloor$.*
2. *Create the candidate set $C_k = \{\boldsymbol{c}_0^k, \boldsymbol{c}_1^k, \ldots, \boldsymbol{c}_{|C_k|}^k\}$, composed by all the possible combinations of binary input sequences of length N with the first k non-zero components and zero values for the last $N-k$ components. See figure*
3. *Form the hypothesis set $H_{k+1} = \{\boldsymbol{h}_0^{k+1}, \boldsymbol{h}_1^{k+1}, \ldots, \boldsymbol{h}_{2|C_k|}^{k+1}\}$ from the sequences in C_k so that for all $i = 1, \ldots, |C_k|$ and for all $j = 1, \ldots, N$:*

$$\boldsymbol{h}_i^{k+1}(j) = \boldsymbol{c}_i^k(j)$$
$$\boldsymbol{h}_{i+|C_k|}^{k+1}(j) = \begin{cases} \boldsymbol{c}_i^k(j) & \text{if } j \neq k+1 \\ 1 & \text{otherwise} \end{cases}$$

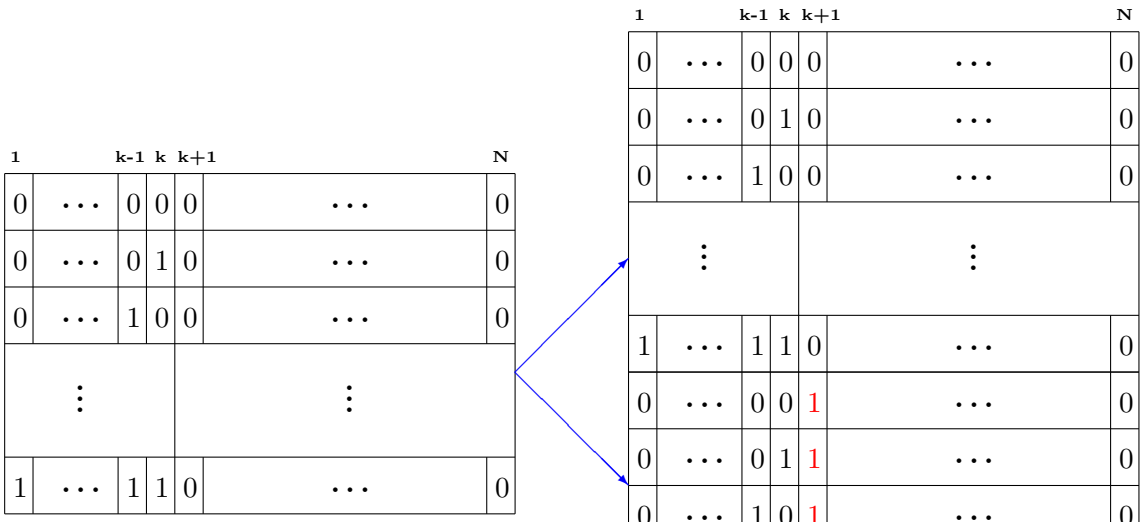
   where $\boldsymbol{c}_i^k(j)$, $\boldsymbol{h}_i^{k+1}(j)$ and $\boldsymbol{h}_{i+|C_k|}^{k+1}(j)$ denotes the $j-th$ component of each sequence respectively.
4. *For each input sequence $\boldsymbol{h}_i^{k+1}$ in H_{k+1}, compute its approximate cost $\hat{V}^{k+1}(x, \boldsymbol{h}_i^{k+1})$.*
5. *Classify H_{k+1} in N_G groups so that each group contains input sequences that lead to a similar slow state z^s predicted $k+1$-steps ahead from the initial state x.*
6. *Sort the sequences in each group of H_{k+1} by a numeric index calculated as $\sum_{j=0}^{N} 2^j \boldsymbol{h}_i^{k+1}(j)$, $\forall i = 1, \ldots, |H_{k+1}|$[1] .*
7. *Find γ_Φ^k such that when each hypotheses sequence $\boldsymbol{h}_i^{k+1}$ is compared with its neighbours applying Theorem 1 with γ_Φ^k and $\sigma_\Phi = 0$ the resulting candidate set C_{k+1} has less than M components.*
8. *Let $k = k+1$.*
9. *Repeat from step 3 until $k = N$.*
10. *Return $\boldsymbol{u}^* = \arg\min_{\boldsymbol{u} \in C_N} V(x, \boldsymbol{u})$.*

Step 3 builds the hypothesis set duplicating the candidates set taking into account that at time step $k+1$ for each candidate there are two possible values that the input can take (0 for $\mathbf{h}_i^{k+1}$ and 1 for $\mathbf{h}_{i+|C_k|}^{k+1}$). Note that in order to simplify the notation, all the input sequences are of dimension N and that by construction, all the inputs of the candidates in C_k are zero for the last $N-k$ steps.

Step 4 of the algorithm can be computationally demanding depending on the function used to approximate the optimal cost to go. In the example section we present an application in which for each candidate the optimal cost to go is approximated by carrying out an open-loop simulation with an explicit controller.

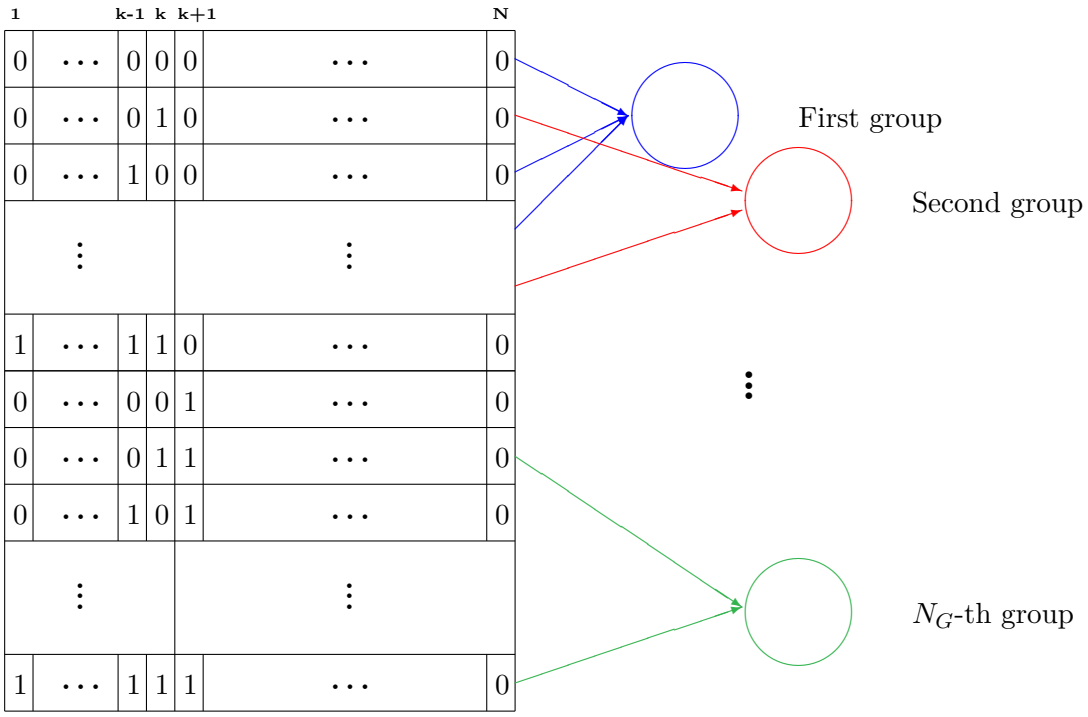
Step 5 of the algorithm groups the hypothesis sequences based on the value of $z^s(k)$ reached so that only sequences which yield a similar $z^s(k)$ should be later compared using

[1] By construction the $N-k$ components after step k of each hypotheses sequence are zero. This implies that sequences with similar last input values will be placed next to each other after sorting.



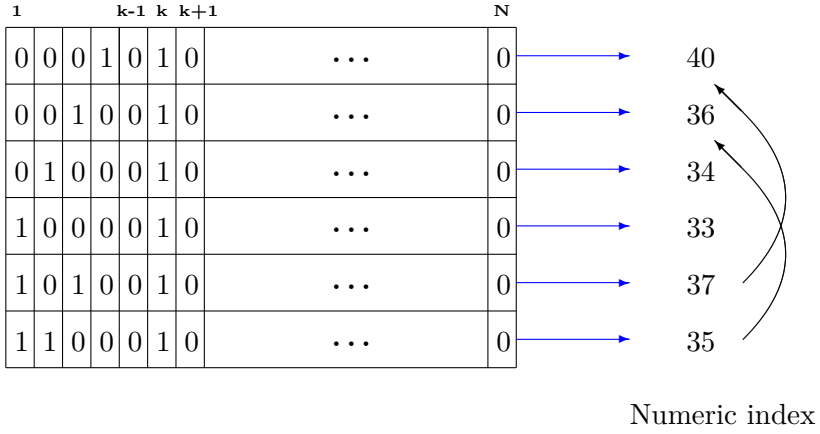
Step 2: Creation of set $\mathbf{C_k}$

Step 3: Formation of set $\mathbf{H_{k+1}}$



Step 5: Classification in N_G groups from set $\mathbf{H_{k+1}}$

Figure 2.2.1: Representation of step 2,3 and 5 of Algorithm 1.



Step 6: Ordering a group from the N_G groups.

Figure 2.2.2: Representation of step 6 of Algorithm 1.

the screening criterium. In doing so, it is more likely to achieve a small $\|x_a - x_b\|$ which would help to reject sequences. Thus this step is based on solving a classification problem in which the hypotheses sequences $\mathbf{h}_i^{k+1}$ are grouped depending on their corresponding value of the slow state of the $k+1$ steps ahead state $x_i^{k+1}(k+1)$. The complexity of this step depends on the dimension of $z^s(k)$, that is, the dimension of the slow dynamics. The number of elements of each group and the number of groups depends on the classification procedure. In the example section we consider a single dimension slow dynamic and propose to sort the hypotheses sequences by the value of their slow mode and divide them into a predefined fixed number of groups N_G with the same number of sequences. There are other low computational burden classification procedures such as classifying the sequences by their distance to a predefined set of a priori chosen points in the slow state space, possibly leading to groups with different number of sequences.

The purpose of step 6 of the algorithm is to ensure that the sequences that are to be compared using the screening criterium, in addition to yield a similar $z^s(k)$ also reach a similar $z^f(k)$. This helps to compare only sequences that lead to a small $\|x_a - x_b\|$, which in turn makes easier to reject sequences. This is based on sorting the sequences of each group accordingly a numerical index, which it is related to the value of $z^f(k)$, so that only sequences with a similar numerical index will be compared.

In step 7 the value of γ_Φ^k can be found explicitly once the estimated cost and predicted state differences between each of the neighbour sequences are computed because these values do not depend on γ_Φ^k. To this end, the comparisons are assumed to be simultaneous, that is, even if one sequence is rejected by one neighbour, it is still compared with its other neighbour.

Algorithm 1 can be modified to take into account different comparing procedures that pair sequences of H_{k+1}. In fact, instead of doing any classification or sorting, each sequence could be compared to each other hypotheses sequence. This procedure would possibly lead to a higher value of γ_Φ^k but, as commented before, that would lead to an unaffordable

number of comparisons. The procedure presented in Algorithm 1 provides a trade-off between optimality and computational cost by reducing the comparisons to those candidates closest in a particular sense.

Algorithm 1 does not consider explicitly constraint handling, but it can be modified to do so. Hard constraints on the input sequence values are very easy to deal with by simply rejecting those sequences that violate the constraints. Such rejection can be done before step 9, rejecting for example those sequences that have a high switching frequency. Note that this strategy has been used before in other works like [105]. This has the benefit of reducing the number of possible sequences to those which are feasible, thus making the problem easier to solve. Constraints on the state are more difficult to be tackled in this way and should be included in the performance cost and penalizing their violation like in exact penalty function methods ([74]).

Note that the algorithm does not need the bounds of Assumption 3, but it will provide better results if the cost-to-go function is a good approximation of the optimal cost-to-go.

One property of the proposed heuristic algorithm is that the computational complexity of the algorithm depends linearly on M and N, and also the computational burden is quite consistent and hence it can be tuned by means of M and N depending on the computation time available. The degree of suboptimality depends greatly on the values of M and N and the approximation error of the cost-to-go function.

2.3 Example

The algorithm presented in the previous section will be applied to a simulated model of a process containing a slow first order dynamics plus a fast oscillating second order one. The transfer function model, akin to that of the pH control in a photobiorreactor ([10, 77]), is:

$$G(s) = \frac{K}{\tau s + 1} \cdot \frac{w_n^2}{s^2 + 2\delta w_n s + w_n^2} \tag{2.28}$$

where $K = -1.6$, $\tau = 1320$ (seconds), $w_n = 0.05$ (radian/seconds) and $\delta = 0.085$. The output variable is the pH deviation from the operating point, which is 8.5. The manipulated variable is the opening of an on-off valve that controls the injection of CO_2 with a fixed flow into the photobiorreactor with the purpose of feeding the algae in the reactor. The CO_2 forms carbonic acid when dissolved into the water, lowering the pH. However, the CO_2 is consumed by the algae, thus rising the pH unless more CO_2 is injected.

The control objective is to regulate the pH to a reference value because the pH affects the growing rates and living conditions of the algae. As a secondary objective, the number of commutations of the valve will be minimized.

Note that the 2% settling time of the fast oscillating dynamics in (2.28) is approximately $\frac{3.9}{\delta \omega_n} = 917.6$ s, whereas the slow overdamped dynamic in (2.28) has an approximate 2% settling time of about $4\tau = 5280$ s. It is clear that this system can be described as a PFMS, which is also evident in the impulse response of the complete system (2.28) shown in figure (2.3.1).

In order to apply the proposed MPC control law, a state space discrete time model of (2.28) is used. The sampling time is 30 seconds. The state space representation has been

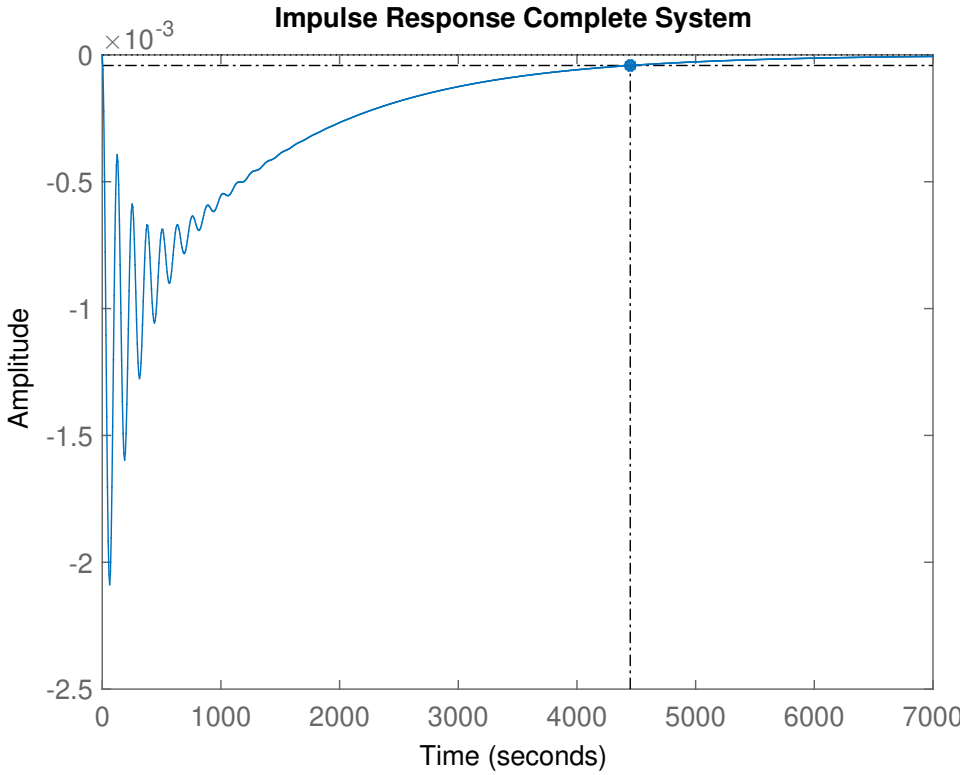


Figure 2.3.1: Impulse response and settling time of system (2.28).

chosen so that the slow and fast dynamics are decoupled using the Jordan canonical form. The slow dynamics state will be used to create the groups in the step 5 of algorithm (1).

To satisfy the control objectives, the stage cost of the MPC controller penalizes the square of the deviation plus a fixed cost if the valve changes its state, that is:

$$\ell(k) = J_{sp} + \alpha J_{sw} \tag{2.29}$$

where r is the target pH, $J_{sp} = (y(k) - r)^2$ is the cost of the tracking term, and $J_{sw} = |u(k) - u(k-1)|$ is the switching effort term (note that this term is one or zero for each stage depending on whether the valve switches or not) and the constant α provides a tuning knob to balance both terms. Note that this stage cost satisfies Assumption 2.

The prediction horizon is chosen to be $N = 100$, which is more than two time constants of the slow dynamics. This implies that the MPC optimization problem can be posed as a BQP with 100 binary variables which could be solved using standard optimization methods. The computational burden of that approach would be much greater than the one of the proposed algorithm.

The computation of the approximated costs in algorithm 1 require the selection of the approximation of the optimal cost to go $\hat{\ell}_k^N(x_k)$. Ideally, this approximation should be equal to the optimal cost to go, but this is very difficult in practice because it would imply solving an optimization problem similar to the original one and the approximation of the cost to go must be computed in real time for all the input sequences in the candidate solutions set. The strategy used in this example is to compute the cost to go using closed-loop simulations with explicit on-off controllers that can be computed in real time.

Note that the simulations required to compute the approximated costs of each candidate input sequence are independent of each other, thus the computations can be done in parallel. In fact, the problem is suited to be solved using general-purpose computing on graphics processing units (GPGPU) instead of using multi-core CPU implementations ([30, 53]). Using this parallel programming paradigm allows the computation of the optimal cost to go approximation for all the candidate input sequences in real time, with a computation time that is a very small fraction of the overall computation time required when a multi-core CPU implementation is used. Moreover, GPGPU computing using NVIDIA CUDA capable GPUs can be done in Matlab, thus it can be integrated easily with the implementation of algorithm 1.

Two possible explicit controllers have been considered: a relay controller that regulates the pH to the desired value with a sampling time of 30 seconds and a proportional controller that regulates the pH deviation to zero with a sample time of 300 seconds in which the continuous control signal is implemented using a 10-bit pulse frequency modulation (PFM) converter with a sampling time of 30 seconds.[2] Figures 2.3.2 and 2.3.3 show a simulation of system (2.28) in closed-loop with the relay controller and the proportional controller with $Kp = -3$. In the simulations, the initial pH is 8.5 and the reference switches from 8 to 8.2 at time 4500 seconds and from 8.2 to 7.8 at time 9000 seconds. The total simulation length is 14400 seconds. For the relay controller the accumulated reference tracking cost J_{sp} is 3.7052 and the total number of commutations is 176. For the proportional controller, the accumulated reference tracking cost J_{sp} is 2.7818 and the total number of commutations is 273. In this case, it can be seen that the proportional controller provides a better performance, hence it will be used in the optimization algorithm to estimate the optimal cost to go.

Although the computational complexity grows with N, the parameter which has the most influence on the performance of the algorithm is the maximum allowed candidate set size M. The computational burden grows linearly when it is increased, thus it is important to keep M as low as possible. However, the suboptimality of the solution and the values of γ_{Φ}^{k} are also determined by M. In order to demonstrate the trade-off between optimality and computational burden, over a hundred problems have been solved with different initial conditions and references for eight different values of M.

On the other hand, a small number of groups N_G leads to a range of variation of $z^s(k)$ within each group larger than the one obtained with higher values of N_G. Algorithm (1) assumes that the sequences in each group have a similar $z^s(k)$, thus large groups are to be avoided. On the other hand big values of N_G result in very small groups in which it can be difficult to find suitable pairs of sequences for the screening criterion of theorem (1). Therefore, the number of groups has been chosen as $N_G = M/256$, as a group size of 256 have been found to work well in this example. Furthermore, different values of M have been used to exemplify some aspects of the algorithm.

The problems for the comparison have been generated choosing a random pH reference and assuming that the system pH is equal to the target reference to focus on the regulation performance, that is, in this set of simulations the transient cost is zero. Note that the transient cost does not provide a lot of information on the performance of the proposed

[2]The proportional controller calculates the input as $u(k) = u_r + K_p(r - y(k))$ where u_r is the steady state opening (between 0 and 1) corresponding to the desired reference and is calculated using the static gain of system (2.28).

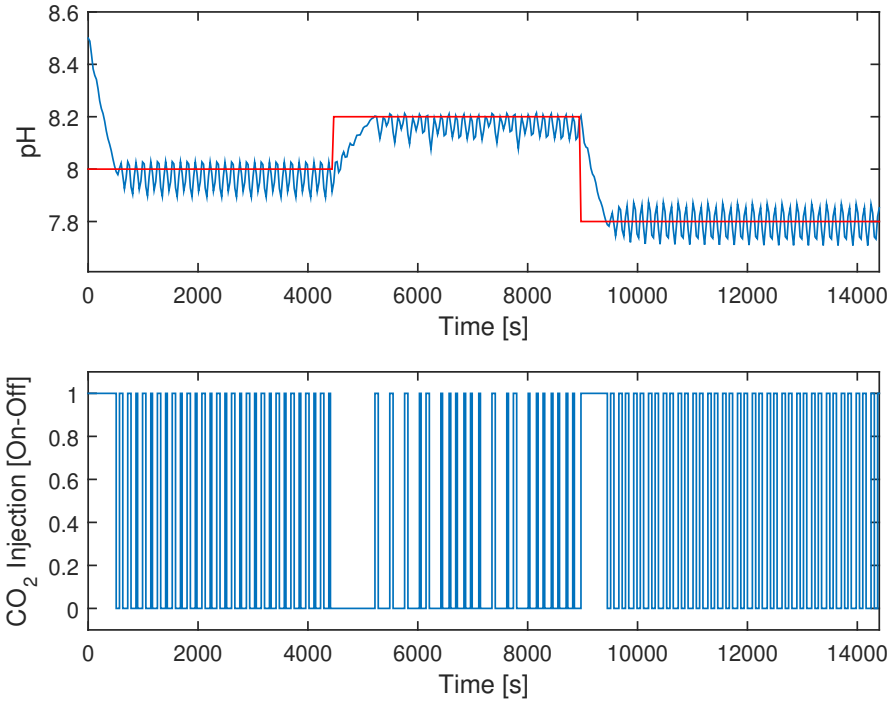


Figure 2.3.2: System controlled by a relay.

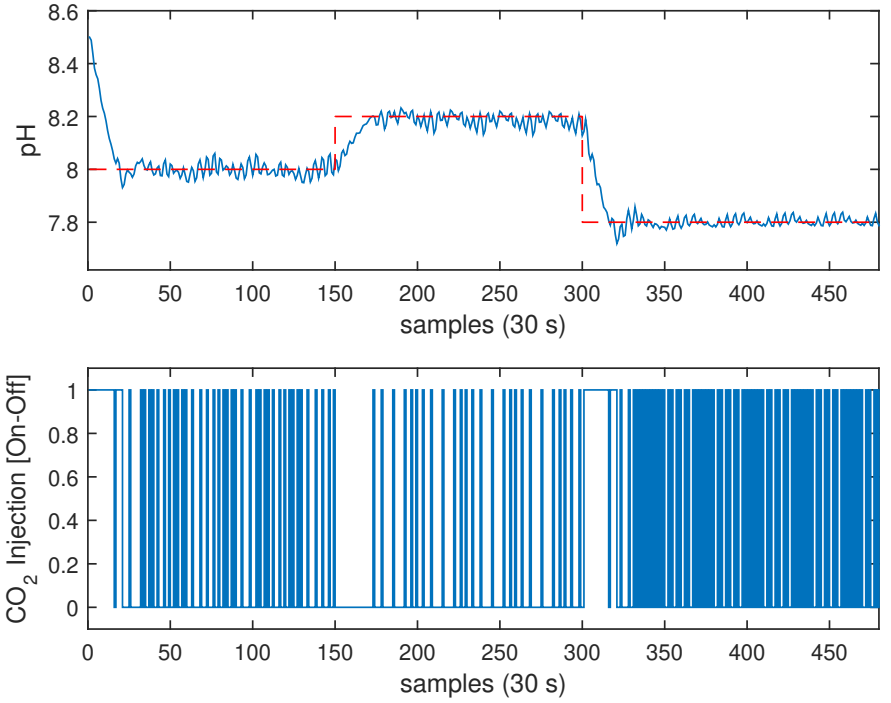


Figure 2.3.3: System controlled by a proportional controller with $K_P = -3$ and pulse frequency modulation.

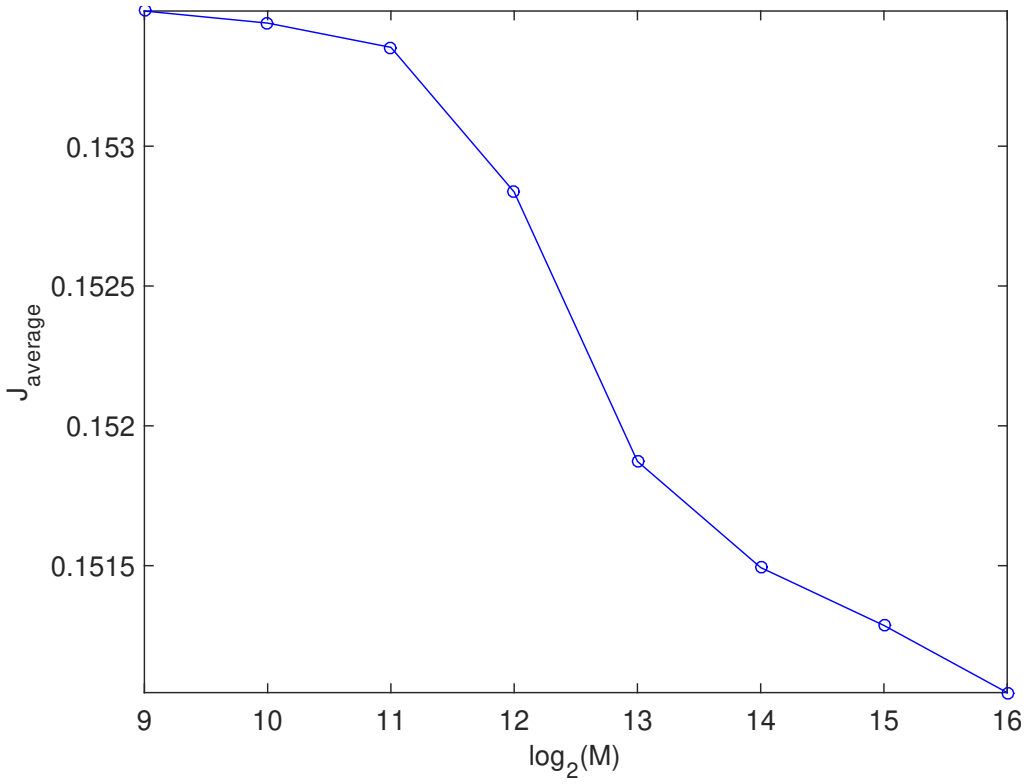


Figure 2.3.4: Relation between $\bar{J}$ and $\log(M)$.

controller because there is a single input that can only take two possible values, and hence, it only starts switching once the set point is reached. For each value of M 100 optimization problems were solved, and the average cost of the 100 different optimization problems and the mean value of the parameter γ_Φ^k have been computed. Figures 2.3.4 and 2.3.5 show the results of these tests. It can be seen in figure 2.3.4 that the mean cost decreases with M.

Figure (2.3.5) shows how increasing the value of M leads to greater values of the average γ_Φ^k. If M is increased, the difference between the slow state projection of neighbouring sequences is generally smaller. This smaller difference results in larger γ_Φ^k values that still meet the screening criterion.

A set of 100 problems with random initial states and random set points have been generated for increasing values of the prediction horizon in order to obtain computation timing and suboptimality measurements. Note that in this sort of problems transient cost is different to zero. Table 2.1 shows the mean value and the standard deviation of the computation time and the suboptimality average for the set of random problems solved with PFMS algorithm and $M = 2^{15}$. The optimal solution for each of the same 100 random is necessary in order to calculate the suboptimality average, thus they are provided by SCIP solver based on SoPlex [64] with zero duality gap[3]. It can be seen that the proposed strategy is a quasi-exhaustive heuristic algorithm, quite consistent in the computational

[3]The optimal solution has been obtained only up to $N = 50$ because it takes too much time to compute it for larger values of N (note that for $N = 50$ the average computation time of SCIP is 345.96 seconds and the standard deviation is 1214.93 seconds).

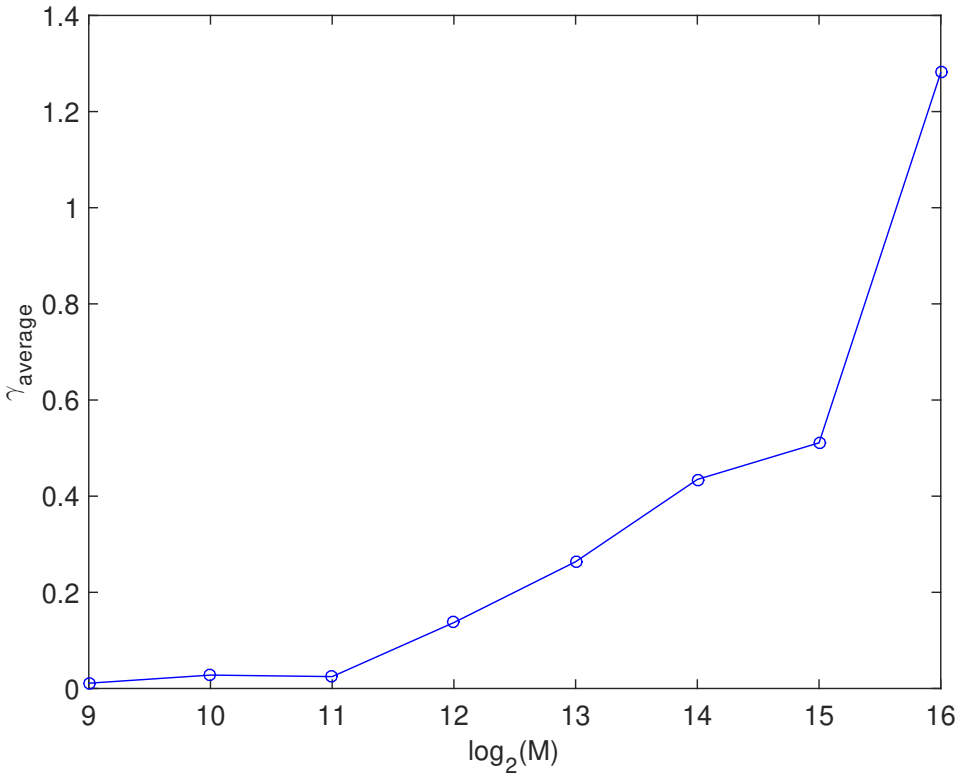


Figure 2.3.5: Relation between $\bar{\gamma}_{\Phi}^{k}$ and $\log(M)$.

burden, which leads to a low standard deviation in the computing times. This consistency in the computational burden allows a fine tuning of the maximum M which yields the most accurate solution. Furthermore, the average suboptimality of the proposed algorithm is very low even for long values of N.

Table 2.2 shows the average and standard deviation of the computation time and average suboptimality of the proposed algorithm computed for the set of one hundred random problems for different values of M and $N = 50$. It can be seen that the average computing times increases in a close to linear trend as M increases and that, as expected, the average suboptimality decreases as M increases.

Figures 2.3.6 and 2.3.7 show closed-loop simulations with set-point changes for two differ-

	PFMSA ($M = 2^{15}$)		
N	$\mathbf{t_{ave}}$	$\sigma_{\mathbf{t}}$	$\%_{\mathbf{sub\text{-}opt}}$
30	2.9877	0.1895	0.3544
35	3.9249	0.0903	0.2787
40	4.8481	0.0957	0.4985
45	5.7933	0.0947	0.2007
50	6.7910	0.1118	0.2265

Table 2.1: Computational timing average and standard deviation and subotimality average PFMSA with $M = 2^{15}$.

	PFMSA ($N = 50$)		
M	$\mathbf{t_{ave}}$	$\sigma_{\mathbf{t}}$	$\%_{\mathbf{sub\text{-}opt}}$
2^8	0.3130	0.0076	1.1290
2^9	0.3726	0.0068	0.5861
2^{10}	0.4025	0.0170	0.5853
2^{11}	0.5848	0.0222	0.4260
2^{12}	0.9731	0.0327	0.2503
2^{13}	1.8153	0.0466	0.3256
2^{14}	3.4488	0.0685	0.2963
2^{15}	6.8292	0.1741	0.2265
2^{16}	14.6569	0.1722	0.2176
2^{17}	36.4092	1.2354	0.2330
2^{18}	102.7905	2.8205	0.2527

Table 2.2: Computational time average, standard deviation and sub-optimality average analysis of PFMSA with $N = 50$.

Control	M	α	J_{sp}	n_{sw}	$J(0)$	$J(0.001)$	$J(0.002)$	$J(0.005)$
PFMSA	2^{15}	0	2.6355	278	**2.6355**	2.9135	3.1915	4.0255
		0.001	2.6565	241	2.6565	**2.8975**	3.1385	3.8615
		0.002	2.8523	132	2.8523	2.9843	**3.1163**	3.5123
		0.005	2.96	79	2.96	3.039	3.197	**3.355**
P ($K_p = -3$)			2.7818	273	2.7818	3.0548	3.3278	4.1468
Relay			3.7052	176	3.7052	3.8812	3.9572	4.5852

Table 2.3: Closed-loop performance comparison.

ent α values, $M = 2^{15}$ and $N_G = M/256$. The optimization problem is solved in less than 10 seconds on a computer with Intel Core i7-4790 CPU and a Matlab implementation of algorithm 1 using the Parallel Computing Toolbox for computing the optimal cost to go approximation on a mid-range consumer-grade NVIDIA GM206 GPU. The effect of varying α can be seen in figure 2.3.6 for $\alpha = 0.001$ and figure 2.3.7 for $\alpha = 0.005$. It is clear that, for greater values of α, the number of valve switchings is lower.

Finally, a comparison of the tracking cost (column J_{sp}) and the number of switches (column n_{sw}) can be seen in table (2.3) for the MPC controller with different values of α, the relay controller and the P controller with the PFM converter and $K_p = -3$. Notice that on the right side, there is a comparison of the total cost ($J(\alpha)$) for different values of α. For each case of α value, notice that the minimal value of the total cost is obtained with the algorithm implementation calculated with the same α. Also, note that the predictive controller always performs with a lower cost than the relay or proportional controller.

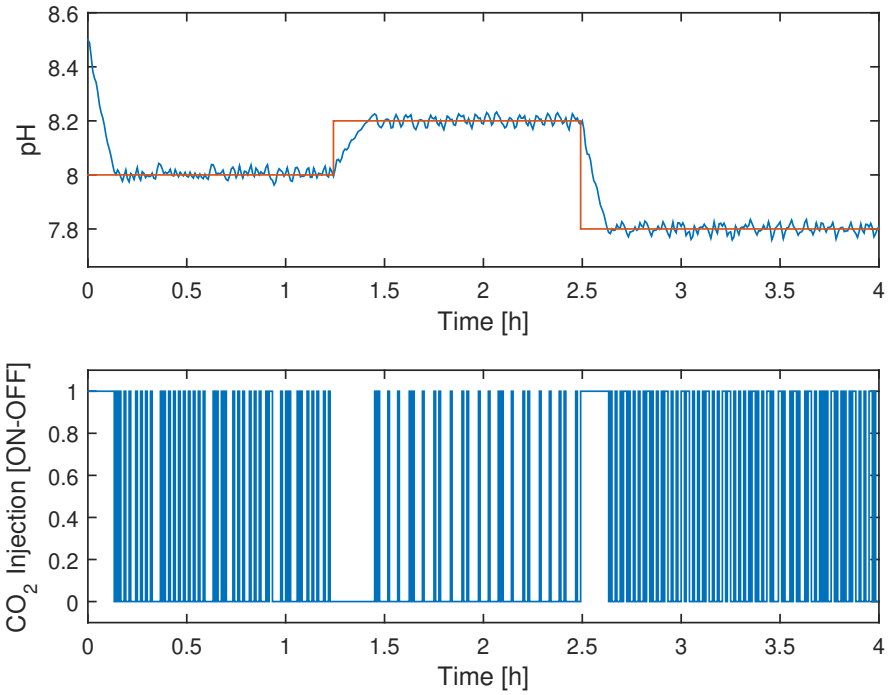


Figure 2.3.6: System controlled by PFMSA with $\hat{\ell}_k^N(x_k)$ computed using a proportional controller, $M = 2^{15}$, $N_G = 2^7$, $N = 100$ and $\alpha = 0.001$.

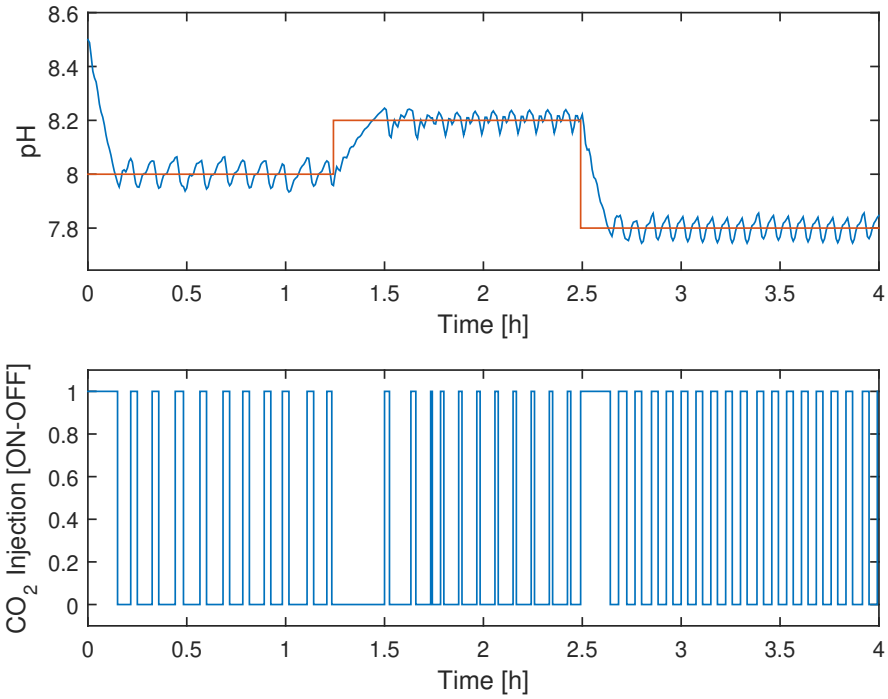


Figure 2.3.7: System controlled by PFMSA with $\hat{\ell}_k^N(x_k)$ computed using a proportional controller, $M = 2^{15}$, $N_G = 2^7$, $N = 100$ and $\alpha = 0.005$.

Chapter 3

Historian data based predictive control

In chapter 2, a MPC controller for systems with binary actuators and partially fading memory was presented. The proposed approach inherits the good properties of MPC such as dealing with linear and nonlinear system with single or multiple inputs/outputs, managing constraints and performance optimization taking into account future predicted trajectories. However, it is a model based approach, which is sometimes hard to obtain, specially for large scale systems like WDN. In addition, even if a model is available, the computation of the cost-to-go of a given candidate can be difficult or cumbersome to obtain.

On the other hand, as presented in chapter 1, complex network systems usually provide a huge amount of data that can be used for identification proposes. In this chapter, a different approach is taken regarding these goals. Instead of first identifying a model and then finding the best future input trajectory based on this model, we limit the set of possible input trajectories to a convex combination of past trajectories with an initial state close (in some sense) to the current state of the system. We will then use the information in the database to estimate the performance of the chosen trajectory using a heuristic linear approximation based on a particular extreme case of direct weight optimization methods [92] that results in the solution of a QP problem (or LP if only regulation around a set point is considered). This is a predictive control strategy in the sense that it uses the future in the past to predict the future evolution of the process. Note that this concept is very general and it does not impose almost any condition on the closed loop trajectories stored in the database, although it would be logical to consider only those that resulted in a good control performance.

A typical assumption is that, although the model is unknown, appropriate data for identifying the system dynamics or a specific control law is available, either from appropriate identification experiments or through the extensive use of a simulator, which may not be possible for some applications. However, we do not assume that the trajectories stored in the database have been chosen or designed to provide sufficient information for a precise identification of the system dynamics or the corresponding control law. This is the case when historian trajectories are available from the closed-loop operation of the system under different conditions, controllers and even control objectives. This issue limits the

applicability of standard identification procedures in which the quality of the information available greatly affects the quality of the resulting identified models.

3.1 Problem formulation

The system considered throughout the chapter will be represented by a discrete time invariant system:

$$x(k+1) = f(x(k), u(k)), \tag{3.1}$$

where $x \subset \mathbb{R}^{n_x}$ is the state, $u \subset \mathbb{R}^{n_u}$ is the input and f is the unknown transition function.

A regulation problem is considered along this chapter, thus the control objective is to regulate the system to the origin while minimizing the performance defined by a stage cost $\ell(\cdot,\cdot)$. Standard model predictive control [25, 62, 88] is based on solving an optimal control finite horizon problem in which the cost of a predicted trajectory of length N is minimized at each time step to obtain an optimal control sequence that is applied in a receding horizon manner. In order to approximate an infinite horizon problem, often a terminal cost function $\ell_N(\cdot)$ is also considered, which leads to the finite horizon performance cost $V : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u N} \to \mathbb{R}$:

$$V(x(k), \mathbf{u}) = \sum_{i=0}^{N-1} \ell(x(k+i|k), u(k+i|k)) + \ell_N(x(k+N|k)), \tag{3.2}$$

where $x(k+i|k)$ is the predicted state obtained applying the future input sequence $u(k+i|k)$ with $i = \{0, \ldots, N-1\}$ from the initial state $x(k)$, $\ell(\cdot,\cdot)$ and $\ell_N(\cdot)$ are convex positive definite functions, N defines the prediction horizon and

$$\mathbf{u} = \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+N-1|k) \end{bmatrix} \tag{3.3}$$

is the optimization variable.

Standard MPC solves an optimization problem based on the model. In this chapter, as the function that models the system f is unknown, a database will be used to obtain a straightforward estimation of the cost V which implicitly predict the behaviour of the system. The database stores different closed loop trajectories. These trajectories contain the state and the input trajectories of different controllers applied in the past. The information stored in the historian database is used to generate a set of candidate trajectories of appropriate length. This set of candidates is built using all the sample times in the historian database for which a N step trajectory is available. Each candidate trajectory q is defined by its state and input after i time steps, $x_q(i), u_q(i)$ with $i = 0, \ldots N$, where $x_q(0)$ is the initial state of the candidate trajectory and they satisfy

$$x_q(i+1) = f(x_q(i), u_q(i)). \tag{3.4}$$

We denote its corresponding input trajectory as

$$\mathbf{u_q} = \begin{bmatrix} u_q(0) \\ u_q(1) \\ \vdots \\ u_q(N-1) \end{bmatrix} \tag{3.5}$$

and its corresponding cost for the objective function considered as

$$V_q(x_q(0), \mathbf{u_q}) = \sum_{i=0}^{N-1} \ell(x_q(i), u_q(i)) + \ell_N(x_q(k+N)). \tag{3.6}$$

In this chapter, we propose to use a control law derived from the control trajectories in the candidates set. Following a receding horizon approach, at each sampling time, the control signal to be applied will be computed as a weighted sum of the initial control signals of the candidates considered, that is

$$u^*(k) = \sum_q \lambda_q^* u_q(0), \tag{3.7}$$

where the optimal values of the weights λ_q are chosen to minimize an estimation of $V(x(k), \sum \lambda_q \mathbf{u_q})$. The following section discuss how to define this estimation and implement the proposed controller.

3.2 Controller formulation

The proposed predictive controller computes the current control actions as a weighted sum of past control actions. In order to minimize the computational burden and to provide good cost estimations based on local data, only a subset of the past control actions in the historian are considered in current control computations. This subset is comprised of closed loop trajectories starting from an initial state close to the current state of the process.

In particular, at time k, we propose to choose the n_Q candidates with an initial state $x_q(0)$ closest to the current state $x(k)$ taking into account a given metric[1]. Once the n_Q candidate trajectories are obtained, the optimal control sequence will be chosen among the convex combination of the control sequences of the candidate trajectories with an initial state equal to $x(k)$, that is, the optimization variables are the n_Q weights λ_q with $q = 1, \ldots, n_Q$ such that:

$$\begin{aligned} & x(k) = \sum_{q=1}^{n_Q} \lambda_q x_q(0), \\ & \sum_{q=1}^{n_Q} \lambda_q = 1, \\ & \lambda_q \geq 0, \quad \forall q \in \{1, \ldots, n_Q\}. \end{aligned} \tag{3.8}$$

For a given choice of candidates weights λ_q, we consider the following estimation of its corresponding cost

$$V\left(x(k), \sum \lambda_q \mathbf{u_q}\right) = V\left(\sum \lambda_q x_q(0), \sum \lambda_q \mathbf{u_q}\right) \simeq \sum \lambda_q V_q. \tag{3.9}$$

This approach is related to direct weight optimization nonlinear identification methods [92]. DWO methods are based on postulating an estimator that is linear in the observed outputs of the estimated function (in this case the predicted cost) and then

[1]Candidates with the same distance value will be randomly selected if necessary so that only n_Q candidates are included in the set.

determining the weights in this estimation by direct optimization of a suitable chosen criterion. Different criteria can be defined depending on the properties of the estimated function in order to take into account the non linearities of the function by penalizing far away points and the effect of noise. In the proposed approach, we assume that for the set of n_Q closest candidates, the function can be approximated by a linear function and we neglect the effect of noise. In practice, using local trajectories is similar to carrying out an online linearization of the dynamics around the current state. Under these assumptions, (3.9) is a valid estimation of the cost for any particular choice of weights (see remark 4 in [92]). The optimization then is carried out to optimize the expected predicted performance as in standard MPC. Note that because of the receding horizon implementation, the estimation of the future cost is recalculated at each sampling time, reducing the effect of the prediction error through feedback.

Summing up, the proposed controller is based on solving the following linear programming optimization problem based on minimizing the upper bound on the trajectory defined by the convex combination of the candidate trajectories that start from the current state:

$$\begin{aligned} \min_{\lambda \in \mathbb{R}^{n_Q}} \quad & \sum_{q=1}^{n_Q} \lambda_q V_q \\ s.t. \quad & x(k) = \sum_{q=1}^{n_Q} \lambda_q x_q(0), \\ & \sum_{q=1}^{n_Q} \lambda_q = 1, \\ & \lambda_q \geq 0, \quad \forall q \in \{1, \ldots, n_Q\}. \end{aligned} \tag{3.10}$$

Problem (3.10) is a LP problem that can be solved using off-the-shelf algorithms, even for large number of optimization variables. In the next section, in order to illustrate the proposed strategy, it is applied to control the Richmond water distribution system.

Remark 1. *We notice that if $V : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u N} \to \mathbb{R}$ is a convex function in $\mathbb{R}^{n_x} \times \mathbb{R}^{n_u N}$ then, in view of Jensen's inequality[18], we have*

$$V\left(x(k), \sum \lambda_q \boldsymbol{u_q}\right) = V\left(\sum \lambda_q x_q(0), \sum \lambda_q \boldsymbol{u_q}\right) \leq \sum \lambda_q V_q. \tag{3.11}$$

Thus, the proposed approach, under the convexity assumption, minimizes an upper bound of the cost for the chosen future input trajectory. Note that V is convex for linear systems and quadratic cost functions, which are widely used in the MPC literature.

Remark 2. *Although we have considered a state feedback setting, the proposed approach can also be used in an output feedback setting in which the output measurements are stored in the database. In addition, constraints can be taken into account using the weighted predicted state and input trajectories. In section 3.3 we present an example in which a constraint on an output of the system (the pressure at one of the demand nodes of the water drinking network) is considered.*

Remark 3. *The cardinality of the candidate trajectories n_Q is important because larger values carry higher computational burdens. Moreover, a large number of candidates increases the distance from $x(k)$ of the last candidate selected with a local linearity loss (if the system were nonlinear). On the opposite side, smaller values of n_Q could produce feasibility problems. Figure 3.2.1 shows the feasibility problem in $\mathbb{R}^2$. On the left side, S_3*

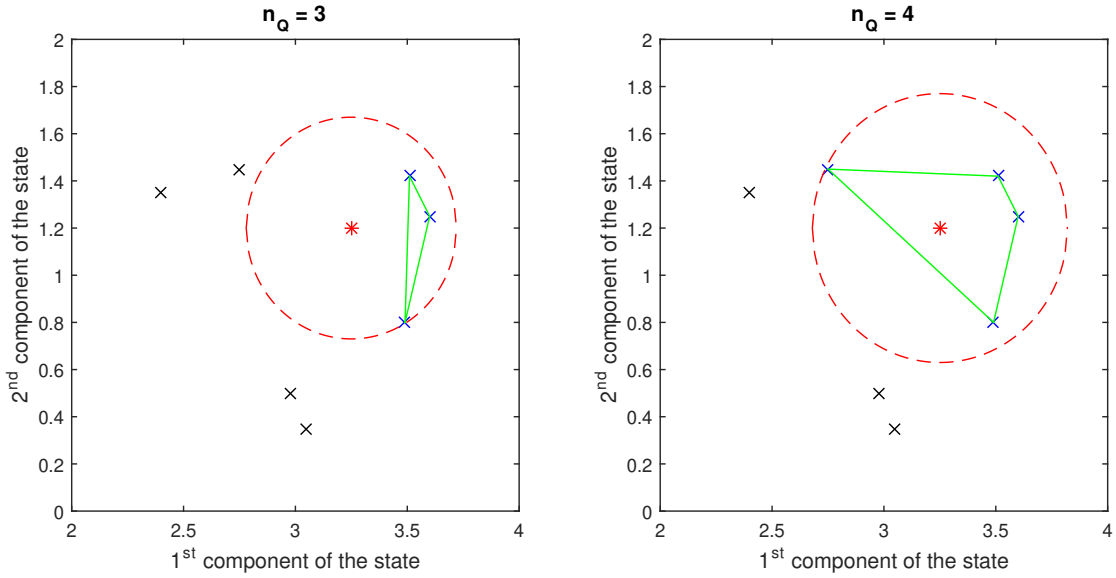


Figure 3.2.1: Feasibility problem in $\mathbb{R}^2$: left infeasible, right feasible.

is the subset contained in the convex envelope formed by states $x_q(0)$ with $n_Q = 3$. It is shown that $x(k) \notin S_3$, so the optimization problem is infeasible. On the right side, S_4 is the subset contained in the convex envelope formed by states $x_q(0)$ with $n_Q = 4$. In this case, $x(k) \in S_4$, so the minimization problem is feasible. The solution to feasibility problems could be to increase the value of n_Q to find a convex envelope that contains $x(k)$. However, sometimes this may not be possible because there is not enough information in the database or because the current state is close to the system operating boundaries and the trajectories in the database operate far from these boundaries. In these cases, a different solution has to be considered. In this approach we propose to apply the $u_q(0)$ of the closest candidate. However, there are other options like using the input corresponding to the nearest point of the convex envelope of the candidates or consider some form of extrapolation procedure.

Remark 4. *Hyperparameters of the trajectories stored in the data base are defined as additional information of the controllers that generate these trajectories. Examples of hyperparameters are the tuning values of a PID controller, reference and hysteresis of a relay controller or the prediction horizon and the matrices that define the cost of a MPC controller. Hyperparameters can be taken into account in the distance function, the cost function, or in the constraints of the optimization problem to improve the performance of the proposed approach. In section 3.3 we present an example in which the information of the reference of the controller of each trajectory stored in the database is used as an hyperparameter in the proposed controller.*

3.3 Application to a water distribution system

The Richmond water distribution system is a well known case study [111, 112] that can be simulated using the EPANET hydraulic simulation software [34]. This case study describes a system that is a good candidate for the historian data based predictive control strategy presented in this chapter which has also been used in a standard MPC framework, see [114]. Although the system is nonlinear and complex, it can be approximated by a

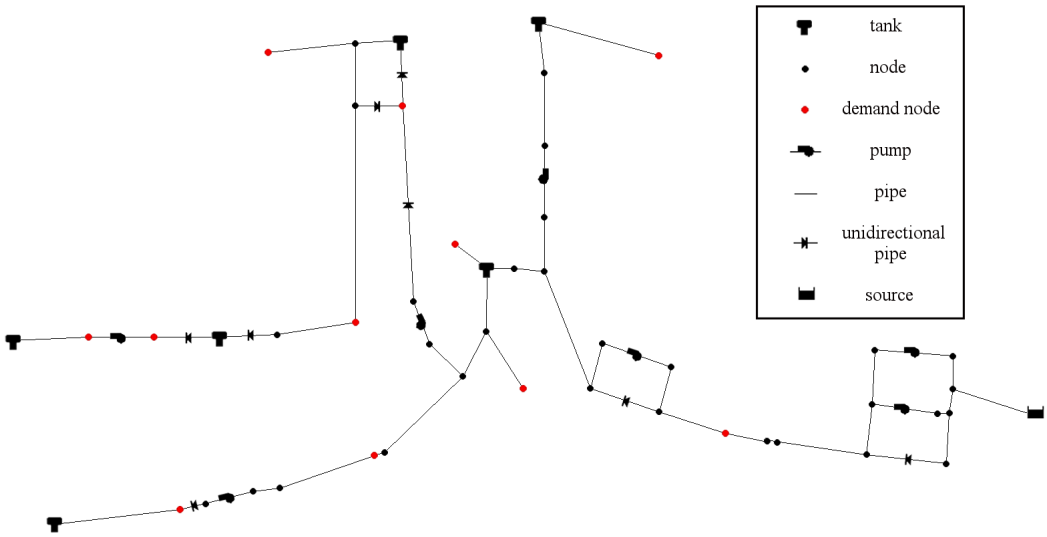


Figure 3.3.1: Richmond water distribution network diagram.

linear water balance based model with sufficiently large sampling times, in this case, one hour. Figure 3.3.1 shows the Richmond water distribution system diagram which is composed by 6 tanks, 7 pumps, 41 nodes of which 11 are demand nodes, 44 pipes of which 8 are unidirectional pipes and 1 source.

Note that for a given tank there are several demand nodes that withdraw water from that tank. Water is introduced by the pumps from a single water source and demand nodes consume this water, lowering the levels in the tanks. The control objective considered is to keep the water levels in each tank around a specified set-point, while satisfying the demands. The state vector x is composed by the levels of the 6 tanks, that is, $x \in \mathbb{R}^6$. Demands are considered disturbances, modelled by the disturbance signal vector $d \in \mathbb{R}^{11}$.

In order to attain the control objective, water flows are used as manipulated variables, thus the input signal vector $u \in \mathbb{R}^7$ contains the water flows that have to be attained using each pump in the system. Note that in the Richmond case study (as in many water distribution systems), pumps are operated with an ON-OFF mode, thus the necessity of a low level switching logic that transform each real component of u into an equivalent logic sequence for each particular pump. In this section, discrete time intervals of 1 hour ($k \in \mathbb{N}$) and low level switching logic intervals of $\frac{1}{24}$ hours = 2.5 minutes are considered. To minimize the number of pump switches, a duty cycle policy consisting on applying the control effort in a single pulse is used.

It is assumed that the water demand is composed by a periodic signal with a random component, that is:

$$d(k) = d_p(k) + d_r(k), \tag{3.12}$$

where $d_p \in \mathbb{R}^{11}$ is a set of periodic signals that satisfy $d_p(k) = d_p(k+T)$, with $T = 24$ hours, obtained from the demand profiles used by Zyl [111, 112] and $d_r(k)$ is a set of random zero mean signals, which added to $d_p(k)$ result in a demand signals with 5% of variation around $d_p(k)$.

The Richmond water distribution system has 6 tanks, however, because of the network

		RbC 1		RbC 2		RbC 3		RbC 4	
Pump	Tank	Level ON	Level OFF	Level ON	Level OFF	Level ON	Level OFF	Level ON	Level OFF
$u1$	$x1$	2.3685	2.9799	2.4785	3.0899	1.5018	2.1132	1.9352	2.5466
$u2$	$x1$	3.0405	3.2513	3.1505	3.3613	2.1738	2.3846	2.6072	2.8180
$u3$	$x1$	2.8888	3.1126	2.9988	3.2226	2.0221	2.2459	2.4555	2.6793
$u4$	$x2$	3.2623	3.5789	3.3323	3.6489	2.5956	2.9122	2.9290	3.2456
$u5$	$x3$	0.7185	1.8850	0.8285	1.9950	0.5852	1.7517	0.6518	1.8183
$u6$	$x4$	1.5907	1.9708	1.7207	2.1008	1.2774	1.6575	1.4340	1.8141
$u7$	$x6$	1.7037	2.1095	1.7837	2.1895	0.7037	1.1095	1.2037	1.6095

Table 3.1: Switching levels (in meters) of the relays of each of the controllers used to generate the data base.

	$x1_r$	$x2_r$	$x3_r$	$x4_r$	$x6_r$
RbC1	2.9403	3.4206	1.3018	1.7808	1.9066
RbC2	3.0503	3.4906	1.4118	1.9108	1.9866
RbC3	1.6403	2.4206	1.1018	1.3108	0.4066
RbC4	2.5069	3.0873	1.2351	1.6241	1.4066

Table 3.2: Set-points (in meters) of the controllers used to generate the data base.

structure, in order to maintain the desired levels, Tank 5 is always full in normal operating conditions. For this reason, only the rest of the tanks are considered for the purposes of computing the control signals. Note, however, that the EPANET simulation uses the whole state information.

The cost function in these examples is the following tracking error penalty stage cost that only depend on x (i.e., in these examples $l(x,u) = l(x)$):

$$\ell(x) = |x_c - x_r|_2^2, \tag{3.13}$$

where $x_c \in \mathbb{R}^5$ are the levels of the 5 tank levels considered of x and $x_r \in \mathbb{R}^5$ are their corresponding reference values. Furthermore, the terminal cost is equal to the stage cost, that is $\ell_N(\cdot) = \ell(\cdot)$.

The database stores the closed loop trajectories of four different controllers, each one based on a different set of relays denoted RbC1, RbC2, RbC3 and RbC4 respectively. Every pump is switched on and off depending on the level of the tank that is directly downstream. Table 3.1 shows this relation and the switching on and off levels for every pump of each controller.

Table 3.2 shows set-points values for each controller. The set-points are obtained as the middle point of the corresponding pump switching on and off levels. In the case of tank 1, denoted as $x1$ which is the first component of x, the set-point is the average of the three middle switching levels of pumps 1, 2 and 3, denoted as $u1$, $u2$ and $u3$ respectively.

	Tank 1	Tank 2	Tank 3	Tank 4	Tank 5	Tank 6
Min. level	1.02	2.03	0.5	1.1	0.2	0.19
Max. level	3.37	3.65	2	2.11	2.69	2.19

Table 3.3: Maximum and minimum safety tank water levels (in meters) in Richmond system.

For each controller, there are 100 trajectories stored in the data base, each one with 96 hours of closed loop simulated operation of the system. Each of the trajectories starts with different random initial values of the tank levels that satisfy the minimum and maximum safety constraints of table 3.3. In addition, the head at the demand node four, denoted $z(k)$, is also included in the database. Head information will be used to include a soft constraint to limit the maximum pressure in that particular demand node to demonstrate that historic data can be used to model complex, possibly nonlinear, outputs and take them into account in the control decision. The amount of information stored in the database is equivalent to 4.38 years of historian information. Although a realistic size for a historian, the database is very small in relation to the dimensionality of the problem which is $\mathbb{R}^6$, which could lead to identification problems.

Using the information of these simulations, over 38000 24-h candidates trajectories are defined, for which at each sampling time, only n_Q will be considered to formulate the controller. The choice of the number of candidates considered depends mainly on the density and quality of the historian trajectories stored in the database. There is a trade-off between complexity, feasibility and relevance. Higher number of candidates in general avoid feasibility issues, but may take into account candidates whose information is not relevant because its initial state is too far away from the current state leading to lower quality cost estimations. In this example, a value of $n_Q = 500$ has been found to be appropriate in regard to the aforementioned trade-off.

3.3.1 Example 1

The example is a closed loop simulation of 120 hours with initial state:

$$x(0) = [3, 2.44, 1.58, 1.5, 0.99, 1.51], \tag{3.14}$$

and the reference used is:

$$x_r = [3.0503, 3.4906, 1.4118, 1.9108, 1.9866]. \tag{3.15}$$

This reference is equal to the reference of one of the controllers used to create the database, in particular the relay based controller 2 (RbC 2). In order to take into account the periodic nature of the demand, the distance function takes the following form:

$$dist_q = \|x(k) - x_q(0)\|_\alpha^2 + \alpha_p \Delta_q(k)^2 \tag{3.16}$$

where

$$\alpha = diag\left(\begin{bmatrix} 1.0831 & 1 & 1.825 & 1.7299 & 1.6667 \end{bmatrix}\right), \tag{3.17}$$

where $diag(\cdot)$ is the diagonal matrix formed with the elements of the vector in the argument and $\alpha_p = 0.2$. The term $\Delta_q(k)$ is the time difference, in hours, between the candidate initial time and the current time. This distance function (3.16) takes into account the periodic nature of the demands of the system, penalizing candidates trajectories that start at a different hour because the demand differs.

The optimization problem solved at each time instant is defined in (3.10) and the control input applied is calculated as in (3.7). A standard LP solver can be used to obtain the solution as, in our case, *linprog* in Matlab. Figure 3.3.2 shows the water levels in the six tanks of the Richmond system during the closed loop simulation using EPANET.

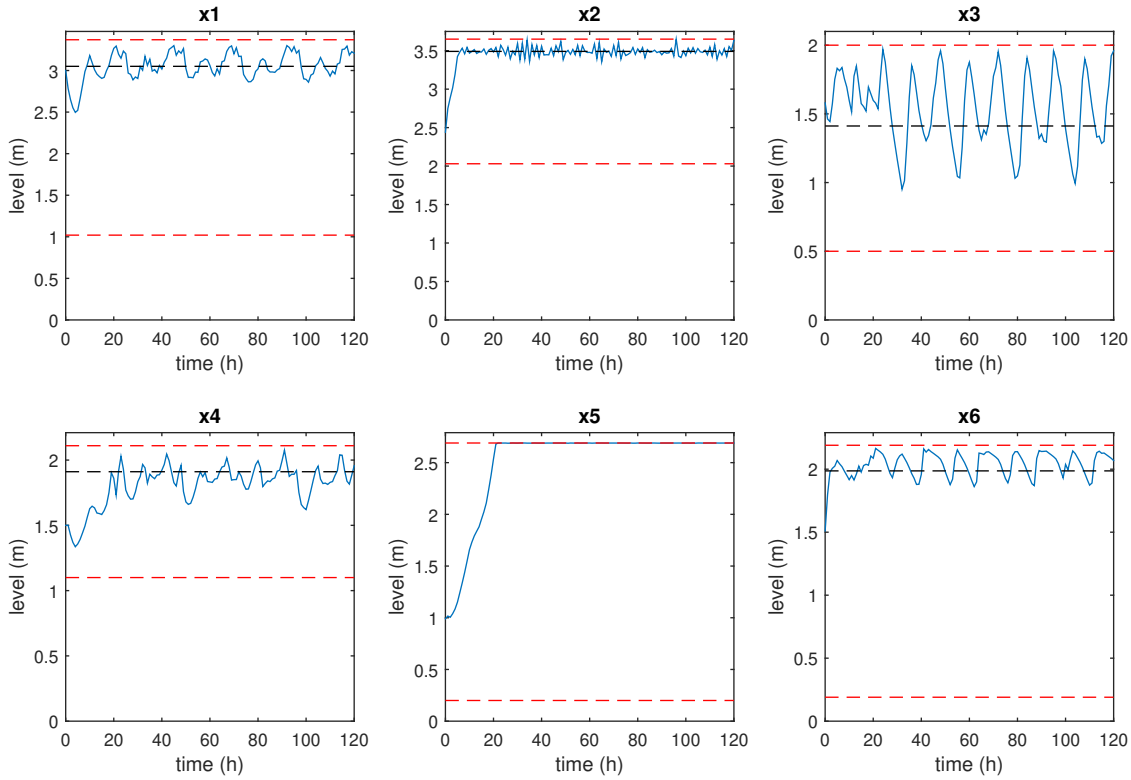


Figure 3.3.2: References (black dash) and closed-loop trajectories (blue solid) of the tank levels with the proposed controller. Maximum and minimum physical level constraints (red dashed) are represented for each tank.

Minimum, maximum and reference levels are represented for each tank (in red dashed and black dashed respectively). Tank 5 is not controllable and there is not any reference signal in its level graphic.

The pump flows, in litres per second, can be observed in figure 3.3.3. Notice that the solution obtained with the proposed controller tends to a quasi periodic behaviour. Since demand signals periodicity produces periodic trajectories and control action stored in the database when system is controlled with relay controllers, the convex combination of the control actions in the database is almost periodic too.

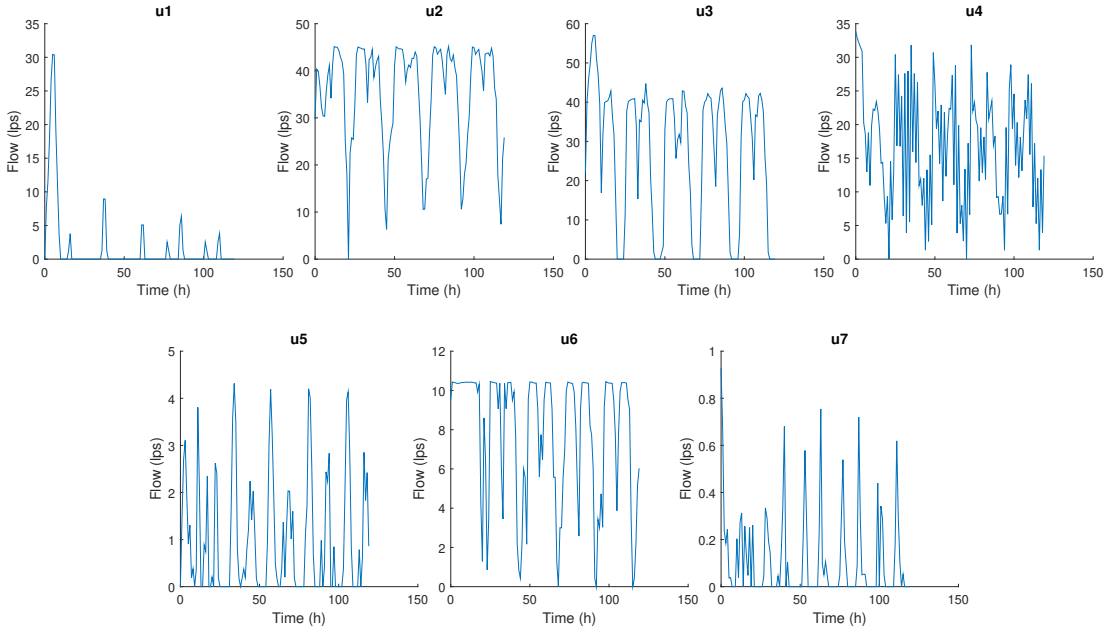


Figure 3.3.3: Closed-loop control actions for each pump with the proposed controller.

To evaluate the performance of the controller it is necessary to take into account the periodic nature of the system. The performance metric will be the summation of the closed loop performance cost over a period of $d(k)$ computed at each hour of the simulation as:

$$J(x(k)) = \sum_{i=0}^{N} \ell(x(k+i)), \tag{3.18}$$

where $x(k+i)$ are the values of the tank levels of the closed loop simulation, $\ell(\cdot)$ is defined as in (3.13) and $N = 24$. Note that in this system, the instantaneous performance cost has no meaning, as it will go up and down as the periodic disturbance changes. The summation of the closed loop performance cost over a period is a sensible choice as it should converge to a constant value when the closed-loop system reaches a quasi steady state periodic trajectory, provided that the controller is working fine. Note that the random part of $d(k)$ has an impact on the behaviour of $J(x(k))$.

Figure 3.3.4 shows the evolution of $J(x(k))$ for two of the relay based controllers and the proposed strategy. The relay based controllers 3 and 4 are not represented because their performance costs are much higher. In particular, their mean cost are 76.99 and 27.7 respectively, while the mean cost of the proposed historian data based predictive control

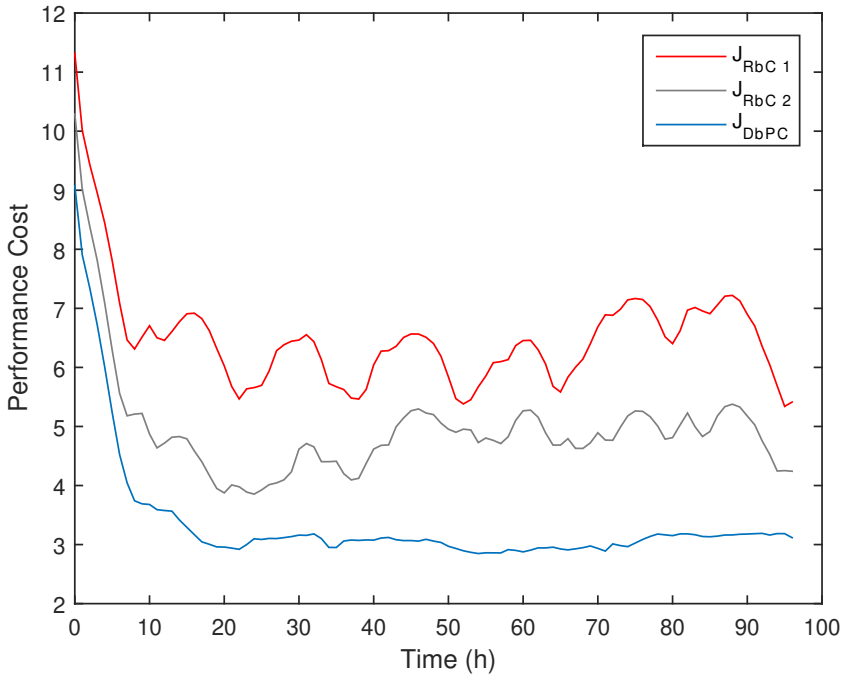


Figure 3.3.4: Performance cost comparison of the proposed controller (J_{DbPC}) and the relay based controllers 1 and 2 (J_{RbC1} and J_{RbC2}).

is 3.36. The historian data based predictive controller improves the controllers included in the data-base, although the performance and behaviour is similar to a relay controller.

3.3.2 Example 2

In this example the proposed strategy is applied taking into account the set-point value of each closed loop trajectory stored in the database. The reference of each relay based controller is shown in table 3.2. According to Remark 4, which presents hyperparameters and the way to use them in the proposed strategy, this example focuses on applying the hyperparameter information in both the distance and the cost function.

Firstly, hyperparameters are used when building the subset of n_Q candidate trajectories. Defining x_q^r as the set-point value of candidate q, the distance function in (3.16), is modified adding a term which penalizes candidate trajectories whose references are far from the reference of the problem x_r, that is:

$$dist_q = \|x(k) - x_q(0)\|_\alpha^2 + \alpha_p \Delta_q(k)^2 + \alpha_r \|x_r - x_q^r\|_2^2, \tag{3.19}$$

where $\alpha_r = 10$ weights the deviation between the current reference and the reference of the candidate.

Secondly, the cost function is also modified to take into account the hyperparameters. In the previous example, the stage cost does not consider the particular target of the candidates. This can affect the set-point tracking capabilities of the controller. One possibility is to add a constraint to optimization problem (3.10) to guarantee that the

convex combination of the candidates corresponding references x_q^r is equal to the target reference of the predictive controller x_r, that is

$$\sum_{q=1}^{n_Q} \lambda_q x_q^r = x_r.$$

This constraint aims at taking into account not only where a given trajectory is, but also where is headed. However, adding this constraint may compromise the feasibility of the optimization problem. For this reason, in this example we propose to add it as a soft constraint modifying the objective function. To this end, at each sampling time the historian based controller solves the following optimization problem:

$$\begin{aligned} \min_{\lambda} \quad & \sum_{q=1}^{n_Q} \lambda_q V_q + \rho_r \left\| x_r - \sum_{q=1}^{n_Q} \lambda_q x_q^r \right\|_2^2 \\ s.t. \quad & x(k) = \sum_{q=1}^{n_Q} \lambda_q x_q(0), \\ & \sum_{q=1}^{n_Q} \lambda_q = 1, \\ & \lambda_q \geq 0, \quad \forall q \in \{1, \ldots, n_Q\}. \end{aligned} \tag{3.20}$$

Any standard QP solver can be used as, in our case, *quadprog* in Matlab. Figure 3.3.5 shows the level trajectories of the closed loop simulation of the proposed control strategy using hyperparameters with a weight $\rho_r = 1$. Figure 3.3.6 shows the closed-loop performance of the proposed controller with and without hyperparameters and the cost of the two best relay based controllers. It can be seen that the use of hyperparameters leads to better set-point regulation and to a lower performance cost.

Figure 3.3.7 shows the number of candidates provided by each relay controller along the whole simulation for the proposed controller with and without the use of the information provided by the hyperparameters.

Note that in both cases most candidates trajectories belong to relay based controller 1 and 2 because their set-points are close to the reference, in fact relay based controller 2 has the same reference and it can be seen that when hyperparameters are considered, almost all candidates belong to this controller. Note also that, even if most of the trajectories are of a single relay controller, the proposed strategy achieves a better performance cost.

3.3.3 Example 3

In this subsection, constraints are taken into account, in particular, a maximum average head constraint in demand node four. Hydraulic head is a specific measurement of liquid pressure above a geodetic datum and it relates the energy in an incompressible fluid to the height of an equivalent static column of the fluid. As mentioned before, the historian data includes the average head in this node for all the trajectories, which we denote as z.

There exist a nonlinear relation between head and flow, which we can observe if we consider the head loss Hazen-William formula[115]:

$$z = \gamma u^{1.852}, \tag{3.21}$$

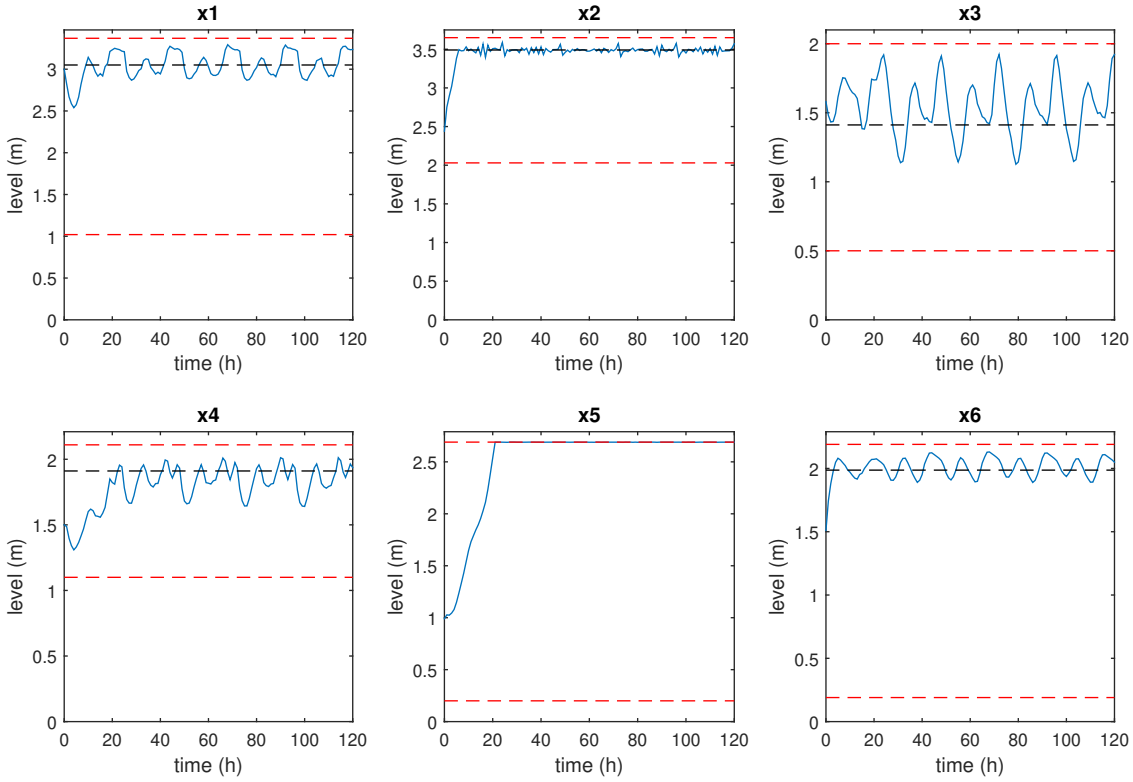


Figure 3.3.5: References (black dash) and closed-loop trajectories (blue solid) of the tank levels with the proposed controller using hyperparameters.

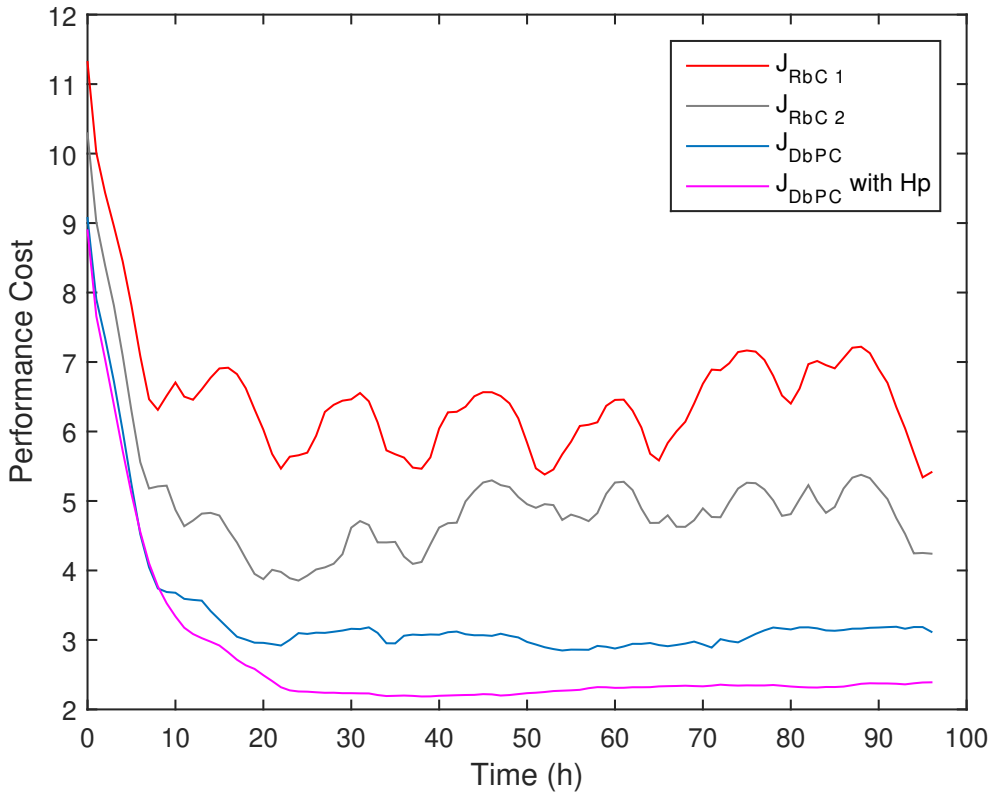


Figure 3.3.6: Performance cost comparison with relay based controllers and historian predictive control with and without hyperparameters.

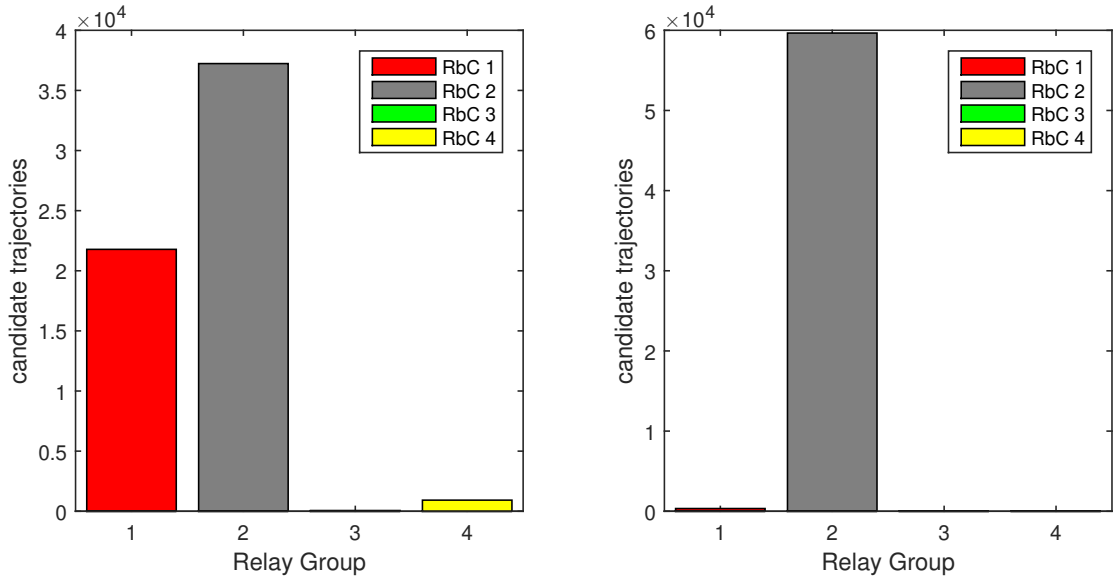


Figure 3.3.7: Comparison of number of candidates provided by each relay based controller along the whole simulation between proposed strategy without hyperparameters (left side) and with hyperparameters (right side).

where γ is a parameter calculated with the length of the pipe, the pipe roughness coefficient and the inside pipe diameter. Despite this nonlinear relation, the estimated average pressure sequence:

$$\hat{z}_{k+i} = \sum_{q=1}^{n_Q} \lambda_q z_q(i), \tag{3.22}$$

provides a good approximation based on local data.

Head constraints are included in the optimization problem as soft constraints (slack variables) instead of hard constraints in order to eliminate feasibility issues caused by this sort of constraints. Note that these feasibility issues are related to the high dimension of the state in relation to the database size in this example. Thus, for lower dimensional systems it could be possible to use hard constraints. The number of slack variables added to the optimization problem is equal to the prediction horizon N to ensure that all the average heads in the estimated average head sequence, denoted as $\hat{z}$ and obtained as the convex combination of the candidate head sequences, satisfy this soft constraints.

Including constraints in z modifies the optimization problem (3.20) in the following form:

$$\begin{aligned}
\min_{\lambda,\tau} \quad & \sum_{q=1}^{n_Q} \lambda_q V_q + \rho_r \left\| x_r - \sum_{q=1}^{n_Q} \lambda_q x_q^r \right\|_2^2 + \rho_z \sum_{i=0}^{N-1} (\tau_i^2 + \tau_i) \\
s.t. \quad & x(k) = \sum_{q=1}^{n_Q} \lambda_q x_q(0), \\
& \sum_{q=1}^{n_Q} \lambda_q = 1, \\
& \lambda_q \geq 0, \qquad \forall q \in \{1, \ldots, n_Q\}, \\
& z_{cons} \geq \left(\sum_{q=1}^{n_Q} \lambda_q z_q(i) \right) - \tau_i, \qquad \forall i \in \{0, \ldots, N-1\}, \\
& \tau_i \geq 0, \qquad \forall i \in \{0, \ldots, N-1\},
\end{aligned} \tag{3.23}$$

where τ is a set of slack variables which lets a small violation of the average head constraints, ρ_z is the weight of the slack variables with respect to the other terms, z_{cons} are the average head constraint values and $z_q(i)$ is the average head of the q-th candidate sequence at instant i[2]. Similar to problem (3.20), a standard QP solver can be used to obtain the solution of (3.23).

The optimization problem solved tracks the same reference in level as in (3.15) and implement the proposed controller with hyperparameters using a number of candidates $n_Q = 500$ and without hyperparameters. The value of the head constraints weight is $\rho_z = 10^5$. For simplicity, head constraint are considered only in one demand node, that is, demand node 4, and the average head constraint values in meters is:

$$z_{cons} = 187.3. \tag{3.24}$$

Figure 3.3.8 shows a comparison between the average head in demand node 4 obtained with and without head average constraints. In case of the proposed controller without constraints, the constraint in average head is clearly violated by the optimal solution.

[2]Notice that with the same slight abuse of the notation used before, we define the head of the q-th candidate trajectory $z_q(i)$ as the i-th row ahead average head after the initial row n_r of the candidate trajectory.

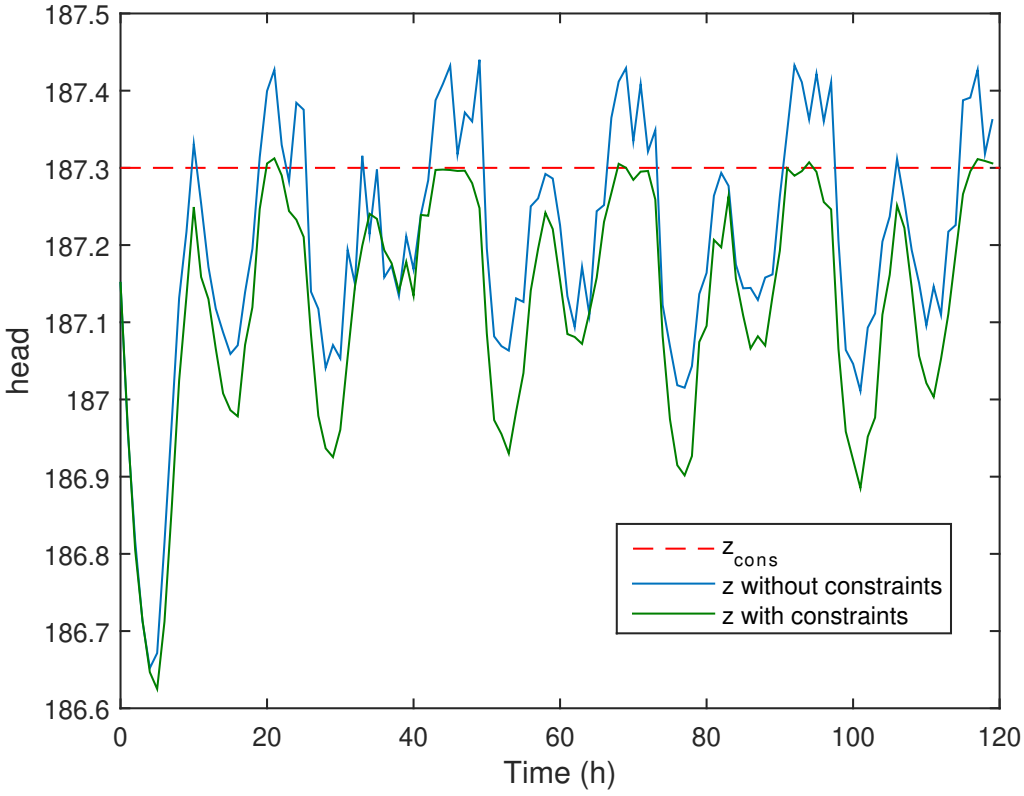


Figure 3.3.8: Comparison between the average head in demand node 4 with and without constraints.

When the controller takes into account this constraint, the optimal solution mostly satisfies it.

Figure 3.3.9 shows the average head estimation signal error for the demand node 4, which is calculated as the difference between the real average head, obtained by simulation, and the estimation of the average head, obtained as in 3.22. We can observe that the error signal has approximately zero mean (0.21%).

3.3.4 Example 4

In this subsection, we consider a set of periodic level reference signals instead of the constant signals used in the previous examples. Typically, water distribution networks are controlled taking into account economic issues such as the tariff pattern related to the electricity price. In the Richmond benchmark, the electricity costs considered had a different tariff during day and night hours. In general, this leads to non-steady level in the tanks, which fill during the night in which pumping water is more economic, and discharge to satisfy the demands during the day. According to this idea and considering the period of the tariff pattern and demand signals, a sinusoidal signal of 24-hours period is taken into account with its maximal at 3 a.m. and its minimal at 15 p.m. in order to build the reference signals for the tank levels. Table 3.4 shows the amplitude and offset for every reference signal considered.

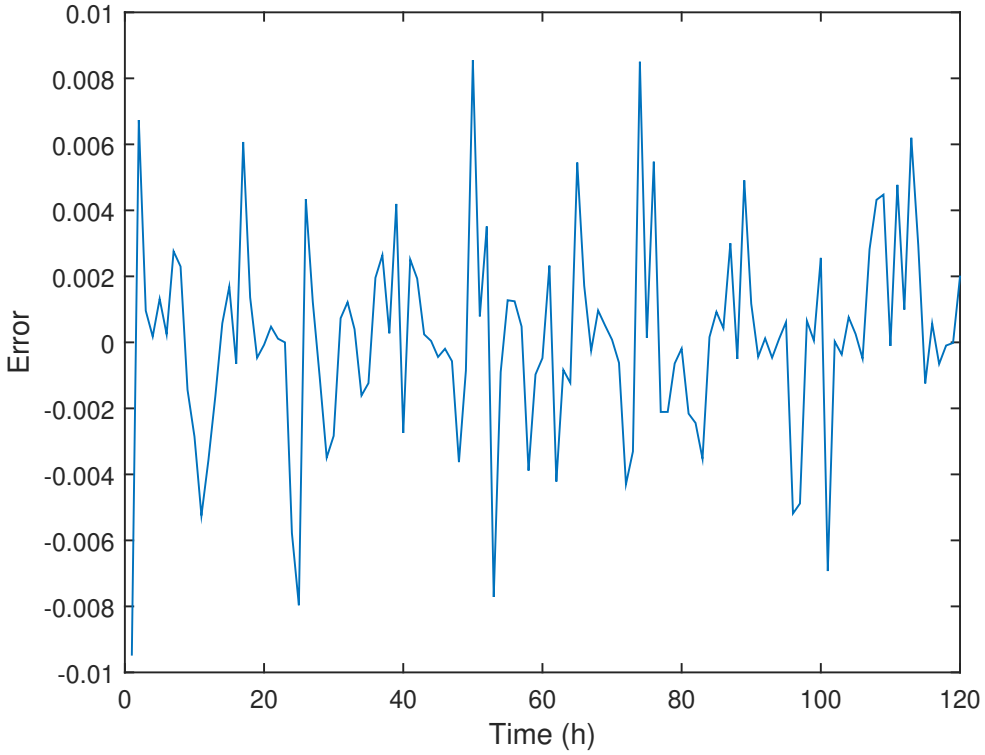


Figure 3.3.9: Difference between the estimated average head and the real average head of demand node 4 .

	Tank 1	Tank 2	Tank 3	Tank 4	Tank 6
Amplitude	1.41	1.07	0.31	0.6	1.58
Offset	2.3453	2.9556	1.2568	1.6108	1.1966

Table 3.4: Amplitude and offset of the sinusoidal reference signal of each controllable tank.

Figure 3.3.10 shows the level references considered and the closed loop trajectories of the proposed controller, together with that of the relay-based controller 4 (which provided the best performance of all the relay based-controllers). This figure shows that the proposed controller provides a quasi-periodic closed-loop trajectory which is almost in phase with the reference, while clearly the relay based trajectory (which is akin to the trajectories of the database) is not correlated. This implies that the proposed strategy does not simply learn (or identify) the control law in the database, but, rather than that, it uses the stored trajectories to fullfill as best as possible the current control objective, which can be different from that used to build the database.

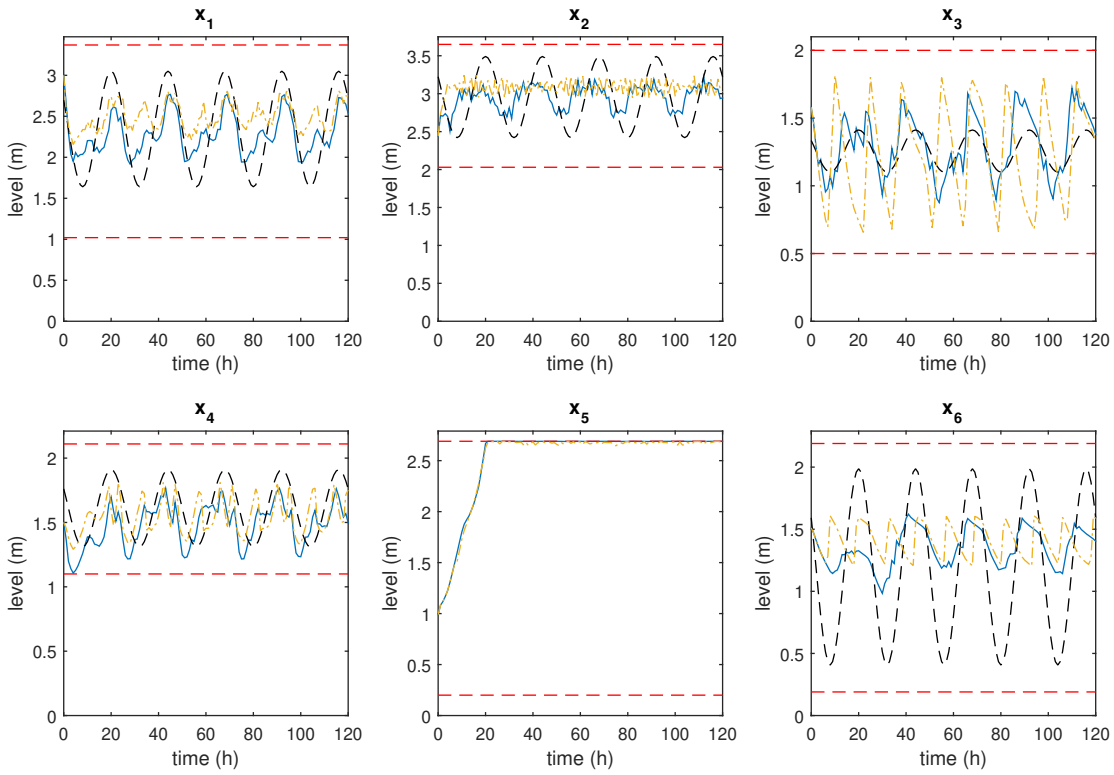


Figure 3.3.10: Periodic reference for every tank (dashed black), closed-loop simulation with the proposed controller (blue) and with the relay-controller 4 (dashed-dotted yellow).

Chapter 4

Offset-free data driven control

The idea of using affine combinations of stored trajectories exposed in chapter 3, resulted in a model free predictive controller which has nice properties when dealing with large scale system like WDN. Nevertheless, the presence of disturbances or noise signals, specially with non zero mean, in real complex systems may lead to some offset issues. These disturbances are included in the trajectories that are stored in the historian process of a noisy system. Since the controller combines noisy trajectories, this approach cannot achieve offset-free tracking, except for the ideal case of a noise and disturbance free database.

Tracking variable set points without steady state error, even in presence of constant disturbances, is one of the most desirable capabilities of a feedback control system. Classical control methods for linear systems achieve this by using integral action controllers, as in the case of the ever popular PI or PID control algorithms. Generally speaking, offset-free control is a well understood problem, although still researched in more ambitious control formulations [63].

In data driven approaches, offset free control is often achieved by following a reference model and assuming an integrator mode [84], controller or set point adaptation [99], reinforcement learning [119] or exploiting the linear dependence in input-output data of linear processes [46].

Following this line of research, in this chapter, a data driven control strategy based on a database of past state trajectories that aim to produce a similar offset free closed loop response in spite of different operating conditions is presented. The control laws used to generate the database trajectories are assumed to be unknown, so that the proposed strategy will learn the underlying unknown control law that obtain similar closed loop responses for different operating conditions. The proposed controller computes the input signal to be applied as an affine combination of the control signals. Zero mean tracking error with minimum variance is achieved under the assumption that the state is measurable, the underlying dynamics are linear and the trajectories of the database share the same error dynamics and are in turn offset free.

With respect to other data driven control approaches, the proposed method does not perform an identification step, avoiding the potential problems that can arise in this phase. Furthermore, being related to the lazy learning techniques ([92]), no training phase is required to learn the underlying control law in the database, thus it possible to include new

data that is made available online. The results are illustrated by means of an application to a well known process control trainer [61, Chap. 4].

4.1 Problem statement

In this chapter we consider a system for which a model of its dynamics is not available, but a particular state representation is known and measurable. This implies that although we propose a model-free approach, some knowledge of the system is needed to define this state. In some cases, the measured state vector corresponds to physical measurements chosen based on first principles; while in others, for those systems in which the state is not completely measurable, the state will be considered composed by present and past values of the system inputs and outputs, following a standard input-output modelling procedure and assuming that an estimation of the order of the system and the delays is known.

Although unknown, we assume that the system is a linear system subject to bounded state and output disturbances, hence, the measured state satisfies the following model

$$x(t+1) = Ax(t) + Bu(t) + w(t) \tag{4.1}$$
$$y(t) = Cx(t) + Du(t) + v(t) \tag{4.2}$$

where t is the discrete time variable, $x(t) \in \mathbb{R}^{n_x}$, $u(t) \in \mathbb{R}^{n_u}$ and $y(t) \in \mathbb{R}^{n_y}$ are the measured state, input and output of the system at time step t respectively, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $C \in \mathbb{R}^{n_y \times n_x}$, and $D \in \mathbb{R}^{n_y \times n_u}$ are the unknown system matrices and $w(t) \in \mathbb{R}^{n_x}$ and $v(t) \in \mathbb{R}^{n_y}$ are unknown state and output disturbances respectively with non zero mean, that is

$$E\{w(t)\} = w^e, \quad E\{v(t)\} = v^e. \tag{4.3}$$

Note that the state and output disturbances include all the discrepancies between measured and real states due to noises and, to some extent, uncertainties and slight non-linearities.

The control objective is to track a reference $r \in \mathbb{R}^{n_y}$ without offset. The stochastic disturbances considered in (4.1) and (4.2) make impossible to reach true offset free control. Thus, by offset free control we mean that y is probabilistically ultimately bounded into a set with mean equal to r [54]. Also, to ensure a well posed control problem we assume the following:

Assumption 4. *The system given by (4.1) and (4.2) is assumed to have full state and output controllability. Furthermore, the reference $r \in \mathbb{R}^{n_y}$ is assumed to be reachable.*

4.1.1 Historian database

In this chapter, instead of using a model to define the controller, we present a procedure to take a decision on behalf of the information stored in a historian database. This historian database has a large number of past offset free state, input, output and reference trajectories. Each trajectory stored in the database, which may be of different length, represents the closed loop behaviour of the system given by (4.1) and (4.2) controlled with a different unknown control law and constant reference. In addition, we assume that the disturbances of each stored trajectory are characterized by a possibly different mean value. Using this framework, the disturbance may account for time-varying and state dependent perturbations.

If the measured state for each trajectory is given by $\tilde{x}$ and its corresponding steady state value is denoted as $\tilde{x}^e$, we assume that all the trajectories in the historian database satisfy the following property:

Assumption 5. *It is assumed that the dynamics of the trajectories of the database satisfy*

$$\tilde{x}(t+1) - \tilde{x}^e = A_c(\tilde{x}(t) - \tilde{x}^e) + \tilde{\tau}(t), \tag{4.4}$$

where A_c is a Schur stable matrix and $\tilde{\tau}(t)$ a zero mean error term with bounded covariance.

The objective of the approach proposed in this chapter is to learn from the historian database the underlying controller defined by Assumption 5, while preserving its offset free property in the presence of different mean perturbations. Note that using a standard function approximation procedure to determine a function that relates state, reference and output would yield a static controller that would not provide offset free constant reference tracking for different mean perturbations.

There are different areas of applicability for the proposed control scheme depending on the origin of the historian database. The trajectories stored can be obtained from real operation (which may include manual operation and different controllers) of the system in the past, or from dedicated tests. In the first case, the controller objective is to learn the underlying control law that has provided good performance in the past, in spite of changing operating conditions. This procedure may be of interest in complex systems for which great amounts of data are available.

In the second case, using closed-loop testing may be a benefit, for example when trying to control a open-loop unstable system, because identification is avoided. In this case, because the transient closed loop response will depend on those of the trajectories stored in the database, it is important to store trajectories that exhibit a good control performance so that the controller inherits it. The design criteria can be anything from a performance cost to transient response measures (e.g., overshoot) that can be used to characterize the good trajectories. In some sense, tuning is carried out using extensive off-line experimentation, which depending on the application, may or may be not possible or be less appropriate than using standard identification modelling techniques. This procedure is shown in the temperature control application example.

Besides closed-loop testing and real past behaviours, it is also possible to include open-loop tests for stable systems in which for a predefined input sequence, obtained for example from step tests with filtering or lead/lag, the reference is defined for the steady state reached. In addition, these results can be combined with trajectories obtained from past operation or closed-loop testing.

Remark 5. *The requirement that all the trajectories of the database satisfy (4.4) could seem very restrictive but it is consistent with the standard control practice in which the process is desired to have the same performance in spite of the different operating conditions. For a real process this implies that different controllers used to generate the historian database were characterized by a similar closed-loop dynamic behaviour (for example, similar rise time and overshoot). Note also that the zero mean error term is not bounded, which provides a certain degree of robustness for closed-loop trajectories with slightly different error dynamics. Furthermore, the bounded covariance implies that the probability of getting high errors is small.*

4.1.2 Building the candidate set

The information stored in the historian database is used to generate a set of possible candidate tuples $j \in S$. Each tuple consists of the inputs, states, outputs and corresponding reference of a particular trajectory and sampling time of the historian database. These tuples will be used to define the optimization problem that has to be solved at each sampling time to implement the proposed controller and is built off-line using all the trajectories available in the database excluding those that are the first stored element of a particular trajectory. The reason of this is that, in the proposed strategy, for each candidate tuple, it is necessary to know the state and input value of the previous sample time in its corresponding trajectory.

For each candidate, we denote as $x_j(0), u_j(0), y_j(0), w_j(0)$ and $v_j(0)$ the state, input, output and disturbances at the corresponding sample time of the candidate, and $x_j(k)$, $u_j(k)$, $y_j(k)$, $w_j(k)$ and $v_j(k)$ the corresponding values shifted k sample times (i.e. $x_j(-1)$ denotes the state at the sample time before the candidate's corresponding sample time). Moreover, for each candidate, we denote as r_j the corresponding reference of its trajectory (note that the reference does not depend on the time because it is assumed constant for each trajectory). The database does not store the values of the disturbances, and hence the variables $w_j(\cdot)$ and $v_j(\cdot)$ are unknown for the controller. Although the disturbances are not stored in the database, their effects are indirectly stored by means of the state and output measurements.

Sampling time index variable t refers to the sampling time of a real trajectory, either from the historian data base or online implementation, in particular, in the controller implementation $x(t)$ refers to the current state measurement. On the other hand, sampling time index variable k refers to displacement relative to a candidate of the set S. Note that the only values of k needed to define the proposed controller are $k = 0$ and $k = -1$. In the proof of the main theorem, $k = 1$ is also used.

To clarify how to define the candidates from the trajectories stored in the historian database, consider figure 4.1.1, which shows an example of a one dimensional state trajectory with 5 sampled values, $\tilde{x}(1)$ to $\tilde{x}(5)$ that reach its corresponding target state $\tilde{x}^e$. From this trajectory, four different state candidates can be defined, one for each state in the trajectory that has a predecessor. In the figure, the possible candidates are denoted as $x_j(0)$ with $j \in a, b, c, d, e$. In this particular case, $x_a(0)$ cannot be included in S because $x_a(-1)$ does not exists. The value of each candidate is defined by a different sample time; that is

$$\begin{aligned} x_a(0) &= \tilde{x}(1) \\ x_b(0) &= \tilde{x}(2) \\ x_c(0) &= \tilde{x}(3) \\ x_d(0) &= \tilde{x}(4) \\ x_e(0) &= \tilde{x}(5). \end{aligned} \tag{4.5}$$

The corresponding previous state of each candidate are also defined by the states in this trajectory; that is

$$\begin{aligned} x_a(-1) &= NA \\ x_b(-1) &= \tilde{x}(1) \\ x_c(-1) &= \tilde{x}(2) \\ x_d(-1) &= \tilde{x}(3) \\ x_e(-1) &= \tilde{x}(4). \end{aligned} \tag{4.6}$$

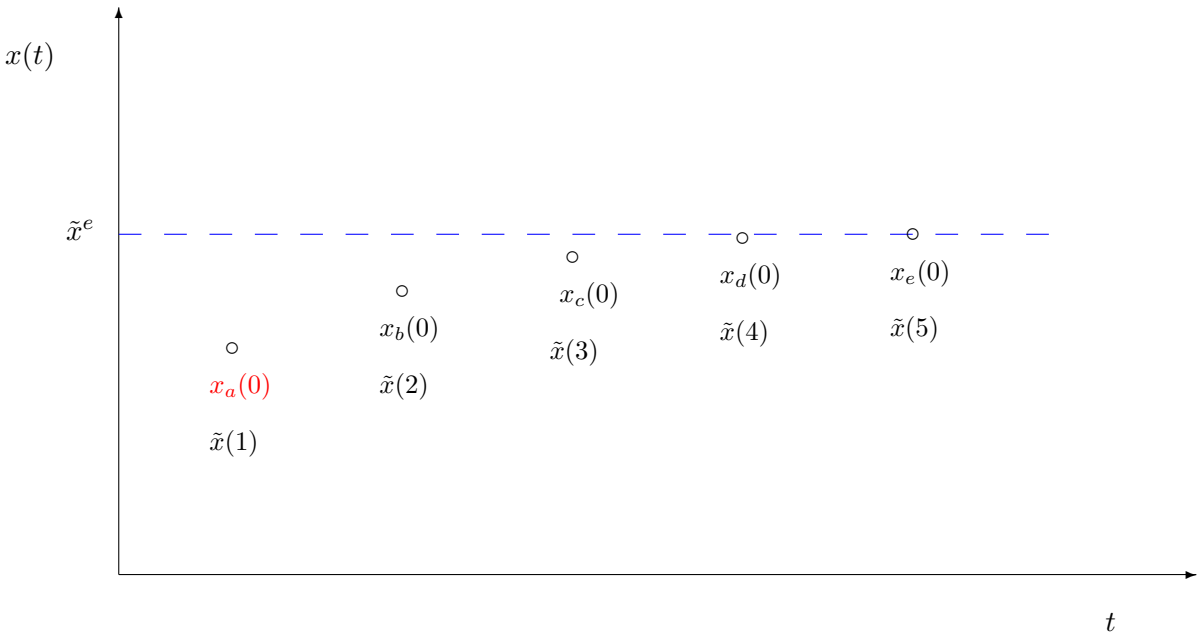


Figure 4.1.1: Example of the notation employed to describe the candidates that conform the set S.

With this notation, the candidates satisfy

$$\begin{array}{rcl} x_j(k+1) & = & Ax_j(k) + Bu_j(k) + w_j(k) \\ y_j(k) & = & Cx_j(k) + Du_j(k) + v_j(k). \end{array} \tag{4.7}$$

For each candidate, we denote its corresponding reference as r_j which has a constant value for the whole trajectory, thus shared with the other candidates that are from the same trajectory. In addition, for each candidate, we denote the constant mean disturbance values of its corresponding trajectory as w_j^e and v_j^e; that is

$$E\{w_j\} = w_j^e, \quad E\{v_j\} = v_j^e. \tag{4.8}$$

Taking into account that all the trajectories in the database satisfy Assumption 5, the candidates also satisfy:

$$x_j(k+1) - x_j^e = A_c(x_j(k) - x_j^e) + \tau_j(k). \tag{4.9}$$

Assumption 5 and (4.9) implies a direct consequence for all the candidates trajectories. Given $0 < p_j \leq 1$ and $\Omega_j \subset \mathbb{R}^{n_x+n_u}$, there exists N_j such that for its corresponding future state and input trajectories satisfy

$$Pr\left[x_j(k) - x_j^e \in \Omega_j\right] \geq p_j, \ \ \forall k \geq N_j \tag{4.10}$$

where Ω_j is a probabilistic ultimate bound ([54]) which is a neighbourhood around the steady state x_j^e whose size is related to the covariance of $\tau_j(k)$. Notice that N_j depends on the initial state of the candidate trajectory $x_j(0)$. We also remark that if the disturbances $\tau_j(k)$ are bounded, it is possible to find a deterministic ultimate bound, that is a Ω_j which satisfies (4.10) with probability $p_j = 1$ [15, Chap. 4].

Remark 6. *The size of the candidates set influences the learning capabilities of the proposed approach and it is also directly related to the computational burden. On the other hand, as in other data based and learning approaches, the dimension of the state vector (and other variables) affects the necessary size of the set S. From a practical point of view, this leads to use the minimum dimension state representation necessary for the control objectives considered.*

4.2 Steady state characterization

Although the proposed controller is based on state feedback, we have considered an output reference tracking problem. In this section, we consider the notion of steady state for the output equal to a given reference r. This characterization will be used on the proof of the main result of this chapter.

The pair (x^e, u^e), with $x^e \in \mathbb{R}^{n_x}$, $u^e \in \mathbb{R}^{n_u}$, represents a steady state and steady control input if and only if

$$\begin{array}{rcl} x^e & = & Ax^e + Bu^e + w^e \\ r & = & Cx^e + Du^e + v^e. \end{array} \tag{4.11}$$

We assume that each of the trajectories of the database has been generated to track a particular reference and moreover, we assume that the control input trajectory drives the output to this reference in spite of the disturbances. Thus, similarly to (4.11), a pair (x_j^e, u_j^e), with $x_j^e \in \mathbb{R}^{n_x}$, $u_j^e \in \mathbb{R}^{n_u}$, represents a steady state and steady control input for the trajectory of the j-th candidate if and only if

$$\begin{array}{rcl} x_j^e & = & Ax_j^e + Bu_j^e + w_j^e, \\ r_j & = & Cx_j^e + Du_j^e + v_j^e, \end{array} \tag{4.12}$$

The following assumption is a necessary controllability condition and it is required to ensure that a steady state pair (x_j^e, u_j^e) exists for a given reference r.

Assumption 6. *Let G be defined as:*

$$G = \begin{bmatrix} \mathrm{I} - \mathrm{A} & -B \\ C & D \end{bmatrix}. \tag{4.13}$$

It is assumed that G has full column rank.

It is possible to provide a characterization of the pair (x_j^e, u_j^e), through theorems 2 and 3 given in the following.

Theorem 2. *Suppose that assumption 6 holds, and that G is a square matrix. Then, the pair (x_j^e, u_j^e) is uniquely given by a linear expression of $r_j - v_j^e$ and w_j^e.*

Proof. The equality constraints (4.12) can be rewritten as

$$\begin{bmatrix} \mathrm{I} - \mathrm{A} & -B \\ C & D \end{bmatrix} \begin{bmatrix} x_j^e \\ u_j^e \end{bmatrix} = \begin{bmatrix} w_j^e \\ r_j - v_j^e \end{bmatrix}. \tag{4.14}$$

Since it is assumed that G has full column rank, the previous system of equations has a unique solution equal to

$$\begin{bmatrix} x_j^e \\ u_j^e \end{bmatrix} = G^{-1} \begin{bmatrix} w_j^e \\ r_j - v_j^e \end{bmatrix}, \tag{4.15}$$

and this completes the proof. □

Theorem 3. *Suppose that assumption 6 holds, and that the number of columns of G is larger than the number of rows, and that the steady state pair (x_j^e, u_j^e) is defined as the solution of*

$$\begin{aligned} \min_{x_j^e, u_j^e} \quad & \frac{1}{2}\|x_j^e\|_Q^2 + \frac{1}{2}\|u_j^e\|_R^2 \\ s.t. \quad & \\ & G \begin{bmatrix} x_j^e \\ u_j^e \end{bmatrix} = b_j^e, \end{aligned} \tag{4.16}$$

where $b_j^e = \begin{bmatrix} w_j^e \\ r_j - v_j^e \end{bmatrix}$, $Q > 0$ and $R > 0$. Then the pair (x_j^e, u_j^e) is uniquely given by a linear expression of $r_j - v_j^e$ and w_j^e.

Proof. Let z_j^e be

$$z_j^e = \begin{bmatrix} x_j^e \\ u_j^e \end{bmatrix}, \tag{4.17}$$

and $\boldsymbol{0}_{a,b}$ the zero matrix with a rows and b columns. It is well known ([18, Chap. 10]) that if the block diagonal matrix

$$\Upsilon = \begin{bmatrix} Q & \boldsymbol{0}_{n_x,n_u} \\ \boldsymbol{0}_{n_u,n_x} & R \end{bmatrix} \tag{4.18}$$

is strictly definite positive and G has full rank, then the optimal value for z_j^e, defined as in (4.17) and solution to (4.16), is given by the following system of equations

$$\begin{bmatrix} \Upsilon & G^\top \\ G & \boldsymbol{0}_{n_G,n_G} \end{bmatrix} \begin{bmatrix} z_j^e \\ \beta_j^e \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_{n_z,1} \\ b_j^e \end{bmatrix}, \tag{4.19}$$

where $n_z = n_x + n_u$ is Υ dimension, $n_G < n_z$ is the number of rows of G and β_j^e denotes the optimal value for the dual variables of the optimization problem. From

$$\Upsilon z_j^e + G^\top \beta_j^e = \boldsymbol{0}_{n_z,1} \tag{4.20}$$

we obtain

$$z_j^e = -\Upsilon^{-1} G^\top \beta_j^e. \tag{4.21}$$

From (4.19) we also have

$$G z_j^e = b_j^e. \tag{4.22}$$

Substituting (4.21) in (4.22), we obtain the optimal value for β_j^e

$$-G\Upsilon^{-1}G^\top \beta_j^e = b_j^e$$

$$\beta_j^e = -(G\Upsilon^{-1}G^\top)^{-1}b_j^e. \tag{4.23}$$

From (4.21) and (4.23) results

$$z_j^e = \Upsilon^{-1}G^\top(G\Upsilon^{-1}G^\top)^{-1}b_j^e. \tag{4.24}$$

We now denote

$$M_\Upsilon = \Upsilon^{-1}G^\top(G\Upsilon^{-1}G^\top)^{-1}, \tag{4.25}$$

to get

$$z_j^e = M_\Upsilon b_j^e. \tag{4.26}$$

Thus, it is shown that there is a linear relationship between vector b_j^e and vector z_j^e. □

4.3 Offset-free data driven control

In this chapter, we propose to use a control law derived from the control signals in the candidates set S. The control signal to be applied at time t will be computed as a weighted sum of the initial control signals of every candidate of S, that is

$$u(t) = \sum_{j\in S} \lambda_j u_j(0). \tag{4.27}$$

The following sections discuss the conditions that $\{\lambda_j\}$ must meet in order to recover the underlying closed-loop properties of the trajectories stored in the historian database. Offset free with minimum variance must be achieved for all possible references r and mean disturbance values w^e, v^e, not only those included in the candidate's data.

At a first approximation, suppose that we compute $\{\lambda_j\}$ so that the current state, output and reference are a combination of the candidates as in chapter 3, that is

$$x(t) = \sum_{j\in S} \lambda_j x_j(0), \tag{4.28}$$

$$1 = \sum_{j\in S} \lambda_j. \tag{4.29}$$

$$y(t) = \sum_{j\in S} \lambda_j y_j(0), \tag{4.30}$$

$$r = \sum_{j\in S} \lambda_j r_j. \tag{4.31}$$

Taking into account (4.27) and (4.28) in the state equation (4.1),

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + w(t) \\ &= A\sum_{j\in S} \lambda_j x_j(0) + B\sum_{j\in S} \lambda_j u_j(0) + w(t) \\ &= \sum_{j\in S} \lambda_j \left(Ax_j(0) + Bu_j(0)\right) + w(t) \\ &= \sum_{j\in S} \lambda_j x_j(1) + \sum_{j\in S} \lambda_j (w(t) - w_j(0)). \end{aligned} \tag{4.32}$$

We notice that approximating the next state $x(t+1)$ as a combination of the next states of the candidates yields an estimation error given by

$$\sum_{j \in S} \lambda_j (w(t) - w_j(0)), \tag{4.33}$$

which is not guaranteed to have zero mean. This means that if we use a control strategy similar to that exposed in chapter 3, the state in $t+1$ will be a combination of the states of every trajectory in S, but it will have an offset caused by the approximation error even if the trajectories in the database are offset free.

To obtain an offset free control, additional constraints have to be imposed on $\{\lambda_j\}$ in order to force the controller to have memory, in the sense that the previous state and applied control signal have to be related with the previous ones of every trajectory in S through the values of $\{\lambda_j\}$. The constraints

$$x(t-1) = \sum_{j \in S} \lambda_j x_j(-1), \tag{4.34}$$

$$u(t-1) = \sum_{j \in S} \lambda_j u_j(-1), \tag{4.35}$$

take this issue into account. The next properties demonstrate that the estimation error of the next state obtained using a set of weights that satisfy the above mentioned constraints has a zero mean error term. This property will be used to prove offset free tracking.

Property 3. *Assuming that (4.27), (4.28), (4.29), (4.34) and (4.35) hold; then*

$$x(t+1) = \sum_{j \in S} \lambda_j x_j(1) + e_x(t), \tag{4.36}$$

where $e_x(t)$ is a zero mean error term.

Proof. From the state equation (4.1) shifted backwards and the constraints (4.28), (4.34) and (4.35),

$$\begin{aligned} w(t-1) &= x(t) - Ax(t-1) - Bu(t-1) \\ &= \sum_{j \in S} \lambda_j x_j(0) - A \sum_{j \in S} \lambda_j x_j(-1) - B \sum_{j \in S} \lambda_j u_j(-1) \\ &= \sum_{j \in S} \lambda_j \left(x_j(0) - Ax_j(-1) - Bu_j(-1) \right) \\ &= \sum_{j \in S} \lambda_j w_j(-1). \end{aligned} \tag{4.37}$$

From (4.27) and (4.28), as mentioned before, we obtain the estimated value of the next time step

$$\begin{aligned} x(t+1) &= \sum_{j \in S} \lambda_j x_j(1) + \sum_{j \in S} \lambda_j (w(t) - w_j(0)) \\ &= \sum_{j \in S} \lambda_j x_j(1) + w(t) - \sum_{j \in S} \lambda_j w_j(0). \end{aligned} \tag{4.38}$$

Defining the prediction error as

$$e_x(t) = x(t+1) - \sum_{j \in S} \lambda_j x_j(1), \tag{4.39}$$

and substituting in (4.38), we have

$$e_x(t) = w(t) - \sum_{j \in S} \lambda_j w_j(0). \tag{4.40}$$

From (4.37) we obtain

$$-w(t-1) + \sum_{j \in S} \lambda_j w_j(-1) = 0. \tag{4.41}$$

Adding this equality to (4.40) yields

$$\begin{aligned} e_x(t) &= w(t) - w(t-1) - \sum_{j \in S} \lambda_j (w_j(0) - w_j(-1)) \\ &= \Delta w(t) - \Delta w(t-1) - \sum_{j \in S} \lambda_j (\Delta w_j(0) - \Delta w_j(-1)), \end{aligned} \tag{4.42}$$

where $\Delta w(\cdot) = w(\cdot) - w^e$ and $\Delta w_j(\cdot) = w_j(\cdot) - w_j^e$. Notice that $e_x(t)$ is a zero mean error term because, by construction, $\Delta w_j(\cdot)$ and $\Delta w(\cdot)$ have zero mean. □

The weights $\{\lambda_j\}$ obtained can be used not only to obtain an estimation of the future state, but also to estimate the mean value of the current perturbations with zero mean error. This property is proved next.

Property 4. *Assuming that (4.27), (4.28), (4.29), (4.30), (4.34) and (4.35) hold; then*

$$w^e = \sum_{j \in S} \lambda_j w_j^e + e_w(t), \tag{4.43}$$

$$v^e = \sum_{j \in S} \lambda_j v_j^e + e_v(t), \tag{4.44}$$

where $e_w(t)$ and $e_v(t)$ are zero mean error terms.

Proof.

$$\begin{aligned} x(t) &= Ax(t-1) + Bu(t-1) + w(t) \\ &= Ax(t-1) + Bu(t-1) + w^e + \Delta w(t). \end{aligned} \tag{4.45}$$

Thus,

$$\begin{aligned} w^e &= x(t) - Ax(t-1) - Bu(t-1) - \Delta w(t) \\ &= \sum_{j\in S} \lambda_j (x_j(0) - Ax_j(-1) - Bu_j(-1)) - \Delta w(t) \\ &= \sum_{j\in S} \lambda_j w_j(-1) - \Delta w(t) \\ &= \sum_{j\in S} \lambda_j (w_j^e + \Delta w_j(-1)) - \Delta w(t) \\ &= \sum_{j\in S} \lambda_j w_j^e + \sum_{j\in S} \lambda_j \Delta w_j(-1) - \Delta w(t) \\ &= \sum_{j\in S} \lambda_j w_j^e + e_w(t), \end{aligned} \tag{4.46}$$

where $e_w(t)$ is a zero mean error term. In a similar way

$$\begin{aligned} y(t) &= Cx(t) + Du(t) + v(t) \\ &= Cx(t) + Du(t) + v^e + \Delta v(t). \end{aligned} \tag{4.47}$$

Thus,

$$\begin{aligned} v^e &= y(t) - Cx(t) - Du(t) - \Delta v(t) \\ &= \sum_{j\in S} \lambda_j (y_j(0) - Cx_j(0) - Du_j(0)) - \Delta v(t) \\ &= \sum_{j\in S} \lambda_j v_j(0) - \Delta v(t) \\ &= \sum_{j\in S} \lambda_j (v_j^e + \Delta v_j(0)) - \Delta v(t) \\ &= \sum_{j\in S} \lambda_j v_j^e + \sum_{j\in S} \lambda_j \Delta v_j(0) - \Delta v(t) \\ &= \sum_{j\in S} \lambda_j v_j^e + e_v(t), \end{aligned} \tag{4.48}$$

where $e_v(t)$ is a zero mean error term. □

The control objective is to drive the output to the reference r. Taking into account Assumption 5, this implies that, for each candidate, the state and the input have to reach the corresponding steady state values x_j^e, u_j^e. In the next result, it is proved that the evolution of the deviation of the state from its target state can be estimated from the weighted evolution of the deviation of each of the candidate trajectories from its corresponding target states with zero mean error. This relation will be used to prove convergence and zero mean tracking error.

Theorem 4. *Suppose that*

$$\begin{bmatrix} x^e \\ u^e \end{bmatrix} = M \begin{bmatrix} w^e \\ r - v^e \end{bmatrix}, \tag{4.49}$$

$$\begin{bmatrix} x_j^e \\ u_j^e \end{bmatrix} = M \begin{bmatrix} w_j^e \\ r_j - v_j^e \end{bmatrix}, \tag{4.50}$$

and that (4.27), (4.28), (4.29), (4.30) (4.31), (4.34) and (4.35) holds; then

$$x(t+1) - x^e = \sum_{j \in S} \lambda_j (x_j(1) - x_j^e) + \eta(t), \tag{4.51}$$

where $\eta(t)$ is a zero mean error term.

Proof. From property 4 it holds that

$$\begin{aligned}
\begin{bmatrix} x^e \\ u^e \end{bmatrix} &= M \begin{bmatrix} w^e \\ r - v^e \end{bmatrix} \\
&= M \begin{bmatrix} \sum\limits_{j \in S} \lambda_j w_j^e + e_w(t) \\ \sum\limits_{j \in S} \lambda_j r_j - \sum\limits_{j \in S} \lambda_j v_j^e - e_v(t) \end{bmatrix} \\
&= \sum_{j \in S} \lambda_j M \begin{bmatrix} w_j^e \\ r_j - v_j^e \end{bmatrix} + M \begin{bmatrix} e_w(t) \\ -e_v(t) \end{bmatrix} \\
&= \sum_{j \in S} \lambda_j \begin{bmatrix} x_j^e \\ u_j^e \end{bmatrix} + M \begin{bmatrix} e_w(t) \\ -e_v(t) \end{bmatrix}.
\end{aligned} \tag{4.52}$$

From the previous equation,

$$x^e = \sum_{j \in S} \lambda_j x_j^e + e_s(t), \tag{4.53}$$

where $e_s(t)$ is a zero mean error term. From property 3 and subtracting (4.53) to (4.36) we finally obtain

$$\begin{aligned}
x(t+1) - x^e &= \sum_{j \in S} \lambda_j (x_j(1) - x_j^e) + e_x(t) - e_s(t) \\
&= \sum_{j \in S} \lambda_j (x_j(1) - x_j^e) + \eta(t),
\end{aligned} \tag{4.54}$$

where $\eta(t) = e_x(t) - e_s(t)$. □

Theorem 4 and the stability of the error dynamics of all the candidates[1] will be used in the following to prove the main result of the chapter. Since the disturbances $w(t)$ and $v(t)$ are random variables with non zero mean, offset free tracking will be attained if we can prove that the closed loop trajectory converges to a neighbourhood of x^e.

Theorem 5. *Under the assumptions of Theorem 4, there exist $\gamma \in (0,1)$ such that*

$$\|x(t+1) - x^e\|_P \leq \|\psi(t)\|_P + \sqrt{\gamma} \|x(t) - x^e\|_P, \tag{4.55}$$

where $\psi(t)$ is a zero mean error term, which implies that the state converges to a neighbourhood of x^e.

[1] This is a direct consequence of assumption 5.

Proof. Under the assumptions of Theorem 4 we have that

$$x(t+1) - x^e = \sum_{j \in S} \lambda_j (x_j(1) - x_j^e) + \eta(t). \tag{4.56}$$

From Assumption 5 we also have

$$x_j(k+1) - x_j^e = A_c(x_j(k) - x_j^e) + \tau_j(k), \tag{4.57}$$

where A_c is Schur stable and $\tau_j(t)$ has zero mean. Therefore, each trajectory satisfies

$$x_j(1) - x_j^e = A_c(x_j(0) - x_j^e) + \tau_j(0). \tag{4.58}$$

Substituting (4.58) into equation (4.56):

$$x(t+1) - x^e = \eta(t) + \sum_{j \in S} \lambda_j (A_c(x_j(0) - x_j^e) + \tau_j(0)), \tag{4.59}$$

We now denote $\xi(t)$ the aggregation of all the error terms. That is,

$$\xi(t) = \eta(t) + \sum_{j \in S} \lambda_j \tau_j(0). \tag{4.60}$$

With this notation,

$$x(t+1) - x^e = \xi(t) + A_c \sum_{j \in S} \lambda_j (x_j(0) - x_j^e). \tag{4.61}$$

Since $x(t) = \sum\limits_{j \in S} \lambda_j x_j(0)$ and taking into account (4.53), we obtain

$$\begin{aligned} x(t+1) - x^e &= \xi(t) + A_c \sum_{j \in S} \lambda_j (x_j(0) - x_j^e) \\ &= \xi(t) + A_c(x(t) - x^e) + A_c e_s(t). \end{aligned} \tag{4.62}$$

Aggregating again the error terms in $\psi(t) = \xi(t) + A_c e_s(t)$ we have

$$x(t+1) - x^e = \psi(t) + A_c(x(t) - x^e), \tag{4.63}$$

where $\psi(t)$ is a zero mean error term. This is enough to ensure the existence of a probabilistic ultimate bound set ([54]). Consider now the weighted norm $\|x(t+1) - x^e\|_P$. The triangle inequality yields

$$\begin{aligned} \|x(t+1) - x^e\|_P &= \|\psi(t) + A_c(x(t) - x^e)\|_P \\ &\leq \|\psi(t)\|_P + \|A_c(x(t) - x^e)\|_P. \end{aligned} \tag{4.64}$$

Since A_c is assumed to be Schur stable, there is $P > 0$ and $\gamma \in (0, 1)$ such that

$$A_c^\top P A_c < \gamma P, \tag{4.65}$$

Taking into account this into (4.64)

$$\|x(t+1) - x^e\|_P \leq \|\psi(t)\|_P + \sqrt{\gamma}\|x(t) - x^e\|_P. \tag{4.66}$$

This means that the trajectory converges to a neighbourhood of x^e. □

Notice that the distance to the desired steady state x^e decreases at each sample time provided that

$$\|\psi(t)\|_P < (1 - \sqrt{\gamma})\|x(t) - x^e\|_P. \tag{4.67}$$

From here we infer that the size of the set in which $x(t)$ is ultimately bounded can be characterized by an upper bound on $\|\psi(t)\|_P$.

4.4 Controller formulation

In this section, a general formulation for the proposed strategy and an implementation procedure (see Algorithm 2) are presented. Furthermore, we focus on some details of the algorithm that provide a simplification in its implementation and a relaxation on some theoretical assumptions made in sections 4.2 and 4.3.

The objective of the proposed controller is to minimize at each sampling time the the variance of the tracking error which following the results of the previous section can be defined as

$$\sum_{j\in S} \lambda_j^2 \mathrm{E}\{\|e_j(0)\|^2\}. \tag{4.68}$$

where the $e_j(0)$ represents the part of the error term $\psi(t)$ related to the tracking error in the stored trajectory j. It can be difficult to obtain the expectation $\mathrm{E}\{\|e_j(0)\|^2\}$, but if we assume an upper bound

$$\mathrm{E}\{\|e_j(0)\|^2\} \leq \sigma_j, \ \ \forall j \in S, \tag{4.69}$$

then the optimization problem to solve is to minimize

$$\sum_{j\in S} \sigma_j^2 \lambda_j^2, \tag{4.70}$$

subject to the equality constraints presented in the previous section. Furthermore, if it is considered that the values for σ_j are all equal to an unknown value σ, the optimization problem can be rewritten as

$$\lambda_j^*(t) = \arg\min_{\lambda_j} \sum_{j\in S} \lambda_j^2 \tag{4.71a}$$

$$s.t. \sum_{j\in S} \lambda_j x_j(0) = x(t) \tag{4.71b}$$

$$\sum_{j\in S} \lambda_j x_j(-1) = x(t-1) \tag{4.71c}$$

$$\sum_{j\in S} \lambda_j u_j(-1) = u(t-1) \tag{4.71d}$$

$$\sum_{j\in S} \lambda_j y_j(0) = y(t) \tag{4.71e}$$

$$\sum_{j\in S} \lambda_j r_j = r \tag{4.71f}$$

$$\sum_{j\in S} \lambda_j = 1 \tag{4.71g}$$

Using the solution obtained with the previous problem, the control signal to be applied is

$$u(t) = \sum_{j\in S} \lambda_j^*(t) u_j(0) \tag{4.72}$$

where $\lambda_j^*(t)$ are obtained every sampling time from the solution of (4.71a) taking into account the current values of $x(t)$, $x(t-1)$, $u(t-1)$ and $y(t)$. This control law can be obtained using a simple explicit equation and will result in an offset free tracking trajectory as shown in the previous section.

4.4.1 Reducing the candidate set

Given a database, a set S of candidate trajectories that includes all the information available can be obtained. In general, the cardinality of this set can be very high if the database is large (note that for each trajectory a number of candidates can be obtained with different initial states along such trajectory). In this section, we propose to use, at each sampling time t, not all the candidates available, but a reduced subset denoted $\hat{S}(t)$. In particular, we propose to use only the n_c candidates closer, in a sense, to the current state of the system. The cardinality of $\hat{S}(t)$ becomes a tuning parameter that provides a trade-off between the amount of information used, the computational burden and the estimation error due to nonlinearities. In practice, there are several reasons that justify using *local* information including the high computational burden with a large database and the low information value between trajectories or data repetition. In addition, using local information reduces the estimation error produced when the proposed approach is applied to a nonlinear system. Using local information is akin to carrying out a linearisation in the current state [20].

In order to reduce the candidate set, a selection criteria has to be specified. We propose to use a distance function that evaluates the trajectories stored in the database with respect the current situation taking into account not only the current state, but also the current output, the reference and the past state and input; that is, all the information used to define the optimization problem of the proposed controller. This information is condensed in the following vectors

$$z(t) = \begin{bmatrix} r(t) \\ y(t) \\ x(t) \\ x(t-1) \\ u(t-1) \end{bmatrix} \quad z_j = \begin{bmatrix} r_j \\ y_j(0) \\ x_j(0) \\ x_j(-1) \\ u_j(-1) \end{bmatrix} \tag{4.73}$$

with $j \in S$. At each sampling time, the n_c candidates from S that yield the lowest value of a given weighted distance function are selected. This distance function can be defined as

$$d_f(z(t), z_j) = \|z(t) - z_j\|_\alpha, \tag{4.74}$$

where α is the weight matrix that is tuned to normalize and prioritize each entry of the deviation vector.

Algorithm 2. *Reducing the candidates set implies that at each sampling time, the following procedure has to be implemented*

1. *Build $z(t)$ as in (4.73).*
2. *For each candidate $j \in S$ compute a suitable distance function $d(z(t), z_j)$ (e.g., like (4.74)).*
3. *Build $\hat{S}$ from the n_c closest candidates.*
4. *Solve problem (4.71) using the constraints defined by $\hat{S}$.*
5. *Compute and apply the control signal $u(t)$ using (4.72) with $\hat{S}$.*

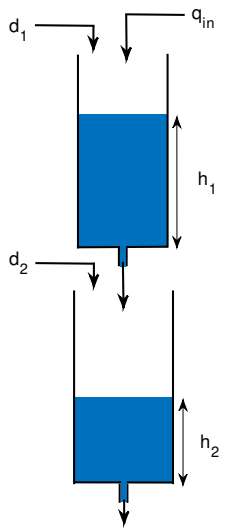


Figure 4.5.1: Two-tanks System

Problem (4.71) can be posed as a quadratic programming problem subject to equality constraints:

$$\begin{aligned} \lambda^*(t) = \arg \quad & \min_{\lambda} \quad \lambda^\top \lambda \\ & \text{s.t.} \quad H\lambda = b, \end{aligned} \tag{4.75}$$

where λ is a vector that includes the n_c weights, matrix $H \in \mathbb{R}^{n_h \times n_c}$ depends on the reduced candidate set $\hat{S}$ and vector $b \in \mathbb{R}^{n_h}$ depends on the current sample time data $z(t)$. The number of equality constraints is $n_h = 2 \cdot n_x + n_u + 2 \cdot n_y + 1$, which in general is much lower than the number of candidates of the reduced set n_c (see subsection 10.1.1. in [18]). The solution to this optimization problem is well known and given by

$$\lambda^*(t) = H^\top (HH^\top)^{-1} b. \tag{4.76}$$

Note that this solution has to be calculated at each sampling time, because matrix H changes with the candidates selected and vector b depends on the current measurements. Note however that the most time consuming calculation is the inversion of matrix $HH^\top$, whose dimension is n_h. This implies that the proposed procedure avoids the use of iterative optimization algorithms and can be implemented on a wide range of applications. In the next section we apply this procedure to a scaled laboratory process with fast dynamics.

4.5 A simulated example with two-tank system

In this section a simulated example will be used to illustrate the proposed strategy. Two tanks system with gravity discharge is considered, which is shown in Figure 4.5.1. The input flow is $q_{in}(t)$ (measured in m^3/s) and the water levels in each tank are $h_1(t)$ and $h_2(t)$ respectively (measured in meters). The continuous time dynamics of the system are assumed to be given by the following nonlinear differential equations:

$$\begin{aligned} \dot{h_1}(t) &= A_1^{-1} \left(q_{in}(t) - k_1 \sqrt{h_1(t)} \right) + d_1(t) \\ \dot{h_2}(t) &= A_2^{-1} \left(k_1 \sqrt{h_1(t)} - k_2 \sqrt{h_2(t)} \right) + d_2(t) \end{aligned} \tag{4.77}$$

where $A_1 = 0.07m^2$ is the section of the upper tank, $A_2 = 0.08m^2$ is the section of the lower tank, $k_1 = 0.007m^3/(\sqrt{m} \cdot s)$ and $k_2 = 0.012m^3/(\sqrt{m} \cdot s)$ are the discharge coefficients of the upper and lower tank respectively and $d_1(t)$ and $d_2(t)$ are additive disturbances measured in m/s. The state vector x is formed by the heights in meters of both tanks, so $x = [h_1, h_2] \in \mathbb{R}^2$. The control action is the input flow of the upper tank, thus $u = q_{in} \in \mathbb{R}$. The output of the system y is the height of the lower tank, that is $y = h_2 \in \mathbb{R}$. Besides the full nonlinear model given in (4.77), a linearised model of the system around the operating point $h_1^o = 1$, $h_2^o = 0.3403$ and $q^o = 0.007$ will be considered for some of the simulations.

The database has been build using different controllers in closed-loop operation. PI controllers has been selected because they produce trajectories without offset. The values of the controller's parameters are randomly obtained in the following ranges:

$$K_p = [0.0045, 0.0055] \quad T_i = [7000, 9000].$$

The intervals for K_p and T_i have been chosen so that all closed loop trajectories in the database are stable but with different degrees of overshoot and settling time, with the purpose of showing the offset free property of the proposed strategy, i.e., no design criteria for the transient response has been taken into account. The sample time used is 10 minutes. The database has 1000 trajectories and all of them have a time length of 1000 minutes. Each trajectory included in the database start, with a random stable pair $\{x_j^e, u_j^e\}$ defined by an initial random value of the reference r_j. At time $t = 0$, a random reference value is chosen. The random reference values, and consequently the heights of the lower tank, are in the interval $h_2(t), r(t) \in [0.2, 1.5]$. The upper tank height is in the range $h_1 \in [0, 4.1]$ and the input flow $q_{in} \in [0.0006, 0.017]$. The disturbances have non zero mean, being randomly selected in the range $E\{d_1\}, E\{d_2\} \in [0.01, 0.03]$.

In order to get the subset S of partial trajectories of the database, a distance function like 4.74 is taken into account. Notice that the output $y(t)$ is included as a part of $x(t)$ because, in this example, $x(t) = [h_1(t); h_2(t)]$ and $y(t) = h_2(t)$. The data driven controller considers the κ nearest points calculated with the proposed distance function, so $j = [1, \ldots, \kappa]$.

The first simulation considers the linearised model of (4.77) to both generate the database and for the closed loop simulated experiments. In this simulation, the initial state and control action are those of the operating point. During the simulation time, which is 4000 seconds, two step changes are given. At time $t = 2000$ there is a sudden increment in the mean value of the disturbances, from an initial value set to $\mathrm{E}\{d_1(t)\} = \mathrm{E}\{d_2(t)\} = 0.005$ to 0.02 after $t = 2000$. Figure 4.5.2 shows the result of the proposed simulation. The upper plot shows the reference signal $r(t)$ and the level of the lower tank $h_2(t)$, plotted in orange color. The lower plot shows the input signal $q_{in}(t)$ (plotted in orange color). It can be seen that offset free tracking is achieved through all the simulation.

Notice that the reference signal is set to 1.7 meter after time $t = 3000$, which is a reference value out of the range of references of the trajectories contained in the database. However, the proposed controller reaches the desired steady state even if the database does not store any information about them.

The same closed loop simulation is then run considering the nonlinear model of the plant and a database obtained in the same conditions, but closing the loop this time with the nonlinear model (4.77). It can be observed that the proposed strategy with nonlinear model obtains offset free tracking (blue plot of figure 4.5.2) as in the previous case with

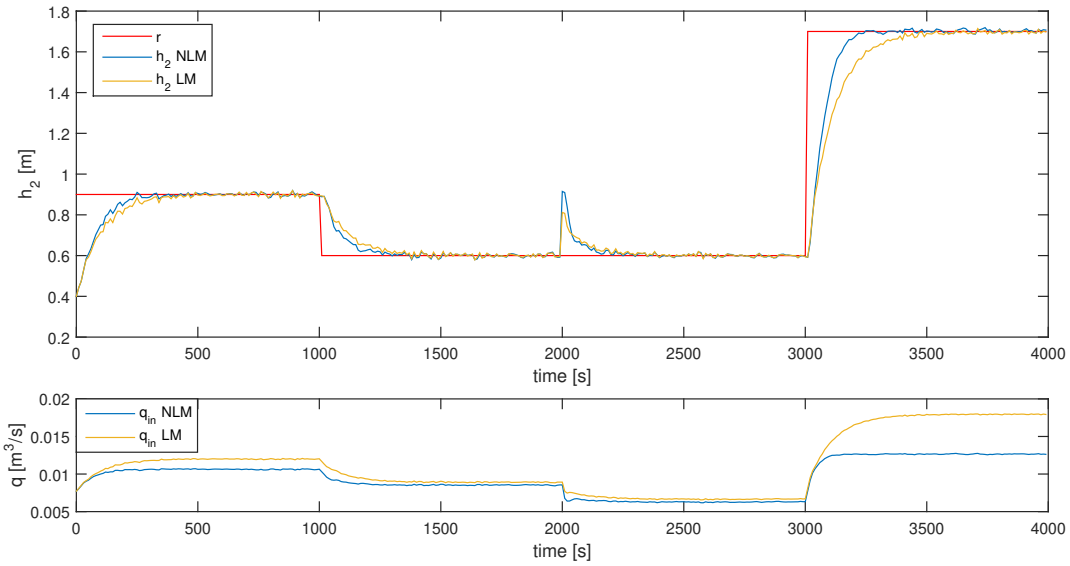


Figure 4.5.2: Closed loop simulation with the proposed controller for the linearized (orange plot) and nonlinear model (blue plot).

a linearized model. The reason for this, is that, although based on a linear model like (4.1), the proposed controller is designed to reject non zero mean disturbances. Thus it can compensate for discrepancies between the dynamics registered in the database and the one shown by the process. For a more exhaustive study of this case, a total of 100 set point tracking and disturbance rejection simulations have been simulated for the non linear model and the linearized. In each simulation, the initial state, step amplitude, and disturbance amplitude are randomly chosen within the ranges used for simulations in figure 4.5.2. The average steady state error in the linearized simulations is $2.28 \cdot 10^{-9}$ and $1.39 \cdot 10^{-4}$ in the nonlinear case. Then, a set of 100 simulations for reference values outside the database range have been performed, being the average steady state error $3.23 \cdot 10^{-9}$ for the linear case and $6.24 \cdot 10^{-2}$ for the nonlinear case. This confirms offset free tracking in the linear case and that, in practice, the offset in nonlinear systems can be very small, especially when the set point to be tracked is within the database range.

Finally, figure 4.5.3 shows two different simulations to demonstrate the necessity of memory constraints (4.34) and (4.35) and offset free trajectories in the database. First simulation is with a similar controller to the proposed data driven controller but removing the memory constraints (named *without memory*). Second simulation is with the proposed data driven controller but with a database in which the trajectories are obtained with proportional (P) controllers (named *P-database*). It can be observed that both cases show obvious steady state errors. Thus, memory constraints and a database with offset free trajectories are the main ingredients to get an offset free tracking with the proposed strategy which with both ingredients can recover the underlying control law.

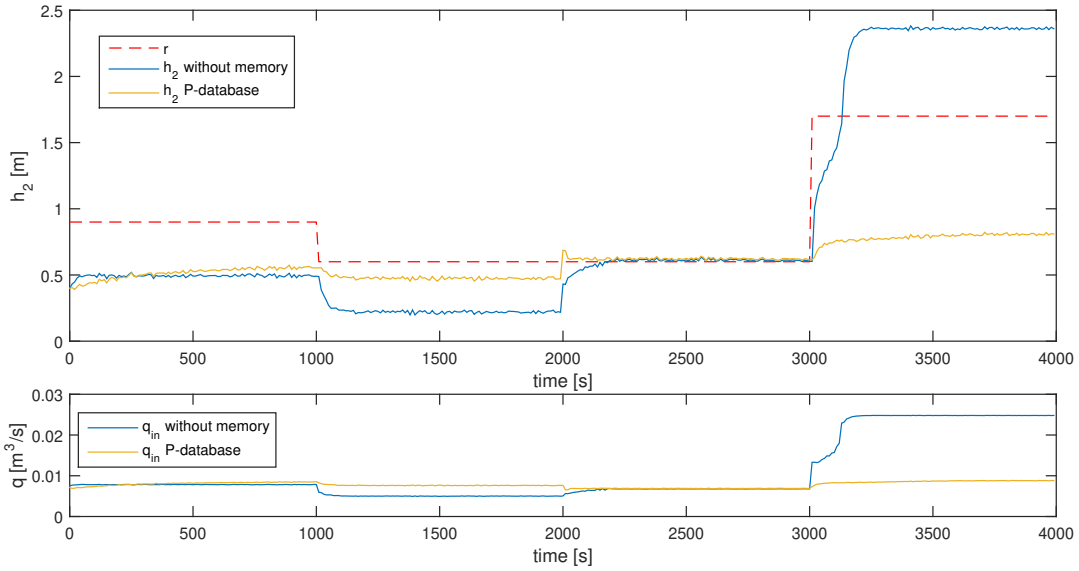


Figure 4.5.3: Closed loop simulation with *without-memory* (blue plot) and *P-database* (orange plot).

4.6 Application to the Feedback Process Trainer 37-100

The proposed controller has been tested on the Feedback Process Trainer 37-100 (see figure 4.6.1), a renewed version of the Feedback PT-326 [61]. In this equipment, an axial fan is used to circulate air through a heating element inside a propylene tube. The heating element can be controlled using a voltage input and the air temperature is measured at the end of the tube by a bead thermistor. The system exhibits air and tube thermal time constants. The dynamic characteristics of the system can be changed by manually changing the fan speed using a potentiometer. The controlled variable in these experiments is the voltage output of the bead thermistor, whereas the manipulated variable is the voltage input that controls the heating element.

Although an scaled laboratory process, this equipment is a challenging test bed for data driven algorithms or other complex control methods because of its fast dynamics, that require sampling times of a few hundredths of a second. A sampling time of 0.07 seconds has been used through all the experiments shown in this section. The delay is neglected because it is lower than the sample time, thus no delay compensation has been taken into account.

Following the nomenclature, the input u is the voltage that controls the heating element, that is $u(t) = V(t) \in [0, 10]$ volts, and the output y is the temperature measured by the sensor represented in a voltage, that is $y(t) = V_T(t) \in [0, 10]$ volts. As it is well known that the process control trainer can be characterized by a first order model, in this experiments the state is equal to the output, so $x(t) = y(t)$.

A total of 300 eight minute trajectories have been generated with the fan speed potentiometer set to 50%. Each trajectory is defined by a random initial set point and a step set point change of random amplitude, computed in a way such that the initial and final

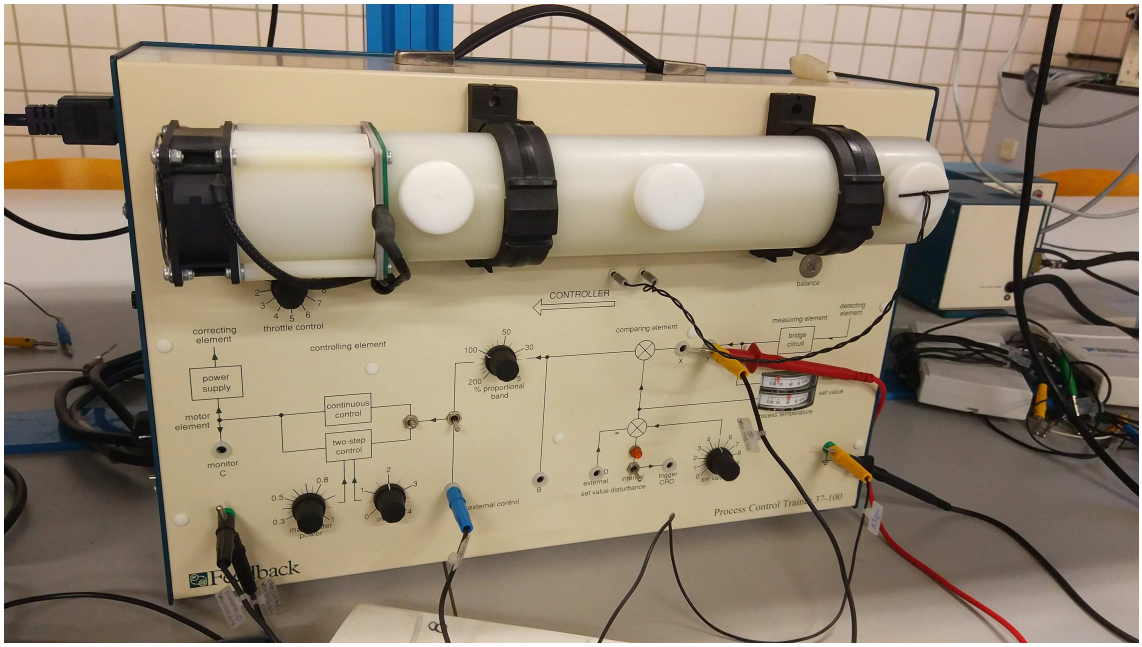


Figure 4.6.1: Feedback Process Trainer 37-100 unit.

set point values differ between 1 and 8 volts. For each trajectory 4 different PI controllers have been tested (each one for 2 minutes, changing from the initial set point value to the step value and back again to the initial value). Table 4.1 shows the parameters for each PI controller while Figure 4.6.2 shows the behaviour of the transient response for each PI controller with 4 different closed-loop tracking experiments. The database comprises a total of 300 two minute closed loop offset free trajectories which results in 51400 different candidates in S. It is noteworthy that the database took 40 hours to be generated, thus the ambient temperature changed quite a bit during the morning and night hours. This implies that the process dynamics are not constant through the database leading to different perturbations mean values for each trajectory.

	$\mathbf{PI_1}$	$\mathbf{PI_2}$	$\mathbf{PI_3}$	$\mathbf{PI_4}$
$\mathbf{K_p}$	1	1	1	1
$\mathbf{K_i}$	1.5	1.07	0.64	0.21

Table 4.1: Parameters for PI controllers of the database.

Following the procedure presented in Algorithm 2, the distance function (4.74), with $\alpha = I$ has been used to select the $n_c = 6000$ nearest points to the current state in the database. It is noteworthy that the solution of (4.71) and the control law (4.72) can be computed within the sampling time of 0.07 seconds in Matlab with a Intel Core-i3 running Windows.

Figure 4.6.3 shows the results of a set point tracking experiment with two reference changes using the proposed approach. The controller achieves offset free tracking in each set point value (plotted in red), despite the obvious noise and disturbances. It can be seen how the controller adjusts continually, in a clear trend, the control effort to keep the controlled variable near the set point. The reason of this trend is the heating of the propylene tube, much slower than the heating of the air, but nonetheless able to affect the controlled

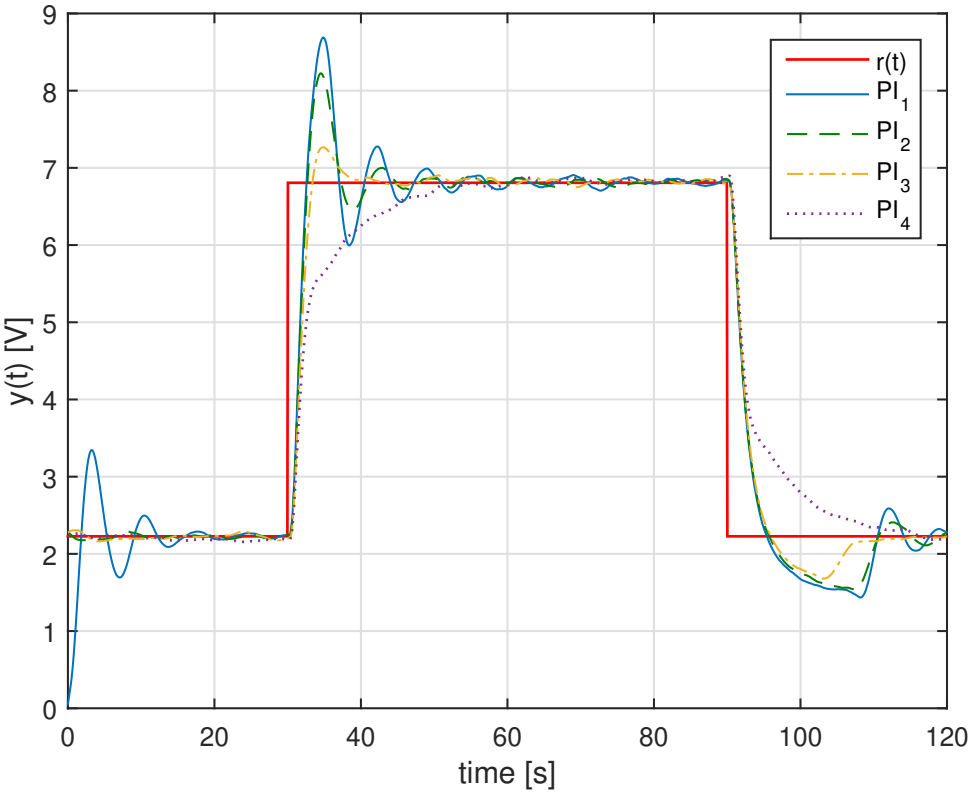


Figure 4.6.2: Tracking experiments in closed-loop with the PI controllers that generate the database.

variable.

To demonstrate the disturbance rejection properties of the controller, three different experiments have been considered. First, a constant error of amplitude 2 volts is added artificially to the temperature measure after 3 minutes. Figure 4.6.4 shows the trajectory of the measured temperature and how it converges again to the reference value because the proposed controller successfully rejects the disturbance. Note that the input has to modify its steady state value to compensate for the effect of the additive disturbance.

Second, the fan speed potentiometer has been increased from 50%, the value used to generate the database trajectories, to 80%. As a result of the increased fan speed, the temperature drops and the controller is forced to raise the voltage applied to the heater. After the disturbance is rejected, the fan speed is changed back to its previous value, which is again another disturbance that it is also effectively rejected. Figure 4.6.5 shows the experimental results. It is noteworthy that the changes induced in the system by increasing the fan speed are more severe than the additive measurement disturbance included in the previous simulation. Despite this disturbance, the nonlinearity of the system and the variations in the ambient temperature, the controller is able to track set point changes and reject disturbances.

Finally, an even more difficult case is shown in figure 4.6.6 where the fan speed was reduced from 50% to 30%. This case is more difficult than the previous one because in addition to

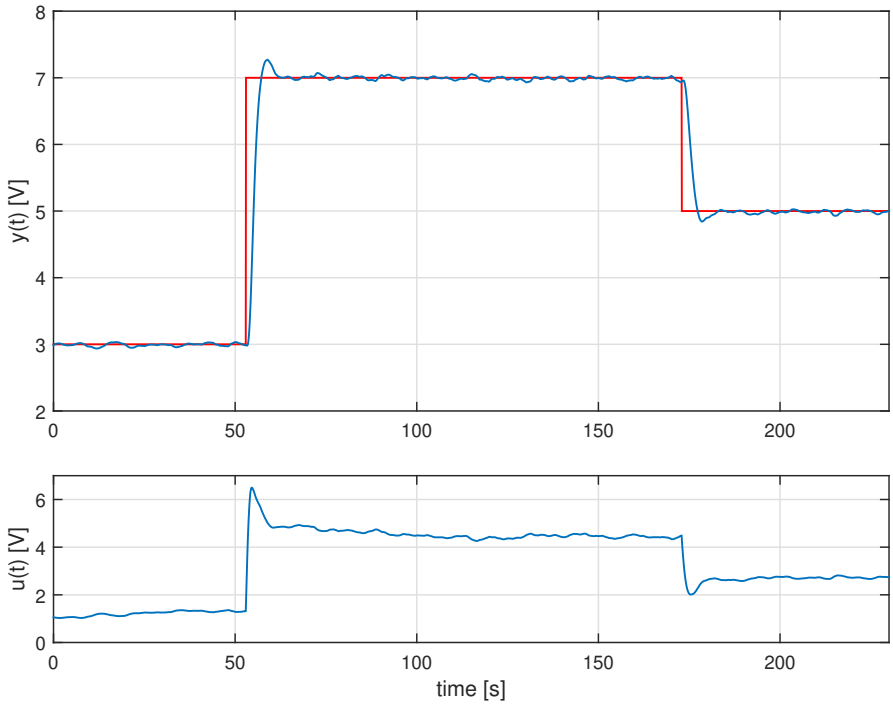


Figure 4.6.3: Tracking test results on the Feedback Process Trainer 37-100.

the changes in the process dynamics, the dead-time is increased. Nevertheless, as shown in figure 4.6.6, the controller is able to track the set-point with minimal steady state error while compensating the slow drifts in the temperature.

In order to study the effect of the number of candidates n_c on the closed-loop performance, we will compare the tracking error of a set of controllers with n_c taking values $n_c = \{1000, 1500, 2000, 2500, 3000\}$. For each controller, 15 closed-loop tracking experiment with length 60 seconds (857 samples) have been carried out with different reference values, in particular, five experiments with reference $r = 3.5$, five with reference $r = 5$ and another five with reference $r = 7$. For each experiment the mean value of the squared error is computed for 172 samples once the closed-loop system has converged to its corresponding reference.

Figure 4.6.7 shows the steady state error for the 75 experiments as magenta dots. Furthermore, the average of the steady state error for the 5 closed-loop experiments with the same constant reference signal and n_c is computed. The blue dashed-dotted line shows this result for reference $r = 3.5$, the red dashed line represents the average of the experiments with $r = 5$ and the green dotted one is with $r = 7$. The average of the 15 experiments with the same n_c is represented with the black solid line. We can observe that the steady state error for all the experiments is always upper-bounded by $2.5 \cdot 10^{-2}$. Moreover, the steady state error decreases when n_c is increased which is the expected behavior.

It is important to remark that reducing too much the number of candidates has a negative impact on the performance of the controller. In those cases, the controller is forced to extrapolate, which results in worse control. Figure 4.6.8 shows a tracking experiment

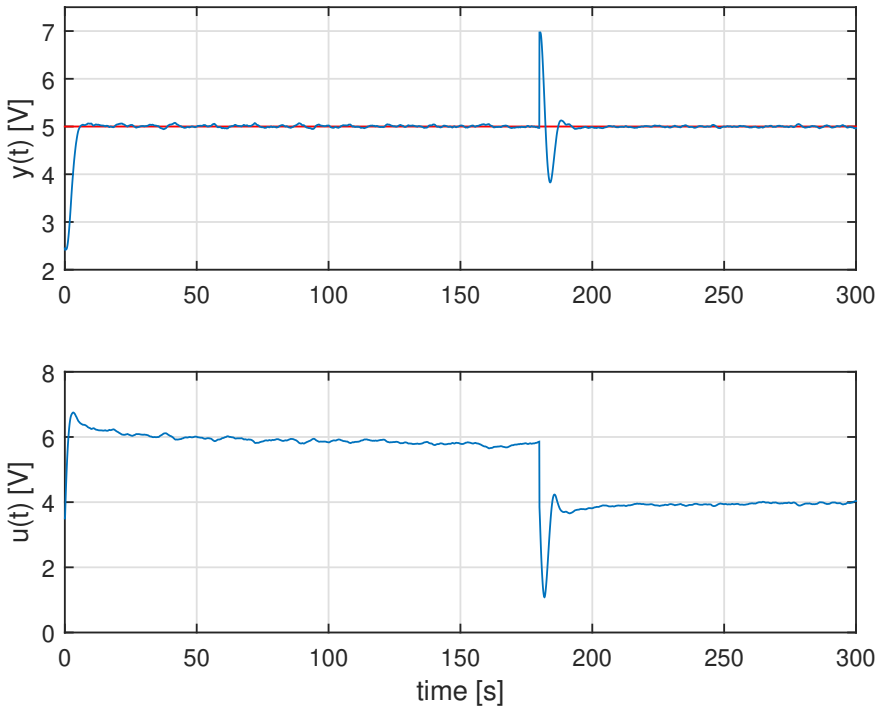


Figure 4.6.4: Disturbance rejection test with the Feedback Process Trainer 37-100: Artificial additive disturbance.

where only the 50 closest candidates are considered. In this experiment, the average tracking error is quite large (almost 0.4 volts with a standard deviation of 0.36), but as shown in table 4.2, these numbers drop as expected if the experiment is repeated with an increasingly higher number of candidates (note that these experiments are different of that of figure 4.6.7 as they contain several set point changes, hence the greater average error and standard deviation).

n_c	50	100	500	1500
$\bar{e}$	0,3998	0,1423	0,0889	0,0872
σ	0,3686	0,1487	0,1434	0,1480

Table 4.2: Average tracking error and standard deviation in varying set-point tracking experiment for low numbers of candidates.

Finally, it is demonstrated that the proposed strategy performance is strongly dependent on the database trajectories considered. Figure 4.6.9 shows the output of the data-based controller in closed loop using two different databases for the same reference signal (dashed red). In the first case, a more oscillating PI controller with parameters $K_p = 1$ and $K_i = 1.5$ is used to obtain the database. On the other hand, a PI controller less aggressive with parameters $K_p = 1$ and $K_i = 0.21$ generates the second database. Note that the closed-loop trajectory of the data-based controller reaches the reference independently of the database used, however its behaviour in the transient response is inherited from the database trajectories.

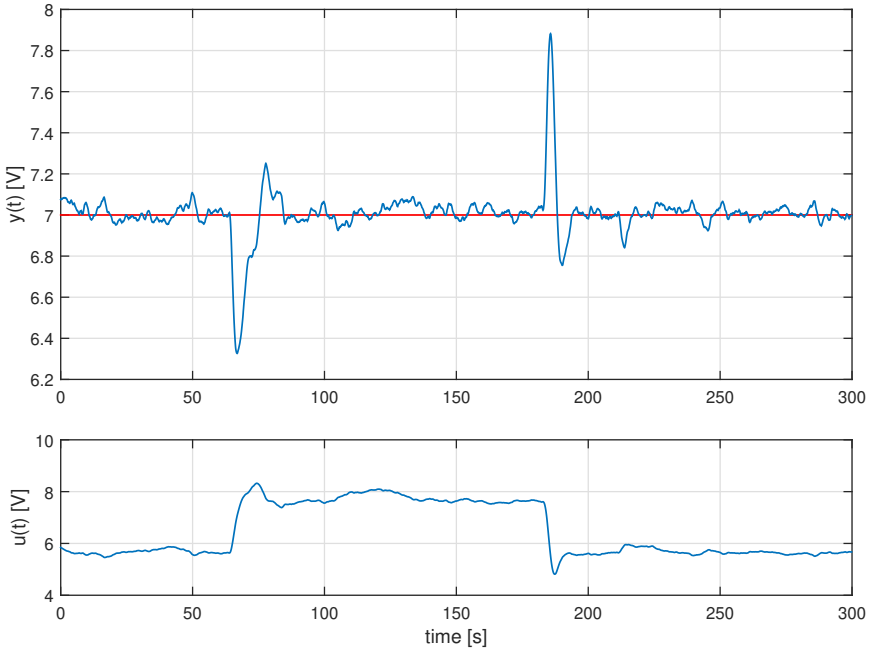


Figure 4.6.5: Disturbance rejection test the Feedback Process Trainer 37-100: Fan speed increased.

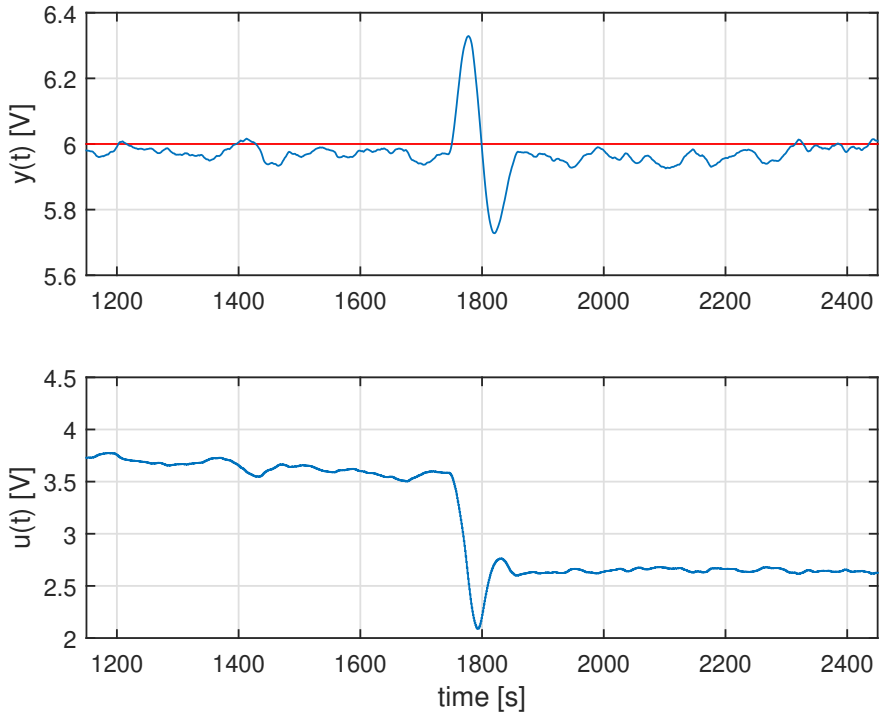


Figure 4.6.6: Disturbance rejection test the Feedback Process Trainer 37-100: Fan speed decreased.

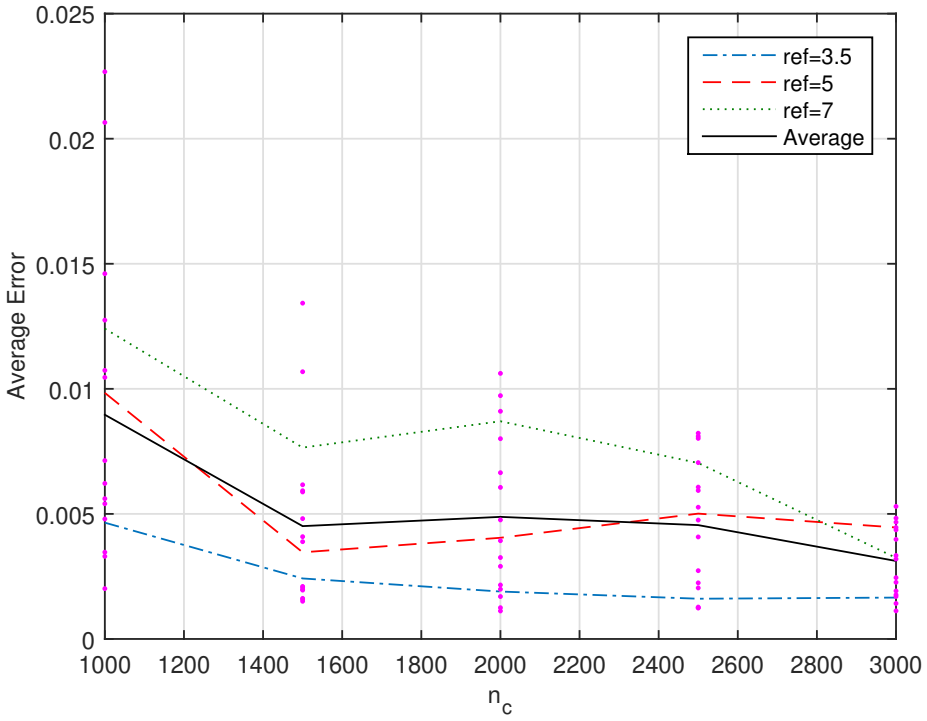


Figure 4.6.7: Average steady state error against the number of candidates n_c for different value of the reference.

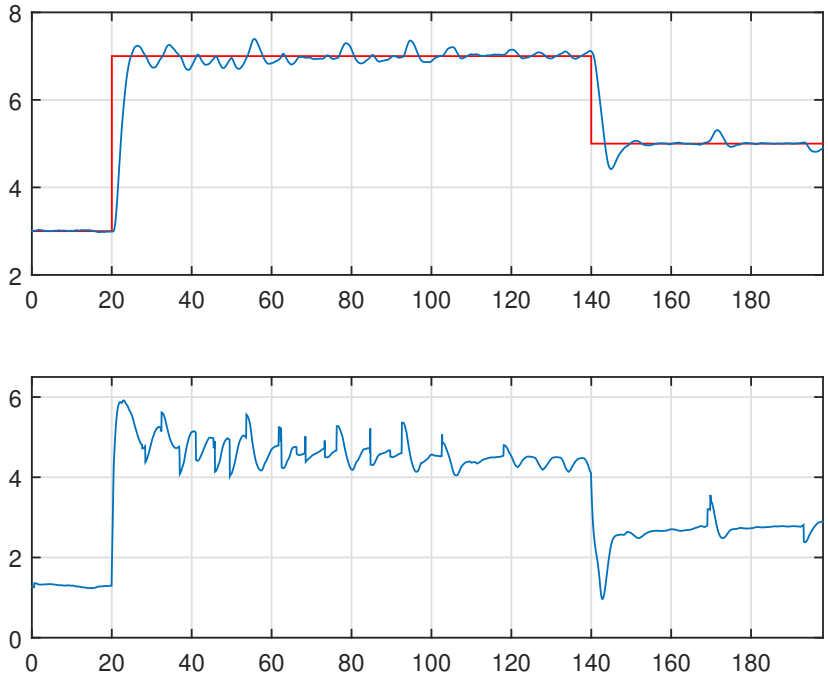


Figure 4.6.8: Tracking experiment with a low number of candidates ($n_c = 50$).

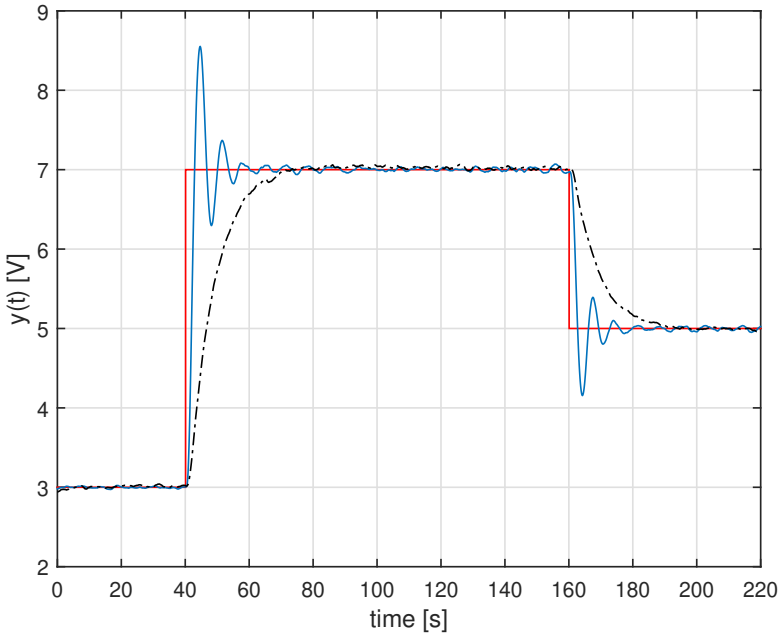


Figure 4.6.9: Comparison of the proposed controller with a database using only a PI controller with $K_p = 1$ and $K_i = 1.5$ (solid blue) and with a database using only a PI controller with $K_p = 1$ and $K_i = 0.21$ (dashed-dotted black). Red dashed line represents the reference signal.

Chapter 5

Data based predictive control via direct weight optimization

Multi-objective optimization (MOO), also known as multi-criteria or multi-performance, is a different paradigm to deal with optimization problems. In recent years, control research community has paid attention to these techniques, as demonstrate the overviews [39, 60] of MOO application in control engineering. In MPC different optimization criteria depending on the minimization objective can be considered, thus MOO can be combined with MPC as in [49, 52], or even with data driven approaches as in [118, 120].

Regarding large complex systems, different objectives with contradictory performance are present. An example is WDNs, where economic performance and demand satisfaction objectives are to be met. The first one proposes to spend as less as possible electricity while trying to consume it in the low tariff period (usually at night) in order to reduce the cost of pumping water. The second one sets that it is necessary to satisfy client's water demands, so to fill up water tanks in the network over safety levels is priority. Thus, multi-performance is required in these kind of systems.

Previous chapters have shown that, typically, data based techniques have focused on a single objective. Identification methods try to obtain a model from data with low estimation errors (i.e. [14, 50, 71]) which allows MPC approaches to predict future behaviour of the system. On the other hand, in model-free strategies, the only information available on the system is past trajectories which are directly used to achieve the final control objective as in [27, 38, 43, 94, 100, 107].

In this chapter we propose to use a prediction model based on identification via direct weight optimization ([20, 92]) which are based on postulating an estimator that is linear in the observed outputs and then determining the weights in this estimator by direct optimization of a suitably chosen criterion. In particular, taking MOO philosophy, we propose to optimize the weights of this linear combination taking into account simultaneously performance and estimation error. For unconstrained systems, dynamic programming is used to obtain an explicit linear solution of a finite or infinite horizon optimal control problem. To obtain the explicit solution, a procedure to exploit the structure of the resulting quadratic optimization problem based on the matrix inversion lemma ([116]) is also provided. If constraints are taken into account, we propose to solve online a con-

strained quadratic optimization problem. In this case, the optimization problem can take into account local information to improve the performance.

5.1 Problem formulation

We consider a system described by an unknown time-invariant discrete-time model

$$z = f(x, u), \tag{5.1}$$

where $x \in X \subseteq \mathbb{R}^{n_x}$ is the system state, $u \in U \subseteq \mathbb{R}^{n_u}$ is the control input vector, $z \in X$ is the successor state, and f is the unknown transition function. In our setting, we will treat the system generating the data as if it were linear, although it may not be in reality if f is nonlinear.

The control objective is to regulate the system to the origin while minimizing the performance cost defined by the following quadratic stage cost

$$\ell(x, u) = x^\top Q x + u^\top R u,$$

where Q and R are the tuning parameters of the controller.

If function f were known, this problem could be solved using off-the-shelf techniques, for example dynamic programming. Instead in this chapter we propose to use the information stored in a database to predict the behavior of the system and solve simultaneously the control design problem. This database stores M different triplets of data corresponding to a state x_q, an input u_q, and the corresponding successor state z_q.

We assume that the samples collected in the data-set satisfy the following equation

$$z_q = f(x_q, u_q) + w_q, \tag{5.2}$$

where w_q models measurement errors that we assume independent from x_q, u_q.

Each data triplet (x_q, u_q, z_q) is denoted as a "candidate". The main idea is to estimate the dynamics of the system using a weighted linear combination of the candidates that is consistent with the current state and input ([92, 20]). The weight of each candidate is $\lambda_q \in \mathbb{R}$.

Lemma. Consider a set of M weights $\lambda_q \in \mathbb{R}$ and candidates (x_q, u_q, z_q) that satisfy (5.2) where (x_q, u_q) are independent for all q. If $f : X \times U \to X$ is linear and each entry w_{q_i} of vector w_q is a zero-mean, i.i.d. random variable with variance σ^2, then

$$f\left(\sum_{q=1}^{M} \lambda_q x_q, \sum_{q=1}^{M} \lambda_q u_q\right) = \sum_{q=1}^{M} \lambda_q z_q + e,$$

where the estimation error e is a random vector whose entries are zero-mean, i.i.d. random variables with variance $\sum_{q=1}^{M} \lambda_q^2 \sigma^2$.

Proof.

Since f is linear,

$$f\left(\sum_{q=1}^{M} \lambda_q x_q, \sum_{q=1}^{M} \lambda_q u_q\right) = \sum_{q=1}^{M} \lambda_q f(x_q, u_q).$$

From (5.2), if follows that

$$f(x_q, u_q) = z_q - w_q,$$

and hence

$$f\left(\sum_{q=1}^{M} \lambda_q x_q, \sum_{q=1}^{M} \lambda_q u_q\right) = \sum_{q=1}^{M} \lambda_q (z_q - w_q).$$

By defining $e = -\sum_{q=1}^{M} \lambda_q w_q$ we have that

$$f\left(\sum_{q=1}^{M} \lambda_q x_q, \sum_{q=1}^{M} \lambda_q u_q\right) = \sum_{q=1}^{M} \lambda_q z_q + e.$$

and taking into account that each entry w_{q_i} of vector w_q is a zero-mean, i.i.d. random variable with variance σ^2, then e is a random vector whose entries are zero-mean, i.i.d. random variables with variance $\sum_{q=1}^{M} \lambda_q^2 \sigma^2$. □

In the following section we propose to solve an optimization problem to determine the optimal candidates weights λ_q to minimize both the performance cost and the estimation error variance following a predictive control approach based on dynamic programming.

5.2 Unconstrained explicit data based predictive control

In this section we present a data-based predictive control for unconstrained systems. The controller is based on solving a dynamic programming problem over a finite horizon in which the optimization variables are, at each iteration i, the optimal weights λ_{iq} for each candidate q.

Following a dynamic programming approach for linear systems and quadratic costs, given a state x we assume that the cost to go is a quadratic function of the state,

$$J_i(x) = x^\top P_i x, \tag{5.3}$$

where P_0 defines the terminal cost of the data-based predictive controller. Then, $J_i(x)$ is defined recursively as follows:

$$J_{i+1}(x) = x^\top Q x + u^\top R u + J_i(z), \tag{5.4}$$

with

$$x = \sum_{q=1}^{M} \lambda_{iq}^*(x) x_q, \quad u = \sum_{q=1}^{M} \lambda_{iq}^*(x) u_q, \quad z = \sum_{q=1}^{M} \lambda_{iq}^*(x) z_q,$$

and

$$\begin{aligned} \lambda_{iq}^*(x) \quad = \quad & \arg \min_{\lambda_{i1} \ldots \lambda_{iM}} u^\top R u + J_i(z) + \beta \sum_{q=1}^{M} \lambda_{iq}^2 \\ \text{s.t.} \quad & x = \sum_{q=1}^{M} \lambda_{iq} x_q, \\ & u = \sum_{q=1}^{M} \lambda_{iq} u_q, \\ & z = \sum_{q=1}^{M} \lambda_{iq} z_q, \end{aligned} \tag{5.5}$$

where $\beta > 0$ is a tuning parameter to provide a trade-off between performance and estimation error.

Note that the cost function of problem (5.5), which is used to calculate the optimal weights $\lambda_{iq}^*(x)$, includes a term that is proportional to the variance of the estimation error, $\sum_{q=1}^{M} \lambda_{iq}^2$. This term, however, is not included in the definition of the optimal cost of the proposed controller, $J_i(x)$.

Optimization problem (5.5) is solved recursively from $i = 1, \ldots, N$. The proposed unconstrained explicit data-based predictive control is then defined as

$$u^*(x) = \sum_{q=1}^{M} \lambda_{N-1,q}^*(x) u_q.$$

5.2.1 Multi-parametric solution of the proposed optimization problem

In this section, an explicit solution of problem (5.5) is provided using a multi-parametric approach in order to compute $J_{i+1}(x)$ from $J_i(x)$. To this end, we define the optimization vector

$$\lambda_i = [\lambda_{i1} \ldots \lambda_{iM}]^\top \in \mathbb{R}^{M \times 1},$$

and the following matrices obtained from the database

$$\begin{array}{rcl} X_q & = & [\ x_1 \quad x_2 \quad \ldots \quad x_M\ ], \\ U_q & = & [\ u_1 \quad u_2 \quad \ldots \quad u_M\ ], \\ Z_q & = & [\ z_1 \quad z_2 \quad \ldots \quad z_M\ ]. \end{array}$$

By treating x as a vector of parameters, optimization (5.5) is equivalent to the following parametric optimization problem

$$\begin{array}{rll} \lambda_i^*(x) & = \arg\min\limits_{\lambda_i} & \lambda_i^\top U_q^\top R U_q \lambda_i + \lambda_i^\top Z_q^\top P_i Z_q \lambda_i \\ & & +\lambda_i^\top \beta_M \lambda_i \\ & \text{s.t.} & x = X_q \lambda_i, \end{array} \tag{5.6}$$

where $\beta_M = \beta I$.

Problem (5.6) can be rewritten as

$$\begin{array}{ll} \min\limits_{\lambda_i} & \lambda_i^\top H_i \lambda_i \\ \text{s.t.} & T\lambda_i = Sx, \end{array} \tag{5.7}$$

where

$$\begin{array}{rcl} H_i & = & U_q^\top R U_q + Z_q^\top P_i Z_q + \beta_M \in \mathbb{R}^{M \times M}, \\ T & = & X_q \in \mathbb{R}^{n_x \times M}, \\ S & = & I \in \mathbb{R}^{n_x \times n_x}. \end{array}$$

If $H_i \succ 0$, the solution of problem (5.7) can be obtained solving by the following set of linear equations obtained from the Karush-Kuhn-Tucker conditions:

$$\begin{bmatrix} H_i & T^\top \\ T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \lambda_i \\ \tau_i \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ Sx \end{bmatrix}, \tag{5.8}$$

where τ_i are the Lagrange multipliers of the equality constraints in (5.7).

Since $H_i \succ 0$, from (5.8) the optimal value of vector λ_i can be obtained as a explicit function of τ_i,

$$\lambda_i^*(\tau_i) = -H_i^{-1}T^\top \tau_i. \tag{5.9}$$

From (5.8) and (5.9) we obtain that τ_i must satisfy the following equation

$$T(-H_i^{-1}T^\top \tau_i) = Sx.$$

Assuming that the linear quadratic constraint qualification (LICQ) condition holds so that $TH_i^{-1}T^\top$ is invertible, we get

$$\tau_i^*(x) = -(TH_i^{-1}T^\top)^{-1}Sx. \tag{5.10}$$

By substituting (5.10) in (5.9) we obtain

$$\lambda_i^*(x) \;=\; H_i^{-1}T^\top(TH_i^{-1}T^\top)^{-1}Sx = K_{i+1}x,$$

where

$$K_{i+1} = H_i^{-1}T^\top(TH_i^{-1}T^\top)^{-1}S. \tag{5.11}$$

Taking into account (5.3) and (5.4) it follows that

$$P_{i+1} = Q + K_i^\top(U_q^\top R U_q + Z_q^\top P_i Z_q)K_i. \tag{5.12}$$

Equation (5.12) provides an iterative procedure to obtain the value of P_{i+1} from P_i. The proposed unconstrained explicit data-based predictive control is then defined as

$$u^*(x) = U_q K_N x.$$

5.2.2 Efficient computation of the inverse of H_i

The computation of matrices K_{i+1} and P_{i+1} requires the computation of the inverse of H_i. This can be a cumbersome problem when the number of rows in the database M is high. We present next a procedure to compute this inverse efficiently profiting from the structure of the matrix.

Matrix H_i can be parametrized as follows

$$H_i = \beta_M - \Theta^\top \Upsilon \Theta,$$

where

$$\Theta = \begin{bmatrix} U_q \\ Z_q \end{bmatrix}, \qquad \Upsilon = -\begin{bmatrix} R & \mathbf{0} \\ \mathbf{0} & P_i \end{bmatrix}.$$

Considering the matrix inversion lemma, or the Sherman-Morrison-Woodbury formula, particularized for symmetric matrices, we have that

$$\begin{aligned} H_i^{-1} &= (\beta_M - \Theta^\top \Upsilon \Theta)^{-1} \\ &= \beta_M^{-1} + \beta_M^{-1}\Theta^\top(\Upsilon^{-1} - \Theta\beta_M^{-1}\Theta^\top)^{-1}\Theta\beta_M^{-1} \\ &= \frac{1}{\beta}I + \frac{1}{\beta^2}\Theta^\top(\Upsilon^{-1} - \frac{1}{\beta}\Theta\Theta^\top)^{-1}\Theta. \end{aligned}$$

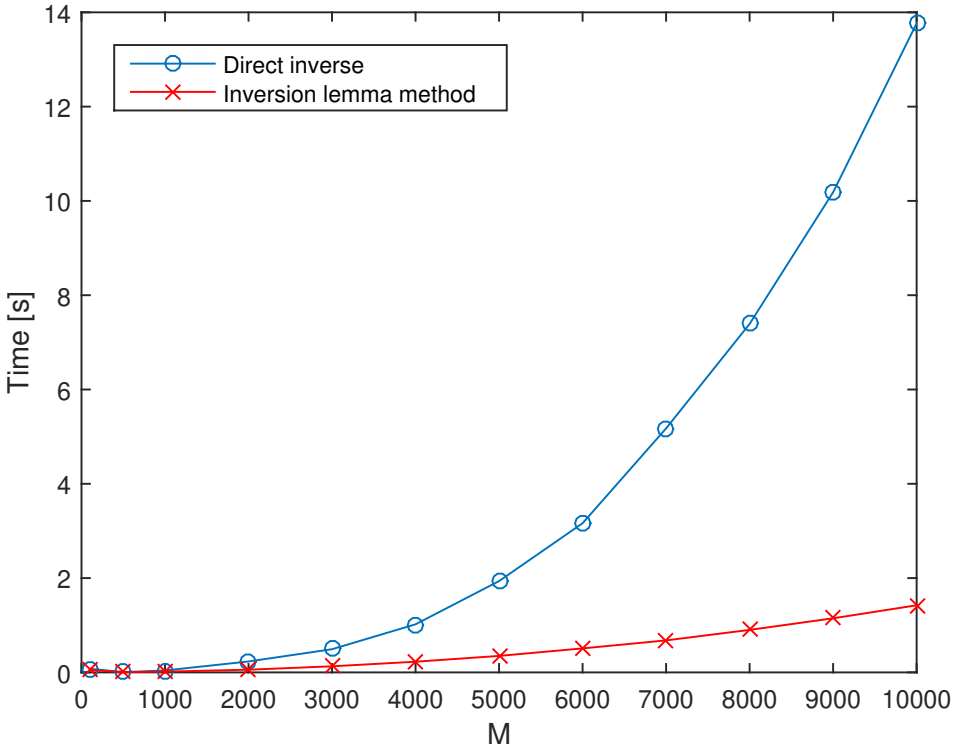


Figure 5.2.1: Comparison between direct inverse of H_i and inverse obtained by the inversion lemma.

The dimension of $(\Upsilon^{-1} - \frac{1}{\beta}\Theta\Theta^\top)$ is $(n_u + n_x) \times (n_u + n_x)$ which is much lower than the dimension M of H_i. This implies that using the inversion lemma is more efficient than a direct inversion. Figure 5.2.1 shows the comparison between the relation of computational time and M, matrix H_i dimension, for both techniques, direct inversion and inversion lemma method.

5.3 Constrained single-step data based predictive control

The proposed control-synthesis procedure can also be applied to constrained systems. However, in this case the resulting optimal control law would be a piece-wise linear function, and the corresponding optimal cost to go, a piece-wise quadratic function ([9]). This implies that a dynamic programming approach cannot be easily applied. Instead, a finite horizon open-loop constrained optimal control problem can be formulated and solved. In this case, for each prediction step j and candidate q a weight λ_{jq} must be optimized so that for each prediction step $j \in \{1, \ldots, N-1\}$ the following constraints are satisfied in order to obtain a state and input prediction

$$x_j = \sum_{q=1}^{M} \lambda_{jq} x_q, \quad u_j = \sum_{q=1}^{M} \lambda_{jq} u_q, \quad z_j = \sum_{q=1}^{M} \lambda_{jq} z_q,$$

with $x_{j+1} = z_j$ and x_0 equal to the current measured state. Taking into account that the number M of possible candidates can be very large, we propose to use a prediction

horizon of one step in order to consider state and input constraints, that we define as linear constraints by taking the sets X and U as polyhedra.

Solving an online optimization problem at each time step following a receding horizon approach allows us to take into account the current measured state in the definition of the optimization problem, in particular, in the estimation procedure. To this end we propose that, at each sampling time, only a subset of the candidates $G(x)$ is taken into account (for example, the closest candidates in the state space) and in addition, that the cost that depends on the square of the candidate weight for each candidate $\beta_q(x)$ depends also on the current state (for example, proportionally with the distance between the current state x and the candidate state x_q). Using close candidates and penalizing those that are far, in some sense, aims at obtaining a better local approximation of the system dynamics.

For a given state x, we propose to solve online the following optimization problem to determine the optimal candidates weights $\lambda_q^*(x)$ that minimize a cost function that depends on both the performance and the estimation error variance:

$$\begin{aligned} \min_{\lambda_q,\, q\in G(x)} \quad & x^\top Qx + u^\top Ru + z^\top Pz + \sum_{q=1}^{M} \beta_q(x)\lambda_q^2 \\ \text{s.t.} \quad & x = \sum_{q\in G(x)} \lambda_q x_q, \\ & u = \sum_{q\in G(x)} \lambda_q u_q \in U, \\ & z = \sum_{q\in G(x)} \lambda_q z_q \in X. \end{aligned} \tag{5.13}$$

The optimization problem (5.13) is solved at each sampling time k for the current state x. The proposed unconstrained explicit data-based predictive control law is then defined as

$$u^* = \sum_{q\in G(x)} \lambda_q^*(x) u_q.$$

5.4 Example 1

In this section proposed approach will be applied to obtain an approximation of the optimal linear quadratic regulator for the double integrator system, that is

$$f(x,u) = Ax + Bu \tag{5.14}$$

with

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

To generate the controller a database of $M = 1000$ rows is generated. Each row contains a random initial state x_q whose components belong to the interval $[-10, 10]$ and a random control action u_q in the interval $[-1, 1]$. The successor state z_q is generated as (5.2 using model (5.14) where w_q is a random vector whose components have zero mean and belong

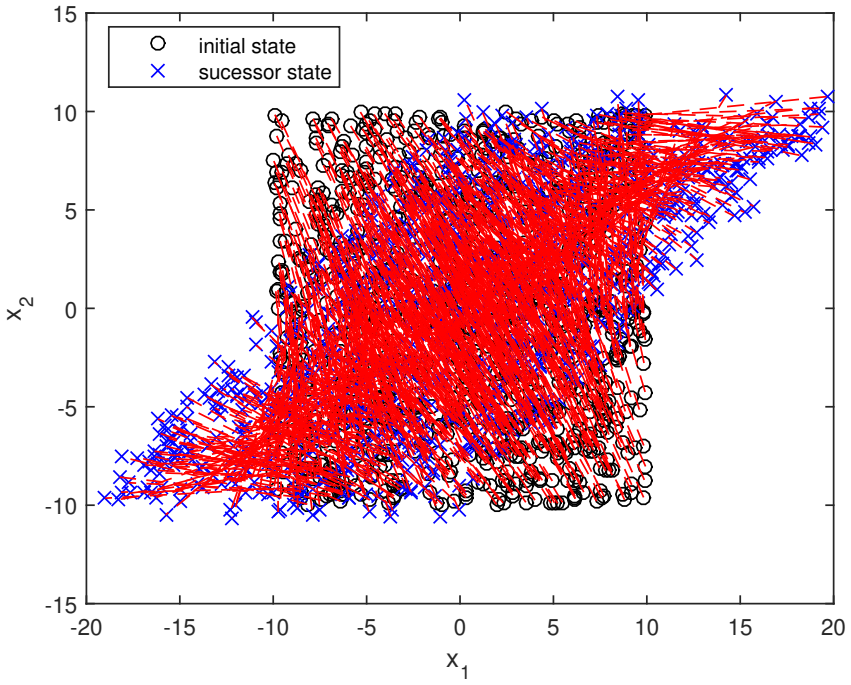


Figure 5.4.1: Representation of the Database. In x_1 and x_2 axis are represented the first and the second components of the current (and the successor) state.

to the interval $[-0.1, 0.1]$. Figure 5.4.1 shows the representation of the information of x_q and z_q states stored in the database.

We group set of $G = 10$ consecutively rows from the database and form a grouped database of size $M_G = M/G = 100$ whose rows are the average of the values of each set, in order to reduce the computational burden because of the lower information used from the grouped database. An additional advantage is that the effect obtained is similar to filter the noise w_q. Figure 5.4.2 shows the representation of the information of x_q and z_q states stored in the grouped database

The control objective is defined by the following matrices.

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$R = 10, \quad P_0 = Q.$$

Using the iterative explicit solution of the proposed controller, we can define the error e as

$$e = |P_{LQR} - P_i| \tag{5.15}$$

where P_{LQR} is the matrix solution of the Riccati equation, obtained with a discrete LQR controller and model (5.14) without noise.

Considering the least square method in order to get an estimation of the matrices of the model, denote as $\hat{A}$ and $\hat{B}$, it is possible to define the following error

$$e_{LS} = |P_{LQR} - P_{LS}| \tag{5.16}$$

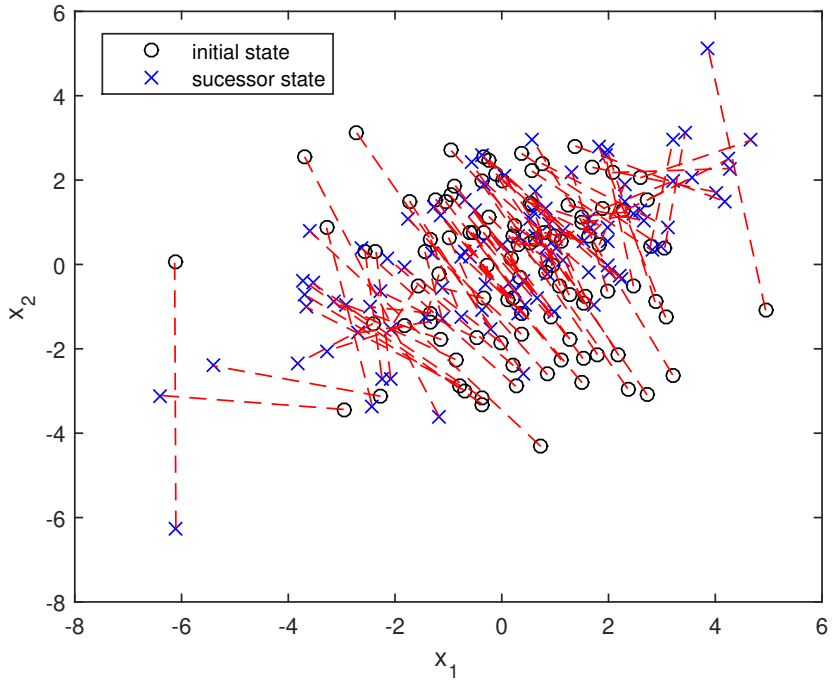


Figure 5.4.2: Representation of the grouped Database with the same representation as in Figure 5.4.1.

where P_{LS} is the matrix solution of the Riccati equation, obtained with a discrete LQR controller and the model

$$z = \hat{A}x + \hat{B}u \tag{5.17}$$

Figure 5.4.3 shows the evolution of the iterative explicit method error with respect the number of iterations with a fixed value of $\beta = 20$. Notice that the dashed line represent the error obtained with the estimation of the least squared method which is greater than the proposed controller method when the number of iteration is higher than a certain amount.

The best value for β parameter depends on the model issues and the disturbance characteristics. Figure 5.4.4 shows the relation between β parameter an the explicit P_i error when it converges for the model and the disturbance specified.

Notice that, in this case, only for β values in the interval $[17, 23]$, the error of the proposed method is lower than the error obtained with the least square method.

5.5 Example 2

In this section the proposed approach is applied to control a quadruple-tank system which is shown in figure 5.5.1. The systems is made of four tanks that are filled from a storage tank located at the bottom of the plant through two three-way valves. The tanks at the top (tanks 3 and 4) discharge into the corresponding tank at the bottom (tanks 1 and 2, respectively). The inlet flows of the three-way valves are the manipulated variables of the

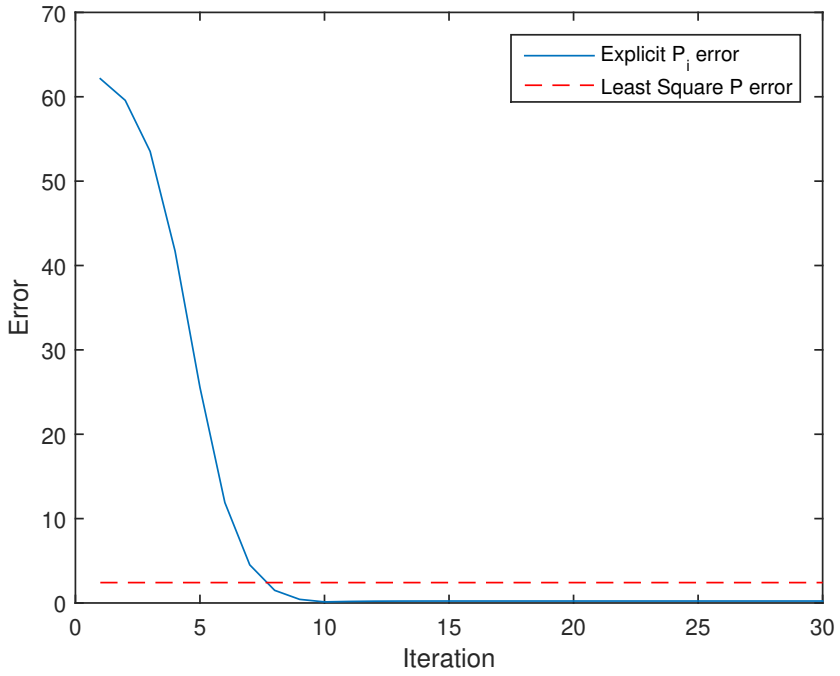


Figure 5.4.3: Comparison between explicit P_i error and number of iterations with $\beta = 20$.

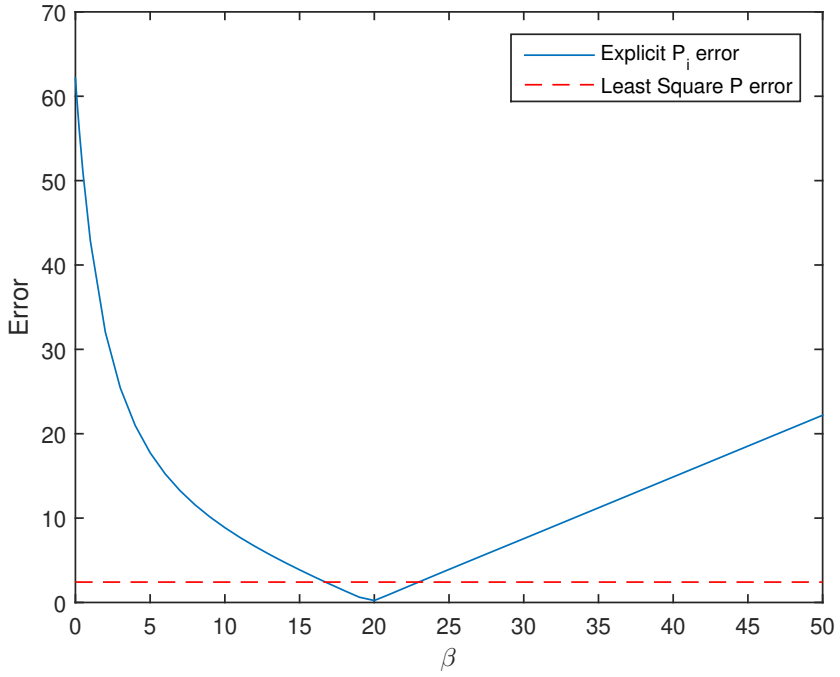


Figure 5.4.4: Comparison between explicit P_i (converged) error and β.

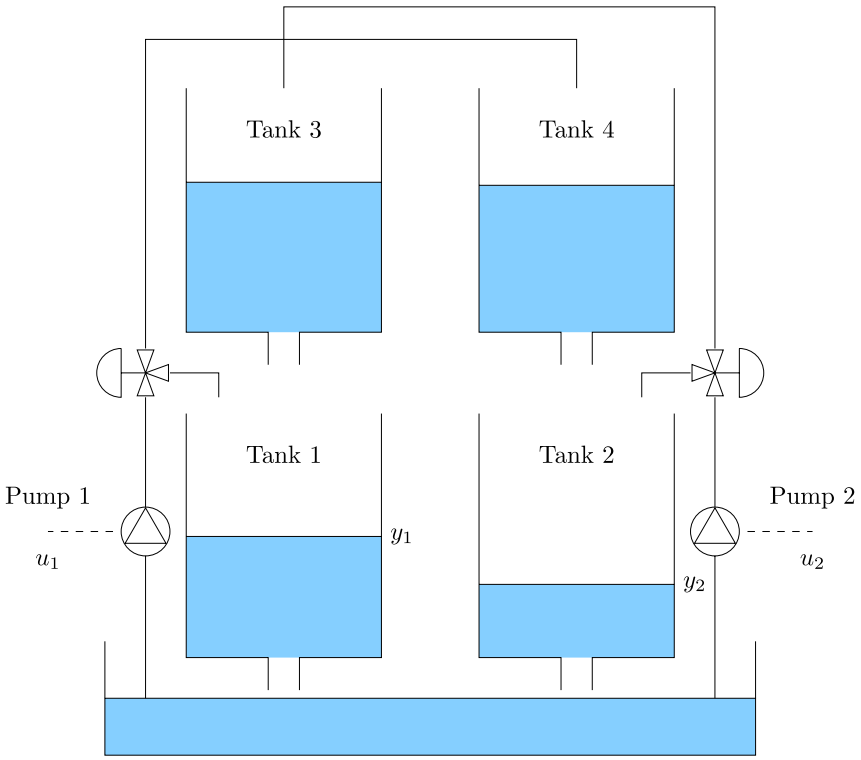


Figure 5.5.1: Quadruple tank plant system diagram.

real plant.

In order to generate data and test the controllers in closed-loop, a simulation model has been used. This model is based on the quadruple-tank process used in ([2]) and is given by the following differential equations:

$$\begin{aligned} S\dot{h}_1(t) &= -a_1\sqrt{2gh_1(t)} + a_3\sqrt{2gh_3(t)} + \gamma_a q_a(t), \\ S\dot{h}_2(t) &= -a_2\sqrt{2gh_2(t)} + a_3\sqrt{2gh_4} + \gamma_b q_b(t), \\ S\dot{h}_3(t) &= -a_3\sqrt{2gh_3(t)} + (1-\gamma_b) q_b(t), \\ S\dot{h}_4(t) &= -a_4\sqrt{2gh_4(t)} + (1-\gamma_a) q_a(t), \end{aligned} \tag{5.18}$$

where h_i and a_i with $i \in \{1, \ldots, 4\}$ refer to the water level and the discharge constant of tank i, respectively, S is the cross section of the tanks, q_j and γ_j with $j \in \{a, b\}$ denote the flow and the ratio of the three-way valve of pump j, respectively, and g is the gravitational acceleration. In this simulation we use the parameters of the quadruple tank process benchmark presented in ([2]) shown in Table 5.1 including maximum and minimum level and flow limits.

The level and flow variables define the following vectors

$$\mathbf{h} = [h_1 \quad h_2 \quad h_3 \quad h_4]^\top, \qquad \mathbf{q} = [q_a \quad q_b]^\top.$$

In order to design the proposed controller, a database of $M = 1000$ candidate triplets $(\mathbf{h}_q, \mathbf{q}_q, \mathbf{h}_{fq})$ is generated using the continuous-time model (5.18) and sampling every 5 seconds the solution,

$$\mathbf{h}_{fq} = \mathbf{h}(0) + \int_0^5 \frac{d\mathbf{h}(t)}{dt} dt + w_q$$

Table 5.1: Parameters of the plant.

	a_1	a_2	a_3	a_3	γ_a	γ_b
value	1.31e-4	1.51e-4	9.57e-4	8.82e-4	0.3	0.4
unit	m^2	m^2	m^2	m^2	-	-
	S	$h_{\max}$	$h_{\min}$	$q_{\max}$	$q_{\min}$	
value	0.06	1.3	0.3	3	0	
unit	m^2	m	m	m^3/h	m^3/h	

Table 5.2: Target references.

Reference r	2	3	4
h_1^r	0.30	0.50	0.90
h_2^r	0.30	0.75	0.75
h_3^r	0.30	0.30	1.06
h_4^r	0.30	1.18	0.58
q_a^r	1.11	2.20	1.53
q_b^r	1.35	1.36	2.54

with $\mathbf{h}(0) = \mathbf{h}_q$, $\mathbf{q}(t) = \mathbf{q}_q$ for all t. Each entry w_{qi} of vector w_q is a zero-mean, i.i.d. random variable with uniform distribution inside the interval $[-0.01, 0.01]$ meters that models both measurement errors and perturbations. The level $\mathbf{h}_q$ and flow values $\mathbf{q}_q$ in each triplet are generated randomly between the maximum and minimum levels with a uniform distribution.

A tracking experiment is defined where a set of reference changes in the levels of the tanks, must be followed by manipulating the inlet flows. Each reference is defined by a target equilibrium point $(\mathbf{h}_r, \mathbf{q}_r)$ where

$$\mathbf{h}^r = [h_1^r \quad h_2^r \quad h_3^r \quad h_4^r]^\top, \qquad \mathbf{q}^r = [q_a^r \quad q_b^r]^\top,$$

with $r \in \{2, 3, 4\}$.

The initial levels in the experiment are 0.65 for all tanks. Reference changes occur every 3000 seconds. The initial reference is $(\mathbf{h}^2, \mathbf{q}^2)$, which changes to $(\mathbf{h}^4, \mathbf{q}^4)$, then to $(\mathbf{h}^3, \mathbf{q}^3)$ and then back to $(\mathbf{h}^2, \mathbf{q}^2)$. The references are the same used in the benchmark although in different order. The level and flow values of each reference are given in Table 5.2.

First, the proposed unconstrained explicit controller is applied. To this end, for each reference r a different controller is designed defining as state, input and prediction variables the deviation of the levels and flows from the corresponding target reference; that is,

$$x = \mathbf{h} - \mathbf{h}^r, \quad u = \mathbf{q} - \mathbf{q}^r, \quad z = \mathbf{h}_f - \mathbf{h}^r. \tag{5.19}$$

By applying the change of variables of (5.19), for each reference r a different database including the M candidates (x_q^r, u_q^r, z_q^r) is calculated. These data are used to solve the finite horizon optimal control problem explicitly using a dynamic programming approach.

The controller design parameters are $Q = I$, $R = 0.01I$, $P = 10I$, $N = 4$ and $\beta_q = 1 \quad \forall\, q \in \{1, \ldots, M\}$. These parameters are used for the three different references. With a slight abuse of notation, we denote by P^r and K^r the matrices P_N and K_N for each

reference r with $N = 4$. This implies that at each sampling time k, the controller is defined as

$$\mathbf{q}(k) = \mathbf{q}^r + U_q^r K^r (\mathbf{h}(k) - \mathbf{h}^r),$$

where r is the appropriate reference.

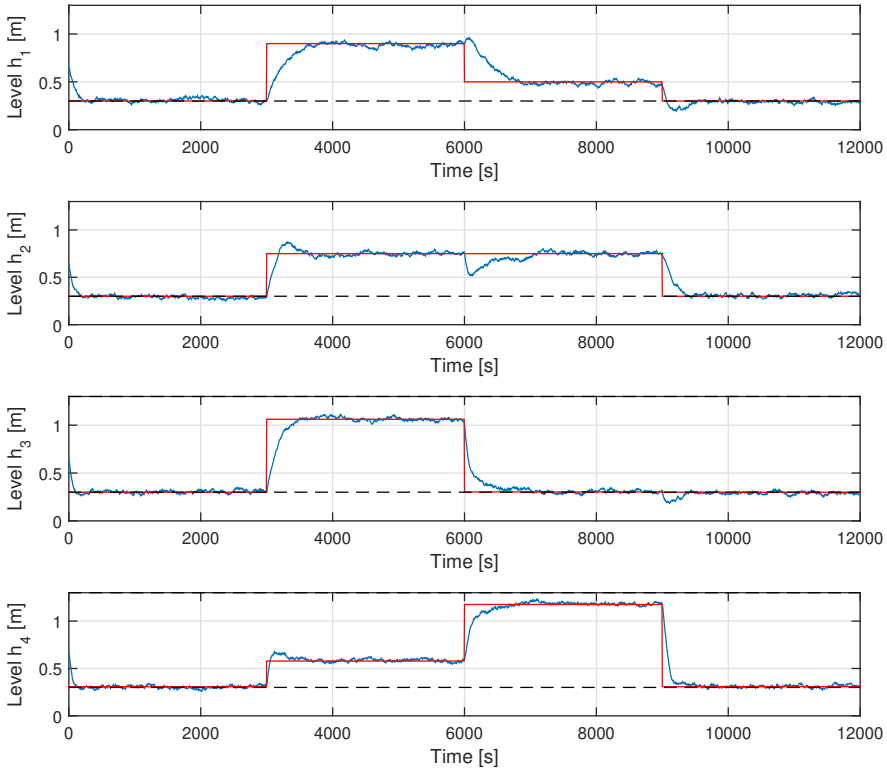


Figure 5.5.2: Tank levels with unconstrained explicit controller.

To carry out the simulation, the nonlinear model is used, including a bounded additive random perturbation in each level with uniform distribution inside the interval $[-0.01, 0.01]$. Figures 5.5.2 and 5.5.3 show the level and flow closed-loop trajectories for the unconstrained explicit controllers. It can be seen that the controller does not satisfy the input constraints, in particular, every time there is a reference change. In addition, from time 9000 to 9500 seconds, the level of the third tank falls below 0.3 meters during the transient.

Next, we apply the constrained predictive controller with prediction horizon one. For each reference, a different optimization problem is defined using the corresponding candidates (x_q^r, u_q^r, z_q^r) error variables and P^r as terminal cost in (5.13). In this simulation, the set of candidates taken into account at each sampling time k is obtained by selecting the 250 closest candidates to the current state x. In addition, for each candidate $q \in G(x))$, the parameter $\beta_q(x)$ is equal to the squared distance between the candidate and the current state, that is,

$$\beta_q(x) = (x(k) - x_q)^\top (x(k) - x_q)$$

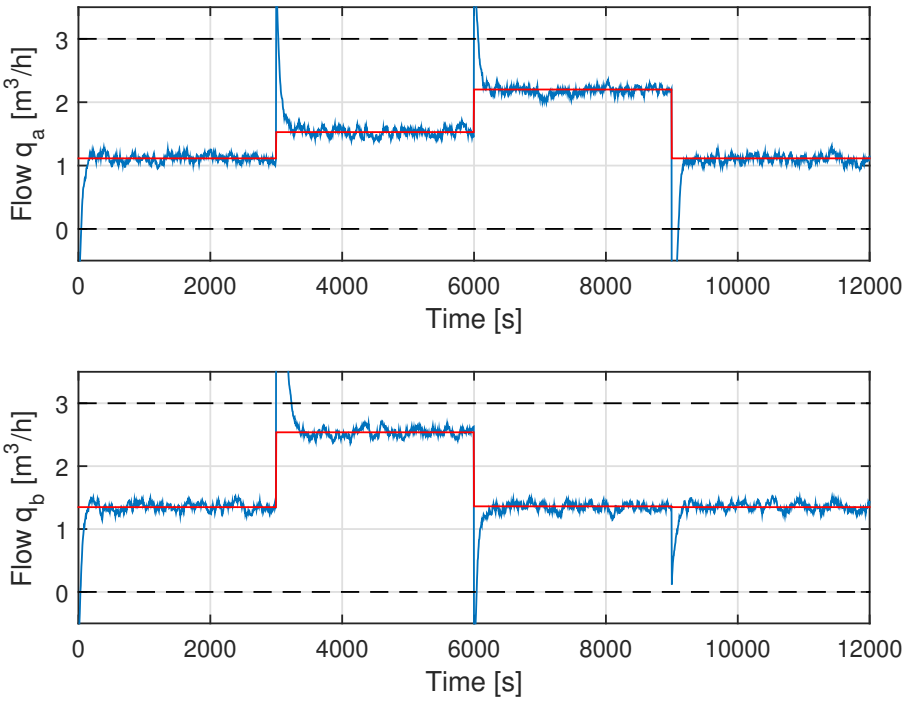


Figure 5.5.3: Pump flows with unconstrained explicit controller.

Limiting the set of candidates and penalizing the weights of candidates far away from the current state may provide more smooth predictions ([92, 20]).

Figures 5.5.4 and 5.5.5 show the level and flow closed-loop trajectories for the constrained implicit controller.

It can be seen that the level constraints are taken into account and constraint violations are reduced when compared with the unconstrained controller.

The error of the direct weight optimization prediction for all the tank levels throughout this simulation is lower than 0.1 meters. This error depends on the deviation of the levels and flows from the corresponding target references considered. The mean value of the errors is lower than 0.001 meters. Finally, the same simulation has been carried out with a MPC based on the nonlinear model of the system to compare the performance. The difference between the predictive controller with horizon one based on direct weight optimization and the nonlinear controller is below 5%, although the number of sampling times in which the minimum level constraints are violated is lower in the simulation carried out by using the nonlinear MPC controller.

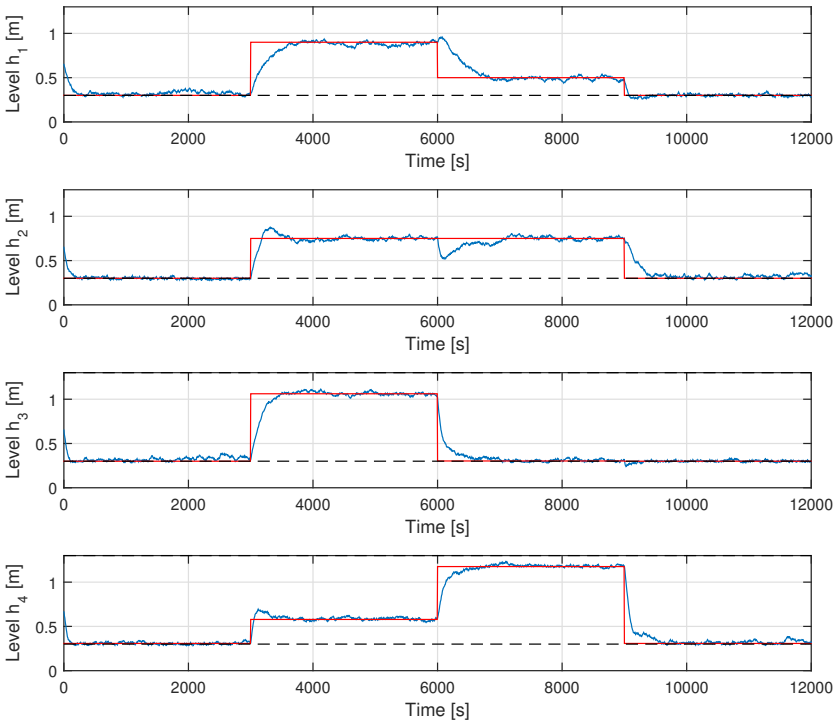


Figure 5.5.4: Tank levels with constrained implicit controller.

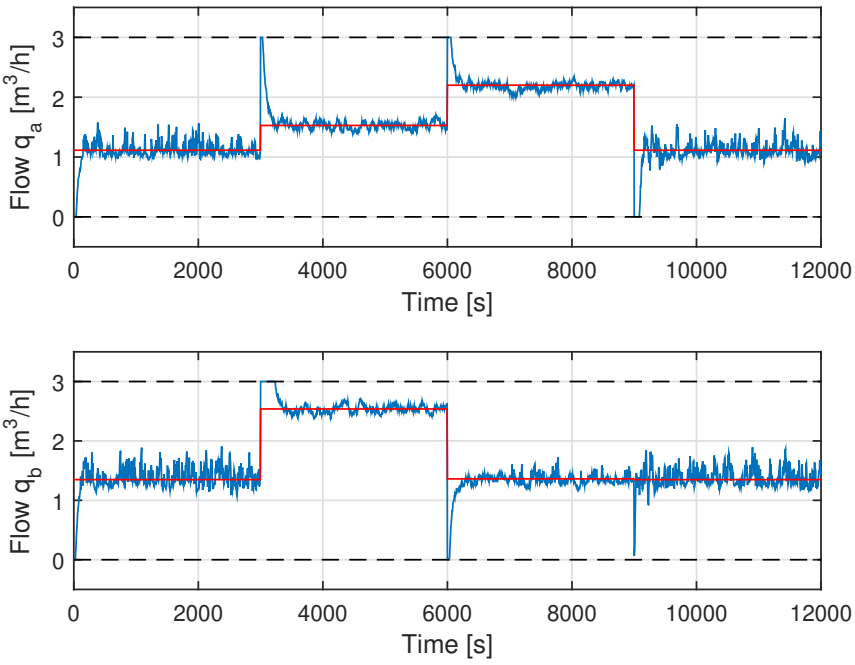


Figure 5.5.5: Pump flows with constrained implicit controller.

Chapter 6

Learning-based predictive control for MIMO systems

Noise and disturbances affect every real application. White noise measured in all electronic devices, noise induced by different actuators of the system, disturbances created by external agents or by random variables of the system are example of noise presence in real plants. Disturbances can affect the performance of controllers, even to reach the extreme case of becoming unstable in closed-loop. In MPC, uncertainties produced by discrepancies between the model and the system have an adverse effect in the controller. To avoid this, a common assumption is to consider a perfect match between the model and the plant with a given bound on the uncertainty, typically additive. Moreover, full state measurement is not an usual quality of real systems which poses problems that aggravate uncertainties such as setting the model order or estimating the state length.

Set membership identification [102] can be used to derive multi-step prediction models and for quantifying their associated uncertainty bounds. Specifically tailored robust MPC controllers have also been proposed for regulation [104], which are able to include explicitly multi-step predictors and prediction error bounds into the MPC controller formulation. Recently, a novel approach has been proposed for linear systems, that addresses model identification, model mismatch quantification, and MPC design in a unitary and consistent fashion [103]. These contributions focus on single input and single output (i.e., SISO) systems, and they have been tested on linear academic examples taken from the related literature.

Following this line of work, in this chapter, we present a joint work with Enrico Terzi, Marcello Farina, Lorenzo Fagiano and Ricardo Scattolini from the *Politecnico di Milano* carried out during an academic stay, in which the issue of robustness in data based predictive control is considered. In particular, a learning-based approach for robust predictive control design for multi-input multi-output (MIMO) linear systems is presented. This joint work is an extension of the results presented in [103] and has been tested on a realistic benchmark example, i.e., the quad-tank system [2]. As a first step, we consider the extension to linear MIMO systems, testing the approach on a linearised system model. Secondly, we discuss about some issues and possible solutions arising in the application of our approach on nonlinear systems and we test the control scheme on the nonlinear simulator. Notably, the uncertainty models are identified based on regressors built from

data of the nonlinear system, thus ensuring robustness of the controller also on the plant tested. Finally, we introduce a modification in the MPC controller that is able to compensate for the static mismatch between the (linear) model and the (nonlinear) plant. The extension of the approach to non-square systems is also discussed.

6.1 Problem formulation

Consider a discrete-time, linear time invariant, MIMO system of order n with input $u \in \mathbb{R}^{n_u}$, output $z \in \mathbb{R}^{n_y}$ and measured output $y \in \mathbb{R}^{n_y}$ described by the autoregressive equations:

$$\begin{array}{rcl} z(k+1) & = & {\theta_z^{(1)}}^T \phi_z^{(1)}(k) + v(k) \\ y(k) & = & z(k) + d(k), \end{array} \tag{6.1}$$

where $v \in \mathbb{R}^{n_y}$ is an additive process disturbance, $d \in \mathbb{R}^{n_y}$ is an additive measurement noise and $\phi_z^{(1)}(k) \in \mathbb{R}^{(n_y+n_u)n}$ is the regressor defined as:

$$\begin{array}{rcl} \phi_z^{(1)}(k) & = & [z^T(k), \ldots, z^T(k-n+1), u^T(k-1), \ldots, \\ & & u^T(k-n+1), u^T(k)]^T \end{array}$$

The matrix $\theta_z^{(1)}$ contains the real system parameters, that are unknown, as well as the order of the system n.

Assumption 7 (System and signals)**.**

- *System* (6.1) *is asymptotically stable*
- $u(k) \in \mathbb{U} \subset \mathbb{R}^{n_u} \, \forall \, k \in \mathbb{Z}$, *where* $\mathbb{U}$ *is a compact and convex set.*
- *the process disturbances* $v(k)$ *and the output noises* $d(k)$ *are bounded, i.e.*

$$|v_i(k)| \leq \bar{v}_i \quad |d_i(k)| \leq \bar{d}_i \quad \forall \, k \in \mathbb{Z}, \, \forall i = 1, \ldots, n_y$$

  where $\bar{d}_i > 0$ *are known and* $\bar{v}_i > 0$ *are possibly not known (and assumed unknown in the sequel).*

In this chapter, for simplicity, we also assume that the system is square (i.e., $n_u = n_y$) and that the input-output static gain matrix (denoted μ) is invertible.

A dataset collected from the plant is available, composed of N_s input-output pairs (u, y), and it is assumed that the input signal in the database excites all the system modes to ensure its identifiability.

The problem addressed in this chapter consists of designing a theoretically sound robust model predictive controller based on identified multi-step prediction models, endowed with a tight prediction error bound estimated from data.

6.2 Tracking MPC with learned models

In this section we extend the procedure for the design of an MPC robust controller for tracking proposed in [103] to MIMO systems.

6.2.1 Available models

The proposed approach requires the definition of a state-space simulation model with state $X(k) = [z^T(k), \dots, z^T(k-o+1), u^T(k-1), \dots, u^T(k-o+1)]^T \in \mathbb{R}^{(n_y+n_u)o-n_u}$ and dynamics

$$\begin{aligned} X(k+1) &= AX(k) + Bu(k) + Mw(k) \\ z(k) &= CX(k) \\ y(k) &= z(k) + d(k) \end{aligned} \tag{6.2}$$

where o (see later) is selected in the identification phase and the entries $w_i(k)$ of $w(k) \in \mathbb{R}^{n_y}$ are bounded, i.e., $|w_i(k)| \leq \bar{w}_i$. Note that $w(k)$ accounts for various sources of uncertainty, e.g., the model mismatch with the real system and the process disturbances. Also, our approach requires the definition of optimal predictors of $z(k+p)$ (denoted $z_p(k)$), for $p = 1, \dots, \bar{p}$, based on the present system state and on the future control actions $U(k) = \begin{bmatrix} u(k)^T & \dots & u(k+\bar{p})^T \end{bmatrix}^T$:

$$z_p(k) = C_p X(k) + D_p U(k) \tag{6.3}$$

Model (6.2) will be defined through a suitable learning phase. Also, the peculiarity of our approach lies in the fact that (6.3) will not be defined by iterating (6.2) p times, but by means of dedicated identification steps, in order to optimize their multi-step predictive potentialities.
For consistency of notation we define $C_0 = C$ and $D_0 = 0_{n_y,(\bar{p}+1)n_u}$ such that we can write $z(k) = z_0(k) = C_0 X(k) + D_0 U(k)$.
A dedicated observer is designed, that includes the estimate $\hat{w}(k)$ of the disturbance w, derived later on.

$$\hat{X}(k+1) = A\hat{X}(k) + Bu(k) + M\hat{w}(k) + L(y(k) - C\hat{X}(k)) \tag{6.4}$$

$\hat{X}(k)$ is the estimated state and the matrix L is chosen such that the closed-loop matrix $(A - LC)$ is Schur stable. We also define the nominal dynamic system as

$$\bar{X}(k+1) = A\bar{X}(k) + B\bar{u}(k) + M\hat{w}(k) \tag{6.5}$$

where

$$u(k) = \bar{u}(k) + K(\hat{X}(k) - \bar{X}(k)) \tag{6.6}$$

The gain K is defined in such a way that the closed-loop transition matrix $A + BK$ is Schur stable. The corresponding nominal outputs are, for all $p = 1, \dots, \bar{p}$

$$\bar{z}_p(k) = C_p \bar{X}(k) + D_p \bar{U}(k) \tag{6.7}$$

where $\bar{U}(k) = \begin{bmatrix} \bar{u}^T(k) & \dots & \bar{u}^T(k+\bar{p}) \end{bmatrix}^T$. We finally define $\bar{z}(k) = C_0 \bar{X}(k) = \bar{z}_0(k)$. In line with [69], the optimization problem will be formulated by regarding the nominal model (6.5), while the displacement of the real variables $X(k)$, $u(k)$ with respect to $\bar{X}(k)$, $\bar{u}(k)$ will be considered for constraint tightening purposes.

6.2.2 Definition of the cost function

The overall goal of the proposed method is to track the reference output z_{goal}. However, for feasibility purposes, in the optimization problem the reference set point is z_{ref}, which in turn is defined as a further optimization variable. Assuming that a reliable (invertible) estimate $\hat{\mu}$ of the system static gain matrix μ is available, the corresponding input and state references are

$$u_{\text{ref}}(k) = \hat{\mu}^{-1} z_{\text{ref}}(k), \quad X_{\text{ref}}(k) = N z_{\text{ref}}(k) \tag{6.8}$$

where $N = \begin{bmatrix} \mathbf{1}_o \otimes I_{n_y} \\ \mathbf{1}_{o-1} \otimes \hat{\mu}^{-1} \end{bmatrix}$. As a result, the estimate $\hat{w}(k)$ of $w(k)$ is defined according to the steady-state expression $X_{\text{ref}}(k) = AX_{\text{ref}}(k) + Bu_{\text{ref}}(k) + M\hat{w}(k)$, i.e.,

$$\hat{w}(k) = \eta_{zw} z_{\text{ref}}(k) \tag{6.9}$$

where $\eta_{zw} = M^T \left[(I_{(n_y+n_u)o-n_u} - A)N - B\hat{\mu}^{-1} \right]$. Moreover, for consistency, the term $\hat{w}(k)$ will be forced to be bounded, i.e., $|\hat{w}_i(k)| \leq \bar{w}_i$ for all $i = 1, \ldots, n_y$ through dedicated constraints in the optimization problem.
From this we can define, $\forall p \in [0, \bar{p}]$, the consistent set point for each p-steps ahead prediction model as

$$z^p_{\text{ref}}(k) = \begin{bmatrix} C_p & D_p \end{bmatrix} \begin{bmatrix} X_{\text{ref}}(k) \\ \mathbf{1}_{\bar{p}+1} \otimes u_{\text{ref}}(k) \end{bmatrix} \tag{6.10}$$

The cost function to be minimized at each step k is

$$\begin{aligned} J(k) = \sum_{p=0}^{\bar{p}} & \left(\|\bar{z}_p(k) - z^p_{\text{ref}}(k)\|^2_{Q_p} + \|\bar{u}(k+p) - u_{\text{ref}}(k)\|^2_{R_p} \right) \\ & + \|\bar{X}(k+\bar{p}+1) - X_{\text{ref}}(k)\|^2_P + \sigma \|z_{\text{ref}}(k) - z_{\text{goal}}\|^2 \end{aligned} \tag{6.11}$$

where $\bar{X}(k+\bar{p}+1)$ is obtained by iterating the unperturbed state equation (6.5) $\bar{p}+1$ times, i.e.,

$$\bar{X}(k+\bar{p}+1) = A^{\bar{p}+1}\bar{X}(k) + \Gamma \bar{U}(k) + \Gamma_w(\mathbf{1}_{\bar{p}+1} \otimes \hat{w}(k)) \tag{6.12}$$

Also, $\Gamma = \begin{bmatrix} A^{\bar{p}}B & \ldots & B \end{bmatrix}, \Gamma_w = \begin{bmatrix} A^{\bar{p}}M & \ldots & M \end{bmatrix}$, The weights Q_p, R_p, P, and $\sigma > 0$ are defined to guarantee convergence properties, see [103] for details.

6.2.3 Definition of the tightened constraints

Suitable tightened input and output constraints are imposed on variables $\bar{u}(k)$ and $\bar{z}(k) = C\bar{X}(k)$

$$\bar{u}(k) \in \bar{\mathbb{U}}, \quad \bar{z}(k) \in \bar{\mathbb{Z}}, \quad \hat{w}(k) \in \mathbb{W}, \tag{6.13}$$

where $\mathbb{W} = \{w \in \mathbb{R}^{n_y} : |w_i| \leq \bar{w}_i, i = 1, \ldots, n_y\}$ and the sets $\bar{\mathbb{U}}$ and $\bar{\mathbb{Z}}$ are closed and satisfy:

$$\bar{\mathbb{U}} \subseteq \mathbb{U} \ominus K\bar{\mathbb{E}} \tag{6.14a}$$

$$\bar{\mathbb{Z}} \subseteq \mathbb{Z} \ominus C(\bar{\mathbb{E}} \oplus \hat{\mathbb{E}}) \tag{6.14b}$$

Set $\hat{\mathbb{E}}$ is robust positively invariant (RPI) [85] for the system

$$\hat{e}(k+1) = (A - LC)\hat{e}(k) + M(w(k) - \hat{w}(k)) - Ld(k) \tag{6.15}$$

Note that (6.15) describes the evolution of $\hat{e}(k) = X(k) - \hat{X}(k)$. Also, $\bar{\mathbb{E}}$ is the RPI set for

$$\bar{e}(k+1) = (A+BK)\bar{e}(k) + LC\hat{e}(k) + Ld(k) \tag{6.16}$$

where $\bar{e}(k) = \hat{X}(k) - \bar{X}(k)$. To define the terminal constraint set we consider the following auxiliary control law

$$\bar{u}(k) = u_{\text{ref}}(k) + K(\bar{X}(k) - X_{\text{ref}}(k)) \tag{6.17}$$

To compute an invariant set where $(\bar{X}(k), z_{\text{ref}})$ must lie in order to guarantee that constraints (6.13) are verified for all k, we need to define the Maximal Output Admissible Set (MOAS, see [42]) $\mathbb{O}$ for the system

$$\begin{bmatrix} \bar{X}(k+1) \\ z_{\text{ref}}(k+1) \end{bmatrix} = \underbrace{\begin{bmatrix} A+BK & BM_2 + M\eta_{zw} \\ 0_{n_y,(n_u+n_y)o-n_u} & I_{n_y} \end{bmatrix}}_{\mathcal{F}} \begin{bmatrix} \bar{X}(k) \\ z_{\text{ref}}(k) \end{bmatrix} \tag{6.18}$$

that is subject to the auxiliary control law (6.17), where $M_2 = \hat{\mu}^{-1} - KN$. The triplet $(\bar{u}(k), \bar{z}(k), \hat{w}(k))$ is computed as

$$\begin{bmatrix} \bar{z}(k) \\ \bar{u}(k) \\ \hat{w}(k) \end{bmatrix} = \underbrace{\begin{bmatrix} C & 0_{n_y,n_y} \\ K & M_2 \\ 0_{n_y,(n_u+n_y)o-n_u} & \eta_{zw} \end{bmatrix}}_{\mathcal{C}} \begin{bmatrix} \bar{X}(k) \\ z_{\text{ref}}(k) \end{bmatrix} \tag{6.19}$$

In the following we will use the invariant, polytopic inner approximation $\mathbb{O}_\epsilon$ to the MOAS.

6.2.4 The optimization problem

The optimization problem, to be solved at each time instant $k \geq 0$, reads

$$\min_{\bar{X}(k), \bar{U}(k), z_{\text{ref}}(k)} J(k) \tag{6.20a}$$

subject to (6.5), (6.7), (6.8), (6.9), (6.10) and

$$\hat{X}(k) - \bar{X}(k) \in \bar{\mathbb{E}} \tag{6.20b}$$

Also, $\forall p \in [0, \bar{p}]$

$$\bar{u}(k+p) \in \bar{\mathbb{U}}, \quad \bar{z}(k+p) = C_0\bar{X}(k+p) \in \bar{\mathbb{Z}}, \quad \hat{w}(k) \in \mathbb{W} \tag{6.20c}$$

Finally, as a terminal constraint, the following must be fulfilled

$$\begin{bmatrix} \bar{X}(k+\bar{p}+1) \\ z_{\text{ref}}(k) \end{bmatrix} \in \mathbb{O}_\epsilon \tag{6.20d}$$

If available, the solution to the optimization problem (6.20) is denoted $\bar{X}(k|k), \bar{U}(k|k) = (\bar{u}(k|k), \ldots, \bar{u}(k+\bar{p}|k)), z_{\text{ref}}(k|k)$, and $u(k)$ in (6.6) is applied to the system according to the receding horizon principle. Theorem 1 in [103] guarantees convergence and recursive feasibility.

6.3 Learning multi-step models

6.3.1 Definition of multistep prediction models in normal form

In this section we discuss how to obtain multistep models and uncertainty bounds by adapting the original algorithm in [102], [103] in case of MIMO systems.

In particular, we first note that, by iteration of (6.1), the p-steps ahead output value of the real system can be computed. In particular, defining the sequence $\mathcal{V}(k) = [v^T(k), \dots, v^T(k+p-1)]^T$ and the extended regressor vector, containing also future inputs up to time $k+p-1$, $\phi_z^{(p)}(k) =$

$$\begin{aligned}=&[z^T(k), \dots, z^T(k-n+1), u^T(k-1), \dots, \\ &u^T(k-n+1), u^T(k), u^T(k+1), \dots, u^T(k+p-1)]^T\end{aligned}$$

it is possible write:

$$z(k+p) \;=\; {\theta_z^{(p)}}^T \phi_z^{(p)}(k) + {\theta_v^{(p)}}^T \mathcal{V}(k) \tag{6.21}$$

In (6.21) the matrices $\theta_z^{(p)}$ and $\theta_v^{(p)}$ are polynomial combinations of the entries of $\theta_z^{(1)}$, and they are thus unknown. By introducing the vector $\mathcal{D}(k) = [d^T(k), d^T(k+1), \dots, d^T(k+\bar{p}-1)]^T$, expression (6.21) can be re-written as a function of the regressor $\phi_y^{(p)}(k)$, which corresponds to $\phi_z^{(p)}(k)$ where z samples have been replaced by y ones. The resulting expression reads:

$$z(k+p) \;=\; {\theta_y^{(p)}}^T \phi_y^{(p)}(k) + \underbrace{{\theta_d^{(p)}}^T \mathcal{D}(k) + {\theta_v^{(p)}}^T \mathcal{V}(k)}_{\mathcal{H}(k)} \tag{6.22}$$

Inspired by the form of (6.22), that provides directly the p-steps ahead value of output vector z from $\phi_y^{(p)}(k)$, we select an independent model to predict the output vector at each value of $p = 1, \dots, \bar{p}$. Each one of these predictors, that we term "multistep", is of the following form:

$$\hat{z}(k+p) \;=\; {\hat{\theta}^{(p)}}^T \tilde{\phi}_y^{(p)}(k) \tag{6.23}$$

In (6.23) ${\hat{\theta}^{(p)}}^T$ is a matrix $\in \mathbb{R}^{n_y \times (n_y+n_u)o+n_u(p-1)}$, where o is the chosen model order, so that $\tilde{\phi}_y^{(p)}(k) = [y^T(k), \dots, y^T(k-o+1), u^T(k-1), \dots, u^T(k-o+1), u^T(k), u^T(k+1), \dots, u^T(k+p-1)]^T$. Note that, given the definition of $\tilde{\phi}_y^{(p)}(k)$, the matrix ${\hat{\theta}^{(p)}}^T$ can be naturally partitioned into submatrices referred to variable y (denoted with $\hat{\theta}_Y^{(p)}$), to past inputs (denoted with $\hat{\theta}_{\bar{U}}^{(p)}$) and to current and future inputs (denoted with $\hat{\theta}_U^{(p)}$), so that $\hat{z}(k+p) = \begin{bmatrix} {\hat{\theta}_Y^{(p)}}^T & {\hat{\theta}_{\bar{U}}^{(p)}}^T & {\hat{\theta}_U^{(p)}}^T \end{bmatrix} \tilde{\phi}_y^{(p)}(k)$. Model (6.23), though, is not in a form that is suitable for the direct application of the identification algorithm in [103], which requires the model parameters to be in a vector.

We thus reformulate (6.23) in a more convenient way. In particular, considering that $\hat{\theta}_{\bullet i}^{(p)}, i = 1, \dots, n_y$ contains the set of parameters associated to the i-th output, we write

$$\hat{z}(k+p) = \underbrace{\begin{bmatrix} \tilde{\phi}_y^{(p)^T}(k) & 0 & \dots & 0 \\ 0 & \tilde{\phi}_y^{(p)^T}(k) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & \dots & 0 & \tilde{\phi}_y^{(p)^T}(k) \end{bmatrix}}_{\tilde{\Phi}_y^{(p)^T}} \underbrace{\begin{bmatrix} \hat{\theta}_{\bullet 1}^{(p)} \\ \hat{\theta}_{\bullet 2}^{(p)} \\ \vdots \\ \hat{\theta}_{\bullet n_y}^{(p)} \end{bmatrix}}_{\hat{\Theta}^{(p)}} \tag{6.24}$$

In equation (6.24) the predictor contains a unique vector of unknown parameters to be identified, and it thus fits the form required by the algorithm in [103], to which the reader is referred for details.

The outcome of the identification procedure consists, for each prediction step $p = 1, \dots, \bar{p}$ in:

- A nominal model vector, denoted with $\hat{\Theta}^{*(p)}$. It can be straightforwardly recast as in (6.23), and we denote this representation with:

$$\hat{\theta}^{*(p)^T} = \begin{bmatrix} \hat{\theta}_Y^{*(p)^T} & \hat{\theta}_{\bar{U}}^{*(p)^T} & \hat{\theta}_U^{*(p)^T} \end{bmatrix}$$

  .

- A global prediction error bound for each one of the system outputs, i.e. a vector $\hat{\tau}^{(p)}(\cdot) \in \mathbb{R}^{n_y}$ such that:

$$|z_j(k+p) - \hat{z}_j(k+p)| \leq \hat{\tau}_j^{(p)}(\hat{\Theta}^{*(p)}),$$
$$\forall k \in \mathbb{Z},\ j = 1, \dots, n_y$$

  These bounds are termed global since they depend only on the vector of parameters (i.e. the model) and not on the specific regressor. They are valid over a compact set of regressor trajectories of interest, and they enjoy asymptotic convergence properties under suitable assumptions on the informative content of the data collected in such a compact set, see [103] for details.

6.3.2 Definition of the control-oriented models

The models (6.2) and (6.3) are obtained by setting

$$A = \begin{bmatrix} \hat{\theta}_Y^{*(1)^T} & \hat{\theta}_{\bar{U}}^{*(1)^T} \\ I_{(o-1)n_y} \quad 0_{(o-1)n_y, n_y} & 0_{(o-1)n_y, (o-1)n_u} \\ 0_{(o-1)n_u, on_y} & 0_{n_u,(o-1)n_u} \\ & I_{(o-2)n_u} \quad 0_{(o-2)n_u, n_u} \end{bmatrix},$$

$$B = \begin{bmatrix} \hat{\theta}_U^{*(1)^T} \\ 0_{(o-1)n_y, n_u} \\ I_{n_u} \\ 0_{(o-2)n_u, n_u} \end{bmatrix}, C = \begin{bmatrix} I_{n_y} & 0_{n_y,(n_y+n_u)(o-1)} \end{bmatrix}, M = C^T$$

$$C_p = \begin{bmatrix} \hat{\theta}_Y^{*(p)^T} & \hat{\theta}_U^{*(p)^T} \end{bmatrix}, D_p = \begin{bmatrix} \hat{\theta}_{\bar{U}}^{*(p)^T} & 0_{n_y,(\bar{p}+1-p)n_u} \end{bmatrix}$$

.

The maximum amplitude of each of the components $\bar{w}_i$ is identified based on the prediction error bound $\hat{\tau}_j^{(p)}(\cdot)$ suitably employed, with arguments similar to those presented in [103], with special attention to the multivariable nature of the system. In particular, we obtain $\bar{w} = [\bar{w}_1, \ldots, \bar{w}_{n_y}]^T$ as a solution to optimization program (6.25) stated below. Specifically, $\forall p = 1, \ldots, \bar{p}$, $\hat{\Theta}^{*(1,p)}$ represents the 1 step nominal model iterated p times according to (6.2), $c \in \mathbb{R}^{1,n_y}$ is a weighting vector, and $E = \begin{bmatrix} I_{on_y} \\ 0_{(o-1)n_u, on_y} \end{bmatrix}$ is a selection matrix.

$$
\begin{aligned}
\bar{w} &= \arg \min_{w_i \in \mathbb{R}^+} c^T w \qquad (6.25)\\
&\sum_{i=0}^{p-1} \sum_{h=1}^{n_y} \left| C_{j\bullet} A^i M_{\bullet h} \right| w_h + \\
\text{s.t.} \quad & \\
&+ \sum_{i=1}^{on_y} |C_{j\bullet} A^p E_{\bullet i}| \bar{d}_{rem^*(i/n_y)} \quad \geq \hat{\tau}_j^{(p)}(\hat{\Theta}^{*(1,p)}) \\
&\forall j = 1, \ldots, n_y, \forall p = 1, \ldots, \bar{p}
\end{aligned}
$$

6.4 Numerical example

In this section we test the proposed approach on a quadruple-tank system. First, the method developed in the previous sections is applied to a linearized model of the system. Then, after a short discussion on the issues arising in the application of the proposed approach to a nonlinear system, we show the results obtained with a nonlinear simulator.

6.4.1 The quad-tanks case study

We consider the system showed in Figure 5.5.1. The inputs to the system are flowrates q_a and q_b, generated by two pumps. Tanks 1 and 4 are filled with flowrates $\gamma_a q_a$ and $(1-\gamma_a)q_a$, respectively, where $\gamma_a \in [0, 1]$. On the other hand, tanks 2 and 3 have, as input flowrates, $\gamma_b q_b$ and $(1-\gamma_b)q_b$, respectively, where $\gamma_b \in [0, 1]$. Tanks 3 and 4 discharge water into tanks 1 and 2, respectively.

The corresponding dynamical model is

$$
\begin{aligned}
S\dot{h_1}(t) &= -a_1\sqrt{2gh_1(t)} + a_3\sqrt{2gh_3(t)} + \gamma_a q_a(t), \\
S\dot{h_2}(t) &= -a_2\sqrt{2gh_2(t)} + a_4\sqrt{2gh_4} + \gamma_b q_b(t), \\
S\dot{h_3}(t) &= -a_3\sqrt{2gh_3(t)} + (1-\gamma_b)q_b(t), \\
S\dot{h_4}(t) &= -a_4\sqrt{2gh_4(t)} + (1-\gamma_a)q_a(t),
\end{aligned} \qquad (6.26)
$$

h_i is water level of the i-th tank (with $i = 1, \ldots, 4$), a_i is the discharge constant of the i-th tank, S is the cross section of the tanks, q_a and q_b denote the flowrates of pumps 1 and 2, respectively, and γ_a and γ_b are the aperture ratio of the three-way valve after pumps 1 and 2, respectively. g is the gravitational acceleration. The parameters of the plant are given in Table 5.1. Finally, the measurement noise is such that $|d_i(k)| \leq \bar{d}_i = 0.005$ for $i = 1, 2$.

We define $x = [h_1, h_2, h_3, h_4]^T$, $u = [q_a, q_b]^T$, $y = [h_1, h_2]^T$. Note that each of h_1 and h_2 is the output of a second order system. A linearized discrete-time model is obtained with sample time $T_s = 60$ s around the steady-state condition

$$x_{op} = [0.6016, 0.6097, 0.5950, 0.6343]^T$$

$$u_{op} = [1.6, 1.9]^T$$

$$y_{op} = [0.6016 \quad 0.6097]^T.$$

The input constraints are $q_a \in [0.1, 2.6]$ m^3/h and $q_b \in [0.4, 2.9]$ m^3/h, while the outputs must lie within the intervals $h_1 \in [0.1016, 1.1016]$ m and $h_2 \in [0.1097, 1.1097]$ m.

6.4.2 Control of the linearized model

Using data collected from the linearized model we applied the learning procedure described in Section 6.3, setting $o = 2$. In order to learn the uncertainty bound vector $\bar{w}$, the one-step predictor $\hat{\Theta}^{*(1)}$ is iterated according to (6.2), the related guaranteed error bound $\hat{\tau}^{(p)}(\hat{\Theta}^{*(1,p)})$ is computed and problem (6.25) is solved. A plot depicting the evolution of the bounds' amplitude as a function of the prediction horizon is shown in Figure 6.4.1. In this figure lines with marker "x" refer to output 2, while lines without a marker refer to output 1. Dashed lines represent term $\sum_{i=1}^{on_y} |C_{j\bullet}A^pE_{\bullet i}|\bar{d}_{rem^*(i/n_y)}$ accounting for the wrong initialization of the state containing y samples instead of z ones, dotted lines are the term $\sum_{i=0}^{p-1}\sum_{h=1}^{n_y} |C_{j\bullet}A^iM_{\bullet h}|\,\bar{w}_h$ accounting for the disturbance $\bar{w}$ integrated over time, line with circles refer to $\hat{\tau}_j^{(p)}(\hat{\Theta}^{*(1,p)}), j = 1, 2$ and solid lines are the overall error bound $\sum_{i=0}^{p-1}\sum_{h=1}^{n_y} |C_{j\bullet}A^iM_{\bullet h}|\,\bar{w}_h + \sum_{i=1}^{on_y} |C_{j\bullet}A^pE_{\bullet i}|\bar{d}_{rem^*(i/n_y)}$. Note that $rem^*(m/n)$ is the least positive remainder of the division m/n with $m, n \in \mathbb{N}$, however if m is multiple of n, $rem^*(m/n) = n$ will be considered rather than 0.

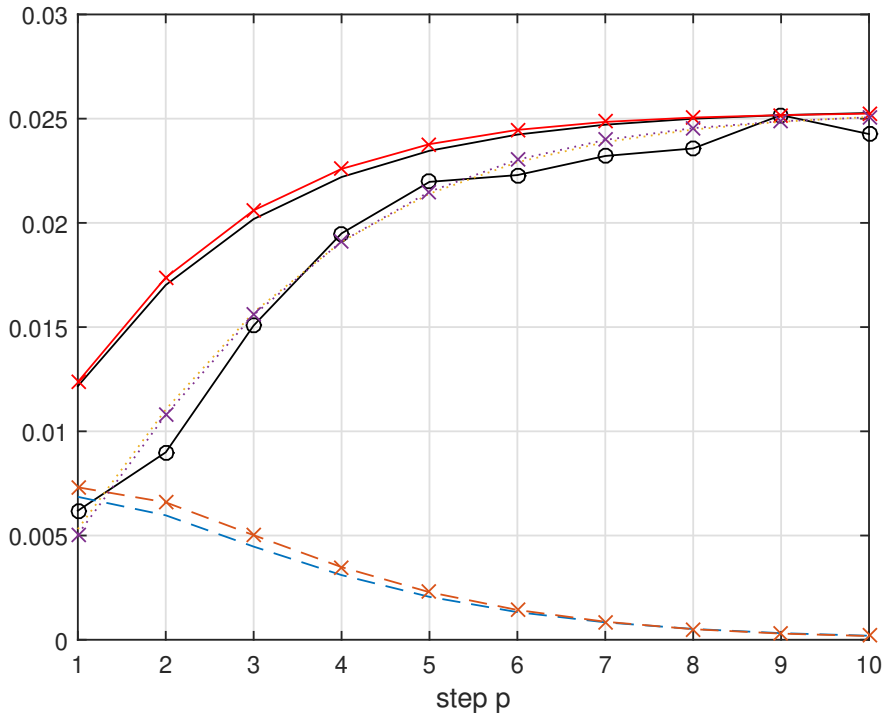


Figure 6.4.1: Trends of the bounds for linearised case.

The MPC controller has a control horizon $\bar{p} = 5$. The Luenberger observer gain L is selected as the Kalman filter stationary gain while the auxiliary control law gain K is selected as a LQ gain, both of them obtained solving the discrete-time algebraic Riccati equations of optimal control theory and with diagonal matrices:

$$Q = \begin{bmatrix} \gamma_x I_{on_y} & 0_{on_y,(o-1)n_u} \\ 0_{(o-1)n_u,on_y} & \gamma_u I_{(o-1)n_u} \end{bmatrix}, R = \gamma_u I_{n_y}$$

where $\gamma_x = 0.25$ and $\gamma_u = 0.1$ for the observer and $\gamma_x = 1$ and $\gamma_u = 0.1$ for the auxiliary control law.

Two piece-wise constant output reference signals are used. Specifically, the one for h_1 takes values $\{0.6016, 0.7, 0.4, 0.95, 1.3\}$, while the set-point for h_2 takes values $\{0.6097, 0.8, 0.4, 0.9, 1.3\}$. The closed-loop trajectories and the corresponding control actions are shown in Figure 6.4.2 and Figure 6.4.3, respectively.

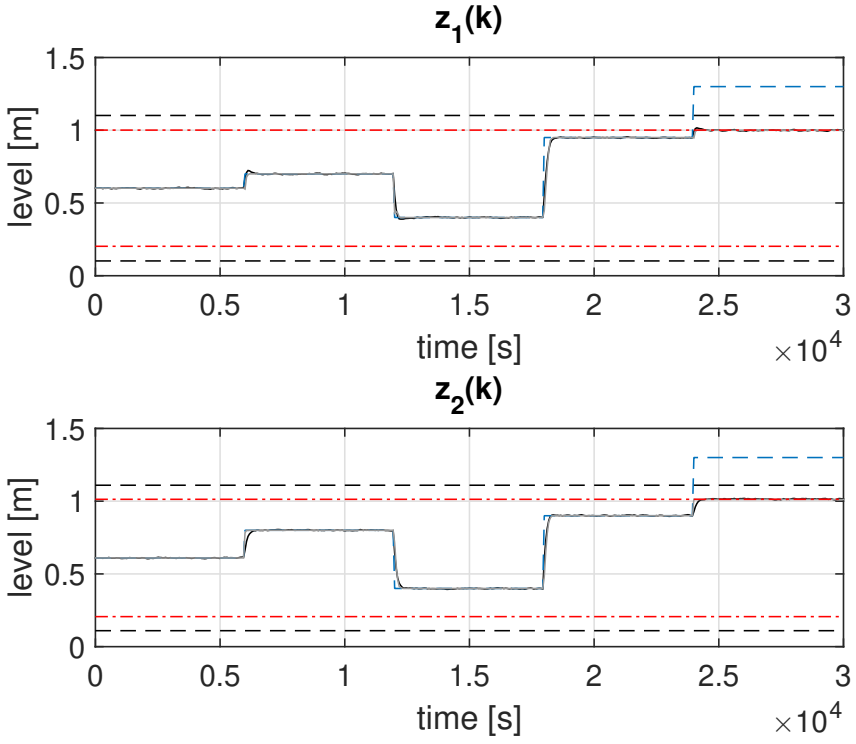


Figure 6.4.2: Outputs from the closed-loop simulation with linearized model.

The original (black dashed) and the tightened (red dashed-dotted) constraints are depicted in both figures. The inputs and their references are showed in solid black and grey, respectively. It can be observed that the reference z_{goal} (dashed) is successfully tracked by the outputs $z(k)$ (solid black).

6.4.3 Application to nonlinear MIMO systems

The method discussed in the previous sections has been applied also on the original nonlinear system, using linear models identified from data generated by the nonlinear simulator

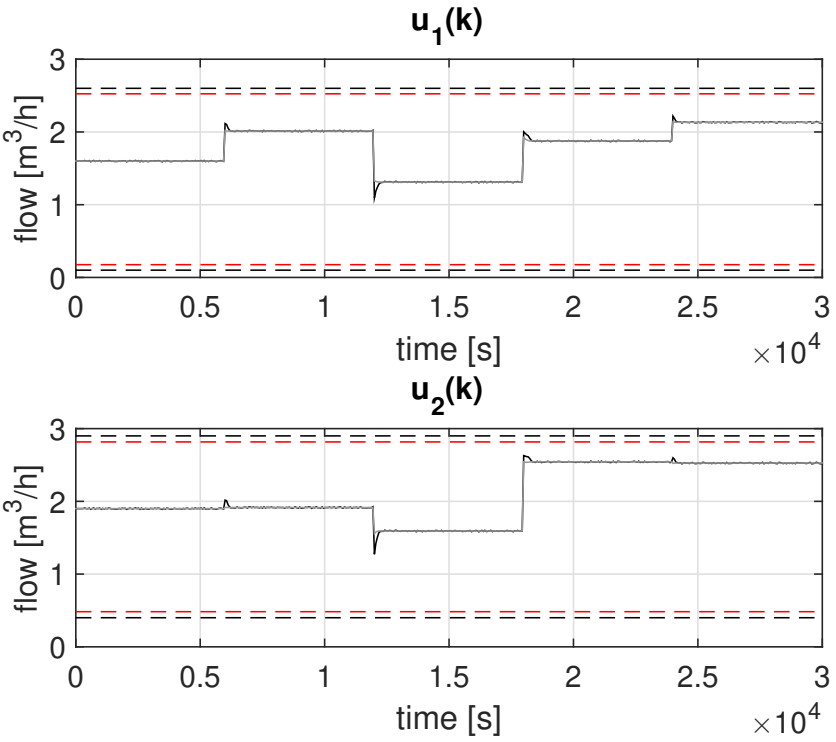


Figure 6.4.3: Control actions from the closed-loop simulation with linearized model.

(6.26). Since the control algorithm has been conceived with a focus on linear systems, a short discussion is due before to show the numerical results.

Identification and control issues in case of nonlinear systems

Using data generated by the nonlinear simulator (6.26), we have identified linear prediction models (6.2) and (6.3), together with the corresponding perturbation bounds $\bar{w}_i$, $i = 1, 2$. However, the main assumptions guaranteeing the soundness of the learning phase (see [103]) require that the used model class includes the model of the system generating the data: this assumption is clearly impossible to be verified in this setting. In fact, the set of regressors used in (6.21) does not include nonlinear functions of input and state variables. In our opinion, however, this has not caused any significant problem in the considered case study which, remarkably, does not display a complex nonlinear dynamics (e.g. multiple equilibria, limit cycles, chaotic behaviour). The mismatch between the linear model and the nonlinear system has indeed been accurately included thanks to the disturbance term $w(k)$, leading to satisfactory simulation results, especially as far as the constraints fulfillment is regarded.

However, in the control design phase a different problem has arised from the fact that the nonlinear static gain is not constant, contrarily to the linear case. This problem has been here addressed by modifying the cost function (6.11), and in particular the final additive term $\sigma\|z_{\text{ref}}(k) - z_{\text{goal}}\|^2$. The idea used here consists of replacing z_{goal} with $\hat{\mu}\mu_{NL}(z_{\text{goal}})^{-1}z_{\text{goal}}$, where $\mu_{NL}(z_{\text{goal}})$ is the nonlinear system input-output static gain, computed on the working conditions defined by z_{goal}.

Numerical results

For the sake of completeness, Figure 6.4.4 shows the trends of the bounds against the prediction horizon for the nonlinear case. Similar to figure 6.4.1, lines with marker "x" refer to output 2, while lines without a marker refer to output 1. In addition, dashed lines represent term $\sum_{i=1}^{on_y} |C_{j\bullet} A^p E_{\bullet i}| \bar{d}_{rem^*(i/n_y)}$ accounting for the wrong initialization of the state containing y samples instead of z ones, dotted lines are the term $\sum_{i=0}^{p-1} \sum_{h=1}^{n_y} |C_{j\bullet} A^i M_{\bullet h}| \bar{w}_h$ accounting for the disturbance $\bar{w}$ integrated over time, line with circles refer to $\hat{\tau}_j^{(p)}(\hat{\Theta}^{*(1,p)}), j = 1, 2$ and solid lines are the overall error bound $\sum_{i=0}^{p-1} \sum_{h=1}^{n_y} |C_{j\bullet} A^i M_{\bullet h}| \bar{w}_h + \sum_{i=1}^{on_y} |C_{j\bullet} A^p E_{\bullet i}| \bar{d}_{rem^*(i/n_y)}$.

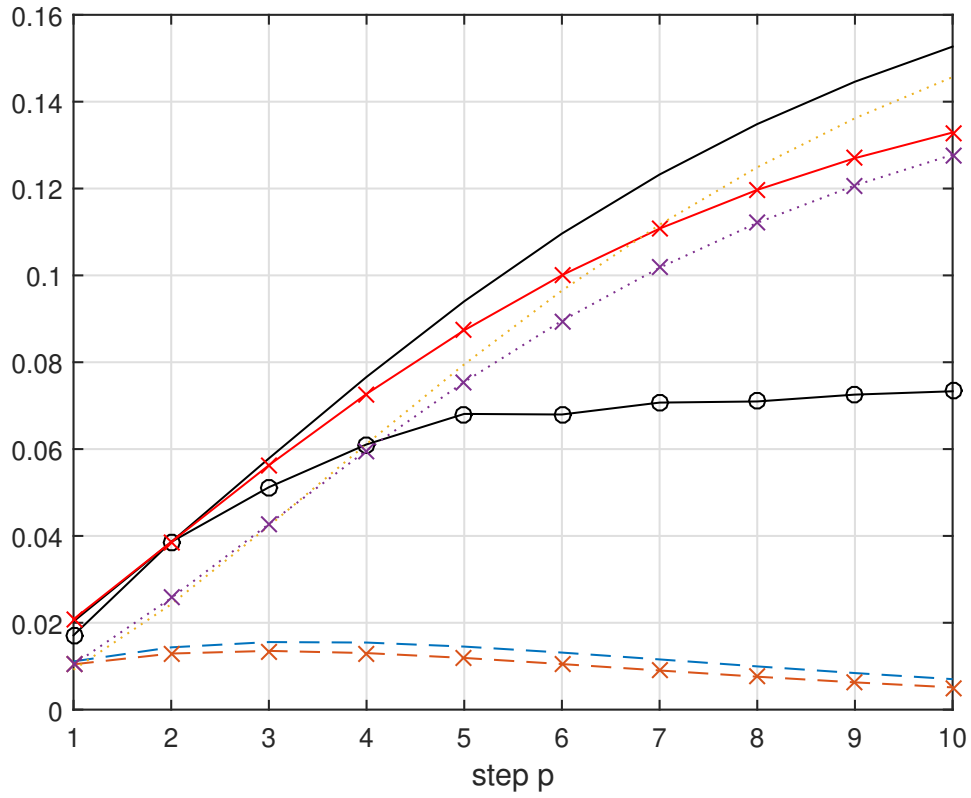


Figure 6.4.4: Trend of the bounds for nonlinear case.

Figure 6.4.5 and Figure 6.4.6 shows the closed-loop trajectories and inputs with the nonlinear model. Recall that the data are collected on the nonlinear plant, and thus the tightened constraints, used to enforce robustness, result to be more restrictive compared to the ones obtained with the linearized simulator. This is visible in Figure 6.4.5. Thanks to the modification of the final goal in the terminal cost, as described in this Section, the output signals are able to track the desired references, reducing the steady state error, without harming the guaranteed theoretical properties of the controller. This fact can be appreciated thanks to Figure 6.4.7, which shows a detailed comparison between the control scheme including (solid black) or not including (dashed-dotted red) such modification, using the same disturbance signals.

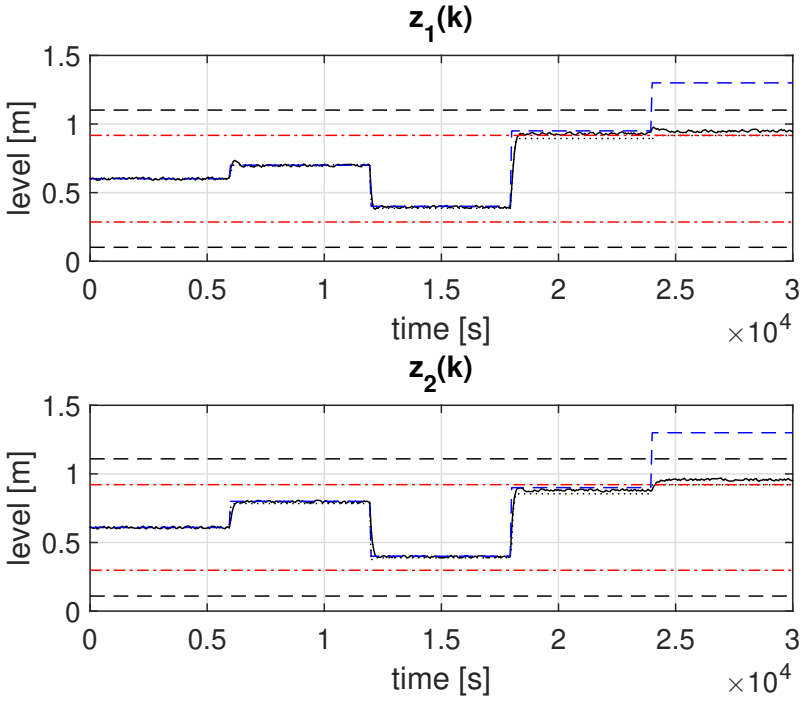


Figure 6.4.5: Outputs from the closed-loop simulation with the nonlinear simulation.

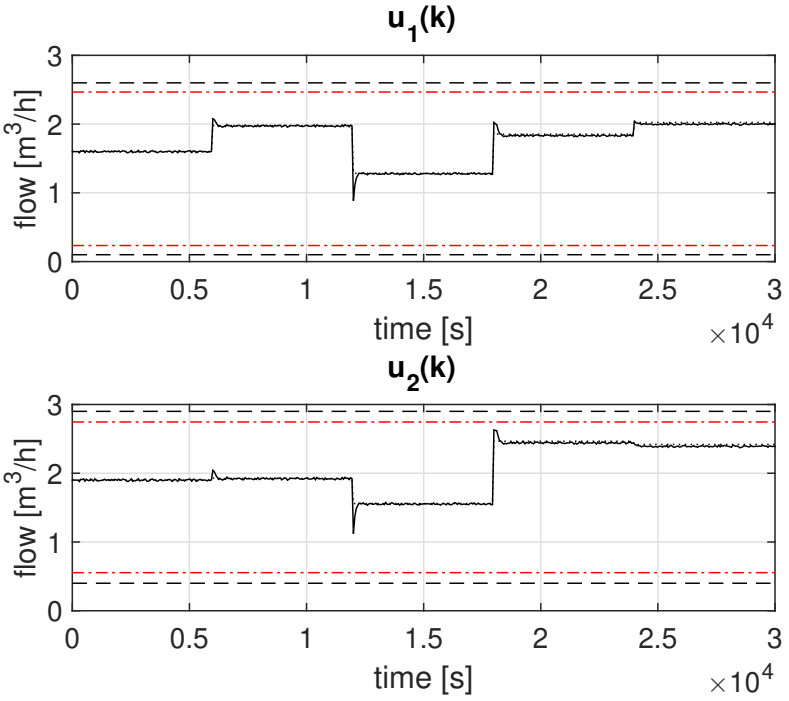


Figure 6.4.6: Control actions from the closed-loop simulation with the nonlinear simulation.

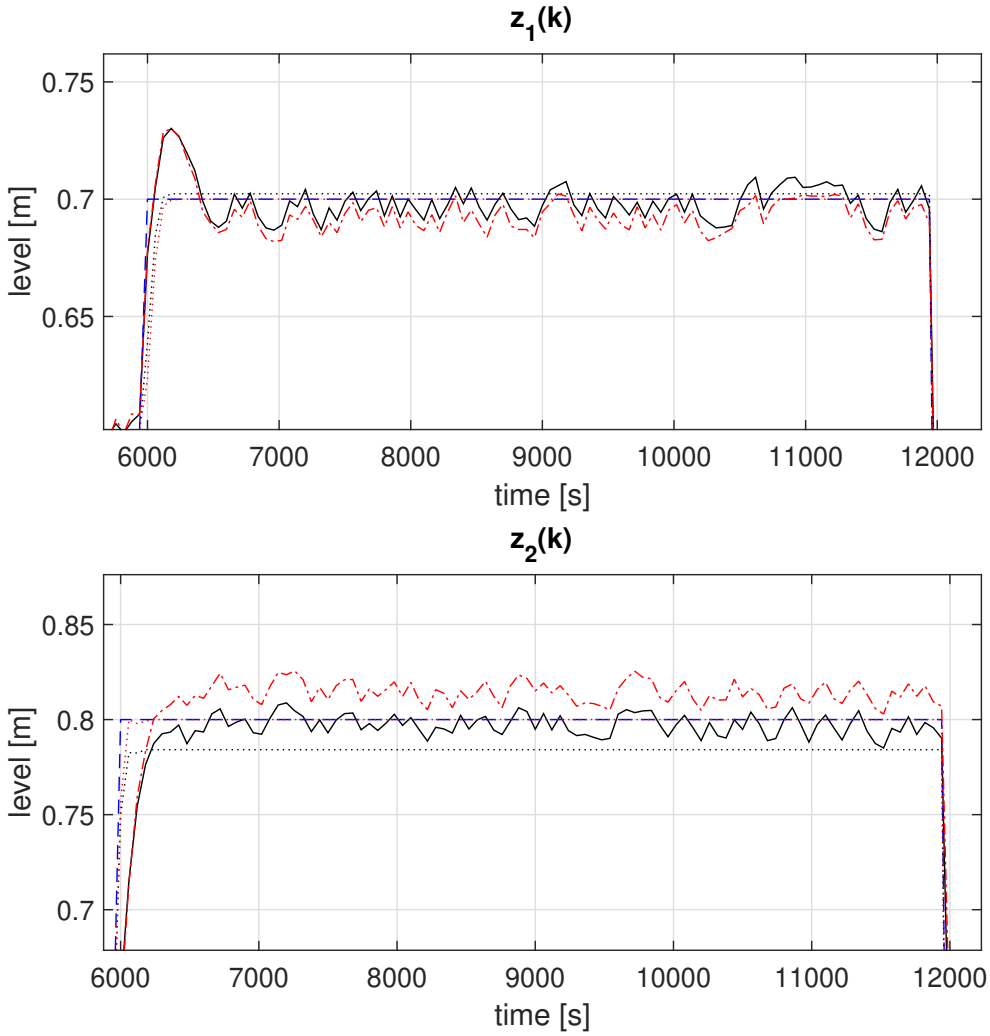


Figure 6.4.7: Outputs from the closed loop-simulation with the nonlinear simulation: focus on a steady state period

Chapter 7

Conclusions and future lines

This thesis has proposed new data based predictive control schemes which use past historian process data of the systems that have to be controlled. Large scale complex infrastructures motivate this research because while it is difficult to obtain accurate models appropriate for control, they offer in general large amounts of historic data. Most of the proposed data based predictive control approaches are based on direct weight optimization techniques which provide model free control laws with different goals. This class of inference methods are based on linear combinations of the available data and are appropriate for control as shown by the results presented in this thesis. In addition, binary quadratic programming problems that arise because on-off actuators (or, in a more general context, discrete actuators) and robust control problems due to disturbances and uncertainties, both associated with water distribution networks, are also treated in this thesis.

It is worthwhile to highlight that a great effort has been made to develop control approaches with a low computational burden that able to cope with complex control objectives. For this reason, besides the applications presented in previous chapters, it could be possible to apply these strategies, which are defined by different objectives, to a large variety of industrial control problems with potentially favourable results.

An important goal of the controllers designed is the capacity to deal with real systems. Although the data based predictive controllers proposed in this thesis are supported theoretically for linear systems, nonlinearities present in all real complex plants have been treated successfully in the examples of this thesis through implicit local linearisation. Another important issue is that these controllers have also been able to cope with uncertainties learning and inferring an underlying control law with integral effect from data or using robust MPC theory combined with data inference techniques.

Finally, it is important to remark that all the provided controllers has been tested in different simulated systems, realistic benchmarks and real applications. In the following, the conclusions of each chapter are presented:

In chapter 2, motivated by industrial large complex systems, we focused on predictive control problems with binary inputs and an efficient heuristic algorithm to solve them was presented. This algorithm constructs iteratively a candidate solutions set and relies on a reduction criterion able to keep the size of this set manageable under the available computing power and storage. In consequence, the proposed algorithm allows to solve

in real time problems that cannot be solved otherwise. Moreover, the class of systems to which this algorithm can be applied is rather wide, as stable systems with dominant dynamics and control applications with on-off actuators are quite common in the industry.

In chapter 3, after we realized the difficulty of estimating the optimal cost-to-go for techniques such as the one presented in chapter 2 and the data availability of the class of systems considered, we proposed a heuristic data based predictive controller focused on optimizing an estimation of the performance over a convex combination of past trajectories. This approach, supported by DWO techniques, can be used in complex systems in which models or enough data for identification are not available. The proposed approach was applied to a water distribution network demonstrating that it is able to consider different issues such as periodic level references and pressure constraints explicitly in its formulation.

In chapter 4, motivated by possible steady state errors of data based approaches, caused by disturbances and uncertainties, we presented an efficient strategy to solve a tracking control problem that it is tailored for systems with an unknown model function and databases generated with control laws with integral effect. The proposed method solves the problem using past closed-loop offset free trajectories and control actions stored in a database while minimizing the variance of the tracking error. To this end, the proposed controller, based on DWO methods, infers and recovers the underlying control law. Moreover, the optimization problem can be solved efficiently which fits with fast dynamic process control problems. The proposed strategy has obtained offset free tracking with a real scaled laboratory process with fast dynamics which shows its effectiveness, even when the actual process dynamics are nonlinear.

In chapter 5, the last approach of this thesis based on DWO was presented. The goal of this approach was to implement data-based predictive controllers which provide a trade-off between different performance objectives, in particular, to reduce identification errors while optimizing the closed-loop performance. With this goal, the provided controllers use the information stored in a database of input and state trajectories. For unconstrained systems, a piecewise linear explicit controller can be obtained by following a multi-parametric approach to solve the dynamic programming problem. When constraints are taken into account, we proposed a single-step predictive controller based on the solution of a quadratic programming problem.

In chapter 6, disturbances and uncertainties were dealt with a learning-based approach for robust MPC. This predictive control approach focused on MIMO systems and allow us to control plants affected by noise, disturbances or discrepancies between real systems and their identified models. The identification algorithm and the resulting controller are endowed with theoretical properties in the linear case, however they proved to be effective also on a nonlinear simulator. Preliminary extensions to address the system nonlinearities have been introduced, improving the static performance of the scheme. The provided controller has been successfully tested on a simulator of the quadruple tank system.

7.1 Future works

In recent years, data based approaches have become an important research topic of the control community. There are a huge number of open future lines of research in this field. In this section, some future works related with the results of this thesis are presented.

In the context of the control strategies presented in this thesis one of the most promising future research lines is the combination of PFMSA and data based approaches, in particular, the use of DWO techniques to estimate the cost-to-go functions. This would allow to tackle large scale complex problems in an efficient manner.

Regarding data based control strategies, it is possible to enumerate different future research topics such as:

- The development of stabilizing designs and estimation error bounds.
- Taking into account issues such as periodic changing set-points (taken into account only in some of the approaches proposed) or large process delays.
- Study of the theoretical properties in the nonlinear framework and study of identification algorithms with models that are nonlinear in the regressors.
- Online maintenance of the database in which over-parametrization versus data reusability would have to be considered, specially when few data are available.
- Design of tuning procedures that take into account a trade-off between estimation and performance using MOO techniques.
- Design of ad-hoc optimization techniques for large prediction horizons.
- Extension of the multi-step prediction models to nonlinear MIMO systems.

List of Figures

List of Tables

Bibliography

[1] T. Alamo, D. Muñoz de la Peña, and E. F. Camacho. An efficient maximization algorithm with implications in min-max predictive control. *IEEE Transactions on Automatic Control*, 53(9):2192–2197, 2008.

[2] I Alvarado, D Limon, D Muñoz de la Peña, JM Maestre, MA Ridao, H Scheu, W Marquardt, RR Negenborn, B De Schutter, F Valencia, et al. A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark. *Journal of Process Control*, 21(5):800–815, 2011.

[3] L. B. Armenio, E. Terzi, M. Farina, and R. Scattolini. Echo state networks: analysis, training and predictive control. *Accepted for European Control Conference 2019, available at arXiv:1902.01618*, 2019.

[4] T. Asakawa and T. Kida. Model predictive control for LPV system using on-off actuator with application to spacecraft attitude maneuver. In *11th IEEE International Conference on Control Automation (ICCA)*, pages 1215–1220, 2014.

[5] A. Aswani, H. Gonzalez, S. Sastry, and C. Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.

[6] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11(1):11–73, Feb 1997.

[7] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning for control. *Artificial Intelligence Review*, 11(1):75–113, Feb 1997.

[8] A. Bemporad. A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control. *IEEE Transactions on Automatic Control*, 61(4):1111–1116, April 2016.

[9] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[10] M. Berenguel, F. Rodríguez, F. G. Acién, and J. L. García. Model predictive control of pH in tubular photobioreactors. *Journal of Process of Control*, 14:377–387, 2004.

[11] D. P. Bertsekas. *Dynamic programming and optimal control*. Athena scientific, Belmont, Massachusetts, USA, 1995.

[12] D.P. Bertsekas. *Reinforcement learning and optimal control*. Athena scientific, Belmont, Massachusetts, USA, 2019.

[13] G. Betti, M. Farina, and R. Scattolini. A robust MPC algorithm for offset-free tracking of constant reference signals. *IEEE Transactions on Automatic Control*, 58(9):2394–2400, 2013.

[14] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[15] F. Blanchini and S. Miani. *Set-theoretic methods in control*. Springer, 2008.

[16] H. G. Bock and K. J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984.

[17] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory*, volume 15. Siam, 1994.

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

[19] J. M. Bravo, T. Alamo, and E. F. Camacho. Bounded error identification of systems with time-varying parameters. *IEEE Transactions on Automatic Control*, 51(7):1144–1150, 2006.

[20] J. M. Bravo, T. Alamo, M. Vasallo, and M. E. Gegúndez. A general framework for predictors based on bounding techniques and local approximation. *IEEE Transactions on Automatic Control*, 62(7):3430–3435, July 2017.

[21] J. M. Bravo, A. Suarez, M. Vasallo, and T. Alamo. Slide window bounded-error time-varying systems identification. *IEEE Transactions on Automatic control*, 61(8):2282–2287, 2015.

[22] M. Brdys and B. Ulanicki. *Operational Control of Water Systems: Structures, Algorithms and Applications*. Prentice Hall, 1994.

[23] X. Bu, Q. Wang, Z. Hou, and W. Qian. Data driven control for a class of nonlinear systems with output saturation. *ISA transactions*, 81:1–7, 2018.

[24] J. P. Calliess. *Conservative decision-making and inference in uncertain dynamical systems*. PhD thesis, University of Oxford, 2014.

[25] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer, second edition, 2007.

[26] E. F. Camacho, D. R. Ramirez, D. Limon, D. Muñoz de la Peña, and T. Alamo. Model predictive control techniques for hybrid systems. *Annual Reviews in Control*, 34(1):21 – 31, 2010.

[27] M. C. Campi, A. Lecchini, and S. M. Savaresi. Virtual reference feedback tuning: a direct method for the design of feedback controllers. *Automatica*, 38(8):1337 – 1346, 2002.

[28] M. Canale, L. Fagiano, and M.C Signorile. Nonlinear model predictive control from data: a set membership approach. *International Journal of Robust and Nonlinear Control*, 24(1):123–139, 2014.

[29] J. Causa, G. Karer, A. Núñez, D. Sáez, I. Škrjanc, and B. Zupančič. Hybrid fuzzy predictive control based on genetic algorithms for the temperature control of a batch reactor. *Computers & Chemical Engineering*, 32(12):3254 – 3263, 2008.

[30] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, and K. Skadron. A performance study of general-purpose applications on graphics processors using cuda. *J. Parallel Distrib. Comput.*, 68(10):1370–1380, October 2008.

[31] R. Chi, X. Liu, R. Zhang, Z. Hou, and B. Huang. Constrained data-driven optimal iterative learning control. *Journal of Process Control*, 55:10 – 29, 2017.

[32] E. G. Cojocaru, J. M. Bravo, M. J. Vasallo, and D. Marín. A binary-regularization-based model predictive control applied to generation scheduling in concentrating solar power plants. *Optimal Control Applications and Methods*.

[33] R. de Rozario and T. Oomen. Data-driven iterative inversion-based control: Achieving robustness through nonlinear learning. *Automatica*, 107:342–352, 2019.

[34] Enviromental Protection Agency. Epanet: Software that models the hydraulic and water quality behavior of water distribution piping systems. `https://www.epa.gov/water-research/epanet`, 2018.

[35] L. Fagiano and C Novara. Learning a nonlinear controller from data: Theory, computation, and experimental results. *IEEE Transactions on Automatic Control*, 61(7):1854–1868, July 2016.

[36] W. Favoreel, B. De Moor, P. Van Overschee, and M Gevers. Model-free subspace-based lqg-design. Proceedings of the 1999 American Control Conference, pages 3372–3376, June 1999.

[37] A. Ferramosca, D. Limon, F. Fele, and E. F. Camacho. L-Band SBQP-based MPC for supermarket refrigeration systems. In *2009 European Control Conference (ECC)*, pages 162–167, 2009.

[38] S. Formentin, D. Piga, R. Tóth, and S. M. Savaresi. Direct learning of LPV controllers from data. *Automatica*, 65:98 – 110, 2016.

[39] A. Gambier and E. Badreddin. Multi-objective optimal control: An overview. In *2007 IEEE International Conference on Control Applications*, pages 170–175. IEEE, 2007.

[40] T. Gao, S. Yin, H. Gao, X. Yang, J. Qiu, and O. Kaynak. A locally weighted project regression approach-aided nonlinear constrained tracking control. *IEEE Transactions on Neural Networks and Learning Systems*, 29(12):5870–5879, Dec 2018.

[41] M. Ghanes, M. Trabelsi, H. Abu-Rub, and L. Ben-Brahim. Robust adaptive observer-based model predictive control for multilevel flying capacitors inverter. *IEEE transactions on industrial electronics*, 63(12):7876–7886, 2016.

[42] E. Gilbert and K. T. Tan. Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Transactions on Automatic control*, 36(9):1008–1020, 1991.

[43] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin. Iterative feedback tuning: theory and applications. *IEEE Control Systems*, 18(4):26–41, Aug 1998.

[44] Z. Hou and S. Jin. Data-driven model-free adaptive control for a class of MIMO nonlinear discrete-time systems. *IEEE Transactions on Neural Networks*, 22(12):2173–2188, Dec 2011.

[45] Z. Hou, S. Liu, and C. Yin. Local learning-based model-free adaptive predictive control for adjustment of oxygen concentration in syngas manufacturing industry. *IET Control Theory & Applications*, 10(12):1384–1394, 2016.

[46] M. Ikeda, Y. Fujisaki, and N. Hayashi. A model-less algorithm for tracking control based on input-output data. *Nonlinear Analysis: Theory, Methods & Applications*, 47(3):1953 – 1960, 2001. Proceedings of the Third World Congress of Nonlinear Analysts.

[47] A. Ingimundarson, J. M. Bravo, V. Puig, T. Alamo, and P. Guerra. Robust fault detection using zonotope-based set-membership consistency test. *International journal of adaptive control and signal processing*, 23(4):311–330, 2009.

[48] A. Jain, F. Smarra, and R. Mangharam. Data predictive control using regression trees and ensemble learning. Proceedings of the 2017 Conference on Decision and Control. IEEE, 2017.

[49] J. Ju, M. S. Chiu, and C. Tien. Multiple-objective based model predictive control of pulse jet fabric filters. *Chemical Engineering Research and Design*, 78(4):581–589, 2000.

[50] R. Kadali, B. Huang, and A. Rossiter. A data driven subspace approach to predictive controller design. *Control Engineering Practice*, 11(3):261 – 278, 2003.

[51] P. Kergus, C. Poussot-Vassal, F. Demourant, and S. Formentin. Frequency-domain data-driven control design in the loewner framework. *IFAC-PapersOnLine*, 50(1):2095 – 2100, 2017. 20th IFAC World Congress.

[52] E. C. Kerrigan and J. M. Maciejowski. Designing model predictive controllers with prioritised constraints and objectives. In *Proceedings. IEEE International Symposium on Computer Aided Control System Design*, pages 33–38. IEEE, 2002.

[53] D. B. Kirk and W. W. Hwu, editors. *Programming Massively Parallel Processors (Third Edition)*. Morgan Kaufmann, third edition edition, 2017.

[54] E. Kofman, J. A De Doná, and M. M. Seron. Probabilistic set invariance and ultimate boundedness. *Automatica*, 48(10):2670–2676, 2012.

[55] D. Laurí, J. A. Rossiter, J. Sanchis, and M. Martínez. Data-driven latent-variable model-based predictive control for continuous processes. *Journal of Process Control*, 20(10):1207 – 1219, 2010.

[56] G. Leitmann. Guaranteed asymptotic stability for some linear systems with bounded uncertainties. *Journal of Dynamic Systems, Measurement, and Control*, 101(3):212–216, 1979.

[57] D. Li, Y. Xi, and F. Gao. Synthesis of dynamic output feedback RMPC with saturated inputs. *Automatica*, 49(4):949 – 954, 2013.

[58] Y. Li, C. Yang, Z. Hou, Y. Feng, and C. Yin. Data-driven approximate q-learning stabilization with optimality error bound analysis. *Automatica*, 103:435–442, 2019.

[59] D. Limon, J. Calliess, and J.M Maciejowski. Learning-based nonlinear model predictive control. *IFAC-PapersOnLine*, 50(1):7769 – 7776, 2017.

[60] G. P. Liu, J. B. Yang, and J. F. Whidborne. *Multiobjective optimisation and control*. Research Studies Press,, 2003.

[61] L. Ljung. *System Identification: Theory for the User*. Prentice-hall, Inc., 1987.

[62] J. Maciejowski. *Predictive control: with constraints*. Pearson education, 2002.

[63] U. Maeder and M. Morari. Offset-free reference tracking with model predictive control. *Automatica*, 46(9):1469–1476, 2010.

[64] S. J. Maher, T. Fischer, T. Gally, G. Gamrath, A. Gleixner, R. L. Gottwald, G. Hendel, T. Koch, M. E. Lübbecke, M. Miltenberger, et al. The SCIP optimization suite 4.0. 2017.

[65] J. M. Manzano, D. Limon, D. Muñoz de la Peña, and J. P. Calliess. Output feedback mpc based on smoothed projected kinky inference. *IET Control Theory & Applications*, 2019.

[66] T. M. Maupong and P. Rapisarda. Data-driven control: A behavioral approach. *Systems & Control Letters*, 101:37 – 43, 2017.

[67] D. Q. Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

[68] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[69] D. Q. Mayne, M. Seron, and S. Raković. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41(2):219–224, 2005.

[70] M. Milanese, J. Norton, H. Piet-Lahanier, and É. Walter. *Bounding approaches to system identification*. Springer Science & Business Media, 2013.

[71] M. Milanese and C. Novara. Set membership identification of nonlinear systems. *Automatica*, 40(6):957–975, 2004.

[72] M. Milanese and C Novara. Unified set membership theory for identification, prediction and filtering of nonlinear systems. *Automatica*, 47(10):2141 – 2151, 2011.

[73] M. Milanese and A. Vicino. Optimal estimation theory for dynamic systems with set membership uncertainty: an overview. *Automatica*, 27(6):997–1009, 1991.

[74] J. Nocedal and S.J. Wright. *Numerical optimization*. Springer-Verlag, New York, 2006.

[75] C. Novara and M. Milanese. Control of MIMO nonlinear systems: A data-driven model inversion approach. *Automatica*, 101:417–430, 2019.

[76] C. Ocampo-Martinez, V. Puig, G. Cembrano, and J Quevedo. Application of predictive control strategies to the management of complex networks in the urban water cycle. *IEEE Control Systems*, 33(1):15–41, 2013.

[77] A. Pawlowski, I. Fernández, J. L. Guzmán, M. Berenguel, F. G. Acién, and J. E. Normey-Rico. Event-based predictive control of pH in tubular photobioreactors. *Computer and Chemical Engineering*, 65:28–39, 2014.

[78] M. Pereira, D. Muñoz de la Peña, D. Limon, I. Alvarado, and T. Alamo. Application to a drinking water network of robust periodic mpc. *Control Engineering Practice*, 57:50–60, 2016.

[79] D. Piga, S. Formentin, and A. Bemporad. Direct data-driven control of constrained systems. *IEEE Transactions on Control Systems Technology*, 26(4):1422–1429, 2018.

[80] W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality.* John Wiley & Sons, 2007.

[81] V. Puig, C. Ocampo-Martinez, R. Pérez, G. Cembrano, J. Quevedo, and T Escobet. *Real-time Monitoring and Operational Control of Drinking-Water Systems.* Springer, 2017.

[82] S. J. Qin. Survey on data-driven industrial process monitoring and diagnosis. *Annual Reviews in Control*, 36(2):220 – 234, 2012.

[83] M. B. Radac and R. E. Precup. Three-level hierarchical model-free learning approach to trajectory tracking control. *Engineering Applications of Artificial Intelligence*, 55:103 – 118, 2016.

[84] M. B. Radac, R. E. Precup, and R. C. Roman. Data-driven model reference control of mimo vertical tank systems with model-free VRFT and Q-Learning. *ISA Transactions*, 73:227 – 238, 2018.

[85] S. Rakovic and K. Kouramas. Invariant approximations of the minimal robust positively invariant set via finite time aumann integrals. *2007 46th IEEE Conference on Decision and Control*, pages 194–199, 2007.

[86] C. E. Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[87] J. B. Rawlings and R. Amrit. Optimizing process economic performance using model predictive control. In *Nonlinear model predictive control*, pages 119–138. Springer, 2009.

[88] J. B. Rawlings and D. Q. Mayne. *Model Predictive Control:Theory and Design.* Nob Hill Publishing, Madison, Wisconsin, 2009.

[89] N. Lawrence Ricker. Predictive hybrid control of the supermarket refrigeration benchmark process. *Control Engineering Practice*, 18(6):608 – 617, 2010.

[90] J. Roll, A. Nazin, and L. Ljung. A non-asymptotic approach to local modelling. In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002*, volume 1, pages 638–643, 2002.

[91] J. Roll, A. Nazin, and L. Ljung. Local modelling with a priori known bounds using direct weight optimization. In *2003 European Control Conference (ECC)*, pages 2138–2143, 2003.

[92] J. Roll, A. Nazin, and L. Ljung. Nonlinear system identification via direct weight optimization. *Automatica*, 41(3):475–490, 2005.

[93] S. Sager, H. G. Bock, and G. Reinelt. Direct methods with maximal lower bound for mixed-integer optimal control problems. *Mathematical Programming*, 118(1):109–149, 2009.

[94] A. Sala and A. Esparza. Extensions to "virtual reference feedback tuning: A direct method for the design of feedback controllers". *Automatica*, 41(8):1473 – 1476, 2005.

[95] U. Schmitz, R. Haber, and F. Lang. Predictive on-off cost minimizing control of a municipal waste water treatment plant. *IFAC Proceedings Volumes*, 38(1):43 – 48, 2005. 16th IFAC World Congress.

[96] D. Selvi, D. Piga, and A. Bemporad. Towards direct data-driven model-free design of optimal controllers. In *Proceedings of the European Control Conference, Accepted for publication.*, 2018.

[97] H. Shah and M. Gopal. Model-free predictive control of nonlinear processes based on reinforcement learning. *IFAC-PapersOnLine*, 49(1):89 – 94, 2016. 4th IFAC Conference on Advances in Control and Optimization of Dynamical Systems ACODS 2016.

[98] D. Sprock and P. Hsu. Predictive discrete time control of switch-mode applications. In *Power Electronics Specialists Conference, 1997. PESC '97 Record., 28th Annual IEEE*, volume 1, pages 175–181 vol.1, 1997.

[99] G. M. Stanley. Big data approximating control (BDAC)—A new model-free estimation and control paradigm based on pattern matching and approximation. *Journal of Process Control*, 67:141–159, 2018.

[100] M. Tanaskovic, L. Fagiano, C. Novara, and M. Morari. Data-driven control of nonlinear systems: An on-line direct approach. *Automatica*, 75:1 – 10, 2017.

[101] W. Tang and P. Daoutidis. Distributed adaptive dynamic programming for data-driven optimal control. *Systems & Control Letters*, 120:36–43, 2018.

[102] E. Terzi, L. Fagiano, M. Farina, and R. Scattolini. Learning multi-step prediction models for receding horizon control. In *2018 European Control Conference (ECC)*, pages 1335–1340. IEEE, 2018.

[103] E. Terzi, L. Fagiano, M. Farina, and R. Scattolini. Learning-based predictive control for linear systems: a unitary approach. *Automatica*, 108:108473, 2019.

[104] E. Terzi, M. Farina, L. Fagiano, and R. Scattolini. Robust predictive control with data-based multi-step prediction models. In *2018 European Control Conference (ECC)*, pages 1710–1715. IEEE, 2018.

[105] J. Thomas. Analytical non-linear model predictive control for hybrid systems with discrete inputs only. *IET Control Theory Applications*, 6(8):1080–1088, 2012.

[106] H. Van Hasselt. Reinforcement learning in continuous state and action spaces. Reinforcement learning, pages 207–251. Springer, 2012.

[107] K. van Heusden, A. Karimi, and D. Bonvin. Data-driven model reference control with asymptotically guaranteed stability. *International Journal of Adaptive Control and Signal Processing*, 25(4):331–351, 2010.

[108] P. Van Overschee and B. De Moor. Subspace algorithms for the stochastic identification problem. *Automatica*, 29(3):649 – 660, 1993.

[109] P. Van Overschee and B. De Moor. A unifying theorem for three subspace system identification algorithms. *Automatica*, 31(12):1853 – 1864, 1995. Trends in System Identification.

[110] P. Van Overschee and B. De Moor. *Subspace identification for linear systems: Theory—Implementation—Applications*. Springer Science & Business Media, 2012.

[111] J. E. Van Zyl. Richmond water distribution network EPANET model. `http://emps.exeter.ac.uk/engineering/research/cws/resources/benchmarks/operation/richmond.php`, 2001.

[112] J. E. Van Zyl, D. A. Savić, and G. A Walters. Operational optimization of water distribution systems using a hybrid genetic algorithm. *Journal of Water Resources Planning and Management*, 130(2):160–170, 2004.

[113] X. Wang, B. Huang, and T. Chen. Data-driven predictive control for solid oxide fuel cells. *Journal of Process Control*, 17(2):103 – 114, 2007.

[114] Y. Wang, J. R. Salvador, D. Muñoz de la Peña, V. Puig, and G. Cembrano. Economic model predictive control based on a periodicity constraint. *Journal of Process Control*, 68:226 – 239, 2018.

[115] G. S. Williams and A. Hazen. *Hydraulic tables: showing the loss of head due to the friction of water flowing in pipes, aqueducts, sewers, etc. and the discharge over weirs*. John Wiley and Sons, New York, 1905.

[116] M. A. Woodbury. Inverting modified matrices. *Memorandum Rept. 42*, 4:pp, 1950. Princeton University, Princeton, NJ.

[117] J. Yang, W. X. Zheng, S. Li, B. Wu, and M. Cheng. Design of a prediction-accuracy-enhanced continuous-time MPC for disturbed systems via a disturbance observer. *IEEE Transactions on Industrial Electronics*, 62(9):5807–5816, Sept 2015.

[118] R. Zhang and J. Tao. Data-driven modeling using improved multi-objective optimization based neural network for coke furnace system. *IEEE Transactions on Industrial Electronics*, 64(4):3147–3155, 2016.

[119] Y. Zhang, S. X. Ding, Y. Yang, and L. Li. Data-driven design of two-degree-of-freedom controllers using reinforcement learning techniques. *IET Control Theory Applications*, 9(7):1011–1021, 2015.

[120] H. Zhao, T. Icoz, Y. Jaluria, and D. Knight. Application of data-driven design optimization methodology to a multi-objective design optimization problem. *Journal of Engineering Design*, 18(4):343–359, 2007.

Glossary

BQP Binary Quadratic Programming. 29

CPU Central Processing Unit. 30, 34

CUDA Compute Unified Device Architecture. 30

DWO Direct Weight Optimization. 6, 8, 10–12, 39, 112, 113

EPANET is a public domain, water distribution system modeling software package developed by the United States Environmental Protection Agency's (EPA) Water Supply and Water Resources Division. iii, 11, 41, 43, 44

GEPOC Estimation, Prediction, Optimization and Control Group (Acronym in spanish). iii

GP Gaussian process. 6

GPGPU General-Purpose computing on GPUs. 30

GPU Graphic Processing Unit. 30, 34

IP Integer Programming. 5

LP Linear Programming. 5, 7, 37, 40, 44

LQG Linear-Quadratic-Gaussian. 9

LTI Linear Time-Invariant. 10

Matlab MAThematical LABoratory software. 74

MILP Mixed-Integer Linear Programming. 15

MIMO Multiple Input Multiple Output (system). 10, 12, 13, 97, 98, 102, 112, 113

MIP Mixed-Integer Programming. 5, 15

MIQP Mixed-Integer Quadratic Programming. 15

MOO Multi-objective Optimization. 81, 113

MPC Model Predictive Control. 1–7, 9–11, 15, 20, 25, 28, 29, 34, 37, 38, 40, 41, 81, 97, 98, 106, 111, 112, 115

P Proportional controller. 34, 72

PFM Pulse Frecquency Modulation. 30, 34

PFMS Partial Fading Memory System. v, 16, 20, 24, 25, 28, 32

PFMSA PFMS Algorithm. 33–35, 113, 115, 119

PI Proportional-Integral controller. 55, 71, 74, 75, 77, 80, 116, 119

PID Proportional-Integral-Derivative controller. 1, 41, 55

QP Quadratic Programming. 5, 37, 48, 51

RPI Robust Positively Invariant (set). 100, 101

SCIP Solving Constraint Integer Programs. 32

SISO Single Input Single Output (system). 97

SoPlex Sequential object-oriented simPlex: Optimization package for solving LP problems. 32

WDN Water Distribution Network. 2, 5, 6, 11, 37, 55, 81