Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Informatica, Gestionale e dell'Automazione

---

Complexity certification and efficient implementation of model predictive control for embedded applications

Ph.D. Dissertation of:
Gionata Cimini

Advisor:
Prof. Gianluca Ippoliti

Coadvisor:
Prof. Sauro Longhi
Prof. Alberto Bemporad

Curriculum Supervisor:
Prof. Francesco Piazza

XV edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Informatica, Gestionale e dell'Automazione

---

Complexity certification and efficient implementation of model predictive control for embedded applications

Ph.D. Dissertation of:
Gionata Cimini

Advisor:
Prof. Gianluca Ippoliti

Coadvisor:
Prof. Sauro Longhi
Prof. Alberto Bemporad

Curriculum Supervisor:
Prof. Francesco Piazza

XV edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

Abstract

Model Predictive Control (MPC) is experiencing an ever growing success in multivariable embedded control, thanks to the optimized performance and the effective handling of system's constraints. However, the high computational burden to solve online the optimization problem is an issue of paramount importance, especially when paired with the low computational power and the fast sampling frequency typical of embedded control. Specifically, the difficulties of guaranteeing that the online optimization problem converges in a prescribed amount of time, without compromising optimality and feasibility, is severely limiting the spread of embedded MPC. This thesis proposes a certification algorithm for dual active-set methods, which is able to certify exactly the worst-case time needed by the solver to reach the optimal solution, given a parametric Quadratic Programming (QP) problem like those that arise in linear MPC. The certification algorithm is therefore used to establish if the MPC problem will be always solved in the allocated time slot on a specific computational unit.

In order to soften the computational issue, several accelerating methods for MPC have been proposed in the past few years. However, they have suffered the lack of a complexity certification as well, because they are shown to improve the average case performance without any guarantee on the worst-case behavior. This thesis presents two novel accelerating methodologies for which the worst-case improvement can be exactly certified. The first is a semi-explicit MPC, combining an online solver with the explicit solution of those polyhedral regions that affect more the worst-case complexity. An algorithm to select the best trade-off between memory and performance improvement is also proposed, which is a further novelty for semi-explicit MPC. The second acceleration method consists in an alternative selection strategy for the violated constraint to add to the active set at each iteration. By exploiting the particular structure of QP problems coming from MPC formulations, the new

strategy brings a certifiable improvement in both the worst-case number of iterations and also the complexity of the single iteration.

Electrical motors and power electronics are some of the most flourishing fields for embedded MPC application. The aforementioned certification and acceleration techniques, are tested here experimentally for the control of a permanent magnet synchronous motor, and the control of several DC-DC converters. The online solution of the MPC problem is demonstrated to be effective even with scarce computational resources and high sampling frequency. Additionally, embedded MPC for precompensated DC-DC converters is also presented and experimentally tested to overcome the issue of a non-modifiable primal controller, very common in power converters. Finally, the issue of estimating the state for multiple DC-DC converters on the same power supply is addressed. A unified nonlinear robust observer for six different converters' topologies is derived and experimentally validated.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Model Predictive Control (MPC) is an advanced control technique that was proposed in the process industry already in the late sixties [1, 2], and received a lot of attention since then [3, 4]. What makes MPC successful in many engineering fields is the capability to optimally handle constrained multivariable systems, by making explicit use of a prediction model of the system. MPC derives the input sequence by solving a finite-time, open-loop, optimal control problem, and applies only the inputs corresponding to the actual time step, discarding the rest of the input sequence. This procedure is then repeated at each time step, in the so-called receding horizon fashion. Despite its long history, MPC is still one of the most flourishing fields in the control community, [5], and embedded MPC has seen an ever growing interest in the very last years [6, 7]. Automotive, aerospace and electrical power systems are just some of the areas where the research in embedded MPC is very active, and these fields are particularly pushing towards more and more computationally efficient algorithms [8–14]. The necessity to solve online an optimization problem makes the computational burden of MPC way higher than many other control algorithms, therefore its application to embedded systems is still a challenge. Indeed, the fast sampling frequency typical of embedded control is usually paired with reduced computational power, imposing severe limits to the diffusion of embedded MPC. In some applications, the use of more powerful computational units, e.g. Field Programmable Gate Arrays (FPGAs), is tolerated, softening the complexity issue [10, 15, 16]. Anyhow, their cost, lack of flexibility and finite precision arithmetic do not make FPGAs the preferred choice for embedding an MPC algorithm [17–19]. For this reason, this thesis will deal with theoretical aspects and applications of embedded MPC imple-

mented on resource-constrained platforms, such as cheap Digital Signal Processors (DSPs), and here follows a brief literature review of the last advances in the topic.

Explicit MPC could be a possible solution for embedded control [20]. The standard formulation of MPC, based on a linear time-invariant prediction model, a quadratic cost function, and affine constraints, can be cast into a Quadratic Programming (QP) problem, whose linear term of the cost function and right hand side of the constraints depend linearly on a vector of parameters. In this case the optimization problem can be pre-solved offline by means of multiparametric Quadratic Programming (mpQP), converting the MPC law into a continuous and piecewise affine (PWA) function of the parameter vector [20, 21]. This off-line explicit solution, made by the polyhedral partitions and gains of the control law, is then stored and used as a look up table in real-time. Therefore, the computational load can be drastically reduced with respect to solve the problem online. However, despite of the efforts in looking for faster and more memory efficient search methods, [22,23], explicit MPC can be applied only when the QP is small enough. Indeed, the logarithmic search time (i.e. when a binary search tree is used [23]), and the polynomial memory occupancy make it a way to go only with few inputs and very short control horizon.

When the complexity of explicit MPC is prohibitive, the only solution is *implicit* MPC, where the optimization problem is solved online at each time step. For linear MPC, this means embedding a QP solver in the control unit, which has to be fast, memory efficient and simple to code. The recent advances in convex optimization and the increasing power of control units, have favored an unrivaled interest for embedded QP solvers in the very last years [24–27]. The literature on the topic is extremely rich, and among several popular approaches, we mention interior-point methods [28,29], gradient projection methods [24,30], the alternating directions method of multipliers (ADMM) [31], and active-set methods [32–35]. For the small-size QP problems that arise in embedded MPC, active-set methods are often considered outperforming in terms of speed and accuracy with respect to other solvers [26, 36–38]. Moreover, the solution can be achieved with very high accuracy and in a finite number of iterations, even in the case of ill-conditioned problems, single-precision arithmetics and without preconditioning [39].

With the purpose to increase the throughput of implicit MPC, several methods to accelerate the QP solvers have been proposed in the last few years. Many of them are closely related to the explicit solution. For instance, qpOASES [34, 40] improves the standard warm-start, by considering the underlying explicit solution, and moving on a straight line in the parametric space between successive QPs. The *Phase I*, typical of primal active-set methods, is avoided by moving on an homotopy path. The number of iterations can be therefore reduced when the successive QPs are related, that is if their solutions are close enough. *Partial enumeration* is gaining popularity as well to accelerate the online optimization. It lies in the middle between explicit and implicit MPC [41–43], as a subset of the explicit solution is stored in a fixed dimensional table, and updated online. The table contains the polyhedral partitions and gains of the most recent used control laws, and the performance improvement depends on the amount of data stored. Another popular semi-explicit method consists in using a partial explicit solution to warm start an online solvers. For instance, in [44], a primal active-set algorithm runs for a fixed number of iterations, allowing for possible sub-optimality. Another method to improve implicit MPC has been recently proposed in [45]. At each time step, a set of the so-called *regions of activity*, containing an incomplete information about the inactive constraints, is identified. They are then used to build a lower dimensional QP, solved faster.

Unfortunately, despite of all the efforts in developing more and more efficient solvers and acceleration techniques, the use of embedded MPC solved online is still limited, especially in those applications with very fast sampling frequencies. For instance, electrical motors and power systems are showing an unattainable interest in MPC, thanks to the intuitive design and tuning of the controller, the enhanced performance, the possibility to include safety constraints and the availability of relatively accurate models for electrical devices [11, 12, 46]. However, embedded MPC is far to become a technology in those fields [12, 47]. The literature splits into two branches when transistor-based systems are addressed. Continuous Control Set (CCS)-MPC involves modulation, and takes control actions in a continuous set that usually corresponds to the duty-cycle of the modulation [48]. On the other hand, Finite Control Set (FCS)-MPC exploits the discrete nature of the switches and manipulates

directly their position with a binary control signal [49–51]. FCS-MPC provides a faster response time, and can reduce switching losses, however the optimization problem has a combinatorial nature, and scales exponentially with the prediction horizon [52,53]. Furthermore, an high control frequency is required (e.g., 40μs in [54]). CCS-MPC performs better with a lower sampling frequency, and drives the inverter at a fixed frequency, possibly higher than the control one [50, 51, 55]. These key factors for embedded implementation make CCS-MPC still the primal choice for industrial applications, however the computational burden of the online optimization is typically considered unmanageable, due to the low-cost platforms and to the sampling frequency that ranges from 1kHz to 10kHz. Successful implementation of explicit MPC exist for electrical motors [56,57], and for power converters [58–61], even though these results are often achieved for simplified MPC formulations, with one-step prediction and/or approximated constraints [56].

1.1 Motivations and contribution

Besides the high computational load, the need to solve online an optimization problem poses another cumbersome limitation for embedded MPC implementation: the iterative nature of the algorithm and the consequent time-varying complexity. Indeed, for a successful implementation of MPC, the optimization problem must be always solved in a prescribed amount of time, usually corresponding to the sampling period or even shorter in multi-purpose control units [62]. While this is trivial to verify in a non iterative control routine, the possibility of failure into providing the input sequence in time can be a decisive factor in preventing the use of MPC. As far as embedded MPC, this issue is even sharper. In fact, by considering the state of the art of the QP solvers' speed and the computational power of cheap control units, the average time to solve an embedded QP problem is often in the same order of magnitude of the maximum time allowed [37]. Therefore, even though extensive simulation campaign can reveal an average time always below the hard constraint, the possibility that a non tested worst-case exceeds it is not remote. This failure in the control routine can be critical for the system if no countermeasures are taken. A fixed amount of iterations is one of the most common solutions, but it can cause infeasibility and

sub-optimality [44]. For this reason, the topic of complexity certification of online solvers is an urgent problem to solve, in order to favor the spread of embedded MPC [62, 63]. This thesis proposes an algorithm that is able to compute exactly the worst-case number of iterations and flops, given the parametric QP formulation of an MPC problem. The algorithm can be therefore used to clearly assess if the embedded MPC controller will be always solved within the prescribed time. The algorithm certifies the complexity of dual active-set methods, which outperform very often other solvers given the problem size typical of embedded MPC [26, 37, 38]. This new algorithm is the first exact certification for QP solvers, as far as we know, and has a twofold contribution. Besides the obvious benefit for embedded MPC implementation, it fills the lack in the literature regarding tight bounds on active-set methods complexity [64]. Indeed, only simulation or probabilistic-based analysis are available for estimating those bounds, and consequently active-set methods are currently considered polynomial algorithms on average but without any tight bound on the worst-case behavior [65–67].

The second part of the thesis proposes two novel acceleration techniques for embedded MPC, which both share the unique feature of having a certifiable improvement in terms of computational complexity. Indeed, for a lot of embedded MPC applications, the improvement in the worst-case is the leading factor, if not the sole one, to assess the effectiveness of an online acceleration. However, due to their theoretical foundations, the methods introduced in the above literature review can improve the average behavior without any guarantee on the worst-case, e.g. [40, 43, 44, 68]. This thesis first presents a novel semi-explicit MPC that is linked with the results provided by the complexity certification algorithm. It differs from the semi-explicit techniques already discussed because the partial explicit control law is not updated online and contains only those regions that are verified to improve the worst-case behavior. Furthermore, the trade-off between memory and speed, difficult to be selected in other methods, can be easily chosen in this case. The lack of a certified improvement is certainly the biggest drawback in the state of the art of acceleration techniques, but not the only one. Indeed, the additional memory occupation and the dependence from the previous QP solution can undermine their implementability on certain applications. Therefore, this thesis proposes another accelera-

tion strategy , which consists in a novel method to select the constraints added to the active set at each iteration. By exploiting the nature of a QP problem derived from MPC, this new selection rule can reduce both the complexity of the single iteration and the number of iterations needed to find the optimal solution. More importantly these improvements can be exactly calculated by means of the certification algorithm. The memory occupation is not affected, and the approach relies only on the current time-step information.

The last part of the thesis presents theoretical advances and experimental results for the control of electrical motors and power converters [37, 69]. In the specific, embedded CCS-MPC solved online on a cheap DSP is addressed for the control of a permanent magnet synchronous motor. Despite the common trend of considering CCS-MPC a solution on cheap boards only if it is solved with offline multiparametric programming, this thesis shows that the online solution is feasible and certifiable. This can shed light on CCS-MPC for electrical motors, which recently has been partially abandoned in this field because considered too computationally demanding, in favor of FCS-MPC [50, 70, 71]. Moreover, the corresponding explicit version of the implemented controller is shown to require too much memory to be implementable on the selected control board. The concept of MPC for pre-compensated systems is then introduced for the control of power supplies, with an hard link to Reference Governor (RG) technique, [72, 73], which has been already successfully used in other engineering fields, e.g. automotive and robotics [74–76]. The performance of standard Voltage Mode Control (VMC) for a DC-DC buck converter is enhanced by an external MPC loop that steers its output voltage reference. The experimental results show the reliability and feasibility of the method for those embedded systems where the primal controller cannot be changed for several reasons, i.e. hard-coded/hardware-based algorithm. The research in control algorithms for power supplies suffers the wide differentiation of converters topologies and configurations, especially in those power supplies composed by several DC-DC converters. Indeed, they are generally nonlinear systems, and advanced controllers require a dedicated function for each converter [77–80]. Using different algorithms on the same computational unit worsen the memory occupation of the controller code, affects its verification and maintenance, and forces the designer to learn

different tuning processes. Therefore, a unified algorithmic environment can be substantially beneficial [81, 82], but it has not been investigated for sensorless control yet [83–85]. The thesis presents a unified current observer suitable for Pulse Width Modulation (PWM)-based algorithms, that can be applied to buck, boost and buck-boost DC-DC converters in both synchronous and asynchronous configuration [86]. The so designed nonlinear observer is unique for its versatility compared to the converter-based approaches present in the literature [87–89], takes parasitics components into consideration and it is robust against unknown load variations. Experimental tests on different DC-DC converters confirm its validity for embedded applications, and it can be valuable tool for improving the embedded implementation of the state observer used in MPC.

The thesis is structured as follows. Chapter 2 presents the notation used throughout the thesis, and some mathematical preliminaries. The linear MPC formulation and the corresponding QP problem are described in Chapter 3. The basics on convex optimization and in particular the operation of a dual active-set method are detailed in Chapter 4. Chapter 5 presents the novel complexity certification algorithm, together with results on well-known MPC problems. Chapter 6 is dedicated to the acceleration techniques, it describes first the novel semi-explicit MPC and then the novel constraints selection strategy for dual active-set solvers. Chapter 7 shows the experimental results regarding the application of embedded MPC and complexity certification for the control of an electrical motor. MPC for power converters is then addressed in Chapter 8, where first MPC for pre-compensated systems and then the novel unified current observer are detailed and tested. Chapter 9 concludes the thesis with a summary of what has been presented, together with possible future developments of the thesis.

Chapter 2

Preliminaries

For the ease of the reader, this chapter goes over some basic mathematical preliminaries that will be used in the rest of the thesis. For a detailed discussion on them, please refer to textbooks on convex optimization [90–92].

2.1 Notation

Let $\mathbb{R}^n$ denote the set of real vectors of dimension n and $\mathbb{N}$ the set of natural integers, respectively. Let $\mathcal{I} \subset \mathbb{N}$ be a finite set of integers and denote by $\#\mathcal{I}$ its cardinality. For a vector $a \in \mathbb{R}^n$, a_i denotes the i-th entry of a, $a_{\mathcal{I}}$ the subvector obtained by collecting the entries a_i for all $i \in \mathcal{I}$. For a matrix $A \in \mathbb{R}^{n \times m}$, A' denotes its transpose, A_i denotes the i-th row of A, $A_{\mathcal{I}}$ the submatrix of A obtained by collecting the rows A_i for all $i \in \mathcal{I}$. We denote by $M_{0,0}$ the empty matrix ($n = 0$, $m = 0$). For a square matrix $A \in \mathbb{R}^{n \times n}$, A^{-1} denotes the inverse of A (if it exists). Given a set $\Theta \in \mathbb{R}^n$, $\mathring{\Theta}$ denotes its interior. Given a function $f(x) : \mathbb{R}^n \to \mathbb{R}^m$, **dom** f denotes its domain.

2.2 Definitions

Definition 2.1 (Convex set)**.** A set $S \in \mathbb{R}^n$ is *convex* if for any $x_1, x_2 \in S$, and any ξ such that $0 \leq \xi \leq 1$, the following equation holds:

$$\xi x_1 + (1 - \xi) x_2 \in S, \tag{2.1}$$

that is, the line segment between any two points of the set belongs to the set itself. ■

Definition 2.2 (Affine set)**.** A set $S \in \mathbb{R}^n$ is *affine* if for any $x_1, x_2 \in S$, and any $\xi \in \mathbb{R}$, the following relation holds:

$$\xi x_1 + (1 - \xi)x_2 \in S, \tag{2.2}$$

that is, the line through any two points of the set belongs to the set itself. Every affine set is also convex. ■

Definition 2.3 (Polyhedron)**.** A *polyhedron* is the set of solutions $x \in \mathbb{R}^n$ to a system of linear equalities and inequalities, such as:

$$\mathbb{P} = \{x \mid Gx \leq b,\ G_e x = b_e\}, \tag{2.3}$$

with $G \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $G_e \in \mathbb{R}^{m_e \times n}$ and $b_e \in \mathbb{R}^m_e$. Polyhedra are convex sets. ■

Definition 2.4 (Convex function)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $\mathbf{dom}\, f$ is a convex set and for any x_1, $x_2 \in \mathbf{dom}\, f$, and any ξ such that $0 \leq \xi \leq 1$, the following equation holds:

$$f(\xi x_1 + (1 - \xi)x_2) \leq \xi f(x_1) + (1 - \xi) f(x_2). \tag{2.4}$$

If $-f$ is convex, then f is *concave*. ■

Definition 2.5 (Affine function)**.** A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is *affine* if it is of the form $f(x) = Gx + b$, with $G \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. In other words, an affine function is a sum of a linear function and a constant. All affine functions are both convex and concave. ■

Definition 2.6 (Quadratic function)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is *quadratic* if it can be written in the form:

$$f(x) :\ \frac{1}{2} x' H x + h' x + k \tag{2.5}$$

with $H \in \mathbb{R}^{n \times n}$ a symmetric matrix, $h \in \mathbb{R}^n$ and $k \in \mathbb{R}$. A quadratic function is convex if and only if $H \geq 0$. In the rest of the thesis $k = 0$ holds, therefore the constant term k will be omitted. ■

Definition 2.7 (Convex optimization)**.** A *convex optimization* problem

is of the form

$$\begin{aligned} \min_{x} \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq b_i, \quad i = 1, \ldots, m, \end{aligned} \tag{2.6}$$

where $f_0, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$ are convex functions. ■

Definition 2.8 (Quadratic programming)**.** A *quadratic programming* (QP) problem minimizes a convex quadratic function over a polyhedron. A QP problem can be written in the following form:

$$\begin{aligned} \min_{x} \quad & \frac{1}{2} x' H x + h' x \\ \text{s.t.} \quad & Gx \leq b \\ & G_e x \leq b_e \end{aligned} \tag{2.7}$$

where $H \in \mathbb{R}^{n \times n}$ is symmetric, $H \geq 0$, $h \in \mathbb{R}^n$, $G \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $G_e \in \mathbb{R}^{m_e \times n}$ and $b_e \in \mathbb{R}^m_e$. ■

Definition 2.9 (Active constraint)**.** Given the QP problem (2.7) and $\bar{x} \in \mathbb{R}^n$, the constraint $G_i x \leq b_i$ is *active* at $\bar{x}$ if the equality $G_i \bar{x} = b_i$ holds, otherwise it is *inactive*. ■

Definition 2.10 (Active set)**.** Given the QP problem (2.7) and $\bar{x} \in \mathbb{R}^n$, the *active set* $\mathcal{A}(\bar{x})$ is

$$\mathcal{A}(\bar{x}) = \{i \in \mathcal{K} \,|\, G_i \bar{x} = b_i\} \tag{2.8}$$

where $\mathcal{K} = \{1, \ldots, m\}$ is the set of constraint indexes. The active-set at the optimal solution x^* of the QP problem (2.7) is called *optimal active-set* and it is be denoted by $\mathcal{A}^*$. ■

Definition 2.11 (Inactive set)**.** Given the QP problem (2.7) and $\bar{x} \in \mathbb{R}^n$, the *inactive set* $\mathcal{I}(\bar{x})$ is

$$\mathcal{I}(\bar{x}) = \{i \in \mathcal{K} \,|\, G_i \bar{x} < b_i\}, \tag{2.9}$$

and the relation $\mathcal{I} = \mathcal{K} \setminus \mathcal{A}$ holds. ■

Definition 2.12 (Polyhedral partition)**.** Given a polyhedron $\Theta \subseteq \mathbb{R}^{n_\theta}$, the collection of sets $\{\Theta_1, \ldots, \Theta_s\}$ is said a *polyhedral partition* of Θ if Θ_i is a polyhedron, $\Theta_i \subseteq \mathbb{R}^{n_\theta}$, $\forall i = 1, \ldots, s$, $\cup_{i=1}^{s} \Theta_i = \Theta$, and $\mathring{\Theta}_i \cap \mathring{\Theta}_j = \emptyset$, $\forall i, j = 1, \ldots, s$, $i \neq j$. ■

Definition 2.13 (Integer piecewise constant function)**.** A function $n : \Theta \to \mathbb{N}$, $\Theta \subseteq \mathbb{R}^{n_\theta}$, is *integer piecewise constant* (IPWC) if there exist a polyhedral partition $\Theta_1, \ldots, \Theta_s$ of Θ and a set of integers $\{n_1, \ldots, n_s\} \subset \mathbb{N}$ such that

$$n(\theta) = \min_{i \in \{1,\ldots,s\}:\ \theta \in \Theta_i} \{n_i\} \tag{2.10}$$

for all $\theta \in \Theta$. Note that the "min" in (2.10) avoids possible multiple definition of $n(\theta)$ on overalapping boundaries $\Theta_i \cap \Theta_j \neq \emptyset$. ■

Definition 2.14 (Piecewise affine function)**.** A function $h : \Theta \to \mathbb{R}^n$, $\Theta \subseteq \mathbb{R}^{n_\theta}$, is said *piecewise affine* (PWA) if there exist an IPWC function $n : \Theta \to \{n_1, \ldots, n_s\}$ defined over a polyhedral partition $\Theta_1, \ldots, \Theta_s$ of Θ and s pairs (F_i, f_i), $F_i \in \mathbb{R}^{n \times n_\theta}$, $f_i \in \mathbb{R}^n$, $i \in \{n_1, \ldots, n_s\}$, such that

$$h(\theta) = F_{n(\theta)}\theta + f_{n(\theta)} \tag{2.11}$$

for all $\theta \in \Theta$. It is said *piecewise constant* if $F_i = 0$, $\forall i \in \{n_1, \ldots, n_s\}$. ■

Chapter 3

Model predictive control

MPC has been addressed as the most impactful methodology for process control in the recent history, as confirmed by the survey paper [93]. In the refining, chemical and petrochemical industry it is now considered a technology. MPC owes this success to its conceptual simplicity, and the ability to effectively handle multivariable systems with input and output constraints. MPC is more a control idea than a specific controller. Indeed, with MPC it is meant the family of controllers that make explicit use of the system model to predict its future behavior and generate the input sequence by applying the receding horizon technique. For this reason there are several and much different extensions of the classical MPC, each one with its own open questions and challenges. For a comprehensive overview on the recent achievements and future developments of MPC, please refer to the survey paper [5]. Among many active areas of research, embedded MPC is seen as one of the more impactful future promises, because of the interest in several engineering areas. The most important open challenge for embedded MPC is controlling those systems with high sampling rates, when running on control units with scarce memory and computational power. This chapter covers the most common formulation used for embedded MPC, with a Linear Time Invariant (LTI) model of the system, whose inputs and outputs are constrained in a polyhedral space. This formulation is widely used because it leads to a QP problem, for which several memory and computationally efficient solvers exist. Formulations based on linear parameter varying systems are common as well, but due to the need to build the QP problem at each iteration, they are more difficult to be used in embedded control [14, 94].

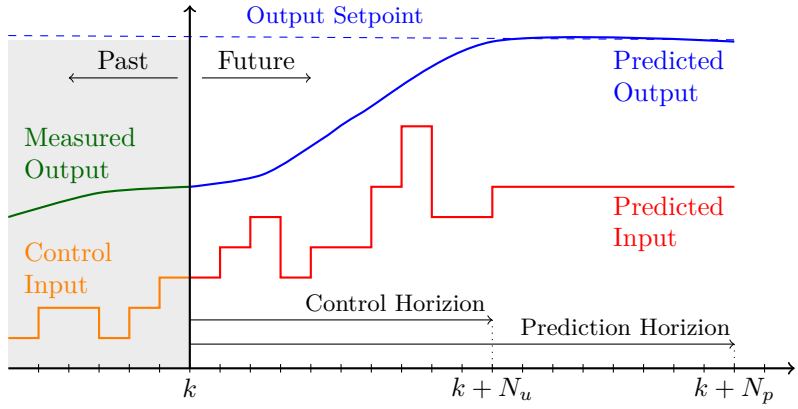


Figure 3.1: The open-loop optimization problem solved by MPC at each the sampling instant k.

3.1 Problem formulation

This section covers briefly the standard LTI-MPC problem. For classic reference textbooks, please refer to [3, 4]. Consider a tracking problem for the discrete-time LTI system in the state-space form:

$$x_{k+1} = Ax_k + B_u u_k \tag{3.1a}$$

$$y_k = Cx_k \tag{3.1b}$$

with $x \in \mathbb{R}^{n_x}$ the state vector, $u \in \mathbb{R}^{n_u}$ the input vector, $y \in \mathbb{R}^{n_y}$ the output vector and A, B_u, and C the dynamics matrices of appropriate dimension. The objective of the controller is to steer the outputs y to track the references y^{ref}. Figure 3.1 shows the operation of MPC at time step k. The set of future inputs is computed by optimizing a cost function of the control performances, such as the tracking error and the control effort. The cost function is minimized subject to the constraints imposed by the dynamics of the system, i.e. the LTI model (3.1), and constraints on both inputs and outputs. Only the inputs at time-step k are actually applied to the system, discarding the rest of the sequence. The procedure is repeated again at the next sampling time, shifting the horizon by one step (receding horizon). The number of future steps for which MPC computes the prediction at time k, is called *prediction*

horizon and it is denoted by N_p. On the other hand, the number of "free moves" of the inputs to be optimized is called *control horizon*, denoted by N_u, and it is chosen such that $1 \leq N_u \leq N_p$. As shown in the figure, for the prediction instants greater then $k + N_u$, the last free move is "frozen" to the past value. This is done to reduce the computational effort required to solve the optimization problem, because the size of the primal vector depends on N_u. Therefore, the optimization problem solved by MPC at each sampling instant is:

$$\begin{aligned}
\min_{\Delta u} \quad & \sum_{i=0}^{N_p-1} \left(\|W_y(y_{k+i+1|k} - y^{\text{ref}}_{k+i+1|k})\|_2^2 + \|W_u(u_{k+i|k} - u^{\text{ref}}_{k+i|k})\|_2^2 \right) + \\
& + \sum_{h=0}^{N_u-1} \|W_{\Delta u} \Delta u_{k+h|k}\|_2^2 && \text{(3.2a)} \\
\text{s.t.} \quad & x_{k|k} = x_k, && \text{(3.2b)} \\
& x_{k+i+1|k} = A x_{k+i|k} + B_u u_{k+i|k}, && \text{(3.2c)} \\
& y_{k+i+1|k} = C x_{k+i+1|k}, && \text{(3.2d)} \\
& \Delta u_{k+i|k} = u_{k+i|k} - u_{k+i-1|k} && \text{(3.2e)} \\
& \Delta u_{k+N_u+j|k} = 0, \quad j = 0, \ldots, N_p - N_u - 1, && \text{(3.2f)} \\
& \Delta u_{k+h|k} \in \mathbb{D}, && \text{(3.2g)} \\
& u_{k+i|k} \in \mathbb{U}, && \text{(3.2h)} \\
& y_{k+i+1|k} \in \mathbb{Y}, && \text{(3.2i)} \\
& i = 0, \ldots, N_p - 1, && \text{(3.2j)} \\
& h = 0, \ldots, N_u - 1, && \text{(3.2k)}
\end{aligned}$$

where W_y, W_u, and $W_{\Delta u}$ are square weight matrices, with $W_{\Delta u}$ invertible, $x_{k+i|k}$ denotes the prediction of the variable x at time $k + i$ based on the information available at time k, x_k is the current state, $\Delta u_{k+i|k}$ is the vector of the input increments, with $u_{k-1|k} = u(k-1)$, $y^{\text{ref}}_{k+i+1|k}$ and $u^{\text{ref}}_{k+i|k}$ are the previewed references for the outputs and the inputs, respectively, and $\mathbb{D}$, $\mathbb{U}$, and $\mathbb{Y}$ are polyhedral sets of constraints on input increments, inputs, and outputs, respectively.

3.2 Condensed Optimization Problem

Problem (3.2) can be cast in a standard QP problem, by arranging the cost function and the constraints such that only the vector of the input increments prediction appears explicitly in the formulation. This is the so-called *condensed form*, and it is usually the preferred one in linear MPC because many QP solvers exist to solve efficiently that optimization problem. In the next, the main steps to derive a condensed formulation starting from the original problem (3.2) are presented. The use of the input increments as optimization variables, eases the design of a cost function for tracking purposes and allows one to weight and constrain also the rate of change of the inputs [3,4]. Define the different predictions on outputs, inputs, and references, relative to time k, in a compact form such as:

$$\Delta \underline{\boldsymbol{u}}_k = \left[\Delta u_{k|k}, \ldots, \Delta u_{k+N_u-1|k}\right]' \tag{3.3a}$$

$$\underline{\boldsymbol{u}}_k = \left[u_{k|k}, \ldots, u_{k+N_p-1|k}\right]' \tag{3.3b}$$

$$\underline{\boldsymbol{u}}_k^{\text{ref}} = \left[u_{k|k}^{\text{ref}}, \ldots, u_{k+N_p-1|k}^{\text{ref}}\right]' \tag{3.3c}$$

$$\underline{\boldsymbol{y}}_k = \left[y_{k+1|k}, \ldots, y_{k+N_p|k}\right]' \tag{3.3d}$$

$$\underline{\boldsymbol{y}}_k^{\text{ref}} = \left[y_{k+1|k}^{\text{ref}}, \ldots, y_{k+N_p|k}^{\text{ref}}\right]' \tag{3.3e}$$

and let the output prediction performed at time k, and relative to the future time instant $k+i$, be defined as:

$$y_{k+i|k} = C\left[A^{k+i}x_k + \sum_{h=k}^{k+i} A^{k+i-1}B_u\left(u_{k-1} + \sum_{j=k}^{h}\Delta u_{j|k}\right)\right]. \tag{3.4}$$

The vector of predicted outputs (3.3d) can be therefore computed as:

$$\underline{\boldsymbol{y}}_k = S_u \Delta \underline{\boldsymbol{u}}_k + S_{u1} u_{k-1} + S_x x_k \tag{3.5}$$

where

$$S_u = \begin{bmatrix} CB_u & 0 & \dots & 0 \\ CB_u + CAB_u & CB_u & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ \sum_{j=0}^{N_p-1} CA^j B_u & \sum_{j=0}^{N_p-2} CA^j B_u & \dots & \sum_{j=0}^{N_p-N_u} CA^j B_u \end{bmatrix}$$

$$S_{u1} = \begin{bmatrix} CB_u \\ CB_u + CAB_u \\ \vdots \\ \sum_{j=0}^{N_p-1} CA^j B_u \end{bmatrix}, \quad S_x = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N_p} \end{bmatrix}. \tag{3.6}$$

Let us denote the cost function of problem (3.2) by J. This cost function can be rewritten in a condensed form as:

$$J = \left(\underline{\boldsymbol{y}}_k - \underline{\boldsymbol{y}}_k^{\text{ref}}\right)' \bar{W}_y \left(\underline{\boldsymbol{y}}_k - \underline{\boldsymbol{y}}_k^{\text{ref}}\right) + \left(\underline{\boldsymbol{u}}_k - \underline{\boldsymbol{u}}_k^{\text{ref}}\right)' \bar{W}_u \left(\underline{\boldsymbol{u}}_k - \underline{\boldsymbol{u}}_k^{\text{ref}}\right) + \\ + \Delta\underline{\boldsymbol{u}}_k' \bar{W}_{\Delta u} \Delta\underline{\boldsymbol{u}}_k, \tag{3.7}$$

with $\bar{W}_{\Delta u}$, $\bar{W}_u$ and $\bar{W}_y$ extended weight matrices of appropriate dimension, in a block-diagonal form, such as:

$$\bar{W}_{\Delta u} = \begin{bmatrix} W_{\Delta u} & 0 & \dots & 0 \\ 0 & W_{\Delta u} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & W_{\Delta u} \end{bmatrix} \tag{3.8a}$$

$$\bar{W}_u = \begin{bmatrix} W_u & 0 & \dots & 0 \\ 0 & W_u & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & W_u \end{bmatrix} \tag{3.8b}$$

$$\bar{W}_y = \begin{bmatrix} W_y & 0 & \dots & 0 \\ 0 & W_y & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & W_y \end{bmatrix}. \tag{3.8c}$$

In order to eliminate the inputs from the cost function, the following transformation, based on the definition of input increment, is introduced:

$$\begin{bmatrix} u_{k|k} \\ u_{k+1|k} \\ \vdots \\ u_{k+N_u-1|k} \\ \vdots \\ u_{k+N_p-1|k} \end{bmatrix} = \underbrace{\begin{bmatrix} I & 0 & \dots & 0 \\ I & I & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ I & \dots & \dots & I \\ \vdots & \ddots & \ddots & \vdots \\ I & \dots & \dots & I \end{bmatrix}}_{T_u} \begin{bmatrix} \Delta u_{k|k} \\ \Delta u_{k+1|k} \\ \vdots \\ \Delta u_{k+N_u|k} \end{bmatrix} + \underbrace{\begin{bmatrix} I \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ I \end{bmatrix}}_{t_u} u(k-1). \tag{3.9}$$

Therefore, by replacing equation (3.5) and the transformation (3.9) into J defined as in (3.7), after some manipulations and discarding those terms that do not depend from the optimization variables $\Delta\underline{\boldsymbol{u}}_k$, the cost function can be rewritten as:

$$J = \frac{1}{2}\Delta\underline{\boldsymbol{u}}_k' \underbrace{2(S_u'\bar{W}_y S_u + T_u'\bar{W}_u T_u + \bar{W}_{\Delta u})}_{H} \Delta\underline{\boldsymbol{u}}_k + \Delta\underline{\boldsymbol{u}}_k' \Bigg(\underbrace{-2S_u'\bar{W}_y}_{h_1} \underline{\boldsymbol{y}}_k^{\text{ref}} + \\ + \underbrace{2S_u'\bar{W}_y S_x}_{h_2} x_k + \underbrace{2(S_u'\bar{W}_y S_{u1} + T_u'\bar{W}_u t_u)}_{h_3} u_{k-1} + \underbrace{2T_u'\bar{W}_u}_{h_4} \underline{\boldsymbol{u}}_k^{\text{ref}} \Bigg) \tag{3.10}$$

that is obviously a quadratic function in the optimization variables $\Delta\underline{\boldsymbol{u}}_k$. It is worth noticing that, in the case the use of input increments as optimization variables is not required, e.g. stabilization problem, a similar procedure can be followed to obtain a quadratic cost function of $\underline{\boldsymbol{u}}_k$.

As far as the constraints, the three polyhedral sets $\mathbb{D}$, $\mathbb{U}$ and $\mathbb{Y}$ can be

written considering the definition of a polyhedron, as in the following:

$$\mathbb{D} = \{\Delta u \mid L_{\Delta u}\Delta u \leq l_{\Delta u}\} \tag{3.11a}$$
$$\mathbb{U} = \{u \mid L_u u \leq l_u\} \tag{3.11b}$$
$$\mathbb{Y} = \{y \mid L_y \Delta u \leq l_y\}. \tag{3.11c}$$

These constraints have to be enforced for the corresponding prediction and control horizons, such as:

$$L_{\Delta u}\Delta u(i) \leq l_{\Delta u} \quad \forall i = k, \ldots, k + N_u \tag{3.12a}$$
$$L_u u(i) \leq l_u \quad \forall i = k, \ldots, k + N_p - 1 \tag{3.12b}$$
$$L_y y(i) \leq l_y \quad \forall i = k + 1, \ldots, k + N_p. \tag{3.12c}$$

Similarly to the derivation of cost function, all the constraints must be written with respect to the optimization variables Δu, exploiting equations (3.5) and (3.9). Therefore, the constraints (3.12) can be collected in the equivalent form:

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix} \Delta \underline{\boldsymbol{u}}_k \leq \begin{bmatrix} b_1 \\ b_2 + s_1 u_{k-1} + s_2 x_k \\ b_3 + s_3 u(k-1) \end{bmatrix}, \tag{3.13}$$

where the first set of inequalities regards the constraints on the inputs increments, the second the constraints on the inputs and the third those on the outputs. After some manipulations, the matrices that compose

the inequality (3.13) are found to be:

$$
G_1 = \begin{bmatrix} L_{\Delta u} & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & L_{\Delta u} \end{bmatrix}, \; b_1 = \begin{bmatrix} l_{\Delta u} \\ \vdots \\ l_{\Delta u} \end{bmatrix}, \; b_2 = \begin{bmatrix} l_u \\ \vdots \\ l_u \end{bmatrix},
$$

$$
G_2 = \begin{bmatrix} L_u & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ L_u & \dots & \dots & L_u \end{bmatrix}, \; s_1 = \begin{bmatrix} -L_u \\ \vdots \\ -L_u \end{bmatrix}, \; s_2 = \begin{bmatrix} -L_y C B_u \\ -L_y C (B_u + A B_u) \\ \vdots \\ -\sum_{j=0}^{N_p-1} L_y C A^j B_u \end{bmatrix},
$$

$$
G_3 = \begin{bmatrix} L_y C B_u & 0 & \dots & 0 \\ L_y C B_u + L y C A B_u & L_y C B_u & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots \\ \sum_{j=0}^{N_p-1} L_y C A^j B_u & \sum_{j=0}^{N_p-2} L_y C A^j B_u & \dots & \sum_{j=0}^{N_p-N_u} L_y C A^j B_u \end{bmatrix},
$$

$$
s_3 = \begin{bmatrix} -L_y A \\ -L_y A^2 \\ \vdots \\ -L_y A_p^N \end{bmatrix}, \; b_3 = \begin{bmatrix} l_y \\ \vdots \\ l_y \end{bmatrix}. \tag{3.14}
$$

Given all the previous assumptions, the optimization problem (3.2) can be cast into the following, equivalent, parametric QP problem:

$$
\begin{aligned} \min_{z} \quad & f(z_k) : \frac{1}{2} z_k' H z_k + z_k' F \theta_k \\ \text{s.t.} \quad & g(z_k) : G z_k - W \theta_k - w \leq 0 \end{aligned} \tag{3.15}
$$

where $\theta_k \in \Theta$ is the vector of parameters defined as

$$
\theta_k = [u_{k-1}, x_{k|k}, \underline{\boldsymbol{y}}_k^{\text{ref}}, \underline{\boldsymbol{u}}_k^{\text{ref}}]', \tag{3.16}
$$

where $\Theta \subset \mathbb{R}^{n_\theta}$ is a bounded set of interest, $z_k \in \mathbb{R}^{n_z}$ is the vector of optimization variables, with $z_k = [\Delta \underline{\boldsymbol{u}}_k, \rho_k]'$ and ρ_k the slack variable

Algorithm 1 MPC controller routine at time-step k

Input: Matrices H, F, G, W, w defining problem (3.15), matrices A, B_u, C defining model (3.1), gains M, L defining Kalman filter (3.17), $y_k, u_{k-1}, \underline{\boldsymbol{y}}_k^{\text{ref}}, \underline{\boldsymbol{u}}_k^{\text{ref}}, \hat{x}_{k|k-1}$.

1: $y_k \leftarrow$ collect the measures from the plant;
2: $\hat{x}_{k|k} \leftarrow \hat{x}_{k|k-1} + M(y_k - C\hat{x}_{k|k-1})$;
3: $\theta_k \leftarrow [u_{k-1}, x_{k|k}, \underline{\boldsymbol{y}}_k^{\text{ref}}, \underline{\boldsymbol{u}}_k^{\text{ref}}]'$;
4: $z_k \leftarrow$ solve
$$\begin{aligned} \min_z \quad & 0.5 z_k' H z_k + z_k' F \theta_k \\ \text{s.t.} \quad & G z_k - W \theta_k - w \leq 0 \end{aligned}$$
5: $\Delta u_k \leftarrow$ extract the first n_u components from z_k;
6: $u_k \leftarrow u_{k-1} + \Delta u_k$;
7: $\hat{x}_{k+1|k} \leftarrow A\hat{x}_{k|k-1} + B_u u_k + L(y_k - Cx_{k|k-1})$;

Output: The control signal u_k to apply to the system, and the estimation $\hat{x}_{k|k-1}$ for the next time step.

for the soft constraints on the outputs, [3], $H \in \mathbb{R}^{n_z \times n_z}$ is a symmetric and positive definite matrix, $F \in \mathbb{R}^{n_z \times n_\theta}$, $G \in \mathbb{R}^{m \times n_z}$, $W \in \mathbb{R}^{m \times n_\theta}$, $w \in \mathbb{R}^m$. For the rest of the thesis, unless differently specified, the solution of the MPC problem will be obtained by solving the parametric QP problem (3.15). However, even though this section has provided the guidelines to build the parametric QP problem starting from the original problem (3.2), many other features can be included to enrich the formulation, e.g. measured disturbances or blocking moves. Indeed, regardless of the particular setup, as long as the model is linear, the cost function is quadratic and the constraints are affine, the original formulation can be always cast into a parametric QP problem.

When controlling a system, the QP problems at different time steps k differs from each other in the vector of parameters θ_k, as the other matrices remain unaltered (this holds true for LTI systems). All the elements of θ_k are commonly available, except for the real state $x_{k|k}$ of the plant, which has to be estimated from the available measures. The standard choice to accomplish this task is a Kalman filter. Let L be the estimator gain, and M the innovation gain, the estimated state $\hat{x}_{k|k}$ can

be computed by implementing the following well-known equations:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + M(y_k - C\hat{x}_{k|k-1}) \tag{3.17a}$$

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k-1} + B_u u_k + L(y_k - Cx_{k|k-1}). \tag{3.17b}$$

Clearly $\hat{x}_{k|k}$ replaces $x_{k|k}$ into the parameters' vector θ_k (3.16). The complete routine to implement an MPC controller online is detailed in Algorithm 1. The conditions for the stability of this control scheme have been formalized in [95], where it is shown that nominal stability of MPC for linear systems is achieved either by adding a terminal cost and constraint, or by extending the prediction horizon.

Chapter 4

Online Optimization

The complexity of solving QP problem (3.15) online, in order to compute the optimal control move u_k for a given θ_k, can limit the use of MPC. In particular, this is a crucial issue for embedded MPC as already anticipated [6, 34]. In the last years, many solutions have been proposed to mitigate this computational burden. Explicit MPC can be used to presolve offline the optimization problem, through mpQP [20, 21], but its memory requirements grow exponentially with the number of imposed linear constraints limiting the approach to small MPC problems (few inputs and short control horizon, few constraints). If the problem is not small enough, the alternative is solving online the optimization, which requires to embed a QP solver in the real-time control board [6]. The QP solver have to be fast and easy to code, require little memory and provide tight bounds on the worst-case execution time [26, 62, 96]. In the case their cost and lack of flexibility are affordable for the specific application, FPGAs with their fast clock rate and parallelized computations can make the speed of the solver a less crucial issue [27, 97, 98]. However, even in that case, the optimization algorithm must be carefully chosen. The literature for QP solvers is very rich, and the recent advances in convex optimization are contributing to its fast development [37, 99]. In addition, the last years have seen a flourishing of toolkits, some of which commercialized, for the rapid prototypation of algorithms tailored to solve embedded MPC problems [25, 40, 100]. Some popular methods have been compared in [101], highlighting their speed and memory occupancy in the case of a control unit with on-board volatile and non-volatile memory. As far as embedded MPC, active-set methods are usually considered outperforming in terms of speed [26, 37, 38, 102]. Moreover, they provide a very accurate solution without the need of

preconditioning when the problem is ill-conditioned [39]. A valuable alternative are interior-point methods, which are less affected by the problem's dimension with respect to active-set methods. However, the overhead for a single iteration is higher, due to the larger number of variables necessary, and therefore, for the size of QP problems arising in embedded applications, active-set algorithms are usually considered faster [103, 104]. The rest of the thesis will be focusing on novel results regarding active-set methods, and their use for embedded MPC. Therefore, in Section 4.1 we recall the generalities of dual active-set methods for solving strictly convex QP's of the form (3.15). In Section 4.2 a well known dual active-set solver, i.e. the Goldfarb-Idnani (GI) algorithm [32], is reconsidered in a parametric formulation that will be useful for the derivation of the complexity certification algorithm of Chapter 5.

4.1 Dual active-set algorithms

In order to find the optimizer of the QP problem (3.15), active-set methods solve a sequence of equality constrained subproblems. The idea is to iteratively make steps towards the solution by solving a reduced problem with only the equality constraints in the current active set, which we indicate as $\mathcal{A}^q$ for the q-th iteration of the solver. The optimal solution is reached when the current active set equals the optimal one, i.e. $\mathcal{A}^q \equiv \mathcal{A}^*$. The reason is that the inactive constraints do not affect the optimal solution, thus the original QP and the reduced equality-constrained QP with share the same optimizer. At every iteration q of the solver, a violated constraint p is added to the active set, and all the *blocking constraints* are removed from it. A constraint is said to be blocking if it belongs to $\mathcal{A}^{q-1}$ and prevents p to be active without violating primal or dual feasibility. The constraint p is selected such that:

$$p \leftarrow \begin{cases} 0 \text{ if } g_i(z^q) \leq 0, \ \forall i \in \mathcal{I} \\ h \in \mathcal{I} \,|\, g_h(z^q) > 0 \text{ otherwise.} \end{cases} \tag{4.1}$$

that is, p is equal to 0 if there are no violated constraints, otherwise it assumes the index of a violated constraint. Let $\mathcal{V}^{q-1}$ be the set of violated constraints at the previous iteration $q-1$. After selecting a

new violated constraint $p \in \mathcal{V}^q \subseteq \mathcal{I}^q$, the step size and direction are derived by solving the following equality constrained QP problem:

$$\begin{aligned} \min_{z} \quad & \frac{1}{2} z^{q\prime} H z^q + \theta' F' z^q \\ \text{s.t.} \quad & G_{\mathcal{A}^q} z^q = W_{\mathcal{A}^q} \theta + w_{\mathcal{A}^q} \end{aligned} \tag{4.2}$$

where the updated working set is $\mathcal{A}^q = \{\mathcal{A}^{q-1} \setminus \mathcal{B}^q\} \cup p$, with $\mathcal{B}^q$ the set of blocking constraints removed at iteration q from $\mathcal{A}^{q-1}$. The primal-dual pair (z^q, π^q), with π the vector of dual variables, is the solution of problem (4.2), if and only if it is the solution of the Karush-Kuhn-Tucker (KKT) system:

$$\underbrace{\begin{bmatrix} H & G'_{\mathcal{A}^q} \\ G_{\mathcal{A}^q} & 0 \end{bmatrix}}_{\text{KKT}(\mathcal{A}^q)} \begin{bmatrix} z^q \\ \pi^q \end{bmatrix} = \begin{bmatrix} -F\theta_k \\ W_{\mathcal{A}^q}\theta_k + w_{\mathcal{A}^q} \end{bmatrix} \tag{4.3}$$

where $\text{KKT}(\mathcal{A}^q)$ the so called *KKT-matrix*. Active-set algorithms can be categorized in two big families: *primal* and *dual* feasible methods. The first start from a primal feasible optimizer z^0 and solve iteratively the KKT system (4.3) until dual feasibility is reached [35, 91], maintaining the primal feasibility in the sub-iterates. A *Phase I* is needed to find a primal feasible z^0, that translates into solving a linear programming problem. On the contrary, dual feasible active-set solvers start from a dual feasible point, which is readily available (i.e. the unconstrained solution) and iterate (4.3) to reach primal feasibility, maintaining dual feasibility in the sub-iterates [32, 33]. This means that dual active-set algorithms produce an infeasible solution if they are stopped prematurely, e.g. if the time slot available to compute the control action is terminated. However, despite this drawback, dual active-set methods are sensibly faster than primal ones because they usually require less iterations, and do not need *Phase I*, saving also the memory to code the linear programming solver [32]. Finally, thanks to the possibility to choose the constraint to add to the active set at each iteration, dual methods are much less affected by the *cycling* problem, that can arise in degeneracy cases [105]. For all these reasons, dual methods are the preferred choice in embedded MPC. Active-set slovers are further categorized depending

on how they solve the KKT system (4.3). As far as the direct solution of KKT matrix is concerned, *range-space* or *null-space* methods are the most common strategies, which refer to the subspaces updated during the iterations. The range-space is defined by vectors that are linear combinations of the rows of G [32,35]. The null-space is defined by vectors orthogonal to the rows of G [34]. Range-space algorithms performs better when few constraints are in the active set, whereas null-space methods take advantage from a large number of active constraints. The computational load of each iteration is often directly proportional to the dimension of either the null-space or the range-space [35]. Algorithm 2 describes the general steps shared by all dual active-set methods that add the violated constraints one by one. Even if this aspect will not be covered in this thesis, dual active-set algorithms are receiving lot of attention also in the fields of mixed integer quadratic programing, and sequential quadratic programming, thanks to their ability to be easily warm-started. These results are of utmost interest for non-linear and hybrid MPC [106–110].

4.2 Goldfarb-Idnani parametric algorithm

In this section we briefly recall the GI algorithm, which is an efficient and numerically stable dual active-set solver for positive definite quadratic programming [32]. In the specific, the GI algorithm is reconsidered here in a parametric version, so to analyze later in Chapter 5 how its iterations for solving (3.15) depend on θ_k. Being a dual range-space solver, it is particularly suited for quadratic programs that arise from MPC problems, where typically $m > n_z$. Range-space approaches solve (4.3) by the explicit inversion of KKT($\mathcal{A}$). Step (4.3) is iteratively solved starting from $\mathcal{A}^{q-1}$ and dropping the blocking constraints one-by-one, until the constraint p can be added to the active set without violating the dual feasibility.

Remark 1. We denote by q the number of the solver's iterations, which is increased each time a constraint is added to the active set, cf. Algorithm 2. Then, we denote by j the index that is increased each time a constraint is either added or dropped from the active set, therefore accounting also for sub-iterations of the algorithm, where clearly $j \geq q$.

Algorithm 2 Generic dual active-set solver

Input: Matrices H, F, G, W, w, θ_k defining problem (3.15).

1: Set $z^0 \leftarrow -H^{-1}F\theta_k$, $\mathcal{A}^0 \leftarrow M_{0,0}$, $\mathcal{I}^0 \leftarrow \mathcal{K}$, $q \leftarrow 0$;
2: $\mathcal{V}^0 \leftarrow \{i \in \mathcal{I}^0 \,|\, G_i z^0 > b_i\}$;
3: choose

$$p \leftarrow \begin{cases} 0 \text{ if } g_i(z^q) \leq 0, \ \forall i \in \mathcal{I}^q \\ h \in \mathcal{I}^q \,|\, g_h(z^q) > 0 \text{ otherwise;} \end{cases}$$

4: **if** $p = 0$ **return** $z^* \leftarrow z^q$; **end if**;
5: $q \leftarrow q + 1$;
6: Find the set of blocking constraints $\mathcal{B}^q$;
7: $\mathcal{A}^q \leftarrow \{\mathcal{A}^{q-1} \setminus \mathcal{B}^q\} \cup \{p\}$;
8: $\mathcal{I}^q \leftarrow \{\mathcal{I}^{q-1} \setminus p\} \cup \{\mathcal{B}^q\}$;
9: Solve

$$\begin{bmatrix} H & G'_{\mathcal{A}^q} \\ G_{\mathcal{A}^q} & 0 \end{bmatrix} \begin{bmatrix} z^q \\ \pi^q \end{bmatrix} = \begin{bmatrix} -F\theta_k \\ W_{\mathcal{A}^q}\theta_k + w_{\mathcal{A}^q} \end{bmatrix};$$

10: **if** Step 9 has no solution **then return** infeasible;
11: **else** go to Step 3;
12: **end if**

Output: The optimal solution z^* or infeasibility codition.

The explicit inversion $\mathrm{KKT}(\mathcal{A}^j)^{-1}$ of system (4.3) is:

$$\mathrm{KKT}(\mathcal{A}^j)^{-1} = \begin{bmatrix} I & -H^{-1}G'_{\mathcal{A}^j} \\ 0 & I \end{bmatrix} \begin{bmatrix} H^{-1} & 0 \\ 0 & \mathcal{S}_H^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -G_{\mathcal{A}^j}H^{-1} & I \end{bmatrix} \tag{4.5}$$

where $\mathcal{S}_H = -G_{\mathcal{A}^j}H^{-1}G'_{\mathcal{A}^j}$ is the Schur complement of H. From (4.5) we define the two operators:

$$G^*_{\mathcal{A}^j} = \left(G_{\mathcal{A}^j}H^{-1}G'_{\mathcal{A}^j}\right)^{-1} G_{\mathcal{A}^j}H^{-1} \tag{4.6a}$$

$$\mathcal{H}^j = H^{-1}\left(I - G'_{\mathcal{A}^j}G^*_{\mathcal{A}^j}\right) \tag{4.6b}$$

where $G^*_{\mathcal{A}^j}$ is the Moore-Penrose pseudoinverse of $G_{\mathcal{A}^j}$ under the transformation $y = H^{1/2}x$ and $\mathcal{H}$ is the inverse Hessian operator in the active set space. The GI solver operation is summarized in Algorithm 3, where at each sub-iteration a new vector z^j and a set of active constraints $\mathcal{A}^j$ are found starting from z^{j-1}, $\mathcal{A}^{j-1}$ and a violated constraint p. The

primal-dual update (z^j, π^j) is defined so to make the constraint p active while maintaining dual feasibility. This is done by setting z^j and π^j as:

$$z^j = z^{j-1} + \alpha^j \Delta z^j \tag{4.7a}$$

$$\pi^j = \begin{bmatrix} \pi^{j-1} \\ \pi_p^{j-1} \end{bmatrix} + \alpha^j \begin{bmatrix} \Delta\pi^j \\ 1 \end{bmatrix} \tag{4.7b}$$

where $\alpha^j \in \mathbb{R}_+$ is the step size, and Δz^j, $\Delta\pi^j$ are the update directions in the primal and dual space, respectively. The primal and dual directions are computed as:

$$\Delta z^j = -\mathcal{H}^{j-1} G_p' \tag{4.8a}$$

$$\Delta\pi^j = -G^*_{\mathcal{A}^{j-1}} G_p' \tag{4.8b}$$

and the step size $\alpha^j = \min\{\alpha_1^j, \alpha_2^j\}$, with

$$\alpha_1^j = \min\left\{ \min_{l \in \{1,\dots,\#\mathcal{A}^{j-1}\}} \left\{ -\frac{\pi_l^{j-1}}{\Delta\pi_l^j} \middle| \Delta\pi_l^j < 0 \right\}, \infty \right\} \tag{4.9a}$$

$$\alpha_2^j = \frac{W_p\theta + w_p - G_p z^{j-1}}{G_p \Delta z^j}, \tag{4.9b}$$

is chosen such that $\pi^j \geq 0$ and, if possible, the p-th constraint becomes active, i.e. $g_p(z^j) = 0$.

Remark 2. The*dual step length* α_1 is the maximum step that can be taken without violating dual feasibility, and the *primal step length* α_2 is the step required to make the p-th constraint active.

Equations (4.9) guarantee that $\alpha^j > 0$. In [32] the authors prove that, being $g_p(z^j)$ the violated constraint, the following relations hold

$$g_p(z^j) > g_p(z^{j+1}) > g_p(z^{j+h}) = 0 \tag{4.10a}$$

$$f(z^j) < f(z^{j+1}) < f(z^{j+h}) \tag{4.10b}$$

between two iterations j and $j+h$ of Algorithm 3 in which constraints are dropped for h sub-iterations, that is $\alpha^{j+i} = \alpha_1^{j+i} < \infty, \forall i = 0, \dots, h-1$, and constraint $g_p(z^j)$ is added at sub-iteration $j+h$, that is $\alpha^{j+h} = \alpha_2^{j+h} < \infty$. The blocking constraint k dropped during one sub-iteration

is selected according to:

$$k = \min \underset{l=1,\dots,\#\,\mathcal{A}^{j-1}}{\arg\min} \left(-\frac{\pi_l^{j-1}}{\Delta\pi_l^j} \middle| \Delta\pi_l^j < 0 \right). \tag{4.11}$$

Three different situations can occur during the j-th sub-iteration:

- if $\alpha^j \equiv \alpha_1^j$ the k-th constraint is blocking and it must be dropped, therefore only a partial step can be taken;
- if $\alpha^j \equiv \alpha_2^j$ a full step can be taken, that is $\mathcal{A}^j = \mathcal{A}^{j-1} \cup \{p^j\}$;
- if $\alpha^j \equiv \infty$ the original QP problem is infeasible.

Computing $\mathcal{H}$ and $G_{\mathcal{A}}^*$ from scratch at each iteration can be too computationally demanding, therefore the Cholesky factorization of $\mathcal{H}$ and the QR factorization of $G_{\mathcal{A}}^*$ are iteratively built by Algorithm 3. Let $H = LL'$ be the Cholesky factorization of the primal Hessian and define:

$$L^{-1} G_{\mathcal{A}^j} = \begin{bmatrix} Q_1^j & Q_2^j \end{bmatrix} \begin{bmatrix} R^j \\ 0 \end{bmatrix} \tag{4.12a}$$

$$J^j = \begin{bmatrix} J_1^j & J_2^j \end{bmatrix} = \begin{bmatrix} L^{-T} Q_1^j & L^{-T} Q_2^j \end{bmatrix}. \tag{4.12b}$$

Then the two operators $\mathcal{H}^j$, $G_{\mathcal{A}^j}$ can be written calculated as:

$$\mathcal{H}^j = J_2^j J_2^{j\prime} \tag{4.13a}$$

$$G_{\mathcal{A}^j}^* = R^{-1,j} J_1^{j\prime}, \tag{4.13b}$$

where $J_1^j \in \mathbb{R}^{n_z \times \#\,\mathcal{A}^j}$, $J_2 \in \mathbb{R}^{n_z \times \#\,\mathcal{I}^j}$, and $J^0 = L^{-T}$, $R^0 = M_{0,0}$. Therefore, the GI algorithm only needs to iteratively update J^j and R^j, so to compute the primal and dual directions as:

$$\Delta z^j = -J_2^j J_2^{j\prime} G_p' \tag{4.14a}$$

$$\Delta\pi^j = -R^{-1,j} J_1^{j\prime} G_p'. \tag{4.14b}$$

The factorizations updates (4.14) can be performed through Householder reflections, or Givens rotations. The latter has been used in the next.

In contrast with primal methods, dual active-set solvers have the degree of freedom to select which constraint becomes active at each iteration, see Step 3 of Algorithm 3. The order the constraints are added to the active set changes the number of iterations, although it does not affect the convergence [111]. As long as the added constraints is violated, condition (4.10) guarantees convergence no matter how $g_p(z^j) > 0$ is selected. In the literature two techniques are commonly used to choose p: the *most-violated* constraint selection rule

$$p \leftarrow \begin{cases} 0 \text{ if } g_i(z^j) \leq 0, \ \forall i \in \mathcal{I} \\ \min \arg\max_{i:\ g_i(z^j)>0} \{g_i(z^j)\} \text{ otherwise,} \end{cases} \tag{4.15a}$$

and the *first-violated* constraint selection rule

$$p \leftarrow \begin{cases} 0 \text{ if } g_i(z^j) \leq 0, \ \forall i \in \mathcal{I} \\ \min_{i \in \mathcal{K}} \{i |\ g_i(z^j) > 0\} \text{ otherwise.} \end{cases} \tag{4.15b}$$

In (4.15a), the "min" before the "arg max" imposes to select the smallest index in case of multiple maximizers (the same rule is adopted in (4.11) for choosing the index k in case of multiple minima). Chapter 6 will detail the aspect of the selection rules for violated constraints, and will present a novel rule which improves the computationally efficiency with respect to the standard ones.

Algorithm 3 Goldfarb-Idnani (GI) QP solver

Input: Matrices H, F, G, W, w, θ_k defining problem (3.15).

1: Compute Cholesky factorization $LL' = H$; $J_2^0 \leftarrow L^{-T}$;
2: $z^0 \leftarrow -J_2^0 J_2^{0\prime} F\theta$, $\mathcal{A}^0 \leftarrow \emptyset$, $q \leftarrow 0$, $j \leftarrow 0$;
3: choose

$$p \leftarrow \begin{cases} 0 \text{ if } g_i(z^j) \leq 0, \ \forall i \in \mathcal{I} \\ h \in \mathcal{I} \,|\, g_h(z^j) > 0 \text{ otherwise;} \end{cases}$$

4: **if** $p = 0$ **then return** $\mathcal{A}^* \leftarrow \mathcal{A}^j$; **end if**;
5: $q \leftarrow q + 1$, $\pi_p^j \leftarrow 0$;
6: $j \leftarrow j + 1$, $\Delta z^j \leftarrow -J_2^j (J_2^j)' G_p'$;
7: **if** $\# \mathcal{A}^j > 0$ **then** $\Delta \pi^j \leftarrow -R^{-1,j} (J_1^j)' G_p'$; **end if**;
8: **if** $(\Delta \pi^j \geq 0 \quad \textbf{or} \quad \# \mathcal{A}^j = 0)$ **then** $\alpha_1^j \leftarrow \infty$;
9: **else**

$$k \leftarrow \min \underset{l=1,\ldots,\# \mathcal{A}^{j-1}}{\arg\min} \left(-\frac{\pi_l^{j-1}}{\Delta \pi_l^j} \Big| \Delta \pi_l^j < 0 \right), \ \alpha_1^j \leftarrow -\frac{\pi_k^{j-1}}{\Delta \pi_k^j};$$

10: **end if**;
11: **if** $\Delta z^j = 0$ **then** $\alpha_2^j \leftarrow \infty$;
12: **else** $\alpha_2^j \leftarrow \dfrac{W_p \theta + w_p - G_p z^{j-1}}{G_p \Delta z^j}$;
13: **end if**;
14: $\alpha^j \leftarrow \min(\alpha_1^j, \alpha_2^j)$;
15: **if** $\alpha^j = \infty$ **then return infeasible**; **end if**;
16: **if** $(\alpha_2^j = \infty)$ **then**
17: $\quad \pi^j \leftarrow \pi^{j-1} + \alpha^j \Delta \pi^j$;
18: $\quad \mathcal{A}^j \leftarrow \mathcal{A}^{j-1} \setminus \{k\}$, update J_1^j, J_2^j, R^j, **go to** Step 6;
19: **end if**;
20: Set

$$z^j \leftarrow z^{j-1} + \alpha^j \Delta z^j;$$

$$\pi^j \leftarrow \begin{bmatrix} \pi^{j-1} \\ \pi_p^{j-1} \end{bmatrix} + \alpha^j \begin{bmatrix} \Delta \pi^j \\ 1 \end{bmatrix};$$

21: **if** $\alpha^j = \alpha_2^j$ **then**
22: $\quad \mathcal{A}^j \leftarrow \mathcal{A}^{j-1} \cup \{p\}$, update J_1^j, J_2^j, R^j, **go to** Step 3;
23: **else if** $\alpha^j = \alpha_1^j$ **then**
24: $\quad \mathcal{A}^j \leftarrow \mathcal{A}^{j-1} \setminus \{k\}$, update J_1^j, J_2^j, R^j, **go to** Step 6;
25: **end if**.

Output: Primal solution $z^* = z^j$, dual solution $\pi^* = \pi^j$, and optimal active set $\mathcal{A}^* = \mathcal{A}^j$, or infeasibility; number $N = q$ of iterations.

Chapter 5

Complexity certification for active-set solvers

In embedded MPC, it is of utmost importance to certify that the QP solver always provides the input sequence within a certain execution time, which must be lower than the sampling interval or lower than the task time reserved in a multi-purpose control unit. Even if this holds true in every situation, embedded MPC is more affected by the issue due to the low computational power and high sampling frequency which make the available time to solve the problem really short. The topic of the complexity certification is being addressed by the very recent literature, as it can be a reason that prevents MPC implementation in real applications [62, 63, 112, 113]. A solution could be to interrupt the solver after a fixed amount of iterations which are known to fit within the prescribed time, [44, 114]. However, the consequent suboptimal or even infeasible solution is not acceptable in lot of situations. Furthermore, fixing a maximum number of iterations is equal to know the worst-case time only with solvers whose iterations have the same complexity. Unfortunately, this does not hold for active-set methods, where the computational load of a single iteration depends on the size of the current active set, and the number of sub-iterations necessary to remove the blocking constraints. Therefore, this chapter presents a novel algorithm able to exactly calculate the computational complexity of a dual active-set method, which translates into knowing the worst-case number of iterations and FLOPs [115]. Besides the needs of embedded MPC, the algorithm has an important role in optimization field as well. Indeed, in spite of the empirical evidence of their efficiency, a theoretical computational complexity of active-set methods that is useful in prac-

tice is not available yet [64, 66, 67]. They are considered, by experience, algorithms that exhibit a polynomial behavior on average [64, 66], but, as shown in the famous Klee-Minty problem, they can even display exponential worst-case number of iterations on contrived problems [65]. This aspect could prevent their use in embedded applications, e.g. in favor of interior-point methods, for which a theoretical polynomial iteration bound is available [28, 116], or gradient projection methods, for which tight worst-case bounds have been derived [24]. The algorithm presented in this chapter builds upon the machinery employed in mpQP to get explicit MPC solutions, where the parameter space is divided into polyhedral regions that share the same optimal active set, and is close in spirit with the multiparametric solution approach developed in [117] for linear programs. Here we consider how "parametric" steps of the GI algorithm propagate in the parameter space during iterations. Even if the algorithm is derived starting from the parametric version of the GI solver, all dual active-set algorithms that add the violated constraints one-by-one into the active set can be certified by the same algorithm. Indeed, as long as the selection rule for the violated constraint to add is the same [118], they perform the same number of iterations. The difference is how they solve (4.3), that translates into a different complexity per iteration. To the best of the author knowledge, this is the first contribution that proposes the exact computation of worst-case flops and iterations for QP solvers. Section 5.1 details the certification algorithm, and Section 5.2 provides results of its application to well known MPC problems.

5.1 Complexity certification algorithm

Given a set Θ of parameters that perturb the QP problem (3.15), the algorithm here derived allows to characterize exactly the worst-case behavior of Algorithm 3 parametrically with respect to θ, in the case the violated constraints are selected according to one of the standard rules (4.15a) or (4.15b).

Theorem 5.1.1. *Let $\Theta \subseteq \mathbb{R}^{n_\theta}$ be a polyhedron. Then a finite number*

$N_{\max}$, *exists such that*

$$N_{\max} = \max_{\theta \in \Theta} N(\theta), \tag{5.1}$$

that is Algorithm 3 terminates in at most $N_{\max}$ *steps for all* $\theta \in \Theta$.

Proof. Convergence of Algorithm 3 for each given $\theta \in \Theta$ is proved in [32, Th. 3]: because of (4.10), the same combination of active constraints $\mathcal{A}^{j-1}$ cannot occur during the iterations of the algorithm; since $q \leq j$ and the number $N_{\mathcal{A}}$ of active set combinations is finite, the algorithm finds the solution in at most $N_{\max} = \max_{\theta \in \Theta}\{N(\theta)\} \leq N_{\mathcal{A}}$ iterations for all $\theta \in \Theta$. □

The question is, how large this $N_{\max}$ can be. Theorem 5.1.1 proves only the existence of such an $N_{\max}$, but in most of the cases its value is much smaller than the number of all the possible combinations of active constraints $N_{\mathcal{A}}$, i.e. a polynomial complexity is observed experimentally on average [64, 66, 67].

Lemma 5.1.2. *Let* $f, g : \Theta \to \mathbb{R}$ *be PWA functions over a polyhedron* $\Theta \subseteq \mathbb{R}^n$. *Then* $h = \min\{f, g\}$ *is also PWA.*

Proof. Let $\{\Theta_1, \ldots, \Theta_s\}$ and $\{\Phi_1, \ldots, \Phi_t\}$ be the polyhedral partitions associated with f and g, respectively, and $\{p_1, \ldots, p_s\}$, $\{k_1, \ldots, k_t\}$ define the corresponding IPWC functions n and m. Let F_i, f_i, $i = 1, \ldots, s$ and G_j, g_j, $j = 1, \ldots, t$, define f and g, respectively. Consider the polyhedra

$$\Psi_{ij}^f = \{\theta \in \Theta : \theta \in \Theta_i \cap \Phi_j,\ (F_i - G_j)\theta \leq g_j - f_i\} \tag{5.2a}$$

$$\Psi_{ij}^g = \{\theta \in \Theta : \theta \in \Theta_i \cap \Phi_j,\ (G_i - F_j)\theta \leq f_j - g_i\} \tag{5.2b}$$

and let $\{\Psi_1, \ldots, \Psi_v\}$ be the set of all polyhedra Ψ_{ij}^f, Ψ_{ij}^g having a nonempty interior. It is easy to show that $\{\Psi_1, \ldots, \Psi_v\}$ defines a polyhedral partition of Θ. By letting $\ell_i = i$, $i = 1, \ldots, v$ the IPWC function $\ell : \Theta \in \{1, \ldots, v\}$ such that

$$\ell(\theta) = \min_{i \in \{1,\ldots,v\}:\ \theta \in \Psi_i} \{i\}$$

we have that

$$h(\theta) = \begin{cases} F_i\theta + f_i & \text{if} \quad \theta \in \Psi_{\ell(\theta)} = \Psi_{ij}^f \quad \text{for some } j \in \{1, \dots, t\} \\ G_j\theta + g_j & \text{if} \quad \theta \in \Psi_{\ell(\theta)} = \Psi_{ij}^g \quad \text{for some } i \in \{1, \dots, s\} \end{cases}$$

and therefore that h is PWA. □

The following Theorem 5.1.3 characterizes the behavior of one sub-iteration j of Algorithm 3 as a function of the parameters θ.

Theorem 5.1.3. *Let $z^{j-1} : \Theta^j \to \mathbb{R}^n$ and $\pi^{j-1} : \Theta^j \to \mathbb{R}^m$ be affine functions on a polyhedron $\Theta^j \subseteq \mathbb{R}^{n_\theta}$, and let $\mathcal{A}^{j-1} \subseteq \mathcal{K}$, $\mathcal{V}^{j-1} \subseteq \mathcal{K} \setminus \mathcal{A}^{j-1}$. The following properties hold:*

i) By letting $p(\theta) = 0$ if no constraint is violated ($\mathcal{V}^{q-1} = \emptyset$), the index $p : \Theta^j \to \mathbb{N} \cup \{0\}$ selected according to (4.15a) *or* (4.15b) *is IPWC;*

ii) The step directions Δz^j, $\Delta \pi^j$ obtained from (4.8) *are PWC;*

iii) the index $k : \Theta^j \to \mathbb{N}$ selected according to (4.11) *is IPWC.*

iv) the step-size $\alpha^j : \Theta^j \to \mathbb{R} \cup \{+\infty\}$ defined in (4.9) *is PWA;*

v) the functions $z^j : \Theta^j \to \mathbb{N}^n$, $\pi^j : \Theta^j \to \mathbb{R}^m$ defined by (4.7) *are PWA.*

Proof. *i)* Let $z^j(\theta) = A_z\theta + b_z$ define the affine function z^j over Θ^j. Then, $g_\ell(z^j) = G_\ell z^j - W_\ell\theta - w_\ell = (G_\ell A_z - W_\ell)\theta + G_\ell b_z - w_\ell$ is also affine, $\forall \ell \in \mathcal{I}$. Let $\bar{G}_\ell = G_\ell A_z - W_\ell$, $\bar{w}_\ell = w_\ell - G_\ell b_z$ and

$$\Theta_\ell^1 = \{\theta \in \Theta^j : \ (\bar{G}_\ell - \bar{G}_h)\theta \geq \bar{w}_h - \bar{w}_\ell, \ \bar{G}_\ell\theta \geq \bar{w}_\ell, \ \forall h \in \mathcal{I}, \ h \neq \ell\} \tag{5.3a}$$

$$\Theta_\ell^2 = \{\theta \in \Theta^j : \ \bar{G}_h\theta \leq \bar{w}_h, \ \bar{G}_\ell\theta \geq \bar{w}_\ell \forall h \in \mathcal{I} \cap \{1, \dots, \ell - 1\}\}. \tag{5.3b}$$

Consider the polyhedra

$$\begin{aligned} \Theta_0 &= \{\theta \in \Theta^j : \ \bar{G}_\ell\theta \leq \bar{w}_\ell, \ \forall \ell \in \mathcal{I}\} \\ \Theta_\ell &= \begin{cases} \Theta_j^1 & \text{if} \quad p = \arg\max\{\bar{G}_\ell\theta - \bar{w}_\ell\} \\ \Theta_j^2 & \text{if} \quad p = \min_{i \in \mathcal{K}}\{i| \ g_i(z^j) > 0\}. \end{cases} \end{aligned} \tag{5.4}$$

The polyhedra $\{\Theta_\ell\}_{\ell=0}^m$ define a polyhedral partition of Θ^j as: (i) they are included in Θ^j by construction, (ii) each $\theta \in \Theta_{p(\theta)}$, where $p(\theta) = 0$ if $\bar{G}_\ell \theta \geq \bar{w}_\ell$, $\forall \ell \in \mathcal{I}$, or $p = \arg\max\{\bar{G}_\ell \theta - \bar{w}_\ell\}$ if rule (4.15a) is used, or $p = \min_{i \in \mathcal{K}}\{i |\ g_i(z^j) > 0\}$ in case (4.15b) is adopted, (iii) the interiors $\mathring{\Theta}_\ell$ are obtained by changing "$\geq$" to "$>$" in (5.4), so no θ can belong simultaneously to two different interiors of polyhedra. Since both rules in (4.15) are equivalent to setting

$$p(\theta) = \min_{\ell \in \{0,\ldots,m\}:\ \theta \in \Theta_\ell} \{\ell\}, \tag{5.5}$$

function p is IPWC.

ii) From (4.6) we have that $\mathcal{H}^{j-1}$ and $G^*_{\mathcal{A}^{j-1}}$ are independent from θ, so the same holds for J_1^j, J_2^j and $R^{-1,j}$. Since G_p depends on p that by (5.5) is IPWC, the step directions obtained from (4.8) are PWC functions.

iii) Since π^{j-1} is an affine function of θ, say $\pi^{j-1}(\theta) = A_\pi \theta + b_\pi$, and $\Delta\pi^j$ is PWC, their negative ratio $-\frac{\pi^{j-1}(\theta)}{\Delta\pi_l^j}$ is PWC over the partition of Θ^j defined by (5.4), for all $l \in \mathcal{P}^{j-1} = \{k_1, \ldots, k_t\} = \{l \in \mathcal{A}^{j-1} : \Delta\pi_l^j < 0\}$, where $t = \#\,\mathcal{P}^{j-1}$. Let $(C_{l,1}, d_{l,1})$, …, $(C_{l,s}, d_{l,s})$ define the corresponding affine terms of the PWC ratios and consider the polyhedra

$$\begin{array}{rcl} \Phi_{i,\ell} & = & \{\theta \in \Theta_\ell :\ (C_{k_i,\ell} - C_{k_h,\ell})\theta \leq d_{k_h,\ell} - d_{k_i,\ell}, \\ & & \forall h = 1, \ldots, t,\ h \neq i,\ \ell = 1, \ldots, s\}. \end{array} \tag{5.6}$$

Each set of polyhedra $\{\Phi_{i,\ell}\}_{i=1}^t$ also defines a polyhedral partition of Θ_ℓ, for all $\ell = 1, \ldots, s$. Since (4.11) is equivalent to setting

$$k(\theta) = \min_{i \in \{1,\ldots,t\}:\ \theta \in \cup_{\ell \in \mathbb{P}^{j-1}} \Phi_{i,\ell}} \{k_i\} \tag{5.7}$$

k is also an IPWC function on Θ^j.

iv) If $\#\,\mathcal{A}^{j-1} = 0$ or $\Delta\pi^j \geq 0$ then $\alpha_1^j(\theta) = +\infty$ for all $\theta \in \Theta^j$, cf. Step 8 of Algorithm 3. Otherwise,

$$\alpha_1^j(\theta) = C_{k(\theta)}\theta + d_{k(\theta)} \tag{5.8}$$

and therefore α_1^j is PWA. Similarly, since z^{j-1} is affine, we have that

$$\alpha_2^j(\theta) = C_{h(\theta)}\theta + d_{h(\theta)} \tag{5.9}$$

with

$$C_{h(\theta)} = \frac{W_{p(\theta)} - G_{p(\theta)} A_z}{G_{p(\theta)} \Delta z^j} \tag{5.10a}$$

$$d_{h(\theta)} = \frac{w_{p(\theta)} - G_{p(\theta)} b_z}{G_{p(\theta)} \Delta z^j} \tag{5.10b}$$

is also PWA. If $\alpha_1^j(\theta) = +\infty$ then $\alpha^j(\theta) = \alpha_2^j(\theta)$ for all $\theta \in \Theta^j$ and so α^j is PWA. Otherwise, Lemma 5.1.2 proves that $\alpha^j = \min\{\alpha_1^j, \alpha_2^j\}$ is PWA. As a result, we consider the set of polyhedra

$$\tilde{\Phi}_{0,\ell} = \{\theta \in \Theta_\ell : \; C_{k_j,\ell} - C_{h,\ell} \leq d_{h,\ell} - d_{k_j,\ell}, \; \forall j = 1, \ldots, t\} \tag{5.11a}$$

$$\tilde{\Phi}_{i,\ell} = \Phi_{i,\ell} \setminus \tilde{\Phi}_{0,\ell}, \; \forall i = 1, \ldots, t \tag{5.11b}$$

such that $\{\tilde{\Phi}_{i,l}\}_{i=0}^t$ defines a polyhedral partition of Θ_ℓ. v) The functions z^j and π^j defined in (4.7) are the sum of an affine and PWA function, and are therefore PWA. □

Theorem 5.1.4. *Let $\Theta \subseteq \mathbb{R}^{n_\theta}$ and z^0, π^0 be affine functions of θ on Θ. Then each iterate z^j, π^j defined in* (4.7) *is PWA for all $j \in \mathbb{N}$ such that Algorithm 3 is executed. Moreover, $N : \Theta \to \{0, \ldots, N_{\max}\}$ is IPWC.*

Proof. Theorem is proven by induction. Since z^0, π^0 are affine, they are also PWA. Assume z^{j-1}, π^{j-1} are PWA functions defined over a polyhedral partition $\{\Theta_i^{j-1}\}_{i=1}^{s^{j-1}}$ of Θ. Therefore, z^{j-1} and π^{j-1} are affine on each polyhedron Θ_i^{j-1}, and by property (v) of Theorem 5.1.3 we have that z^j, π^j are PWA functions defined over a polyhedral partition $\{\Theta_{ih}^{j-1}\}_{h=1}^{t_i^{j-1}}$ of Θ_i^{j-1}. As the collection of sets

$$\left\{\Psi_i^j\right\}_{i=1}^{L^j} = \left\{\Theta_{1h}^{j-1}\right\}_{h=1}^{t_1^{j-1}} \cup \ldots \cup \left\{\Theta_{s^{j-1}h}^{j-1}\right\}_{h=1}^{t_{s^{j-1}}^{j-1}},$$

where $L^j = \sum_{i=1}^{s^{j-1}} t_i^{j-1}$, is a polyhedral partition of Θ^{j-1}, it follows that z^j, π^j are PWA over Θ^{j-1}.

Since by Theorem 5.1.1 the number of recursive partitioning is finite, Θ gets partitioned in a finite number of polyhedral sets $\Psi_1, \ldots, \Psi_M$. Each polyhedron Ψ_i is characterized by either $\mathcal{V}^j = \emptyset$ (optimal solution found) or $\alpha^j = \alpha_1^j = \infty$ (infeasibility) and by the number $N_i \leq N_{\max}$ of recursive splitting it took to get defining Ψ_j. Then, the function N

such that $N(\theta) = N_i$ if $\theta \in \Psi_i$, $i = 1, \ldots, M$ is IPWC. □

The following corollary confirms a well known property of the optimizer z^* proved in [20, Theorem 4].

Corollary 5.1.5. *The multiparametric QP solution vector z^* of (3.15) is PWA with respect to θ over a subset of Θ.*

Proof. Easily follows since, for each $i = 1, \ldots, M$, either $z^*(\theta) = z^{N_i}(\theta)$ or the problem is infeasible, for all $\theta \in \Psi_i$. □

Thanks to all the previous results, we detail next the certification algorithm for active-set methods (3.15). Let T^h be the h-th tuple defined by the following arguments:

$$T^h = (\Theta^h, \mathcal{A}^h, A_z^h, b_z^h, A_\pi^h, b_\pi^h, J_1^h, J_2^h, R^h, q^h) \tag{5.12}$$

where Θ^h are polyhedra where each tuple is defined, and provide a partition of Θ. The algorithm iteratively constructs two lists of tuples corresponding to parameters $\theta \in \Theta$ for which the QP (3.15) has an optimal solution or is infeasible, called $\mathbb{T}$ and $\bar{\mathbb{T}}$ respectively. These lists are build by partitioning Θ in polyhedra depending on the behavior of each consecutive iteration q made by the GI Algorithm 3. The core of the certification algorithm is the way to split a given polyhedron Θ_ℓ, obtained as in (5.4) for the iteration q, in all the possible polyhedra that are either *feasible*, with constraint $p(\theta)$ active at $q+1$, or *infeasible*. Given Θ_ℓ, let Γ_ℓ^i and $\bar{\Gamma}_\ell^j$ define the i-th feasible and the j-th infeasible polyhedra. From Theorem 5.1.4 it follows that $\{\Gamma_\ell^i\}_{i=1}^{n_\gamma} \cup \{\bar{\Gamma}_\ell^j\}_{j=1}^{n_{\bar{\gamma}}}$ define a polyhedral partition of Θ_ℓ, where different polyhedra mean different sequence of constraints dropped before adding the violated constraint $p(\theta)$ to the current active set, or before verifying the QP infeasibility. Therefore, each tuple that describes the status of the solver at q in the particular polyhedron Θ^q is iteratively split into two sets of tuples $\{\mathcal{T}_{\Gamma,\ell}^i\}_{\ell=1,i=1}^{m,\ n_{\gamma,\ell}}$ and $\{\mathcal{T}_{\bar{\Gamma},\ell}^j\}_{\ell=1,j=1}^{m,\ n_{\bar{\gamma},\ell}}$, associated with the sets of polyhedra $\{\Gamma_\ell^i\}_{\ell=1,i=1}^{m,\ n_{\gamma,\ell}}$ and $\{\bar{\Gamma}_\ell^j\}_{\ell=1,j=1}^{m,\ n_{\bar{\gamma},\ell}}$. Let the following equations hold

$$(\tilde{C}, \tilde{d}) = \begin{cases} (C_k, d_k) \text{ if } \alpha^j \equiv \alpha_1^j \\ (C_h, d_h) \text{ if } \alpha^j \equiv \alpha_2^j, \end{cases} \tag{5.13}$$

Algorithm 4 Parametric GI's dual QP iteration

Input: T^q=$(\Theta^q, \mathcal{A}^q, A^q_z, b^q_z, A^q_\pi, b^q_\pi, J^q_1, J^q_2, R^q, q), G, W, w$

Define $\{\Theta_\ell\}_{\ell=0}^m$ as in (5.4) and p as in (5.5);
if $\mathring{\Theta}_0 \neq \emptyset$ **then**
  $T_o = (\Theta_0, \mathcal{A}^q, A^q_z, b^q_z, A^q_\pi, b^q_\pi, J^q_1, J^q_2, R^q, q)$;
end if
for $\ell = 1, \ldots, m$ such that $\mathring{\Theta}_\ell \neq \emptyset$ **do**
  $\mathcal{T}_\ell \leftarrow \{T^q\}$, $\mathcal{T}_{\Gamma,\ell} \leftarrow \emptyset$, $\mathcal{T}_{\bar{\Gamma},\ell} \leftarrow \emptyset$;
  while $\mathcal{T}_\ell \neq \emptyset$ **do**
    extract from $\mathcal{T}_\ell$ a tuple
    $T^j_\ell \leftarrow (\Theta^j_\ell, \mathcal{A}^j_\ell, A^j_{z,\ell}, b^j_{z,\ell}, A^j_{\pi,\ell}, b^j_{\pi,\ell}, J^j_{1,\ell}, J^j_{2,\ell}, R^j_\ell, q)$;
    $\Delta z^j_\ell \leftarrow -J^j_{2,\ell}(J^j_{2,\ell})' G_p$, $\Delta\pi^j_\ell \leftarrow -R^{-1,j}_\ell (J^j_{1,\ell})' G'_p$;
    if $(\Delta\pi^j_\ell \geq 0 \quad \textbf{or} \quad \#\mathcal{A}^j = 0)$ **then** $\alpha^j_{1,\ell} \leftarrow \infty$;
      if $\Delta z^j_\ell = 0$ **then** $\mathcal{T}_{\bar{\Gamma},\ell} \leftarrow \mathcal{T}_{\bar{\Gamma},\ell} \cup \{T^j_\ell\}$; **else**
        Update $A^{j+1}_{z,\ell}, b^{j+1}_{z,\ell}, A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}$ as in (5.14)–(5.15);
        Update $J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, \mathcal{A}^{j+1}_\ell$ adding constraint p;
        Set the tuple
    $T^{j+1}_{1,\ell} \leftarrow (\Theta^j_\ell, \mathcal{A}^{j+1}_\ell, A^{j+1}_{z,\ell}, b^{j+1}_{z,\ell}, A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}, J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, q{+}1)$;
        $\mathcal{T}_{\Gamma,\ell} \leftarrow \mathcal{T}_{\Gamma,\ell} \cup \{T^{j+1}_{1,\ell}\}$;
      end if;
    else
      partition Θ_ℓ as in (5.11), with k as in (5.7);
      if $\Delta z^j_\ell > 0$ **then**
        Update $A^{j+1}_{z,\ell}, b^{j+1}_{z,\ell}, A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}$ as in (5.14)–(5.15);
        Update $J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, \mathcal{A}^{j+1}_\ell$ adding constraint p;
        Set the tuple
    $T^{j+1}_{1,\ell} \leftarrow (\tilde{\Phi}^j_{0,\ell}, \mathcal{A}^{j+1}_\ell, A^{j+1}_{z,\ell}, b^{j+1}_{z,\ell}, A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}, J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, q{+}1)$;
        $\mathcal{T}_{\Gamma,\ell} \leftarrow \mathcal{T}_{\Gamma,\ell} \cup \{T^{j+1}_{1,\ell}\}$;
      end if;
      for $i = 1, \ldots, t$ with $\mathring{\Phi}_{i,\ell} \neq \emptyset$ **do**
        Update $A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}$ as in (5.15);
        Update $J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, \mathcal{A}^{j+1}_\ell$ removing constraint p;
        if $\Delta z^j_\ell = 0$ **then**
          Set the tuple
    $T^{j+1}_{2,\ell} \leftarrow (\tilde{\Phi}^j_{i,\ell}, \mathcal{A}^{j+1}_\ell, A^j_{z,\ell}, b^j_{z,\ell}, A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}, J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, q)$;
        else
          Update $A^{j+1}_{z,\ell}, b^{j+1}_{z,\ell}$ as in (5.14);
          Set the tuple
    $T^{j+1}_{2,\ell} \leftarrow (\tilde{\Phi}^j_{i,\ell}, \mathcal{A}^{j+1}_\ell, A^{j+1}_{z,\ell}, b^{j+1}_{z,\ell}, A^{j+1}_{\pi,\ell}, b^{j+1}_{\pi,\ell}, J^{j+1}_{1,\ell}, J^{j+1}_{2,\ell}, R^{j+1}_\ell, q)$;
        end if;
        $\mathcal{T}_\ell = \mathcal{T}_\ell \cup \{T^{j+1}_{2,\ell}\}$;
      end for;
    end if;
  end while
end for.

Output: Tuple T_o; sets of tuples $\{\mathcal{T}^i_{\Gamma,\ell}\}_{\ell=1,i=1}^{m,\ n_{\gamma,\ell}}$, $\{\mathcal{T}^j_{\bar{\Gamma},\ell}\}_{\ell=1,j=1}^{m,\ n_{\bar{\gamma},\ell}}$.

then, for each new tuples derived from T^{j-1}, the following parametric primal update is performed:

$$\begin{aligned} A_z^j &= A_z^{j-1} + \tilde{C}\Delta z^j \\ b_z^j &= b_z^{j-1} + \tilde{d}\Delta z^j, \end{aligned} \tag{5.14}$$

and the parametric dual update is defined by:

$$\begin{aligned} A_\pi^j &= \begin{cases} \begin{bmatrix} A_\pi^{j-1} \\ 0 \end{bmatrix} + \tilde{C} \begin{bmatrix} \Delta\pi^j \\ 1 \end{bmatrix}, & \textbf{if constraint is added;} \\ A_\pi^{j-1} + \tilde{C}\Delta\pi^j, & \textbf{if constraint is dropped;} \end{cases} \\ b_\pi^j &= \begin{cases} \begin{bmatrix} b_\pi^{j-1} \\ 0 \end{bmatrix} + \tilde{d} \begin{bmatrix} \Delta\pi^j \\ 1 \end{bmatrix}, & \textbf{if constraint is added;} \\ b_\pi^{j-1} + \tilde{d}\Delta\pi^j, & \textbf{if constraint is dropped.} \end{cases} \end{aligned} \tag{5.15}$$

Algorithm 4 characterizes the steps to split a given polyhedron Θ^q at a generic iteration q, and Algorithm 5 iterates Algorithm 4 in order to characterize the entire parameter set Θ. This provides the complete list of optimal and infeasible tuples that partition the Θ into polyhedra characterized by different steps of a dual active-set algorithm to reach the optimal solution, or to detect infeasibility. Therefore the following result holds,

Result 1. Consider the QP problem (3.15) for $\theta \in \Theta$ and let $\mathbb{T}$, $\bar{\mathbb{T}}$ be the lists of tuples generated by Algorithm 5. Then the dual active-set method of Algorithm 3 takes no more than

$$N_{\max} = \max_{T \in \mathbb{T} \cup \bar{\mathbb{T}}} \{q(T)\} \tag{5.16}$$

iterations to solve (3.15) or detect infeasibility for any $\theta \in \Theta$, and exactly $N_{\max}$ iterations for all $\theta \in \Theta(T)$ such that $q(T) = N_{\max}$, where $q(T)$ and $\Theta(T)$ denote the value q and set Θ associated with tuple T, respectively.

Even though the bound $N_{\max}$ is derived analyzing the parametric behavior of the GI algorithm, the same $N_{\max}$ applies to all dual active-set methods that add violated constraints one-by-one, and select p with the same rule used for generating $\{\Theta_\ell\}_{\ell=0}^m$ in (5.4). Indeed, once p is chosen, no matter how system (4.3) is solved the solution of the reduced

equality QP problem remains the same.

Taking a step forward, what really counts in embedded MPC is the worst-case CPU run-time, rather than the number of iterations. This is of particular interest in active-set methods, where the solver iterations have different execution time, and therefore the worst-case number of iterations does not necessarily correspond to the worst-case time. In order to address this issue, the certification algorithm provides also the exact number of flops n_{imp} required to reach the optimal solution in each tuple T^h defined as:

$$T^h = (\Theta^h, \mathcal{A}^h, A_z^h, b_z^h, A_\pi^h, b_\pi^h, J_1^h, J_2^h, R^h, q^h, n_{\text{imp}}^h). \tag{5.17}$$

where n_{imp} is incrementally updated at every sub-iteration j, and may be even further differentiated into the number of atomic and non-atomic operations, like square-roots. Additionally, multiple n_{imp} can be simultaneously computed, extending immediately the certification algorithm to other dual active-set methods for QP's, including range-space or null-space based methods [35,107], so that one can choose the best solver for a given MPC controller based on the one that has the least worst-case execution time. The use of information on n_{imp} allows to exactly certify the worst-case, by computing

$$n_{\text{imp}}^{\max} = \max_{T \in \mathbb{T} \cup \bar{\mathbb{T}}} \{n_{\text{imp}}(T)\} \tag{5.18}$$

where $n_{\text{imp}}(T)$ associates the value n_{imp} with tuple T. The value $n_{\text{imp}}^{\max}$ can be then used to derive the worst-case time, given a particular hardware architecture. Additionally, the following result holds.

Result 2. The certification algorithm also provides the multiparametric solution of (3.15) as a by-product. The convex polyhedra associated with each optimal active set $\mathcal{A}$, corresponding to the union of the regions $\theta(T)$, $T \in \mathbb{T}$, such that $\mathcal{A}(T) = \mathcal{A}$, can be immediately computed as in [20], by imposing the primal and dual feasibility conditions:

$$(GA_z(T) - W)\theta \leq w - Gb_z(T) \tag{5.19a}$$

$$-A_\pi(T)\theta \leq b_\pi(T) \tag{5.19b}$$

for some T such that $\mathcal{A}(T) = \mathcal{A}$.

Given the output of the certification algorithm, Result 2 allows to compute the explicit solution of the MPC problem, so that one can optimally decide whether implicit or explicit MPC should be chosen for the implementation, given the certified complexity and memory occupancy of both the solutions.

Algorithm 5 GI's QP solver certification

Input: Matrices H, F, G, W, w defining problem (3.15) and polyhedral set Θ of parameters.

Compute Cholesky factorization $LL' = H$, $J_2^0 \leftarrow L^{-T}$;
$\mathbb{T} \leftarrow \emptyset$, $\bar{\mathbb{T}} \leftarrow \emptyset$;
$\mathcal{C} \leftarrow \{(\Theta, \emptyset, -H^{-1}F, 0, M_{0,0}, M_{0,0}, M_{0,0}, J_2^0, M_{0,0}, 0)\}$;
while $\mathcal{C} \neq \emptyset$ **do**
  $T^q \leftarrow$ extract tuple from $\mathcal{C}$;
  Execute Algorithm 4 with input data from T^q;
  $\mathbb{T} \leftarrow \mathbb{T} \cup \{T_o\}$;
  $\bar{\mathbb{T}} \leftarrow \bar{\mathbb{T}} \cup \{\mathcal{T}^i_{\bar{\Gamma},\ell}\}_{\ell=1,i=1}^{m,\ n_{\bar{\gamma},\ell}}$;
  $\mathcal{C} \leftarrow \{\mathcal{T}^i_{\Gamma,\ell}\}_{\ell=1,i=1}^{m,\ n_{\gamma,\ell}}$;
end while

Output: List of *optimal* tuples $\mathbb{T}$, list of *infeasible* tuples $\bar{\mathbb{T}}$.

5.2 Examples

The certification Algorithm 5 is tested on four well known MPC problems taken from the Model Predictive Control Toolbox for MATLAB® demos library: an *inverted pendulum* control problem, consisting of controlling a single-input-multi-output inverted pendulum on a cart, with a measured disturbance and input constraints; the *DC motor* control problem, concerning the control of a DC servomechanism under voltage and shaft torque constraints [119]; the *nonlinear demo* control problem, consisting of controlling a multi-input multi-output nonlinear plant with a linear MPC formulation; and the multivariable *AFTI-16* aircraft control problem, characterized by an open-loop unstable pole and constraints on both inputs and outputs [120]. We use here exactly the same settings used in the toolbox, so that the results are reproducible. Table 5.1 collects the dimension of the problems.

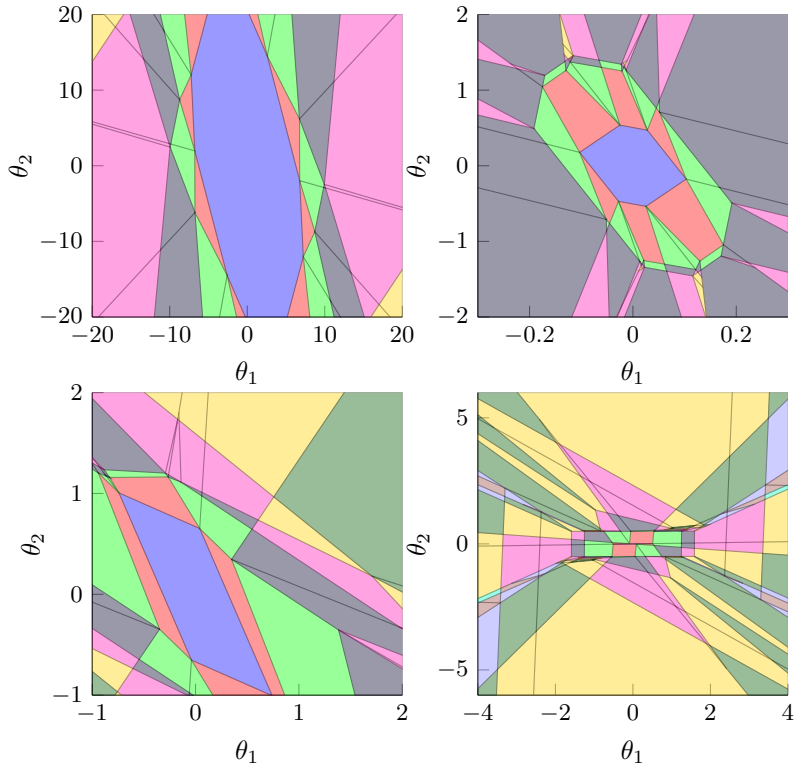


Figure 5.1: Results of the explicit certification algorithm: Partition of the parameter set Θ based on the number of iterations required by the GI QP solver (same color = same number of QP iterations).

To better clarify the operation of Algorithm 5, Figure 5.2 shows a sample 2D section of the polyhedra associated with optimal tuples $\{T^i\}_{i=1}^{\#\mathbb{T}}$ for each tested control problem, where regions corresponding to nodes that share the same number of iterations have the same color. As expected, $\{\bar{T}^i\}_{i=1}^{\#\bar{\mathbb{T}}} = \emptyset$ for all the problems, that is the GI algorithm applied to the four examples is always feasible in the set of interest (all output constraints are treated as soft constraints in the toolbox). Instead by merging the regions that share the same optimal active-set, we get the explicit MPC solution, for which Figure 5.2 shows the same 2D section considered before. Therefore, the final goal of the certification

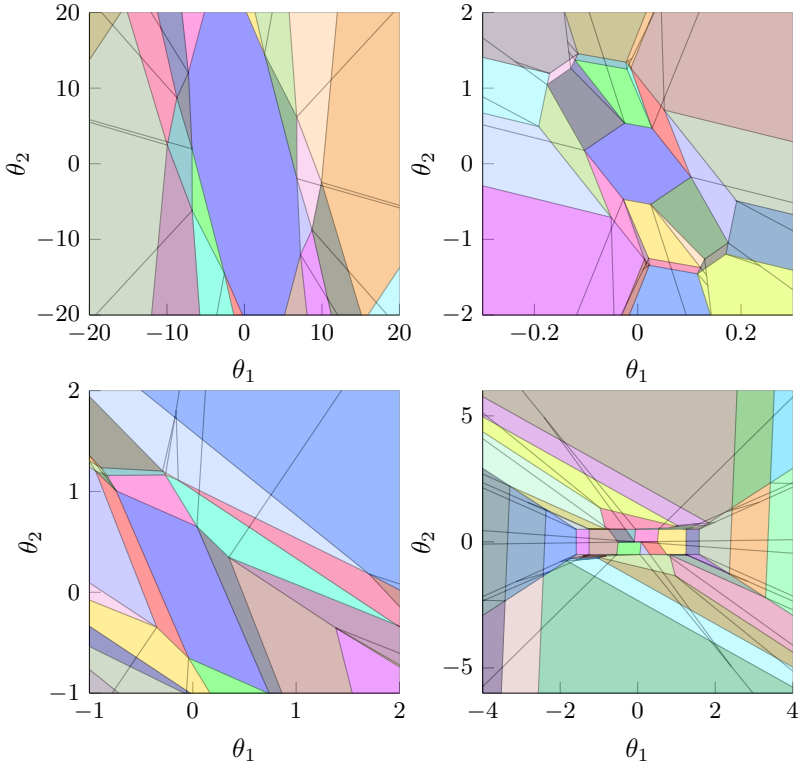


Figure 5.2: Results of the explicit certification algorithm: Partition of the parameter set Θ based on the explicit solution (same color = same optimal active set).

algorithm is to have results on MPC problems, like those collected in Table 5.2. In the specific, the worst-case number of iterations $N_{\max}$, the single and double precision memory allocation m_{imp}, and the worst-case number of flops $n_{\text{imp}}^{\max}$ are reported for implicit MPC solved with the GI algorithm (selecting p as in (4.15a)). The table also collects information on explicit MPC, namely the number of regions of the explicit solution n_r, its memory occupancy m_{exp} in single and double precision, and the worst case number of flops $n_{\text{exp}}^{\max}$.

In order to obtain the results on explicit MPC, let the explicit optimal

Table 5.1: Dimensions of MPC problems for complexity certification algorithm

	inv. pend.	DC motor	nonlinear demo	AFTI 16
n_x	5	4	5	6
n_u	1	1	3	2
n_y	2	2	2	2
N_p	50	10	5	10
N_u	5	2	2	2
n	5	3	6	5
m	10	10	18	12
n_θ	9	6	10	10

Table 5.2: Results from the complexity certification algorithm of GI solver (max violated selection rule (4.15a)).

	inv. pend.	DC motor	nonlin. demo	AFTI 16
Explicit MPC				
n_r	87	67	215	417
$n_{\text{exp}}^{\max}$ $(\pm, *)$	3382	1689	9184	16434
m_{exp} 16|32 bit (kb)	28.2|55.4	15.7|30.3	148.8|296.6	215.0|430.0
Implicit MPC				
$N_{\max}$	11	9	13	16
$n_{\text{imp}}^{\max}$ $(\pm, *, \div)$|sqrt	3809|27	2082|9	7747|37	7807|33
m_{imp} 16|32 bit (kb)	12.2|15.4	10.9|12.9	14.3|19.7	12.5|16.0

control law be the PWA function:

$$u^*_{\text{exp}}(\theta) = \{K_i\theta + c_i, \quad \forall\theta \in \mathbb{P}_i, \quad i = 1, \ldots, n_r\} \tag{5.20}$$

with $K_i \in \mathbb{R}^{n_u \times n_\theta}$, $c_i \in \mathbb{R}^{n_u}$, $\{\mathbb{P}_i\}_{i=1}^{n_r}$ a polyhedral partition of Θ such that $\mathbb{P}_i = \{\theta \in \mathbb{R}^{n_\theta} | E_i\theta \leq e_i, \forall i = 1, \ldots, n_r\}$, where $E_i \in \mathbb{R}^{n_e^i \times \theta}$, $e_i \in \mathbb{R}^{n_e^i}$ are a minimal polyhedral representation of (5.19). The worst-case number of flops $n_{\text{exp}}^{\max}$, needed to find and apply the optimal solution through explicit MPC, and the memory occupancy m_{exp} in kilobytes to store the PWA control law and the needed information for point

location, can be calculated as:

$$n_{\text{exp}}^{\max} = 2n_\theta(n_u + n_r) + \sum_{i=1}^{n_r} n_e^i \tag{5.21a}$$

$$m_{\text{exp}} = \frac{((n_\theta + 1)n_u + n_\theta + 1)n_r}{p_f} \tag{5.21b}$$

with $p_f = 32$ for double precision and $p_f = 64$ for single precision, under the assumption of using the explicit algorithm presented in [22]. For both implicit and explicit MPC, memory allocation also comprises the solver code, that, in the case of explicit MPC, is negligible (< 1 kB), whereas our implementation in C-code of Algorithm 3 takes approximately 8 kB. We count square roots separately from other arithmetic operations, as the associated computation time depends on the architecture. In conclusion, Table 5.2 allows the designer to certify for each problem if implicit MPC will be faster or not with respect to explicit MPC, and to select whether to use one or the other solution in terms of speed and memory occupancy.

Chapter 6

Certified fast embedded MPC

The research in convex optimization led to a set of new efficient solvers with appealing features for embedded MPC, like those covered in Chapter 5. However, in order to favor the spread of MPC in embedded applications, accelerating techniques have been developed as well, and last years count several methods to accelerate online QP solvers. A wide portion of them is closely related to the explicit solution. An example is qpOASES [34, 40], which reduces the number of iterations when the solutions of two successive QP problems are close enough. The Phase I of primal active-set method is therefore avoided by moving on an homotopy path in the parameter space. Another well-known solution is partial enumeration, or semi-explicit MPC, which combines a subset of the underlying explicit solution with an online solver [41–43]. Only some of the polyhedral partitions and gains (usually the ones at the most recent time steps) are stored in a fixed dimensional table, and before running the online solver, a point location is performed in this partial explicit solution. If it is successful, the computation time can be drastically reduced because the online optimization is not performed. The consequent drawback, is the direct dependence of performance from the memory allocated for the look-up table. Recently, the idea of exploiting an incomplete information of inactive constraints by saving a set of regions of activity has been proposed [45]. A number of regions equal to the constraints cardinality is needed in that case. The method has been further improved by showing that, if the cost function is a Lyapunov function for the closed-loop system, the inactive constraints can be derived without storing explicit regions [68]. Another different approach that can be classified as trade-off between explicit and implicit optimization has been presented in [44]. Here a PWA approximation of

the optimal control law is computed offline and used to warm start a primal active-set method with a limit on the maximum number of iterations. The direct drawback is the sub-optimal solution obtained when more iterations are needed.

However, all the aforementioned methods share some disadvantages that can be crucial for embedded applications. First, the trade-off between the stored data and the performance improvement is not usually clear, or even better certifiable [41,44]. Indeed, searching into the partial explicit solution takes time, and with many regions this point location procedure can be too costly to be beneficial. The second, and most important, limitation is the impossibility to guarantee improvements in the worst-case, unless opting for a sub-optimal solutions, e.g. [44]. This is due to the underlying concept of warm-start that is the base of most of the acceleration methods. In the worst-case, such as after a sudden change in the reference or in the measure disturbance, the solutions of successive QP problems can be arbitrary different [40, 68]. Therefore, even if these methods are viable options to reduce the average number of iterations, the behavior in the worst-case can be, potentially, even worse. If there is no guarantee with respect to the worst-case time, the usefulness of an accelerating technique in embedded MPC is very questionable.

By making use of some of the results obtained in Chapter 5, we present in the next two novel accelerating methods for MPC, which overcome the drawbacks discussed above. In Section 6.1 we propose a novel technique that combines implicit and explicit MPC, and that we refer to as Worst Case Partial Enumeration (WCPE)-MPC. As suggested by the name, WCPE-MPC certifies exactly the worst-case time improvement, contrarily to other semi-explicit techniques. Furthermore it provides always the optimal solution, and allows one to optimally select the trade-off between memory occupancy and speed enhancement, difficult in other cases. Then, in Section 6.2 we present a simple yet powerful method to accelerate implicit MPC that exploits the particular structure of the constraints derived from an MPC problem [111]. The developed technique consists of an heuristics-based priority rule for the selection of a violated constraint in dual active-set methods. It can reduce both the number of iterations and the computational demand of the single iteration. One of the advantages with respect to the state

of the art is that this technique relies only on the information coming from the solver at iteration q, therefore it does not depend neither from the explicit solution or the warm-start concept. This degree of freedom allows to combine it with other well known techniques, even WCPE-MPC. However, the main advantage of the method that makes it shining for embedded MPC is the possibility to be exactly certified, by exploiting the tools developed in Chapter 5. Therefore, even though it is an heuristic approach, the eventual improvement for the worst-case can be exactly verified before implementing it.

6.1 Worst-case partial enumeration MPC

Chapter 5 has described how to evaluate when there is a certified improvement into adopting an online solver or the explicit solution, being the memory occupancy and the flops needed to obtain the solution in the worst-case exactly computable. However, in some cases a solution that combines both the strategies can be outperforming. This is the main idea of partial enumeration, where a portion of the multiparametric solution is stored as a table and used in conjunction with an online solver [41]. The WCPE-MPC here proposed builds upon the complexity certification of Algorithm 5, in order to identify those regions that, if stored as explicit solution, will help improving the worst-case number of flops of an implicit approach. Being interested in the worst-case improvement, the partial solution that must be stored is of fixed dimensions, and not updated online, in contrast to what done in [41]. The basic idea of the proposed WCPE-MPC approach is to first search the optimal solution in the partial explicit PWA law, and, if the search fails, an online dual active-set solver is executed. The reason to use the a dual active-set method is the possibility to certify its execution time in those regions where the PWA law is not stored. Define $\bar{\mathbb{P}} = \mathbb{P}(\{1, \dots, n_r\}) \setminus (\emptyset, \{1, \dots, n_r\})$, with $\mathbb{P}(X)$ the power-set of X, and consider the partial explicit control law $u^*_{\text{exp},\mathcal{N}}(\theta)$ derived from (5.20):

$$u^*_{\text{exp},\mathcal{N}}(\theta) = \{K_j\theta + c_j, \quad \forall \theta \in \mathbb{P}_j, \quad \forall j \in \mathcal{N}\} \tag{6.1}$$

with $\mathcal{N} \in \bar{\mathbb{P}}$ the set of indexes corresponding to the affine functions of $u^*_{\text{exp}}(\theta)$ to be included in the partial explicit solution, where $\#\mathcal{N} < n_r$

and $\cup_{j\in\mathcal{N}}\{\mathbb{P}_j\} \subset \Theta$.

For each polyhedron $\mathbb{P}_j$ that defines the partial PWA function (6.1), let C_j be the list of "neighboring" polyhedra of $\mathbb{P}_j$, such as:

$$C_j = \{h \,|\, \mathbb{P}_h \text{ is a neighbor of } \mathbb{P}_j,\, h = 1, \dots, n_r\} \tag{6.2}$$

where $C_j \in \mathbb{N}_+^{n_{c_j}}$, and $\mathbb{P}_h$ is a neighbor of $\mathbb{P}_j$ if they share a facet. Consider the set of indexes $\mathcal{M}$, such as:

$$\mathcal{M} = \{h \mid h \in C_j,\ \forall j \in \mathcal{N}\} \tag{6.3}$$

and define the scalar PWA *descriptor function* $r(\theta) : \mathbb{R}^{n_\theta} \to \mathbb{R}$

$$r(\theta) = \{r_i(\theta) = \bar{K}_i\theta + \bar{c}_i \mid \theta \in \mathbb{P}_i, \quad i \in \mathcal{M}\} \tag{6.4}$$

where $\bar{K}_i \in \mathbb{R}^{n_\theta}$, $\bar{c}_i \in \mathbb{R}$ and $\bar{K}_j \neq \bar{K}_h, \forall h \in C_j,\ j \in \mathcal{N}$. The PWA descriptor function $r(\theta)$ is derived from the PWA optimal control law (5.20), cf. [121]. For each $\mathbb{P}_j$, consider the *ordering function* $O_j(\theta) = \{o_i^j(\theta) \mid i \in C_j\}$, such that:

$$o_i^j(\theta) = \begin{cases} +1 & \text{if } r_j(\theta) \geq r_i(\theta) \\ -1 & \text{if } r_j(\theta) < r_i(\theta), \end{cases} \tag{6.5}$$

with $j \in \mathcal{N}$ and $i \in \mathcal{M}$, and let $S_j = C_j(\bar{\theta})$, with $\bar{\theta} \in \mathring{\mathbb{P}}_j$, $\forall j \in \mathcal{N}$ be pre-calculated set. Given $u^*_{\text{exp},\mathcal{N}}(\theta)$, $r(\theta)$ and S_j, $\forall j \in \mathcal{N}$, Algorithm 6 presents the steps of the proposed WCPE-MPC.

The procedure is very similar to standard semi-explicit MPC, however what makes the difference is how $u^*_{\text{exp},\mathcal{N}}(\theta)$ is selected, so to guarantee an improvement of the worst-case number of flops $n^{\max}_{\text{imp}}$. Given n_r explicit regions, a total of $2^{n_r} - 2$ partial explicit solutions exist. Definition 6.1 and Theorem 6.1.1 allow to reduce the entire set to only those regions that *can* improve the implicit worst-case $n^{\max}_{\text{imp}}$.

Definition 6.1. Given a set of indexes $\mathcal{N} \in \bar{\mathbb{P}}$, let $n^{\max}_{\mathcal{N}}$ be the worst-case number of flops of WCPE-MPC Algorithm 6 with input argument the partial explicit solution $u^*_{\text{exp},\mathcal{N}}$, such that

$$n^{\max}_{\mathcal{N}} = n^{\max}_{\text{exp},\mathcal{N}} + n^{\max}_{\text{imp},\mathcal{N}}, \tag{6.6}$$

Algorithm 6 WCPE-MPC Algorithm

Input: Matrices H, F, G, W, w, θ defining problem (3.15). $u^*_{\text{exp},\mathcal{N}}(\theta) = \{K_j\theta + c_j, \quad \forall\theta \in \mathbb{P}_j, \quad \forall j \in \mathcal{N}\}$, $r(\theta) = \{r_i(\theta) = \bar{K}_i\theta + \bar{c}_i \mid \theta \in \mathbb{P}_i, \quad i \in \mathcal{M}\}$, and $\{S_j\}_{j\in\mathcal{N}}$

1: $j \leftarrow 1$, $f \leftarrow$ TRUE;
2: **while** f **or** $j \leq \#\mathcal{N}$ **do**
3: $\quad O_j(\theta) \leftarrow \{o_i^j(\theta) \mid o_i^j(\theta) = \begin{cases} +1 & \text{if } r_j(\theta) \geq r_i(\theta) \\ -1 & \text{if } r_j(\theta) < r_i(\theta) \end{cases}, \ i \in C_j\}$;
4: $\quad$ **if** $O_j(\theta) = S_j$ **then**
5: $\quad\quad f \leftarrow$ FALSE;
6: $\quad$ **else**
7: $\quad\quad j \leftarrow j + 1$;
8: $\quad$ **end if**
9: **end while**
10: **if** $f =$FALSE **then**
11: $\quad$ **return** $u^* \leftarrow K_j\theta + c_j$;
12: **else**
13: $\quad z^* \leftarrow$ execute Algorithm 3 with inputs H, F, G, W, w, θ;
14: $\quad$ **return** $u^* \leftarrow$ first n_u rows of z^*;
15: **end if**

Output: Optimal control input u^*.

where $n^{\max}_{\text{exp},\mathcal{N}}$ is the required number of flops, in the worst-case, for the point location in the partially explicit solution, and $n^{\max}_{\text{imp},\mathcal{N}}$ is the worst-case number of flops for the online solver. ■

Lemma 6.1.1. *Given $\bar{\mathbb{P}}$ the collection of all the possible sets of indexes to build the partial explicit solution* (6.1)*, let $\{\mathcal{N}_i\}_{i=1}^{n_r-1}$, with $\mathcal{N}_i \in \bar{\mathbb{P}}, \forall\, i \in \{1, \ldots, n_r - 1\}$, be the reduced collection of sets of indexes defined such that*

$$\mathcal{N}_j = \mathcal{N}_i \cup h, \quad \forall j > i, \quad h \in \{1, \ldots, n_r\} \setminus \mathcal{N}_i \tag{6.7a}$$

$$n^{\max}_{\text{imp},\mathcal{N}_j} \leq n^{\max}_{\text{imp},\mathcal{N}_i}, \quad \forall j > i \tag{6.7b}$$

with $i \in \{1, \ldots, n_r - 2\}$ and $j \in \{2, \ldots, n_r - 1\}$. Then $\{\mathcal{N}_i\}_{i=1}^{n_r-1}$ are the only set of indexes for which the corresponding set of partial explicit solutions $u^*_{\text{exp},\mathcal{N}_i}(\theta)$ can guarantee that $n^{\max}_{\mathcal{N}_i} \leq n^{\max}_{\text{imp}}$ hold true.

Proof. Consider the v-th set of indexes $\mathcal{N}_v$ that defines the partial ex-

plicit solution $u^*_{exp,\mathcal{N}_v}(\theta)$. Let $n^{\max}_{\mathcal{N}_v} = n^{\max}_{\text{exp},\mathcal{N}_v} + n^{\max}_{\text{imp},\mathcal{N}_v}$ be the worst-case number of flops of Algorithm 6 with $u^*_{\text{exp},\mathcal{N}_v}(\theta)$ as the input argument, and define the list of tuples $\mathbb{T}_{\mathcal{N}_v} = \{T^i_{\mathcal{N}_v}\}_{i=1}^{\#\,\mathbb{T}_{\mathcal{N}_v}}$, such that

$$\mathbb{T}_{\mathcal{N}_v} = \{T^i \in \mathbb{T} \,|\, \Theta(T^i) \cap \{\mathbb{P}_j\}_{j\in\mathcal{N}_v} = \emptyset,\ \forall i = 1, \dots, \#\,\mathbb{T}\}, \tag{6.8}$$

with $\mathbb{T}$ the list of optimal tuples obtained from Algorithm 5. The flops contribution of the implicit solver to the worst-case $n^{\max}_{\mathcal{N}_v}$ is

$$n^{\max}_{\text{imp},\mathcal{N}_v} = \max\{n_{\text{imp}}(T^i_{\mathcal{N}_v})|\ i = 1, \dots, \#\,\mathbb{T}_{\mathcal{N}_v}\} \tag{6.9}$$

with $n^{\max}_{\text{imp},\mathcal{N}_v} = n_{\text{imp}}(T^s_{\mathcal{N}_v})$. Consider then $\mathcal{N}_h = \mathcal{N}_v \cup \{h\}$, with h an index to be chosen, and the corresponding partial explicit law $u^*_{\text{exp},\mathcal{N}_h}(\theta)$. The relation $n^{\max}_{\text{exp},\mathcal{N}_h} > n^{\max}_{\text{exp},\mathcal{N}_v}$ holds by construction for every h. Therefore, the necessary condition to meet the requirement $n^{\max}_{\mathcal{N}_h} < n^{\max}_{\mathcal{N}_v}$ is that $n^{\max}_{\text{imp},\mathcal{N}_h} < n^{\max}_{\text{imp},\mathcal{N}_v}$ and h is selected such that $T^s_{\mathcal{N}_v} \subset \mathbb{P}_h$. □

Theorem 6.1.1 only provides the necessary condition for $u^*_{\text{exp},\mathcal{N}_j}(\theta)$ to be a partial explicit solution improving the MPC worst-case. This allows to reduce the set of interest for implementing Algorithm 6 from $2^{n_r} - 2$ to $n_r - 1$. However, the "best" partial PWA law among these $n_r - 1$ depends on the particular problem, and more specifically the given computational and memory limits. Therefore, Algorithm 7 is proposed in oder to compute the worst-case number of flops $\{n^{\max}_{\mathcal{N}_i}\}_{i=1}^{n_r-1}$ and memory allocation $\{m_{\mathcal{N}_i}\}_{i=1}^{n_r-1}$ of all the $n_r - 1$ partial explicit laws. This list of the possible implementations of WCPE-MPC can be used by the designer to select the one that provides the best worst-case improvement, given a certain limit to the memory allocation. The worst-case number of flops and occupancy of each partial PWA law is found to be:

$$n^{\max}_{\text{exp},\mathcal{N}} = 2n_\theta n_u + 2n_\theta\,\#\,\mathcal{N} + \sum_{i\in\mathcal{N}} n^i_e \tag{6.10a}$$

$$m_{\text{exp},\mathcal{N}} = \frac{(n_\theta + 1)\,\#\,\mathcal{M} + (n_u n_\theta + n_u)\,\#\,\mathcal{N}}{p_f}. \tag{6.10b}$$

Result 3. Let $m_{\mathcal{N}_i}$ being the memory requirements for WCPE-MPC, when implemented with the i-th partial PWA function of $\{\mathcal{N}_i\}_{i=1}^{n_r-1}$. Then $m_{\mathcal{N}_i} > m_{\mathcal{N}_j}, \forall i > j$. Let $\bar{m}$ be the memory allocation to store the

entire WCPE-MPC, i.e. Algorithm 6 code, GI solver code and matrices, then $m_{\mathcal{N}_i} = \bar{m} + m_{\text{exp},\mathcal{N}_i}$ follows.

Algorithm 7 WCPE Reduction Algorithm

Input: List of optimal tuples $\mathbb{T} = \{T^i\}_{i=1}^{\#\mathbb{T}}$ from Algorithm 5, $u^*_{\text{exp}}(\theta) = \{K_i\theta + c_i,\ \forall\theta \in \mathbb{P}_i,\ i = 1,\dots,n_r\}$ from (5.20), $r(\theta) = \{r_i(\theta) = \bar{K}_i\theta + \bar{c}_i \mid \theta \in \mathbb{P}_i,\quad i \in \mathcal{M}\}$, and the memory occupancy of implicit MPC m_{imp}

1: $\mathcal{N} \leftarrow \emptyset$, $u^*_{\text{exp},\mathcal{N}} \leftarrow \emptyset$, $n^{\max}_{\mathcal{N}} \leftarrow \emptyset$, $m_{\mathcal{N}} \leftarrow \emptyset$, $i \leftarrow 0$, $t \leftarrow \emptyset$;
2: **while** $i < n_r$ **do**
3: $s = \arg\max_{h=\{1,\dots,\#\mathbb{T}\}}\{n_{\text{imp}}(T^h)\}$
4: Extract from $\mathbb{T}$ the tuple $T^s = (\Theta^s, \mathcal{A}^s, A^s_z, b^s_z, A^s_\pi, b^s_\pi, J^s_1, J^s_2, R^s, q^s, n^s_{\text{imp}})$;
5: Remove tuple T^s from $\mathbb{T}$;
6: **for** $v = 1,\dots,\#\mathbb{P}$ **do**
7: **if** $\Theta^s \cap \mathbb{P}_v \neq \emptyset$ **then**
8: $i \leftarrow i + 1$, $t \leftarrow \emptyset$;
9: $\mathcal{N}_i \leftarrow \mathcal{N}_{i-1} \cup \{v\}$;
10: **for** $j = 1,\dots,\#\mathbb{T}$ **do**
11: Extract from $\mathbb{T}$ the tuple $T^j = (\Theta^j, \mathcal{A}^j, A^j_z, b^j_z, A^j_\pi, b^j_\pi, J^j_1, J^j_2, R^j, q^j, n^j_{\text{imp}})$;
12: **if** $\mathcal{A}^j = \mathcal{A}^s$ **then**
13: $t \leftarrow t \cup j$;
14: **end if**
15: **end for**
16: Remove from $\mathbb{T}$ all the tuples indexed by t;
17: $u^*_{\text{exp},\mathcal{N}_i}(\theta) = \{K_j\theta + c_j, \quad \forall\theta \in \mathbb{P}_j, \quad \forall j \in \mathcal{N}_i\}$;
18: Compute $n^{\max}_{\text{exp},\mathcal{N}_i}$, $m_{\text{exp},\mathcal{N}_i}$ as in (6.10);
19: $l = \arg\max_{h=\{1,\dots,\#\mathbb{T}\}}\{n_{\text{imp}}(T^h)\}$
20: Extract from $\mathbb{T}$ the tuple $T^l = (\Theta^l, \mathcal{A}^l, A^l_z, b^l_z, A^l_\pi, b^l_\pi, J^l_1, J^l_2, R^l, q^l, n^l_{\text{imp}})$;
21: $n^{\max}_{\mathcal{N}_i} \leftarrow n^{\max}_{\text{exp},\mathcal{N}_i} + n^l_{\text{imp}}$;
22: $m_{\mathcal{N}_h} \leftarrow m_{\text{imp}} + m_{\text{exp},\mathcal{N}_i}$;
23: **go to** Step 2
24: **end if**
25: **end for**
26: **end while**

Output: List of partial explicit optimal solutions $\{u^*_{\text{exp},\mathcal{N}_i}(\theta)\}_{i=1}^{n_r-1}$ and corresponding $\{n^{\max}_{\mathcal{N}_i}\}_{i=1}^{n_r-1}$ and memory allocation $\{m_{\mathcal{N}_i}\}_{i=1}^{n_r-1}$

6.1.1 Results

In order to show its effectiveness, WCPE-MPC has been tested on the inverted pendulum and nonlinear demo examples introduced in Section 5.2. Figure 6.1 compares the memory occupancy and worst-case flops of WCPE-MPC, implicit and explicit MPC. For both the examples the partial PWA law $u^*_{\text{exp},\mathcal{N}_j}(\theta)$, with $\mathcal{N}_j \in \{\mathcal{N}_i\}_{i=1}^{n_r-1}$, is obtained running offline Algorithm 7. The worst case number of flops for WCPE-MPC has been represented as a function of the increasing memory occupancy $m_{\mathcal{N}_i}$. In the inverted pendulum example (top figure) the complexity of implicit MPC and of all the possible implementations of WCPE-MPC is greater than explicit MPC, which would be the preferred solution with enough memory space, i.e. 55.4 kB in the case of double precision. However, if this occupied memory is a concern, WCPE-MPC implemented with the partial PWA law corresponding to the point labeled p_1, guarantees a memory reduction of 50.07% while worsening the worst-case number of flops only by 7.3%, instead of the 12.6% worsening with the implicit MPC alternative. The results for the nonlinear demo follow the opposite trend. Implicit MPC outperforms explicit MPC not only in memory requirements but also in computational complexity. In this case, the use of WCPE-MPC allows one to further reduce the worst-case execution time, at the price of storing additional data. Indeed, the WCPE-MPC implemented according to the point labeled p_2, guarantees a reduction of 15.4% of the worst-case, by allocating 21.5kB of memory in addition to those required by implicit MPC, in a double precision implementation. The examples show two different scenarios where WCPE-MPC can be successfully used, with exactly computable improvements in memory and complexity that other semi-explicit methods cannot guarantee.

6.2 Efficient constraints selection for dual-active set methods

This section presents another novel method to accelerate online MPC solved by dual active-set algorithm. The idea is to exploit the degree of freedom, offered by dual algorithms, when choosing the violated constraint to add to the active set. As discussed in Chapter 5, two strategies

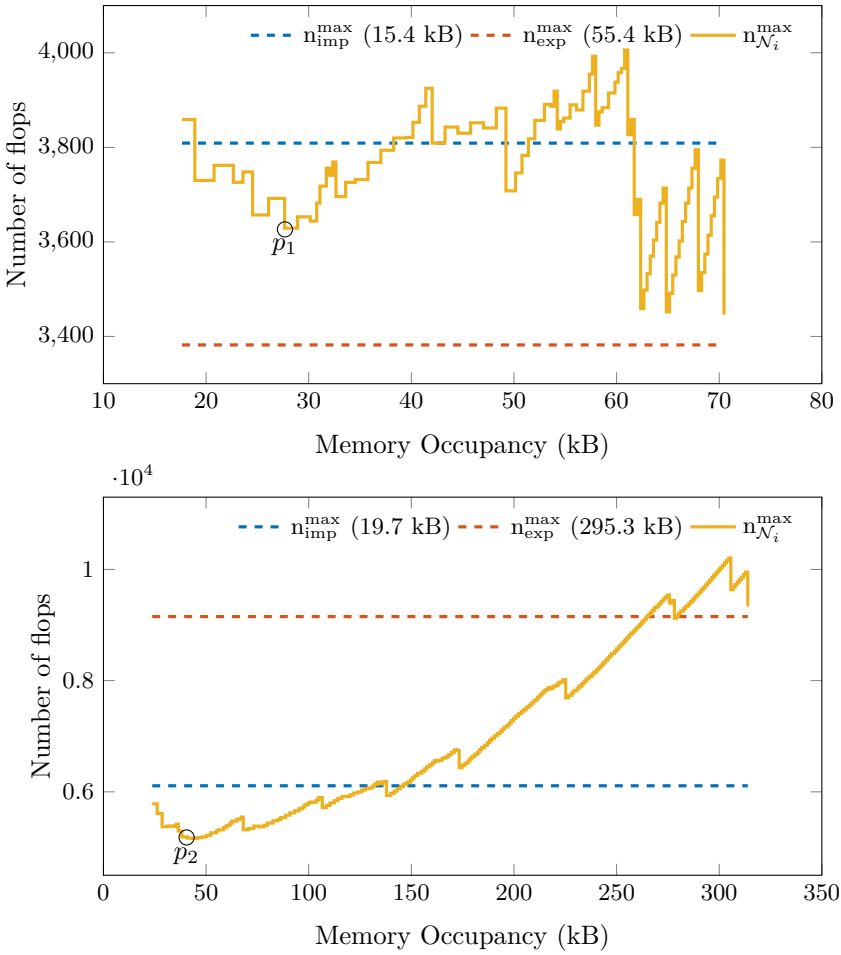


Figure 6.1: Results of the WCPE-MPC approach for inverted pendulum problem (top) and nonlinear demo problem (bottom). Computational complexity (yellow line) of WCPE-MPC is plotted as function of the memory occupancy required to store an increasing number of regions, from 1 to $n_r - 1$, along with the complexity of implicit (blue line) and explicit (red line) MPC, and the corresponding memory requirements stored in brackets. The best tradeoff points between memory and worst-case execution time are circled.

Algorithm 8 Heuristics-based constraint selection

Input: $\mathcal{A}^q \in \mathbb{N}^{n_\mathcal{A}}$ and primal variables z^q at iteration q of Algorithm 3, G, W, θ_k and w from QP (3.15), and the set of indexes $\mathcal{K}_u$, $\mathcal{K}_y$.

$\mathcal{C}^q \leftarrow \emptyset$;
$\mathcal{I}^q \leftarrow K \setminus \mathcal{A}^q$;
for $i = 1 : n_\mathcal{A}$ **do**
  if $\mathcal{A}^q(i) \in \{\mathcal{K}_u^0, \dots, \mathcal{K}_u^{N_u-2}\}$ **then**
    $\mathcal{C}^q \leftarrow \mathcal{C}^q \cup \mathcal{K}(\mathcal{A}^q(i) + N_u)$;
  end if
  if $\mathcal{A}^q(i) \in \{\mathcal{K}_y^0, \dots, \mathcal{K}_y^{N_y-2}\}$ **then**
    $\mathcal{C}^q \leftarrow \mathcal{C}^q \cup \mathcal{K}(\mathcal{A}^q(i) + N_p)$;
  end if
end for
$\mathcal{C}^q \leftarrow \mathcal{C}^q \setminus \mathcal{A}^q$
$\mathcal{V}_1 \leftarrow \{i \in \mathcal{C}^q \,|\, G_i z_{j,i} - W_i \theta_k - w_i > 0\}$;
if $\mathcal{V}_1 \neq \emptyset$ **then**
  return $p \leftarrow \arg\max_{i \in \mathcal{V}_1} \{G_i z_{j,i} - W_i \theta_k - w_i\}$;
else
  $\mathcal{V}_2 \leftarrow \{i \in \mathcal{I}^q \setminus \mathcal{C}^q \,|\, G_i z_{j,i} - W_i \theta_k - w_i > 0\}$;
  if $\mathcal{V}_2 \neq \emptyset$ **then**
    return $p \leftarrow \arg\max_{i \in \mathcal{V}_2} \{G_i z_{j,i} - W_i \theta_k - w_i\}$;
  else
    return $z^* \leftarrow z^q$;
  end if
end if

Output: Violated constraint p to add to the active-set, or optimality condition and z^*.

are commonly used to select the violated constraint p: the *most violated* (4.15a), and the *first violated* (4.15b). Between the two, the most violated is usually preferred, because it turns into fewer iterations (on average) and it is less prone to incur into degeneracy cases [32]. However, checking all the violated constraints at each iteration is costly, especially in those problems derived from MPC where $m > n_z$, and therefore the first violated rule, which is computationally cheaper in the single iteration, could be faster in some cases. Chapter 5 has shown how the two strategies can be both certified, and therefore given an MPC problem it can be exactly computed which one is the best. However, as long as the index p remains IPWC, any selection rule can be

certified. The idea of modifying the strategy to choose the violated constraints is not new. Indeed, it has been used to limit the infeasibility effect of solver interruption, by first searching for p in the most recent prediction horizon [103]. However, the authors are not aware of results regarding the improvement in computational efficiency by using different rules than the standard ones. To this end, here we pursue the idea, and the experimental observation, that the optimal active set $\mathcal{A}^*$ can follow a specific pattern when the QP problem comes from an MPC formulation. Specifically, $\mathcal{A}^*$ is *usually* composed by input and output constraints that are consecutive in time, that is, given the QP problem instance at time k, if a constraint is violated at prediction step i, it will have an high probability to be violated also at prediction step $i+1$. Under this assumption, the set of constraints from where p is selected at each iteration can be dynamically divided into two subsets with different "activation probability". Consider the set of constraints indexes $\mathcal{K}$, partitioned as in the follows:

$$\mathcal{K} = [\underbrace{1, \dots, (n_{cu} \cdot N_u)}_{\mathcal{K}_u}, \underbrace{(n_{cu} \cdot N_u + 1), \dots, m}_{\mathcal{K}_y}]. \tag{6.11}$$

where the set $\mathcal{K}_u$ contains the indexes of $\mathcal{K}$ that regard the input constraints, whereas $\mathcal{K}_y$ contains the indexes of $\mathcal{K}$ that regard the output constraints, and n_{cu}, n_{cy} are the cardinality of constraints imposed on inputs and outputs, respectively. To simplify the notation we are not considering the presence of constraints on input rates $\Delta u_{k+i|k}$, although the extension to this case is straightforward. The sets $\mathcal{K}_u$ and $\mathcal{K}_y$ are further divided into N_u and N_p subsets respectively, such that:

$$\mathcal{K}_u^i = \{\mathcal{K}_u(q) \,|\, q \in \{(i-1) \cdot n_{cu} + 1, \dots, i \cdot N_u\}\}, \tag{6.12a}$$

$$\mathcal{K}_y^j = \{\mathcal{K}_y(q) \,|\, q \in \{(j-1) \cdot n_{cy} + 1, \dots, j \cdot N_p\}\}, \tag{6.12b}$$

$$i = 0, \dots, N_u - 1, \quad j = 0, \dots, N_p - 1$$

The subset $\mathcal{K}_u^i$ represents the input constraints at the i-th step of the prediction, whereas $\mathcal{K}_y^j$ contains the output constraints at j-th step. The ordering of the constraints within these subsets is preserved to guarantee the correspondence of constraints for different prediction instants. As an example, the elements $[\mathcal{K}_u(1), \mathcal{K}_u(n_u + 1), \mathcal{K}_u(2n_u + 1), \dots]$ correspond

to the same input constraint at different prediction steps. This means that when $\mathcal{K}_u(1)$ is added to the working-set, we consider $K_u(n_u + 1)$ as a constraint with higher priority then others.

Let us assume the set of constraints indexes is partitioned as in Eqs. (6.11), (6.12a) and (6.12b). The steps for the proposed constraint selection are summarized in Algorithm 8. At each iteration of the dual active-set, the algorithm extract the subset $\mathcal{C} \subseteq \mathcal{I}$ which represents a set of constraints that must be evaluated with higher priority. Thus a violated constraint is firstly searched into $\mathcal{C}$. If a violated constraint is found, the remaining constraints $\mathcal{C} \setminus \mathcal{K}$ are not evaluated. When searching into these two different priority sets, a selection rule is still needed. In this case the most violated criterion is used, but performed on a reduced set. The proposed strategy is very simple to code, and can be employed in any dual active-set solver. Furthermore it does not require additional memory storage, and *i*) it can reduce the number of iterations, as constraints that are mostly likely to be active at $\mathcal{A}^*$ are added to the current active set, *ii*) it can reduce the time spent to solve the iteration when a violated constraint is found into $\mathcal{C}$, and moreover for the iterations where this does not happen the time is not increased. Being the method based only on the constraints structure, it easily follows that worst-case can benefit from the above advantages as well. Therefore, similarly to WCPE-MPC, this acceleration technique is particularly suited for embedded MPC, and overcomes the limit of well known acceleration methods for which only an improvement of the average case is demonstrated [34, 43, 68]. Moreover, it is straightforward to verify that the index p, selected accordingly to the proposed rule is IPWC, and therefore the complexity of the algorithm can be certified.

6.2.1 Results

A large collection of random MPC problems has been used to demonstrate the good performance of the novel selection strategy, applied to the GI algorithm presented in Chapter 4. However, the improvement in terms of number of iterations and execution time provided by the algorithm are independent from the chosen dual active-set solver. The results have been compared to those obtained with the standard most-violated selection rule.

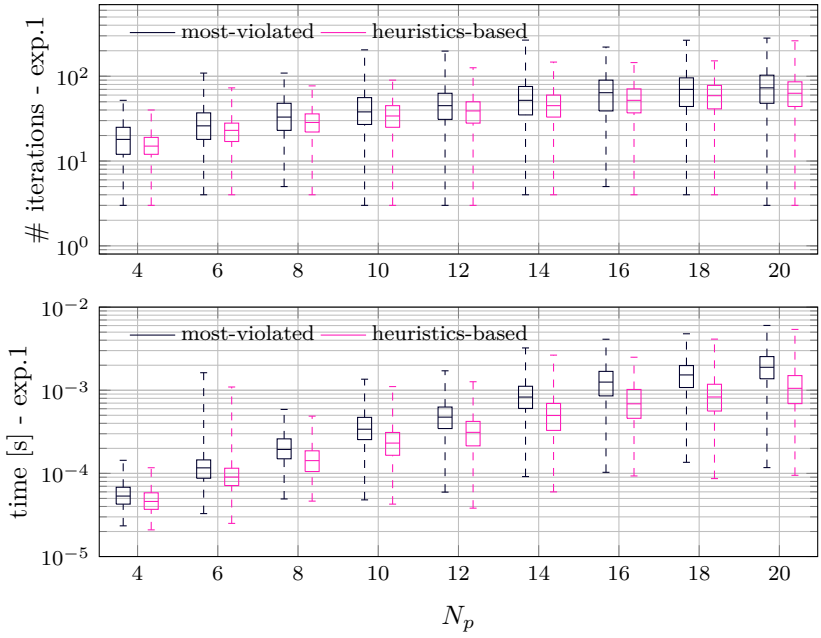


Figure 6.2: Comparison of most-violated and heuristics-based constraint selection strategies, tested with the GI solver. For each prediction horizon N_p, 1000 QP's arising from random MPC problems are evaluated, with $n_x = 6$, $n_u = 3$.

Figures 6.2 and 6.3 collect the results, showing the number of iterations and the time to solve a collection of random QPs coming from MPC problems. The computational environment consists of a PC with an Intel®Core i7-4710MQ CPU @ 2.50GHz. Figure 6.2 refer to problems where $n_x = n_y = 6$ and $n_u = 3$ hold, whereas for figure6.3 the problems have dimensions $n_x = n_y = 12$ and $n_u = 6$. The figure shows the behavior incrementing the prediction horizon, and for each value on the x-axis, a set of 1000 random MPC problems is solved. The results show that the iterations and the computation time are improved with respect to the most violated rule. Table 6.2.1 collects a summary of the relevant results shown by the figures. For instance, the proposed selection strategy guarantees improvements up to 29.57% for the number of iterations, and up to 46.83% for the time required to compute the

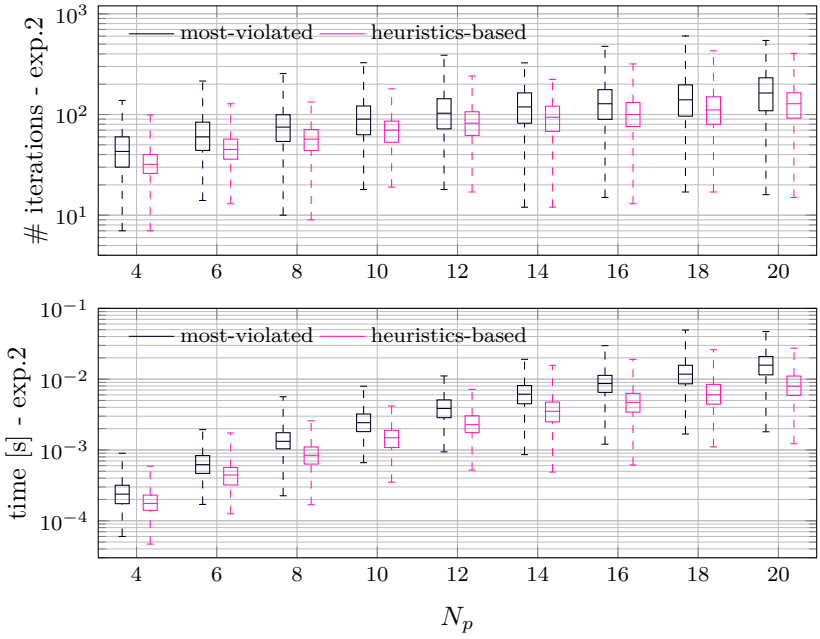


Figure 6.3: Comparison of most-violated and heuristics-based constraint selection strategies, tested with the GI solver. For each prediction horizon N_p, 1000 QP's arising from random MPC problems are evaluated, with $n_x = 12$, $n_u = 6$.

optimal solution.

Table 6.1: Improvements of the proposed algorithm respect to the standard "most violated" rule (4.15a)

	Min [%]	Mean [%]	Max [%]
Experiment 1			
Iterations	16.27	18.06	19.86
Time	14.41	32.49	43.48
Experiment 2			
Iterations	24.03	25.91	29.57
Time	25.64	39.43	46.83

Chapter 7

Model predictive control for electrical motors

Electrical systems are one of the fields where embedded MPC has been widely studied in the very last years. In particular, this chapter will deal with embedded MPC applied to electrical motors [46, 47, 122, 123]. This interest is motivated by the relatively accurate models available for electrical devices, the straightforward tuning, and the need to impose voltage and current constraints for safety and durability reasons [11, 124, 125]. MPC performance are therefore widely appreciated for the control of electrical motors, as confirmed by the recent literature [49, 52, 57, 126]. Hot topics which moved this interest are energy efficiency, fast response, and safety and reliability of the equipments, see for example [127–130] and references therein. Secondly, the International Energy Agency recognized electric motors as the single biggest consumer of electricity [131].

Different electrical motors exist, and this thesis focus on the control of Permanent Magnet Synchronous Motors (PMSMs), which are interesting for their power density, fast torque response and long life spans [12]. As anticipated by the introduction, CCS-MPC and FCS-MPC are the two common strategies used when controlling transistor-based systems [48–50, 132]. Both of them have been widely investigated, highlighting their pros and cons [52, 53, 133]. CCS-MPC is still the primal choice for applications oriented towards industrial use because it drives the switches at a fixed frequency, and decouples sampling and switching frequencies. This translates into less stress for electrical components, i.e. the inverter, that are driven by a modulated signal, such as Pulse Width Modulation (PWM). Moreover, the control frequency

can be slowed down if the response time is satisfactory without affecting steady state ripple.

For these reasons, the control of PMSM drives with implicit CCS-MPC is addressed in this chapter. The results are of particular interest because the computational burden of such controller is typically considered unmanageable in this field, due to the cheap boards and the sampling frequency which ranges from 1kHz to 10kHz. Several contributions have analyzed the use of explicit MPC for electrical motors control, see e.g. [56, 57, 134], however such results hold for simplified MPC formulations, with one-step prediction and/or approximated constraints [56]. Implicit MPC has been demonstrated to be a valid alternative if powerful boards are available, like FPGAs. However, a part from the author's contribution [37], implicit MPC solved on a cheap board for motor control has been rarely addressed in the recent literature. The only contribution this author is aware is [114], where however input constraints are not considered, and sub-optimality and infeasibility are allowed by stopping the solver after a fixed amount of iterations.

The control objective addressed in this chapter is the torque tracking of a PMSM, that we refer to as Model Predictive Torque Control (MP-TC), and the results here proposed enrich those obtained in [37]. The main goal is to demonstrate that an efficient implementation of online QP solvers, like those proposed in the previous part of the thesis, allows one to consider online CCS-MPC as a valuable solution, although it has been almost abandoned in this field, in favor of explicit MPC or FCS-MPC, because considered unmanageable. In the specific, the complexity certification plays a unrivaled role in raising up the interest in CCS-MPC, as its computational feasibility can be exactly demonstrated offline. This aspect is an important novelty with respect to the present state of the art.

The proposed MP-TC has been tested on a commercially available PMSM, controlled by a Texas Instruments DSP, used for power electronics and electrical drives control. The results show that, as expected, MP-TC deeply reduces the memory occupation when compared to explicit MPC. Furthermore, the complexity certification analysis reveals that, for the particular application, the proposed online solver is cheaper also in terms of worst-case number of flops, respect to the explicit counterpart. Section 7.1 describes the mathematical model of a PMSM,

Section 7.2 presents the implementation details of the proposed MP-TC and finally the experimental results on a commercially available brushless motor are shown in Section 7.3.

7.1 Mathematical model

Those control algorithms that rely on the so-called (d, q) reference frame belong the family of Field Oriented Control (FOC) schemes. The approach consists in controlling the stator currents represented by a vector, obtained applying the Park coordinate transformation, which allows for the decoupling and linearization of the dynamics [135–137]. In the specific, FOC-based schemes exploit the fact that in the (d, q) reference frame, synchronously rotating with the rotor, the torque and the flux dynamics are linear and decoupled, and two independent torque and flux control loops can be implemented to drive the motor. Therefore, the electrical model of a PMSM in the (d, q) frame can be expressed by the following equations:

$$\dot{i}_d(t) = -\frac{R}{L_d} i_d(t) + \frac{L_q}{L_d} \omega(t) i_q(t) + \frac{1}{L_d} u_d(t) \tag{7.1a}$$

$$\dot{i}_q(t) = -\frac{R}{L_q} i_q(t) - \left(\frac{L_d}{L_q} i_d(t) + \frac{\lambda}{L_q} \right) \omega(t) + \frac{1}{L_q} u_q(t), \tag{7.1b}$$

where d and q are the subscripts for *direct* and *quadrature* quantities, respectively, L, R, i and u are the stator inductance [H], resistance [Ω], current [A] and voltage [V], respectively. On the other hand, the mechanical dynamics is represented by:

$$\dot{\omega}(t) = \frac{B}{J} \omega(t) + \frac{p}{J} \tau(t) - \frac{p}{J} \tau_l(t) \tag{7.2a}$$

$$\tau(t) = \frac{3}{2} p \left(\lambda i_q(t) + (L_d - L_q) i_d(t) i_q(t) \right), \tag{7.2b}$$

where $\omega(t)$ is the electrical rotor speed [rad/s], $\tau(t)$ is the electrical torque [Nm], J is the inertia coefficient [kg·m^2], λ is the motor flux leakage [Wb], $\tau_l(t)$ is the load torque [Nm], and p is the number of pole pairs. In conclusion, the complete mathematical model of an isotropic

motor, i.e. $L_d \equiv L_q \equiv L$, is:

$$\dot{i}_d(t) = -\frac{R}{L} i_d(t) + \omega(t) i_q(t) + \frac{1}{L} u_d(t) \tag{7.3a}$$

$$\dot{i}_q(t) = -\frac{R}{L} i_q(t) - \omega(t) i_d(t) + \frac{1}{L} u_q(t) - \frac{\lambda}{L}\omega(t) \tag{7.3b}$$

$$\dot{\omega}(t) = \frac{B}{J}\omega(t) + \frac{p}{J} K_t i_q(t) - \frac{p}{J}\tau_l(t), \tag{7.3c}$$

where $K_t = \frac{3}{2} p\lambda$ is the torque constant. Please note that in the rest of the chapter, the assumption of a single pole pair motor holds, i.e. $p = 1$, as for the motor used in experimental results this holds true. Model (7.3) exhibits a nonlinear behavior in the electrical sub-system, due to the coupling between currents and speed. In order to obtain an LTI model to be used in the linear MPC formulation presented in Chapter 3, one can impose a nominal speed $\omega(t) = \omega_0$ in the bilinear terms $\omega(t) i_q(t)$ and $\omega(t) i_d(t)$. Under this assumption, a linearized version of the equations (7.3a)-(7.3b) is derived in the form:

$$\dot{x}(t) = A^c x(t) + B_u^c u(t) + B_v^c v(t) \tag{7.4a}$$

$$y(t) = C^c x(t) \tag{7.4b}$$

where $x(t) = [i_d(t),\, i_q(t)]'$ are the states, $u(t) = [u_d(t),\, u_q(t)]'$ are the manipulated inputs, $y(t) = [i_d(t),\, \tau(t)]'$ are the outputs, $v(t) = \omega(t)$ is the measured disturbance, and the c-apex stands for a continuous-time matrix, with

$$A^c = \begin{bmatrix} -\frac{R}{L} & \omega_0 \\ -\omega_0 & -\frac{R}{L} \end{bmatrix},\; B_u^c = \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \end{bmatrix},\; B_v^c = \begin{bmatrix} 0 \\ -\frac{\lambda}{L} \end{bmatrix},\; C^c = \begin{bmatrix} 1 & 0 \\ 0 & K_t \end{bmatrix}. \tag{7.5}$$

The use of a measured disturbance improves the model accuracy by modeling part of the time-varying behavior in the affine term $v(t)$, while still having an LTI model for control purposes. This is an improvement with respect to more conservative solutions where the effect of the speed is completely linearized around a steady-state point [138].

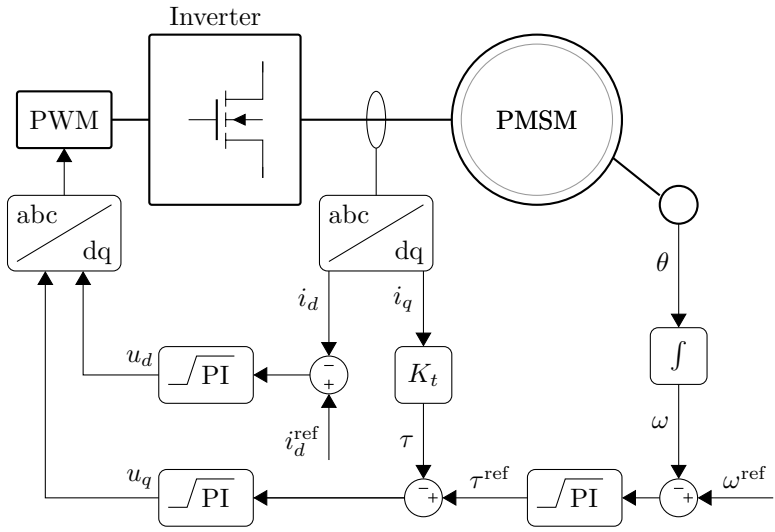


Figure 7.1: Standard field oriented control scheme for a PMSM. Both the speed loop and the two stator currents loops are controlled by linear regulators.

7.2 Control Design

The standard PI-FOC consists of a cascaded scheme with an outer loop regulating the rotor speed (or position), and an inner loop regulating the stator (d, q) currents. The two loops can be considered dynamically decoupled as the electrical dynamics can be up to one order of magnitude faster than the mechanical ones [56]. Therefore, given a reference trajectory ω^{ref} for the rotor speed, the outer loop track it by changing the torque reference signal τ^{ref} for the inner loop. Figure 7.1 shows the control architecture of a standard PI-FOC. When the motor is not supposed to work in flux weakening operation, the direct current component i_d is stabilized at 0, whereas the quadrature component i_q tracks the reference profile i_q^{ref}, obtained by scaling the torque reference τ^{ref} by the torque constant K_t. This control scheme is the most common for isotropic machines where maximum current implies maximum torque, i.e. no field weakening operation. To deal also with this aspect, Maximum Torque per Ampere (MTPA) could be a valid option [55, 138], but this is out of the scope of the work in this thesis.

Starting from Figure 7.1, the MP-TC scheme is obtained by replacing the controller for the electrical subsystem and keeping unchanged the external loop, that results into the control scheme of Figure 7.2. The reason to focus the performance enhancement on the inner loop is its the faster dynamics, and the necessity to impose safety constraints on stator voltages and currents. Given the MPC formulation of Chapter 3, the LTI prediction model for the optimization problem is obtained by the discretization of model (7.4), given sampling time T_s, which results into the discrete time model:

$$x_{k+1} = Ax_k + B_u u_k + B_v v_k \tag{7.6a}$$
$$y_k = Cx_k \tag{7.6b}$$

with $A = e^{A^c T_s}$, $B_u = \int_0^{T_s} e^{A^c \tau} d\tau B_u^c$, $B_v = \int_0^{T_s} e^{A^c \tau} d\tau B_v^c$, and $C = C^c$, which differs from the one used in Chapter 3 only for the measured disturbance. In the specific, the general formulation (3.2) is slightly modified to obtain the optimization problem solved at each time step for MP-TC, such as:

$$\min_{\Delta u} \quad \sum_{i=0}^{N_p-1} \|W_y(y_{k+i+1|k} - y_k^{\text{ref}})\|_2^2 + \sum_{h=0}^{N_u-1} \|W_{\Delta u} \Delta u_{k+h|k}\|_2^2 \tag{7.7a}$$
$$\text{s.t.} \quad x_{k|k} = x_k, \tag{7.7b}$$
$$x_{k+i+1|k} = Ax_{k+i|k} + B_u u_{k+i|k} + B_v v_{k+i|k}, \tag{7.7c}$$
$$y_{k+i+1|k} = Cx_{k+i+1|k}, \tag{7.7d}$$
$$\Delta u_{k+i|k} = u_{k+i|k} - u_{k+i-1|k} \tag{7.7e}$$
$$\Delta u_{k+N_u+j|k} = 0, \quad j = 0, \dots, N_p - N_u - 1 \tag{7.7f}$$
$$v_{k+i|k} = v_k, \tag{7.7g}$$
$$u_{k+i|k} \in \mathbb{U}, \tag{7.7h}$$
$$y_{k+i+1|k} \in \mathbb{Y}, \tag{7.7i}$$
$$i = 0, \dots, N_p - 1, \tag{7.7j}$$
$$h = 0, \dots, N_u - 1. \tag{7.7k}$$

which differs from (3.2) because *i*) the cost function does not weight the error on input tracking, *ii*) there is no preview on the outputs reference *iii*) the constraints on Δu are not needed, *iv*) and the linear model

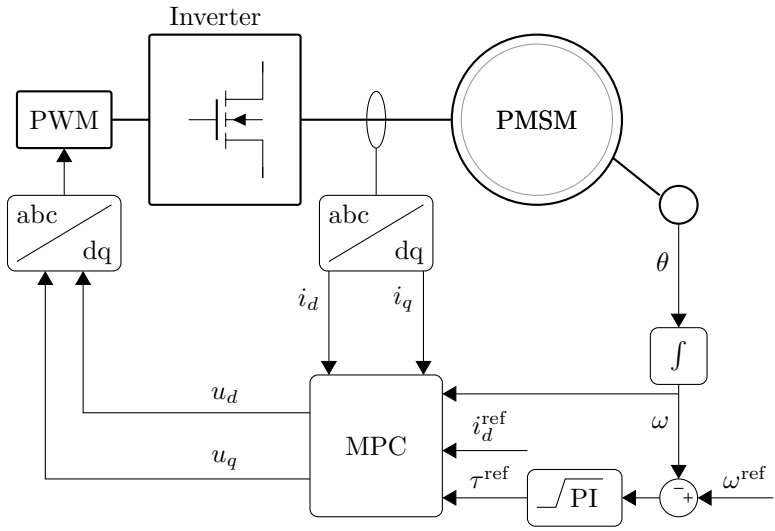


Figure 7.2: The proposed control scheme for PMSM. The speed loop is controlled by a standard regulator, the inner loop is implements the model predictive torque control here introduced.

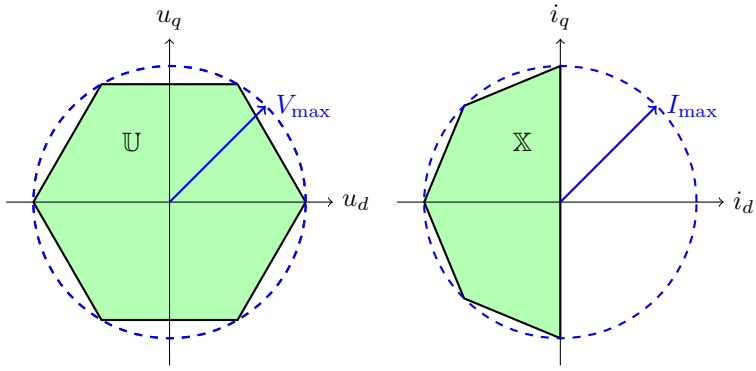
comprises the measured disturbance. By following similar construction steps of the condensed formulation in Section 3.2, problem (7.7) can be cast in the QP problem (3.15), with the parameters vector θ_k equal to:

$$\theta_k = [u_{k-1}, x_{k|k}, y_k^{\text{ref}}, v_k]'. \tag{7.8}$$

Constraints on stator voltages and currents need to be imposed for safety reasons, related to the electrical characteristics of the components [139]. Specifically, the inverter imposes the phase voltage limit, depending on the modulation technique. Given a DC-bus voltage V_{DC}, the limit is found to be:

$$V_{\max} = \frac{V_{\text{DC}}}{\sqrt{3}}, \tag{7.9}$$

for both space vector and pulse-width modulations [138]. Whereas, constraints on the current $I_{\max}$ are imposed to prevent overheating, therefore their violation for short periods of time, eventually caused by constraints' softening, is allowed. Even though current and voltage



(a) Voltage constraints (b) Current constraints

Figure 7.3: Inputs and outputs constraints imposed by the MPC controller. The blue circles represent the original norm constraints of Eq. (7.10). The green regions represent their approximations, and are the polyhedra described by the sets $\mathbb{U}$ and $\mathbb{Y}$, respectively.

limits are bound constraints in the main frame, they get transformed into norm constraints in the (d, q) axis, namely:

$$u \in \tilde{\mathbb{U}} = \{u \in \mathbb{R}^2 : \|u\|_2 \leq V_{\max}\}, \tag{7.10a}$$

$$x \in \tilde{\mathbb{X}} = \{x \in \mathbb{R}^2 : \|x\|_2 \leq I_{\max}\}. \tag{7.10b}$$

In order to retain the QP formulation, the quadratic constraints (7.10) are replaced by polytopic approximations, which give the possibility to exploit all the theoretical results obtained in the previous chapters of this thesis for QP problems [56, 138]. Hexagons can reasonably approximate the feasible region (7.10), resulting into an acceptable trade-off between the accuracy and number of inequality constraints. Figure 7.3 shows such polytopic approximations.

The state vector of the system is fully available, however the use of an observer is usually suggested in MPC to reduce the impact of measurements noise, e.g. due to transistor switching. Furthermore, in embedded MPC is common to have a worst-case solution time for the QP problem that is comparable in order of magnitude to the sampling interval, even

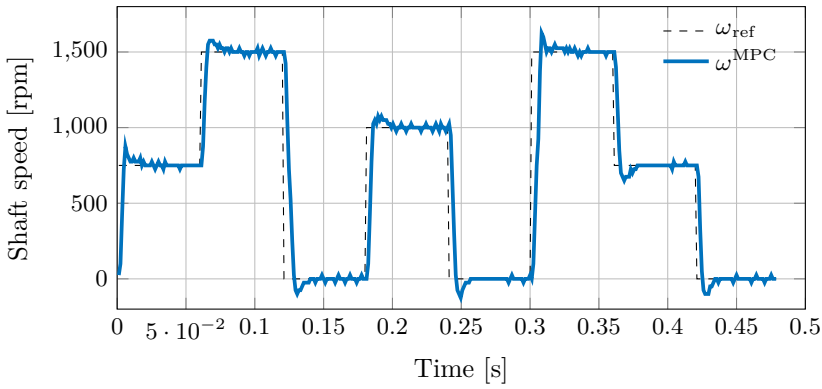


Figure 7.4: Speed tracking performance for the MP-TC controller.

though always feasible. This holds true also for the application here presented, as will be confirmed by the experimental results. Therefore, a delay of one time step is preferably applied to the inputs. This results into a delayed Kalman filter implementation, where the state estimate fed to the MPC problem is obtained as:

$$\hat{x}_{k+1|k} = (A - LC)\hat{x}_{k|k-1} + Bu_{k-1} + G_v v_k + Ly_k \tag{7.11}$$

with L the Kalman gain. Another shrewdness for the MPC implementation,ì regards the minimization of the tracking error in the presence of model uncertainty and noise. Several methods have been proposed in the literature to deal with this issue, and in this work the integral action on the reference vector has been used [140]. The output reference after integral action, denoted by $\tilde{y}^{\text{ref}} = [\tilde{i}_d^{\text{ref}}, \tilde{\tau}^{\text{ref}}]'$, is obtained as:

$$\tilde{i}_d^{\text{ref}}(k) = \tilde{i}_d^{\text{ref}}(k-1) + k_1(i_d^{\text{ref}}(k) - i_d(k)) \tag{7.12a}$$

$$\tilde{\tau}^{\text{ref}}(k) = \tilde{\tau}^{\text{ref}}(k-1) + k_2(\tau^{\text{ref}}(k) - \tau(k)) \tag{7.12b}$$

where k_1, k_2 are scalar parameters tuned in calibration.

Table 7.1: Technosoft MBE.300.E500 PMSM specifications

Parameter	Units	Value
Coil dependent parameters		
Phase-phase resistance	ohm	8.61
Phase-phase inductance	mH	7.13
Back-EMF constant	V/1000 rpm	3.86
Torque constant	mNm/A	36.8
Pole pairs	–	1
Dynamic parameters		
Rated voltage	V	36
Max. voltage	V	58
No-load current	mA	73.2
No-load speed	rpm	9170
Max. cont. current (at 5000 rpm)	mA	913
Max. cont. torque (at 5000 rpm)	mNm	30
Max. permissible speed	rpm	15000
Peak torque (stall)	mNm	154
Mechanical parameters		
Rotor inertia	$\text{kgm}^2 \cdot 10^{-7}$	11
Mechanical time constant	ms	7

7.3 Experimental Results

The implicit MP-TC for PMSMs has been tested on a commercially available device provided by Technosoft SA, namely the MBE.300.E500 motor, whose specifications are collected in Table 7.2. For this application the available control board is an F28335 Delfino DSP by Texas Instruments (TI). This DSP belongs to the TI C2000 series, has a 32-bit, 150 MHz CPU (6.67ns cycle time) and an IEEE-754 single-precision Floating-Point Unit (FPU), and a single hardware multiplier (32x32 bit). This computational unit is commonly used for electrical motors control, and has been chosen to demonstrate the feasibility and certification of implicit MPC on a cheap board. Two interrupts levels manage the controller scheduling: (*i*) a faster loop for current control with 0.3 ms sampling time, and (*ii*) a slow loop for speed control with 1.2 ms sampling time. Therefore, the online QP solver must guarantee that the optimization problem is solved within the 0.3 ms limit, even

Table 7.2: Design parameters for MP-TC applied to Technosoft MBE.300.E500 PMSM

Prediction horizon N	3
Control horizon N_u	1
Voltage limit $V_{\max}$	$24/\sqrt{3}$ V
Current limit $I_{\max}$	1 A
Output weights W_y and P	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Input increments weights $W_{\Delta u}$	$\begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}$
Sampling time T_s	0.3 ms

Table 7.3: Certification of different MP-TC algorithms to control the Technosoft MBE.300.E500 PMSM

	memory	flops		
	16|32 bit (kB)	$(\pm, *, \div)$|sqrt	n_r	$N_{\max}$
MP-TC	12.7 | 16.4 kB	2431|13	–	5
eMP-TC$_1$	50.9 | 100.8 kB	3868	133	–
eMP-TC$_2$	36.6 | 72.3 kB	2737	95	–

in the worst-case. The parameters used for the design of the MP-TC controller are collected in Table 7.2. Table 7.3 shows instead the certification results of the implicit MP-TC, and compare it to an eventual explicit version of the controller, referred to as eMP-TC, in order to validate which one of the two approaches is better to use. With eMP-TC$_1$ we refer to the explicit version of the implicit MP-TC, i.e. with exactly the same setup. Whereas eMP-TC$_2$ denotes a simpler version of the controller, that sacrifices the accuracy into constraints' approximations by replacing the polyhedral constraints in Figure 7.3 with simple box constraints in the (d, q) reference frame, such as:

$$i_d \in [-\epsilon I_{\max}, \epsilon I_{\max}] \tag{7.13a}$$

$$i_q \in [I_{\max}, I_{\max}], \tag{7.13b}$$

which is similar to what done in [56]. The F28335 DSP has a single-access RAM block of 34 kB, therefore the results of Table 7.3 shows that

an eventual implementation of explicit MPC on such board would be infeasible for the lack of memory, even if implemented in single arithmetics and in the simplified and less accurate eMP-TC$_2$ version. On the other hand, implicit MPC-TC fits into the selected board in both single and double precision arithmetics. Moreover, the flops needed in the worst-case to solve the QP problem with implicit MPC are less than those needed by any implementation of explicit MPC. Please note that without the novel results on the certification procedure presented in Chapter 5, the choice between explicit or implicit MPC would have been driven by experience, or extended simulation campaigns. On the contrary, running Algorithm 5 allows to easily select the best option. When none of the solutions is feasible for the selected board, one can check if the acceleration techniques proposed in Chapter 6 make the problem feasible, by certifying exactly the corresponding worst-case time run.

The control algorithm in Figure 7.2, together with an implicit MPC featuring the GI solver presented in Chapter 4, have been implemented in *C*-language on the control board. The control objective is to track a rotor speed trajectory ω^{ref} while minimizing the control effort and imposing input/output constraints.

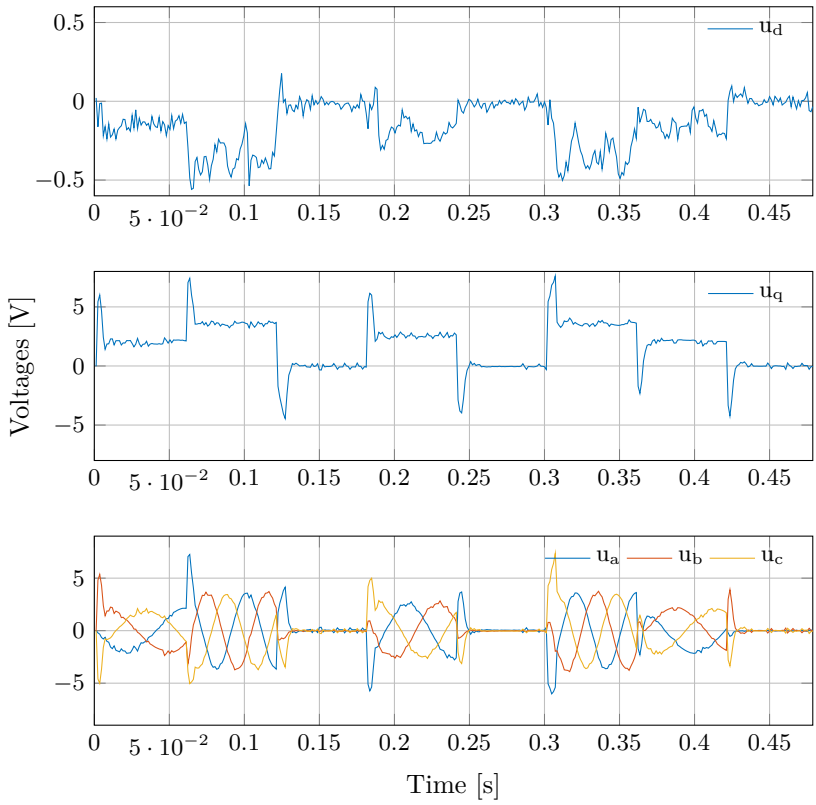


Figure 7.5: Stator voltages for the MP-TC controller when tracking the speed profile of Figure 7.4. From top to bottom: *i*) direct component, *ii*) quadrature component, *iii*) sinusoidal voltages.

The speed tracking results are shown in Figure 7.4, where abrupt changes of reference are requested to the motor in order to operate the system close to the constraints. Figures 7.5 and 7.6 show the stator voltages and currents measured when controlling the motor to track ω^{ref}. The variables are presented both in their original frame, with a sinusoidal behavior, and in the (d, q) coordinates. It is evident that both the current and voltage constraints are correctly imposed by the MP-TC.

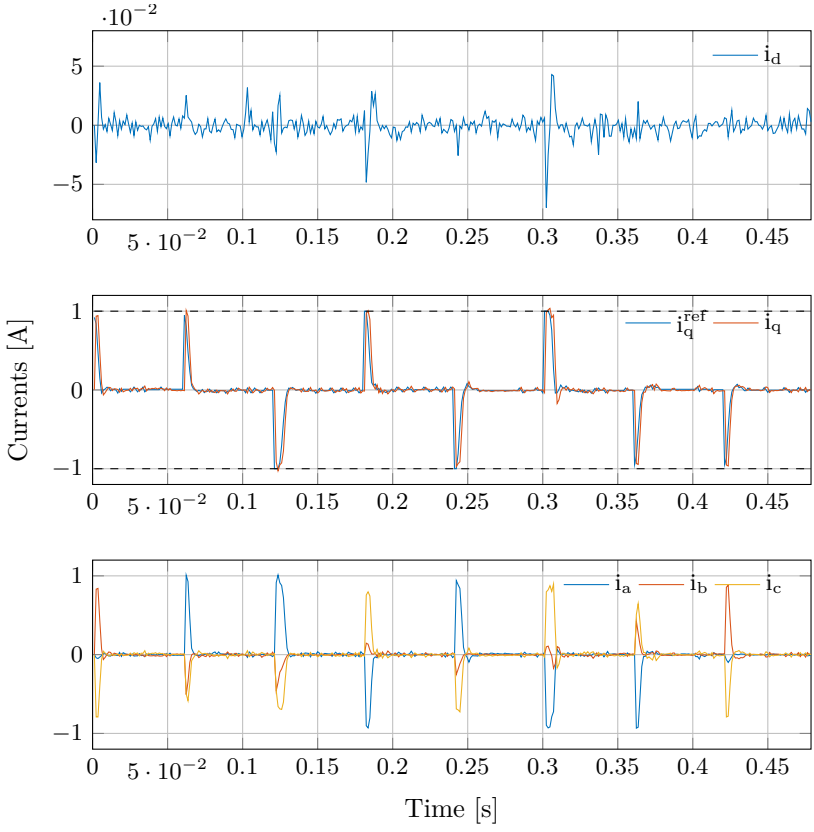


Figure 7.6: Stator currents for the MP-TC controller when tracking the speed profile of Figure 7.4. From top to bottom: *i*) direct component, *ii*) quadrature component, *iii*) sinusoidal currents.

Finally, figure 7.7 shows the control task timing on the DSP obtained through a high precision internal clock, and the number of iterations needed by the solver to obtain the optimal control sequence. To confirm what verified offline with the certification algorithm, the optimal solution is always obtained within the time limit of 0.3 ms. Please note that this measured time refers to the entire control routine, which include Analogic Digital Converter (ADC) sampling, state estimation and the QP solver.

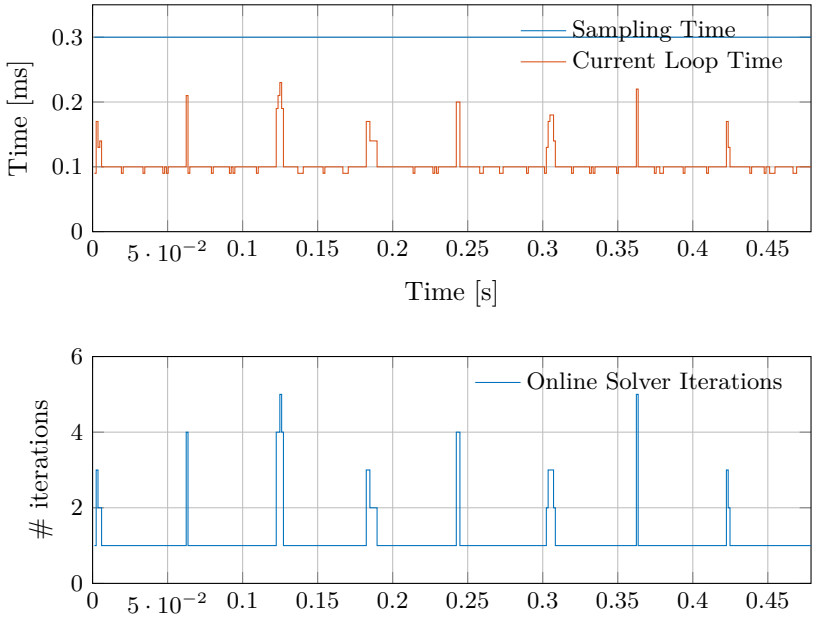


Figure 7.7: Computational burden of MP-TC routine when tracking the speed profile of Figure 7.4. Both the control unit time and the sampling time limit are reported, together with the solver iterations.

Chapter 8

Model predictive control for DC-DC converters

MPC is being widely investigated for the control of Switching Mode Power Supplies (SMPS) [12, 141–143], thanks to the continuously increasing tightening of efficiency and performance requirements (ENERGY STAR®), [11, 58, 144–146]. The development and motivations of MPC in this field share lot of similarities with that discussed in Chapter 7 for electrical motors. Indeed, the distinction between FCS- and CCS-MPC holds for SMPS as well, being transistor-based devices where the discrete nature of the switches can be exploited to optimize between the finite combinations of predicted switches' states [11, 49]. However, despite the possibly improved dynamics, the resulting variable switching frequency and the required high sampling frequency are still a matter of discussion. Therefore, currently industrial applications prefer CCS-MPC, thanks also to decoupling between switching and sampling frequencies which are usually higher in SMPS with respect to electrical motors control. Therefore, here the computational requirements and scarce resources impose more sever limits to the spread of MPC. FPGAs can be a costly alternative also in this field [147]. However, contrarily to electrical motors, explicit MPC is a much more appreciated technology for power converters [58–61], because the mathematical models are usually smaller (e.g. 1 input for lot of converters), allowing a resource efficient implementation as confirmed in [148]. Therefore the direct use of implicit MPC in the field is perceived as less appealing, especially for single-ended supplies [149, 150]. The work of this thesis for MPC in power converters covers two different aspects: the first is CCS-MPC for pre-compensated DC-DC converters, and the second is a unified formu-

lation for current estimation.

Following the idea of RG [72, 73, 120], we introduce an MPC loop that regulates a DC-DC converter by manipulating the reference of the actual controller [151]. A variety of engineering fields experimented the use of RG, e.g. automotive and robotics [74–76]. Only few attempts to improve the control dynamics by modifying the reference can be found in the recent literature regarding DC-DC converters [152, 153]. Nevertheless, power conversion seems to be a area where the control of pre-compensated systems can have an important role. In fact, it is common that the native controller cannot be changed, either because it is hard coded or even hardware based [154]. Designers could also have the necessity to retain the primal controller due to stability and robustness certification procedures [155]. Furthermore, the double and possibly multi-rate controller permits to have the benefits of MPC with a sensibly smaller impact on the computational cost, with respect to its direct application. This thesis presents the design and experimental results of MPC applied to a converter pre-compensated with VMC, which is the simplest of the standard controllers used for power converters. Compared to the state-of-the-art of controllers for such devices [152, 153], the algorithm does not need any current sensor, leaving unaltered the hardware and software setup of standard VMC, and the use of the MPC framework opens the door to all the useful features widely discussed in the previous chapters.

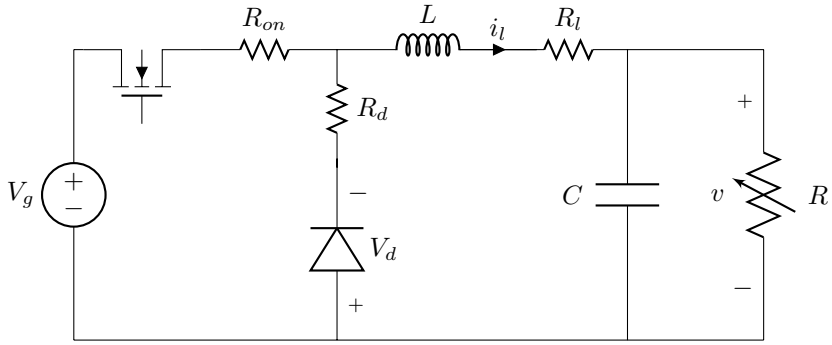
The continuous research for performance improvement in power converters, moved recently the focus on Current Mode Control (CMC), which allows for faster transient response and limited over-current protection [58, 156–159], and the use of MPC for CMC pre-compensated converters has been preliminary investigated by the author in [69]. As already discussed, an MPC framework is almost always based on a state estimator. Even more important, as far as CMC, what gained attention in both academy and industrial applications is sensorless control [83]. Therefore the design of such current observer is the focus of the second part of this chapter. Sensorless CMC (SCMS) improves the reliability of the system, the miniaturization of the device, its efficiency and its cost [84, 144, 160, 161]. However, power converters have a bilinear dynamics, requiring a dedicated nonlinear observer for each configuration. The number of different observers scales easily in this field due

to the combination of different converters' topologies. The literature on current observers for DC-DC converters is huge [85, 87–89, 162, 163], and among them we cite integral state reconstructors [164], extended Kalman filter [59], duty cycle perturbation [88] or input voltage based observers [84, 85, 162]. Despite the strengths and weaknesses of each method, they all have the drawback to be converter's dependent. Here we address the need for an unified current observer for the most common topologies of DC-DC converters. A similar need, on the control side, was recently tackled in [81]. Embedding different control codes, and learning how to tune all of them is a costly operation, which makes extremely helpful having a unified control code. Buck, boost and buck-boost configurations are addressed by the unified observer, in both their synchronous and asynchronous version [165]. The unified observer guarantees the application to Pulse Width Modulated (PWM) converters, an accurate modeling and robustness with respect to load variations, which are required features for SCMC geared towards the industrial use [81, 87, 165, 166]. Industry benefits from such a unified observer because, especially in a single power supply with many different converters, *i*) the time and the memory needed to code different observers is saved, *ii*) calibrators have to learn the tuning of one single algorithm, *iii*) code certification is easier, *iv*) code freezing, e.g. due to a software release, does not represent an issue for hardware changes.

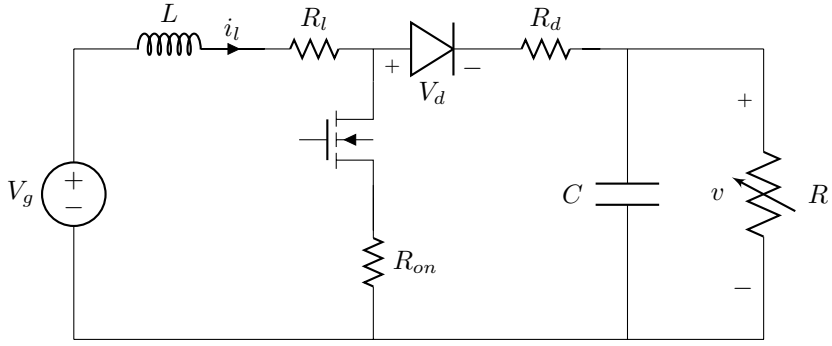
Section 8.1 discusses the modeling of power converters and the basics of standard linear control. Section 8.2 presents the MPC for precompensated VMC converter and Section 8.3 deals with the unified current observer. Both the techniques have been experimentally tested on commercially available hardware from Texas Instruments©.

8.1 Mathematical models and linear control

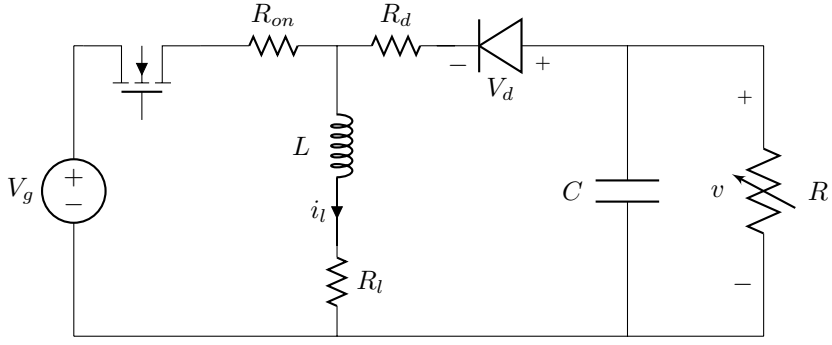
This section briefly covers the modeling for the converters of interest, and presents the basics of the standard VMC and CMC. The electrical schemes are shown in Figures 8.1 and 8.2 in asynchronous and synchronous version, respectively. Synchronous rectification is a step forward into efficiency optimization, with respect to standard asynchronous converters [167–169]. If the converter is controlled by a PWM gating signal, as in most of the cases, its dynamics can be reasonably approx-



(a) Buck Converter

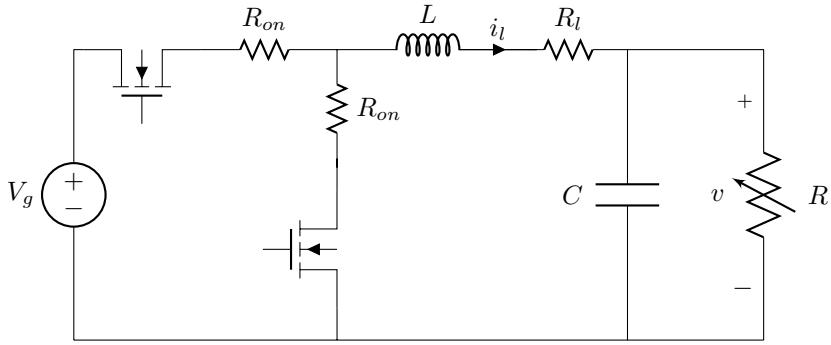


(b) Boost Converter

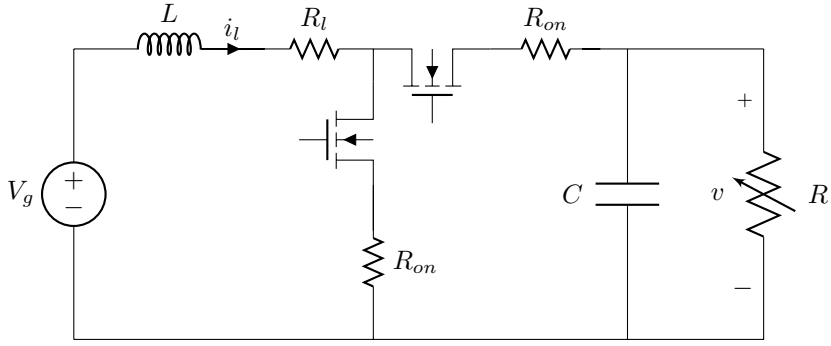


(c) Buck-Boost Converter

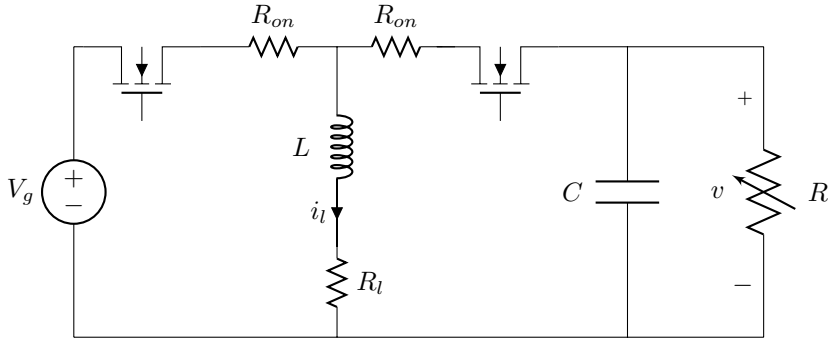
Figure 8.1: Asynchronous DC-DC converters' electrical schemes, with parasitic components taken into account by the unified formulation.



(a) Buck Converter



(b) Boost Converter



(c) Buck-Boost Converter

Figure 8.2: Synchronous DC-DC converters' electrical schemes, with parasitic components taken into account by the unified formulation.

imated by average modeling [165]. For control purposes, an equivalent low-frequency model with variables averaged over one switching time T_s is used. The mathematical models of the converters presented in figures 8.1 and 8.2 can be collected in a bilinear notation, [170], such as:

$$\dot{x}(t) = Ax(t) + \left(B_u + \sum_{i=1}^{p} F_i x_i(t) \right) u(t) + B_v \tag{8.1a}$$

$$y(t) = x(t) \tag{8.1b}$$

where $x \in \mathbb{R}^{n_x}$ is the state vector, with x_i its i-th element, $u \in \mathbb{R}^{n_u}$ is the input vector, $y \in \mathbb{R}^{n_y}$ is the output vector and A, B_u, F_i, B_v are matrices of appropriate dimensions. The index $p \leq n_x$ is the cardinality of the bilinear terms. In the specific, $x_1 \equiv i_l$, $x_2 \equiv v$ and $u \equiv d$, with i_l the inductor current, v the output voltage and d the duty-cycle of the PWM. The main parameters of the models are the input voltage V_g, the inductance L, the capacitance C and the supplied load R. Parasitic effects are also considered, with R_l is the inductor series resistance, R_{on} is the drain-to-source resistance of the switches, V_d is the voltage drop of the diode and R_d is its series resistance. As far as the load, it is the most variable component for a DC-DC converter [171], and the equation $R \equiv \hat{R} + \Delta_R$ holds, with $\hat{R} \in \mathbb{R}_+$ the *nominal value* and $\Delta_R \in \mathbb{R}$ the *unknown perturbation*, such that:

$$\Delta_R^{\min} \leq \Delta_R \leq \Delta_R^{\max}, \tag{8.2}$$

where $\Delta_R^{\min} < \Delta_R^{\max}$ are known bounds on the minimum and maximum variations of the load with respect to the nominal value. Under all the previous assumptions, the dynamics of the six converters is reflected by the following nonlinear system:

$$\dot{x}(t) = Ax(t) + B_u u(t) + F_1 x_1(t) u(t) + F_2 x_2(t) u(t) + B_v, \tag{8.3}$$

where the different matrices are parametrized according to the Table 8.1. Converters' control has to guarantee the regulation of the output power at a desired value. VMC and CMC are the standard controllers for power converters, and nowadays they are considered a technology [87, 165, 172]. The first is the simplest, with a single loop driven

Table 8.1: State space matrices of converters in bilinear form (Eq. 8.3). "AsC" stands for asynchronous converter, and "SC" stands for synchronous converter.

	A	B	F_1	F_2	B_v
Buck AsC	$\begin{bmatrix} -L^{-1}(R_l + R_d) & -L^{-1} \\ C^{-1} & -(C(\hat{R} + \Delta_R))^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1}(V_d + V_g) \\ 0 \end{bmatrix}$	$\begin{bmatrix} -L^{-1}(R_{on} - R_d) \\ 0 \end{bmatrix}$	—	$\begin{bmatrix} -L^{-1}V_d \\ 0 \end{bmatrix}$
Buck SC	$\begin{bmatrix} -L^{-1}(R_l + R_{on}) & -L^{-1} \\ C^{-1} & -(C(\hat{R} + \Delta_R))^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1}V_g \\ 0 \end{bmatrix}$	—	—	—
Boost AsC	$\begin{bmatrix} -L^{-1}(R_l + R_d) & -L^{-1} \\ C^{-1} & -(C(\hat{R} + \Delta_R))^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1}V_d \\ 0 \end{bmatrix}$	$\begin{bmatrix} -L^{-1}(R_{on} - R_d) \\ -C^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1} \\ 0 \end{bmatrix}$	$\begin{bmatrix} L^{-1}(V_g - V_d) \\ 0 \end{bmatrix}$
Boost SC	$\begin{bmatrix} -L^{-1}(R_l + R_{on}) & -L^{-1} \\ C^{-1} & -(C(\hat{R} + \Delta_R))^{-1} \end{bmatrix}$	—	$\begin{bmatrix} 0 \\ -C^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1} \\ 0 \end{bmatrix}$	$\begin{bmatrix} L^{-1}V_g \\ 0 \end{bmatrix}$
Buck-Boost AsC	$\begin{bmatrix} -L^{-1}(R_l + R_d) & -L^{-1} \\ C^{-1} & -(C(\hat{R} + \Delta_R))^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1}(V_d + V_g) \\ 0 \end{bmatrix}$	$\begin{bmatrix} -L^{-1}(R_{on} - R_d) \\ -C^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1} \\ 0 \end{bmatrix}$	$\begin{bmatrix} -L^{-1}V_d \\ 0 \end{bmatrix}$
Buck-Boost SC	$\begin{bmatrix} -L^{-1}(R_l + R_{on}) & -L^{-1} \\ C^{-1} & -(C(\hat{R} + \Delta_R))^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1}V_g \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ -C^{-1} \end{bmatrix}$	$\begin{bmatrix} L^{-1} \\ 0 \end{bmatrix}$	—

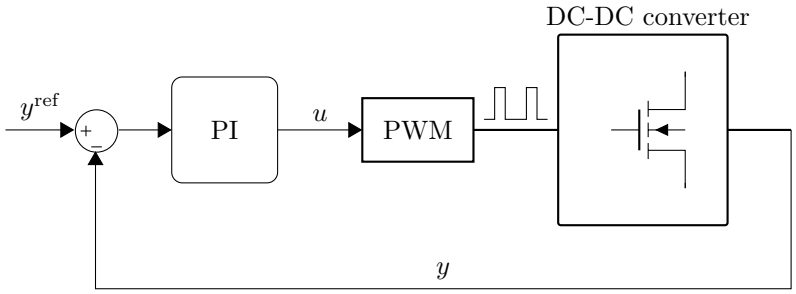
by output voltage error, while the second presents a cascaded structure with inner current and outer voltage control loops. In some applications VMC is preferred in for its simplicity, whereas the faster dynamics makes CMC more suitable for high performance supplies, even though a current sensor (or observer) is required. Anyway, both of them usually rely on linear PI regulators for driving voltage and current loops. The tuning of the PI regulators follows the well known small signal analysis [173], designing the bandwidth and stability margins in order to obtain a satisfactory performance. As a rule of thumb, a controller that exhibits a gain margin of about 10dB and a phase margin greater than 45 degrees is desirable [165, 173], and to mitigate the effect of right-half-plane zeros for some converter topologies, the crossover frequency is kept below 1/3 of the zero frequency [165]. For synchronous converters, where two switches must be actuated, master-slave technique is commonly used, by driving the secondary switch with a complementary signal with respect to the primal one.

8.2 MPC for VMC pre-compensated converter

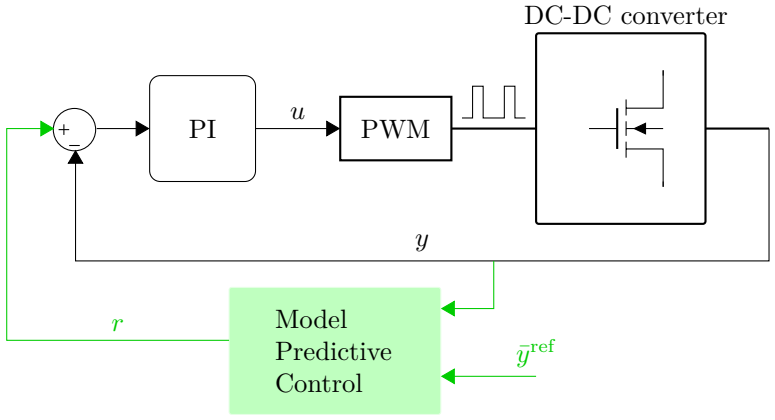
Even though the literature assessed MPC as one of the leading technology for the future controllers in power electronics [12], two motivations can prevent the direct use of MPC as a primal controller:

- a primal controller is already embedded into the physical system, either software or hardware, and cannot be modified;
- the dynamics of the system are too fast and a primal MPC is not feasible, thus a double, multi-rate, loop is preferred.

An MPC regulator for a pre-compensated power converter can overcome the above limits, by enhancing the performance of the primal controller, without substituting it. It is worth noticing that results of this section are based on a simple formulation, that can be the ground for a richer investigation in the field. Indeed, the considered converter is a synchronous DC-DC buck, which exhibits an LTI behavior. However, others DC-DC converter can be studied by applying the small-signal analysis which allows to derive an LTI model around the operative point from the bilinear model 8.1. In order to design this external MPC controller, the



(a) Standard VMC



(b) MPC-VMC

Figure 8.3: Block scheme of the standard VMC 8.3(a), and the MPC applied to the pre-compensated system 8.3(b). The external loop for MPC-VMC is highlighted in green, while the rest of the scheme remains unchanged.

mathematical model of the pre-compensated closed-loop system must be known. It is assumed that MPC is applied to a converter for which a primal VMC is available, as discussed in Section 8.1. This new control scheme is referred to as MPC-VMC. The standard VMC, and the MPC-VMC block schemes are shown in Figures 8.3(a) and 8.3(b), respectively. The derivation of the closed-loop model considers a VMC

implemented with a PID regulator, in the form

$$u_k = \left(K_p + K_i T_s \frac{z}{z-1} + K_d \frac{\delta}{1 + \delta T_s \frac{z}{z-1}} \right) e_k, \tag{8.4}$$

where K_p, K_i and K_d are the proportional, integral and derivative gains, T_s is the sampling frequency, $\delta = \frac{1}{t_f}$ is the derivative filtering term and e_k is the tracking error. The discrete-time state-space model of (8.4) is:

$$x^p_{(k+1)} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix}}_{A^p} x^p_k + \underbrace{\begin{bmatrix} \tilde{K}_i \\ -\tilde{K}_d(1-\alpha) \end{bmatrix}}_{B^p} u^p_k \tag{8.5a}$$

$$y^p(k) = \underbrace{\begin{bmatrix} 1 & 1 \end{bmatrix}}_{C^p} x^p_k + \underbrace{\begin{bmatrix} K_p + \tilde{K}_i + \tilde{K}_d \end{bmatrix}}_{D^p} u^p_k, \tag{8.5b}$$

where the p-superscript stands for *primal controller* and

$$\tilde{K}_i = K_i T_s, \quad \tilde{K}_d = \frac{K_d}{T_s + t_f}, \quad \alpha = \frac{t_f}{t_f + T_s}. \tag{8.6}$$

One can trivially verify that $y^p_k \equiv u_k$ and $e(k) \equiv u^p_k \equiv y^{\text{ref}}_k - y_k$ where $y^{\text{ref}}_k \in \mathbb{R}^{n_y}$ is the reference signal for the controlled system, that is the output voltage reference.

Consider a DC-DC synchronous buck converter, as the one in figure 8.2(a), and let $y^c_k \equiv y_k$ and $x^c_k \in \mathbb{R}^{n_{x_c}}$ be the output and the state vector of the extended system, such that $x^c_k = [x^p_k \; x_k]'$, then the extended open-loop system in state-space form is found to be:

$$x^c_{k+1} = \underbrace{\begin{bmatrix} A^p & 0 \\ BC^p & A \end{bmatrix}}_{A^c} x^c_k + \underbrace{\begin{bmatrix} B^p \\ BD^p \end{bmatrix}}_{B^c} u^p_k \tag{8.7a}$$

$$y^c_k = \underbrace{\begin{bmatrix} DC^p & C_d \end{bmatrix}}_{C^c} x^c_k + \underbrace{\begin{bmatrix} DD^p \end{bmatrix}}_{D^c} u^p_k, \tag{8.7b}$$

with the tracking error $e_k \equiv u^p_k$ as its only input. Considering that $D = M_{0,0}$ implies $D^f = M_{0,0}$, the closed-loop model derived from (8.7)

is:

$$x^c(k+1) = A^f x_k^c + B^f y_k^{\text{ref}} \tag{8.8a}$$
$$y^c(k) = C^f x_k^c \tag{8.8b}$$

where, by setting $d \triangleq (D^c + 1)^{-1}$, the following equations hold

$$A^f = A^c - B^c C^c d \tag{8.9a}$$
$$B^f = B^c - B_c D^c d \tag{8.9b}$$
$$C^f = C^c d \tag{8.9c}$$
$$\tag{8.9d}$$

Model (8.8), with matrices defined as in (8.9), represents the closed-loop system of controlled by a linear PID regulator. With the above derivation, the MPC problem is formulated as:

$$\min_{\Delta y^{\text{ref}}} \quad \sum_{i=0}^{N_p-1} \|W_y(y_{k+i+1|k}^c - \bar{y}_k^{\text{ref}})\|_2^2 + \sum_{j=0}^{N_u-1} \|W_{\Delta u} \Delta y_{k+h|k}^{\text{ref}}\|_2^2 \tag{8.10a}$$
$$\text{s.t.} \quad x_{c,k|k} = x_k^c, \tag{8.10b}$$
$$x_{k+i+1|k}^c = A^f x_{k+i|k}^c + B^f y_{k+i|k}^{\text{ref}}, \tag{8.10c}$$
$$y_{k+i+1|k}^c = C^f x_{k+i+1|k}^c, \tag{8.10d}$$
$$\Delta y_{k+N_u+j|k}^{\text{ref}} = 0, \ j = 0, \dots, N_p - N_u - 1, \tag{8.10e}$$
$$i = 0, \dots, N_p - 1,$$
$$h = 0, \dots, N_u - 1.$$

As already discussed, problem (8.10) can be casted into a parametric QP problem 3.15, with parameter vector:

$$\theta_k = [y_{k-1}^{\text{ref}} \ \ \hat{x}_c(k) \ \ \bar{y}_k^{\text{ref}}]^T. \tag{8.11}$$

The solution z^* of the unconstrained QP problem is analytic in this case, and equal to:

$$z^* = H^{-1} F \theta_k. \tag{8.12}$$

Table 8.2: Hardware and control specifications for experimental VMC-MPC tests

Parameter	Value	Units
DC-DC Buck Converter		
Input Voltage Range	4.75-14	V
Output Voltage Range	0.7-3.6	V
Switching Frequency	400	kHz
L	0.9	μH
R_l	2.2	mΩ
C	470	μF
R_{on}	3.6	mΩ
R	1	Ω
Controllers' Parameters		
Switching Frequency	400	kHz
Primal Control Frequency	400	kHz
MPC Control Frequency	100	kHz
K_p	0.0195	
K_i	350	
Prediction horizon	10	
Control horizon	5	
Measured output weight	5	
Manipulated variable rate weight	0.1	

Being z^* the optimal sequence of input increments, only the first n_u components are considered and applied to the system. Thus, the solution of the unconstrained MPC problem reduces to a matrix vector product, where the first n_u rows of $H^{-1}F$ are computed offline and stored. However, for such high speed problems as power converters, even the unconstrained solution does not represent a negligible cost for low-power embedded boards. The advantage of the double, multi-rate, loop is the possibility to run MPC in a slower task respect to the primal controller. Considering both the solution time of the QP problem and the computation of the estimated state through a delayed Kalman, the complexity c of the control algorithm can be explicitly computed as

$$c = (2n_p - 1)n_u + (2n_y + 2n_u + 2n_{x_c} - 1)n_{x_c}. \tag{8.13}$$

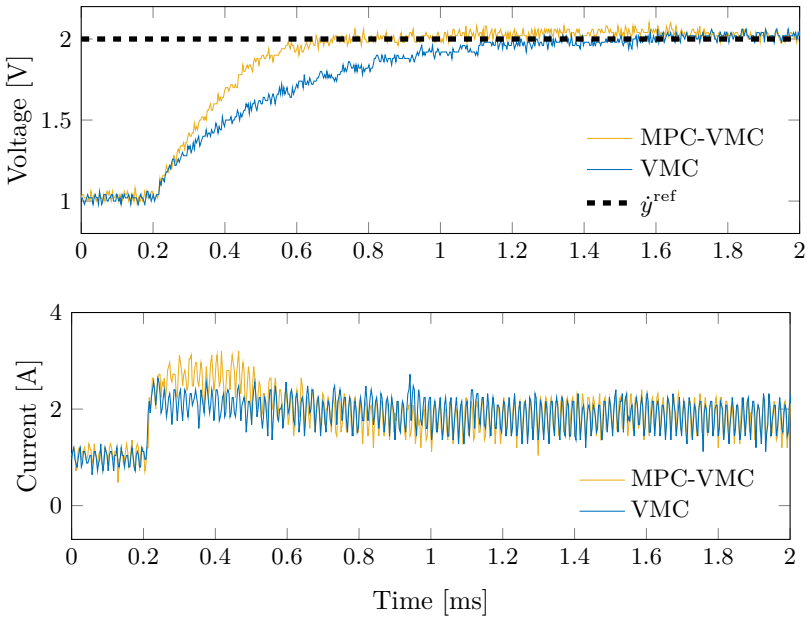


Figure 8.4: Experimental results, with an increasing output voltage step. Comparison between standard VMC and and MPC-VMC. From top to bottom: the output voltage; the inductor current.

With a very similar procedure MPC for a CMC pre-compensated DC-DC converter can be obtained, and preliminary results about its application can be found in [69].

8.2.1 Experimental results

MPC-VMC has been experimentally tested, and compared to the standard VMC. The PTD08A010WAD 10 A synchronous buck converter has been used, which is commercially available by Texas Instruments©. The values of its main components are listed in Table 8.2. The control scenario consists into supplying a 1 Ω-4 W load with 1 V and 2 V DC voltage. The input supply is 9 V, and the switching frequency is set to 400 kHz. Preliminary simulation tests on rapid prototyping software, in the specific PSIM by Powersim© Inc [128, 174], ease the controller

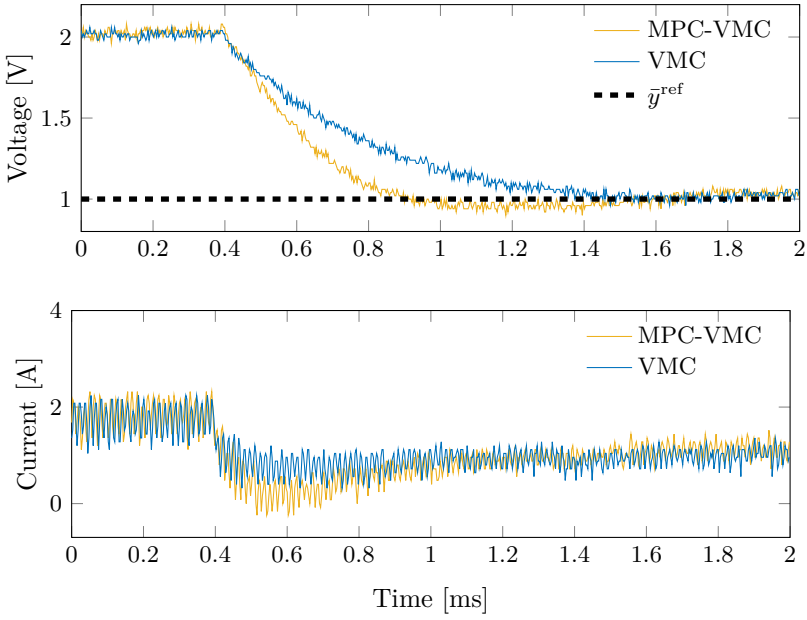


Figure 8.5: Experimental results, with a decreasing output voltage step. Comparison between standard VMC and and MPC-VMC. From top to bottom: the output voltage; the inductor current.

tuning, whose design parameters are collected in the second part of Table 8.2. PSIM allows for the design of accurate circuitry models, including parasitic effects, and digital control algorithms. The primal controller runs at 400 kHz, and has been tuned according to small-signal analysis, cf. Section8.1. The MPC applied to the pre-compensated system is running at 100 kHz, to demonstrate the advantages obtained even when the reference is changed at a slower frequency respect to the primal controller one. Two different tests have been proposed. The first consists in a positive step in the reference voltage from 1 V to 2 V. The second one consists in a negative step from 2 V to 1 V. For both tests the MPC-VMC performance are compared to standard VMC. Obviously, for both VMC and MPC-VMC, the primal controller is the same, namely a PI regulator with the same design parameters. Test acquisitions are collected with a Tektronix DPO3014 Digital Phosphor Oscilloscope, and

Table 8.3: Experimental Improvements of MPC-VMC respect to standard VMC under step variations

		VMC	MPC-VMC
Positive Step	τ_r[ms]	0.735	**0.3024**
	τ_s[ms]	0.917	**0.3683**
Negative Step	τ_r[ms]	0.725	**0.3928**
	τ_s[ms]	0.995	**0.4482**

Table 8.4: Performance Comparison of VMC and MPC-VMC Under Unknown Load Variations for a Positive Reference Step

Load	VMC		MPC-VMC	
R[Ω]	τ_r[ms]	τ_s[ms]	τ_r[ms]	τ_s[ms]
0.2	0.51	0.65	**0.31**	**0.39**
0.6	0.67	0.84	**0.38**	**0.49**
1	0.72	0.91	**0.41**	**0.53**
1.4	0.74	0.93	**0.42**	**0.55**
1.8	0.76	0.96	**0.44**	**0.56**

the entire control routine has been coded on the F28335 Delfino DSP by Texas Instruments©, whose specifications have been already introduced in Section 7.3. Two control interrupts regulate the execution of the primal controller and the MPC loop. Figures 8.4 and 8.5 present the experimental results for the increasing and decreasing step in the output voltage reference, respectively. The quantitative comparison of the results is detailed in Table 8.3. During the positive step signal MPC-VMC guarantees a reduction of 58.86% and 59.84% of the rise time and the settling time, respectively. During the negative step, the rise time and the settling time are reduced by 45.82% and 54.95% respectively. Fig. 8.4 and Fig. 8.5 show also the inductor current which has a faster dynamics in MPC based control, but still keeping the transient peak restrained as expected.

To verify the sensitivity to parameters uncertainties, the two algo-

rithms have been tested under load variations. The tuning of the two controllers is the same of the above experiment, namely Table 8.2, but the load value has been changed from a minimum of $0.2\,\Omega$ to a maximum of $1.8\,\Omega$. Table 8.4 collects the results when comparing the standard VMC and MPC-VMC in several perturbed scenarios, during a positive voltage reference step. It is evident that MPC-VMC is always able to improve the performance, even when the load R is significantly changed with respect to the nominal value used for controller tuning.

8.3 Unified robust current observer

This section shows that it is possible to unify the formulation of nonlinear observers for synchronous and asynchronous buck, boost, and buck-boost converters. A single observer which can be used on the same control unit to estimate the currents of six different types of converters is obtained. Code efficiency and its memory occupation, calibration and certification are facilitated. These advantages are of utmost importance in MPC framework, especially when sensorless control is required. Without any restriction, the observer is proposed for a standard CMC algorithm, as shown in Figure 8.6, but it can be directly used in any CMC-based control scheme. The yellow block in the figure represents a current observer, which is designed according to results presented in the following theorem.

Theorem 8.3.1. *Consider the asynchronous and synchronous buck, boost and buck-boost converters and their bilinear mathematical representation as in Equation* (8.1)*, parametrized with Table 8.1. Let the observed current $\hat{x}_1$ and the observed voltage $\hat{x}_2$ have an estimation error $\tilde{x}_1$ and $\tilde{x}_2$, respectively, defined as follows:*

$$\tilde{x}_1 = \hat{x}_1 - x_1 \tag{8.14a}$$

$$\tilde{x}_2 = \hat{x}_2 - x_2 \tag{8.14b}$$

being x_2 the measurable output voltage, and x_1 the unmeasurable current. Denote with $\hat{A}$ the nominal dynamics matrix, i.e. :

$$\hat{A} \equiv A|_{\Delta_R=0}. \tag{8.15}$$

Define

$$\rho\left(\Delta_R\right) = -\frac{\Delta_R}{(\hat{R}+\Delta_R)\hat{R}} \tag{8.16}$$

and $\bar{\rho} = \max\left(\rho(\Delta_R^{\min}), \rho(\Delta_R^{\max})\right)$*. Given* K *and* α *two strictly positive gains, the nonlinear observer*

$$\dot{\hat{x}} = \hat{A}\hat{x} + Bu + F_1\hat{x}_1 u + F_2\hat{x}_2 u + B_v + W\left(x_2, \tilde{x}_2\right) \tag{8.17}$$

with $\hat{x}$ *the estimated state vector, and*

$$W\left(x_2, \tilde{x}_2\right) = \begin{bmatrix} 0 \\ -K\tilde{x}_2 + \eta\left(x_2, \tilde{x}_2\right) \end{bmatrix} \tag{8.18}$$

where $\eta\left(x_2, \tilde{x}_2\right) \equiv C^{-1}\operatorname{sgn}(\tilde{x}_2)(\bar{\rho}|x_2|+\alpha)$*, is asymptotically convergent to the real state, and robust with respect to bounded variations* $\Delta_r^{\min} \leq \Delta_R \leq \Delta_R^{\max}$*.*

Proof. The error dynamics, obtained by replacing the observer's function (8.17) into the error equation (8.14), is found to be

$$\dot{\tilde{x}}(t) = \hat{A}\tilde{x} + F_1\tilde{x}_1 u + F_2\tilde{x}_2 u + W + \begin{bmatrix} 0 & C^{-1}\rho(\Delta_R) \end{bmatrix}^T x_2. \tag{8.19}$$

Let $u' \equiv 1 - u$, and consider the Lyapunov function

$$V = \frac{1}{2}\left(L\tilde{x}_1^2 + C\tilde{x}_2^2\right) \tag{8.20}$$

which is used for each converter. For all the asynchronous converters the derivative of Equation (8.20) along the solution of the error dynamics is equivalent, namely:

$$\begin{aligned} \dot{V} = -(R_L + uR_{on} + u'R_d)\tilde{x}_1^2 - \hat{R}^{-1}\tilde{x}_2^2 + \rho(\Delta_R)x_2\tilde{x}_2 + \\ -CK\tilde{x}_2^2 + C\eta\left(x_2, \tilde{x}_2\right)\tilde{x}_2. \end{aligned} \tag{8.21}$$

Similarly, all the synchronous converters share the same Lyapunov deriva-

tive along the solution of the error dynamics, that is

$$\dot{V} = -\left(R_L + R_{on}\right)\tilde{x}_1^2 - \hat{R}^{-1}\tilde{x}_2^2 + \rho(\Delta_R)x_2\tilde{x}_2 - CK\tilde{x}_2^2 + C\eta\left(x_2, \tilde{x}_2\right)\tilde{x}_2. \tag{8.22}$$

For both the Lyapunov derivatives in (8.21) and (8.22), the following relation holds:

$$\dot{V} < \rho(\Delta_R)x_2\tilde{x}_2 + C\tilde{x}_2\eta \leq \bar{\rho}x_2|\tilde{x}_2| + C\tilde{x}_2\eta = -\alpha|\tilde{x}_2|, \tag{8.23}$$

which proves the Theorem 8.3.1. □

In brief, the robust nonlinear observer (8.17) can be used for any of the six converters' topologies presented in Section 8.1. They have been found to have a Lyapunov derivative along the errors' dynamics that can be enforced to be strictly negative by the same nonlinear observer function. Equations (8.17) and (8.18), parametrized with the values in Table 8.1, are used for the observer's implementation. The code is simple, and it requires only sums, products and *if-then* operations, therefore the computational burden of the observer can be exactly computed. In the worst-case, that is with all non-empty matrices in Equation (8.17), 35 flops are required to estimate the current. Compared with respect to a well known non-robust observer, based on averaging model, the computational requirements increase only by 7 flops [135]. The digital implementation of the observer requires an approximated discrete time version of Equations (8.17) and (8.18). The backward difference equation $\dot{\hat{x}}(t) \approx \frac{\hat{x}(k)-\hat{x}(k-1)}{T_c}$ can be used to accomplish the task, where $T_c \geq T_s$ stands for the control period. When dealing with very high sampling frequency, the approximation of continuous time control law with backward difference guarantees satisfactory performance, [84]. The observer requires the design of $\Delta_R^{\max}$, α and K in Eqs. (8.17) and (8.18). The selection of $\Delta_R^{\max}$ is a straightforward step, as it represents the maximal variation allowed for the load to be supplied. This value is used to derive $\bar{\rho}$ and consequently the W matrix of the observer, as in Equation (8.18). $\alpha \propto \frac{1}{t_s}$ where t_s is the rising time of the current estimation, and K is the weight for the estimation error. For both α and K a trial and error campaign is necessary. Generally, a greater value is better for performance improvement, but it may cause overshoots and oscillations

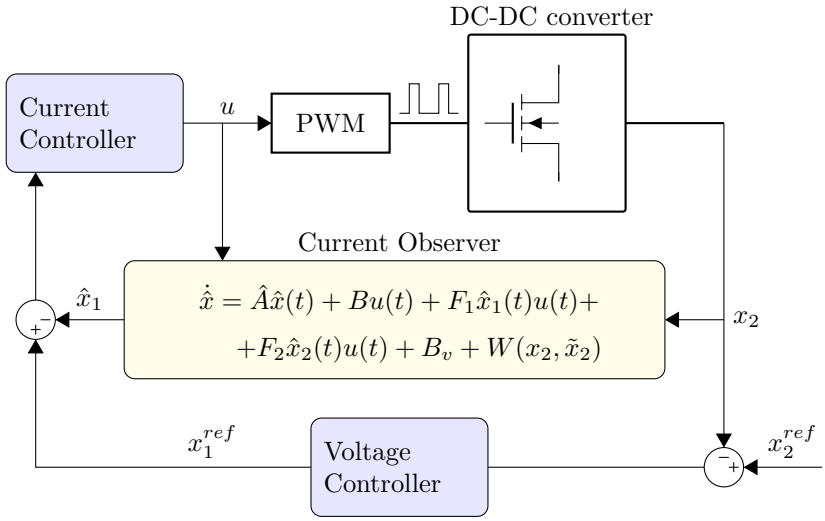


Figure 8.6: Sensorless control scheme. x_2 is the output voltage, x_1 is the inductor current and u is the duty cycle of the PWM.

around the steady state value. For this reason, the design phase can be carried out in a simulation environment, as it is clarified in Section 8.3.1.

8.3.1 Experimental results

The performances of the nonlinear robust current observer have been widely tested in both simulation (PSIM software) and experimental setups, and please refer to [78, 86, 127] for a more detailed analysis of the results. The parameters of the designed power converters are collected in Table 8.1. As far as simulation results, an asynchronous buck, an asynchronous boost and a synchronous buck-boost (in step-up mode), have been used to assess the observer's performance. The control and the switching frequencies have been set both to 350 kHz. All the parameters are collected in Table 8.5. The control scheme in Figure 8.6 is used to regulate the output voltage of the converters. The outer and inner loops implement linear PI-regulators, which have been tuned with the

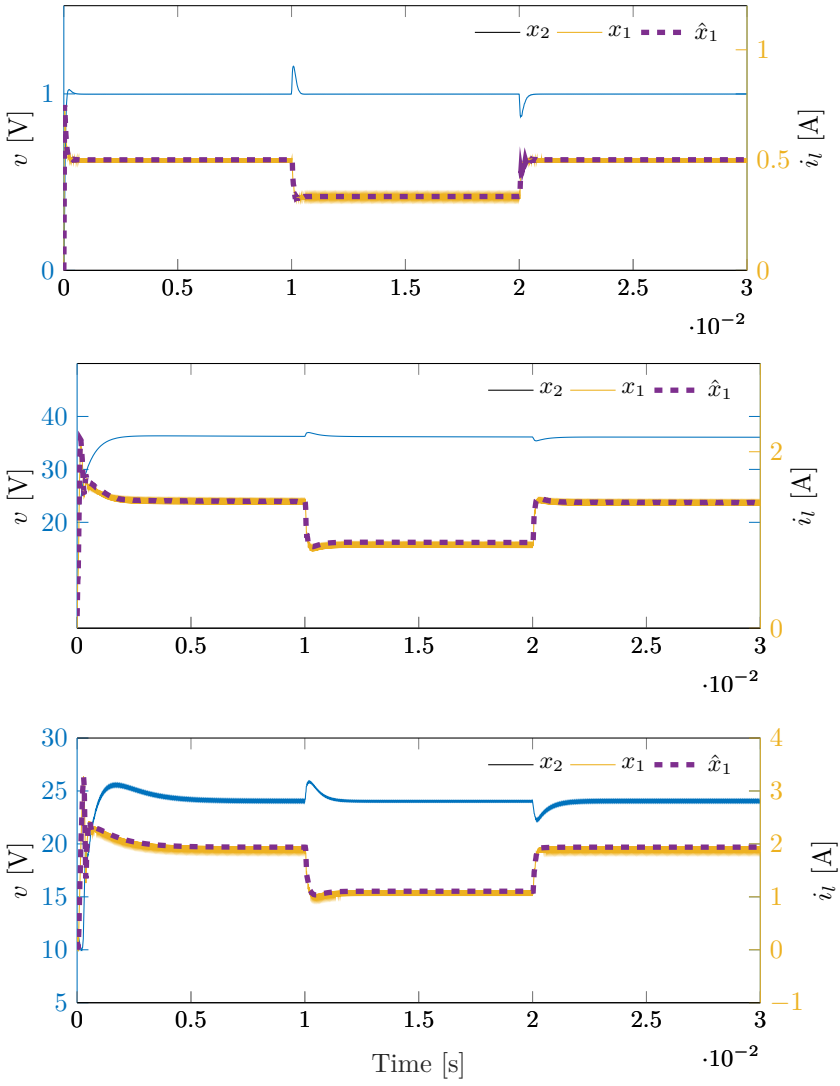


Figure 8.7: Simulation results for the proposed SCMC on output voltage and comparison between real and estimated current are presented. From top to bottom: *i*) asynchronous buck, *ii*) asynchronous boost and *iii*) synchronous buck-boost in step-up mode.

!htbp

Table 8.5: Parameters of the DC-DC converters, for both asynchronous and synchronous topologies.

Variable	Units	Buck	Boost	Buck-Boost
V_g	[V]	9	24	12
v_{ref}	[V]	1	36	24
L	[μH]	0.9	750	150
R_l	[Ω]	0.002	0.2	0.2
R_{on}	[Ω]	0.004	0.09	0.09
C	[μF]	460	40	20
V_d	[V]	0.7	0.7	0.7
R_d	[Ω]	0.3	0.3	0.3
$\hat{R}$	[Ω]	2	40	40
ΔR	[Ω]	1	20	30

well known small-signal analysis, cf. Section 8.1, in order to guarantee a 10 dB gain margin and more than 50° phase margin. The simulation scenario consists of a voltage tracking starting from a quiescent condition. The tests include unknown step changes in the load value, in order to confirm the robustness of the observer. The only measure available is the output voltage, which is sensed synchronously with the beginning of the PWM period. Figure 8.7 shows the output voltage tracking, and the comparison between real and estimated current for the three converters tested. From 0.01 s to 0.02 s a step change in the load value is applied, according to the quantities in Table 8.1. The figures show that the sensorless control correctly steers the output voltage to the desired value and, the stabilization is minimally affected by the load change. More importantly, the observer correctly estimates the real current value, even under an unknown load change. It is worth mentioning that this step variation on the load is high, being the 50% of the nominal value.

For what concern the experimental tests, the setup is the same used to show the MPC-VMC operation in of Section8.2.1, namely the Digital Power Experimenter Kit (TMDSDCDC2KIT) with a PTD08A010WAD 10-A synchronous buck converter. The experimental test consists again in supplying a 1-Ω@4-W load with 1-V and 2-V DC voltage, with a 9 V supply. The synchronous buck converter is switched by a PWM at a frequency of 350 kHz, which is equal to the control frequency. The

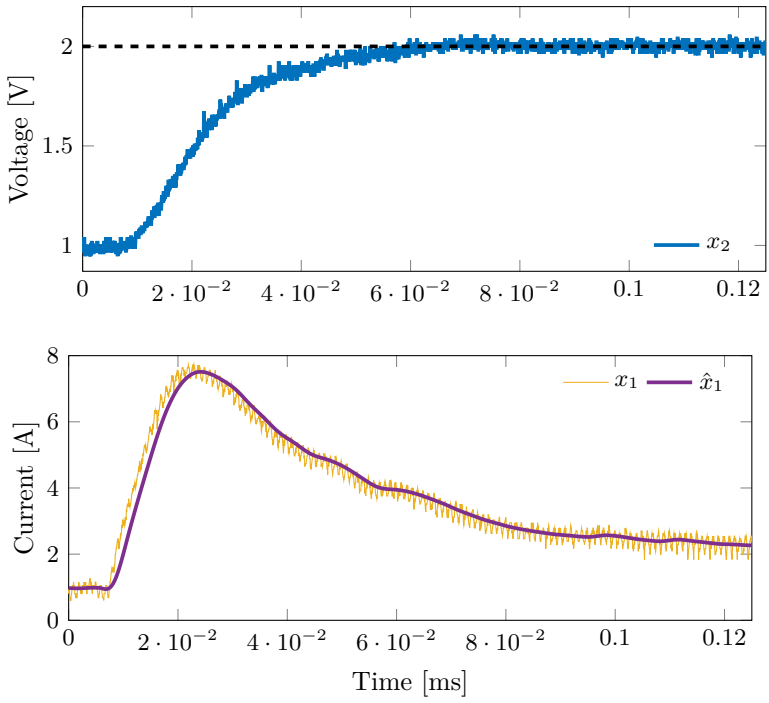


Figure 8.8: Experimental results for voltage step change. From up to bottom, the output voltage and the current. Real current (cyan) and estimated current (black) are shown in the bottom figure.

development control unit is the F28335 Delfino DSP. In order to test the robustness of the proposed observer, a disturbance load of $1\,\Omega$@$5\,\mathrm{W}$ is connected in parallel to the nominal load $\hat{R}$ and actuated by a digital switch. In this configuration, a variation of the 50% of the nominal load is applied when the switch is turned on/off. The observer design parameters have been derived through simulation analysis, and they guarantee a fast response of the observer, without causing transient overshoots and steady state oscillations. In the specific, $K = 1$, $\alpha = 10^{-4}$ and $\Delta_R^{\max} = 1$ (corresponding to a robustness against a variation of 100% of the load value) have been set.

Another key factor for a successful implementation is the computational complexity. The sum of ADC acquisition, control and observer

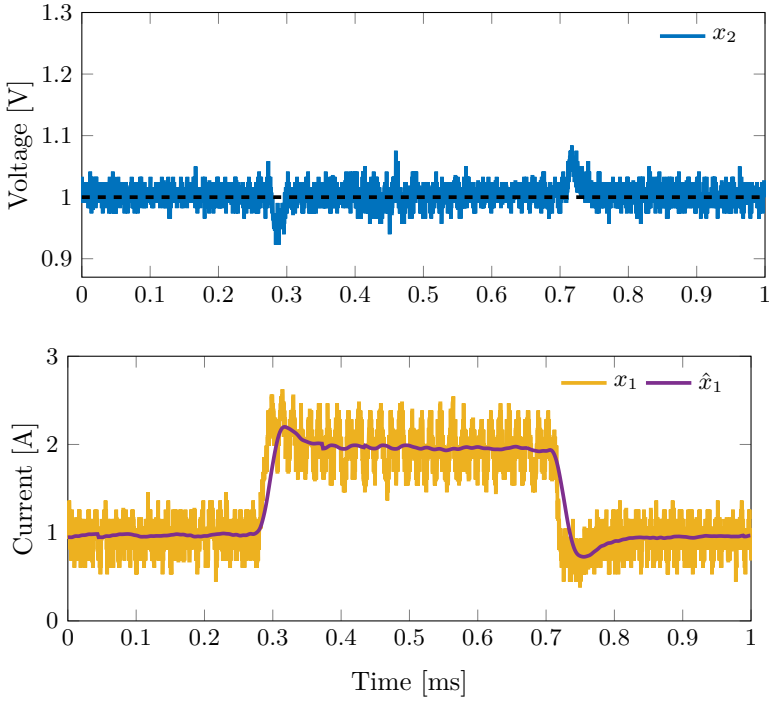


Figure 8.9: Experimental results for load step change. From up to bottom, the output voltage and the current. Real current (cyan) and estimated current (black) are shown in the bottom figure.

times must be less than the sampling interval for a feasible implementation. The run time at each step was measured with an high precision internal clock of the TI-F28335 device, and it is less than 0.5 µs. Figures 8.8 and 8.9 show the oscilloscope acquisitions of the voltage and load step-changes. Output voltage and input current have been acquired with a Tektronik DPO3014 oscilloscope. The voltage reference perform a step from 1V to 2V in Figure 8.8, and the load variation is the 50% of the nominal value in Figure 8.9 from 0.3 ms to 0.7 ms. In order to verify the correct estimation of the inductor current, $\hat{i}_l$ digital value is acquired from the DSP through a serial communication with an host PC. The simulation results obtained in the first part of the section are confirmed

by experimental test, as the voltage is successfully regulated and the estimation of the load current is correct even under load variations of 50% of the nominal value.

Chapter 9

Conclusions

This thesis has laid the foundations of exact complexity certification for embedded MPC applications, by proposing an algorithm to exactly calculate the worst-case time needed to solve a Quadratic Programming (QP) problem. Hopefully, the solver certification and the novel acceleration techniques here proposed will help the spread of Model Predictive Control (MPC) in embedded systems, by making the safe implementation in terms of maximum computational time a reality. The thesis has presented novel theoretical results and algorithms for embedded MPC, as well as the experimental application for the control of electrical drives and power converters, two fields that would certainly benefit from certifiable and efficient MPC implementation. The following sections summarize the main contributions of the thesis, and propose possible future developments.

9.1 Summary and Impact of the thesis

After covering the basics of MPC and embedded optimization in the first chapters, Chapter 5 has proposed the exact complexity certification of dual active-set solvers for parametric QP problems. By iteratively partitioning the parameter space in polyhedral regions that share the same number of iterations, the worst-case complexity in terms of maximum number of iterations and flops can be computed. This result fulfills the lack of tight bounds for active-set algorithms, for which only average and probabilistic results were available before in the literature. Moreover, having a QP solver for which the worst-case number of flops can be exactly computed is crucial for promoting MPC in embedded appli-

cations. Indeed, certifying the impossibility to incur in a QP instance which solution takes more than the available time is paramount for production oriented controllers, and it has chance to favor the growth in popularity of MPC for embedded systems in the next future. In addition, the certification algorithm helps choosing the fastest solver among the family of dual active-set methods and explicit MPC. This exact comparison in terms of flops changes for the better the controller design procedure, which currently relies on experience or extensive simulation campaigns to select the best solver. The chapter has detailed the theoretical foundations and the algorithm steps, verified on well-known QP problems.

Two novel acceleration techniques for MPC have been then presented in Chapter 6. With the aim of being impactful for embedded MPC, the focus of the two methods is having a certified complexity and, more importantly, improving the worst-case time needed to solve the QP problem. Indeed, in embedded MPC the usefulness of algorithms that improve the average case is questionable, as if the worst-case exceeds the time bound, the controller cannot be implemented no matter improvements in the average time. Therefore, a semi-explicit MPC with partial enumeration and a constraint selection strategy for dual active-set methods have been proposed, with the unique feature of both having a certifiable improvement in the worst-case, contrarily to the current literature. The particular semi-explicit MPC here presented differs from other semi-explicit algorithms also because the trade-off between memory occupancy and performance improvement can be optimally chosen. On the other hand, the acceleration by a different selection of active constraints is independent from the explicit solution and from the optimizer at the previous time step, making it usable in parallel with other acceleration methods.

The second part of the thesis is focused on experimental applications. In Chapter 7, some of the above mentioned algorithms have been used to control a permanent magnet synchronous motor, proposing a model predictive torque control solved online on a cheap computational unit. Contrarily to the trend of considering MPC for electrical motors doable only if it is solved by finite control set MPC or offline multiparametric programming, the thesis has shown that MPC with an online QP solver is feasible on a board commonly used for motor control. The certification

algorithm has also demonstrated that, for the particular application, the MPC solved online is even faster (and obviously memory cheaper) than the corresponding multiparametric version. This result can shed light on online MPC for electrical motors, which is not currently receiving much interest in the field because considered unmanageable. Even though its implementation is not easy due to the high sampling frequency, the availability of the exact certification can clear all the doubts about an eventual complexity out of the time bounds.

Finally, Chapter 8 has addressed some of the issues in the application of MPC for DC-DC power converters. Being systems with very high sampling frequencies and cheap computational boards, implicit MPC can be rarely used as the primal controller. Therefore the thesis has proposed the application of MPC to a pre-compensated system, regulated in the specific by the standard voltage mode control. Besides the obvious computational advantages obtained by a multi-rate controller, the proposed control architecture overcomes the common issue, in power converters, of having an hardware-based primal controller that cannot be changed. Additionally, the problem of multiple DC-DC converters on the same board has been addressed, from the point of view of algorithm complexity. In particular, the thesis has focused on sensorless control which is becoming very popular in the field to avoid the use of a current observer. Many types and configurations of converters usually coexist in power supplies, and therefore multiple non-linear observers are required, one for each converter. A unified nonlinear observer, robust to bounded load variations, has been therefore proposed and experimentally tested.

9.2 Future developments

Given the novelty of the topics discussed in the thesis, many extensions to the work are possible. In a first stage, the complexity certification of primal active-set methods can be investigated. Even though dual solvers are faster and more suited for embedded MPC, some applications may prefer the use of primal algorithms because their execution can be interrupted without incurring in infeasible sub-iterations. Nonetheless, warm-start techniques are easier to be implemented for primal algorithms. In some particular MPC formulations the corresponding QP problem does not have general constraints, that is when only inputs

or input rates are weighted in the cost function. Efficient solvers exploit the box-constrained structure of these cases, sensibly improving the computational cost. Therefore, the worst-case certification of such solvers would be of much interest.

For what concerns the experimental results presented in Chapter 7, the mechanical subsystem of the electrical motor can benefit from embedded MPC application as well, allowing for the optimization of the whole motor dynamics and the preview on the speed (or position) profile. The objective may be to investigate if, even with a more complex MPC formulation accounting for the whole dynamics, the online solution is still a successful option in cheap control units.

The use of MPC for DC-DC converters is another open topic where the results here presented can be used as a starting point for future research. As an example, the MPC for pre-compensated systems is a valid idea that can be used to control more complex power converters, and account for different types of primal controllers. Moreover, the unified current observer can be reformulated in the case of direct actuation of the switches, which is becoming popular for its fast transient response. Those algorithms exploiting the discrete nature of the transistors, such as finite control set MPC, can benefit of such a unified observer code to lower the memory and ease the design.

Bibliography

[1] M. Rafal and W. Stevens, "Discrete dynamic optimization applied to on-line optimal control," *AiChE Journal*, vol. 14, no. 1, pp. 85–91, 1968.

[2] E. B. Lee and L. Markus, "Foundations of optimal control theory," *SIAM series in applied mathematics*, vol. 11, no. 1, pp. 93–95, 1969.

[3] J. M. Maciejowski, *Predictive Control: With Constraints*. Pearson Education, 2002.

[4] E. Camacho and C. Bordons, *Model Predictive Control*, ser. Advanced Textbooks in Control and Signal Processing Series. Springer London, Limited, 2004.

[5] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967 – 2986, 2014.

[6] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, March 2010.

[7] S. Di Cairano, H. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinating active front steering and differential braking in the tire sideslip angles domain," *IEEE Trans. Contr. Systems Technology*, vol. 21, no. 4, pp. 1236–1248, Jul. 2013.

[8] L. Del Re, F. Allgöwer, L. Glielmo, C. Guardiola, and I. Kolmanovsky, *Automotive Model Predictive Control: Models, Methods and Applications*, ser. Lecture Notes in Control and Information Sciences. Springer London, 2010.

[9] S. Di Cairano, D. Yanakiev, A. Bemporad, I. Kolmanovsky, and D. Hrovat, "Model predictive idle speed control: Design, analysis, and experimental evaluation," *IEEE Trans. Contr. Systems Technology*, vol. 20, no. 1, pp. 84–97, 2012.

[10] E. N. Hartley and J. M. Maciejowski, "Field programmable gate array based predictive control system for spacecraft rendezvous in elliptical orbits," *Optimal Control Applications and Methods*, vol. 35, no. 7, pp. 585–607, 2015.

[11] S. Vazquez, J. Leon, L. Franquelo, J. Rodriguez, H. Young, A. Marquez, and P. Zanchetta, "Model predictive control: A review of its applications in power electronics," *IEEE Industrial Electronics Magazine*, vol. 8, no. 1, pp. 16–31, Mar. 2014.

[12] S. Kouro, M. Perez, J. Rodriguez, A. Llor, and H. Young, "Model predictive control: MPC's role in the evolution of power electronics," *IEEE Industrial Electronics Magazine*, vol. 9, no. 4, pp. 8–21, Dec 2015.

[13] N. Prakash, G. Cimini, A. Stefanopoulou, and M. Brusstar, "Assessing Fuel Economy from Automated Driving: Influence of Preview and Velocity Constraints while Following a Lead Vehicle Associated with Federal Test Procedures," in *ASME 2016 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, Oct 2016, p. *Accepted for Publication*.

[14] C. G., K. Y., and S. A., "Model Predictive Control for Real-time Position Tracking of a Catenary-free Tram," in *World Congress of the International Federation of Automatic Control*, Oct 2016, p. *Under Review*.

[15] J. L. Jerez, P. J. Goulart, S. Richter, G. A. Constantinides, E. C. Kerrigan, and M. Morari, "Embedded online optimization for model predictive control at megahertz rates," *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3238–3251, Dec 2014.

[16] O. Gulbudak and E. Santi, "FPGA-Based Model Predictive Controller for Direct Matrix Converter," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 7, pp. 4560–4570, July 2016.

[17] M. Rubagotti, P. Patrinos, and A. Bemporad, "Stabilizing linear model predictive control under inexact numerical optimization," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1660–1666, June 2014.

[18] E. N. Hartley, J. L. Jerez, A. Suardi, J. M. Maciejowski, E. C. Kerrigan, and G. A. Constantinides, "Predictive Control Using an FPGA With Application to Aircraft Control," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 1006–1017, May 2014.

[19] P. Patrinos, A. Guiggiani, and A. Bemporad, "A dual gradient-projection algorithm for model predictive control in fixed-point arithmetic," *Automatica*, vol. 55, pp. 226 – 235, 2015.

[20] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[21] A. Bemporad, "A multiparametric quadratic programming algorithm with polyhedral computations based on nonnegative least squares," *IEEE Trans. Automatic Control*, vol. 60, no. 11, pp. 2892–2903, 2015.

[22] M. Baotić, F. Borrelli, A. Bemporad, and M. Morari, "Efficient on-line computation of constrained optimal control," *SIAM Journal on Control and Optimization*, vol. 47, no. 5, pp. 2470–2489, 2008.

[23] P. Tøndel, T. Johansen, and A. Bemporad, "Evaluation of piecewise affine control via binary search tree," *Automatica*, vol. 39, no. 5, pp. 945 – 950, 2003.

[24] P. Patrinos and A. Bemporad, "An accelerated dual gradient-projection algorithm for embedded linear model predictive control," *IEEE Trans. Automatic Control*, vol. 59, no. 1, pp. 18–33, 2014.

[25] A. Domahidi and J. Jerez, "FORCES Professional," embotech GmbH (http://embotech.com/FORCES-Pro), Jul. 2014.

[26] A. Bemporad, "A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control," *IEEE Trans. Automatic Control*, vol. 61, no. 4, pp. 1111–1116, 2016.

[27] M. Rubagotti, P. Patrinos, A. Guiggiani, and A. Bemporad, "Real-time model predictive control based on dual gradient projection: Theory and fixed-point FPGA implementation," *International Journal of Robust and Nonlinear Control*, vol. 26, no. 15, pp. 3292–3310, 2016.

[28] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

[29] R. J. Vanderbei, "LOQO: an interior point code for quadratic programming," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 451–484, 1999.

[30] S. Richter, C. Jones, and M. Morari, "Computational complexity certification for real-time MPC with input constraints based on the fast gradient method," *IEEE Trans. Automatic Control*, vol. 57, no. 6, 2012.

[31] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[32] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Mathematical Programming*, vol. 27, no. 1, pp. 1–33, 1983.

[33] R. A. Bartlett and L. T. Biegler, "QPSchur: A dual, active-set, Schur-complement method for large-scale and structured convex quadratic programming," *Optimization and Engineering*, vol. 7, no. 1, pp. 5–32, 2006.

[34] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *Interna-*

tional Journal of Robust and Nonlinear Control, vol. 18, no. 8, pp. 816–830, 2008.

[35] P. Gill, N. Gould, W. Murray, M. Saunders, and M. Wright, "A weighted Gram-Schmidt method for convex quadratic programming," *Mathematical Programming*, vol. 30, no. 2, pp. 176–195, 1984.

[36] N. I. Gould and P. L. Toint, "A quadratic programming bibliography," 2000. [Online]. Available: ftp://ftp.numerical.rl.ac.uk/pub/qpbook/qp.pdf

[37] G. Cimini, D. Bernardini, A. Bemporad, and S. Levijoki, "Online model predictive torque control for Permanent Magnet Synchronous Motors," in *Industrial Technology (ICIT), 2015 IEEE International Conference on*, March 2015, pp. 2308–2313.

[38] A. Forsgren, P. Gill, and E. Wong, "Primal and dual active-set methods for convex quadratic programming," *Mathematical Programming*, pp. 1–40, 2015.

[39] P. Giselsson and S. Boyd, "Metric selection in fast dual forward backward splitting," *Automatica*, 2014.

[40] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[41] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Fast, large-scale model predictive control by partial enumeration," *Automatica*, vol. 43, no. 5, pp. 852 – 860, 2007.

[42] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear Model Predictive Control: Towards New Challenging Applications*, ser. Lecture Notes in Control and Information Sciences, D. R. L. Magni, F. Allgower, Ed., vol. 384. Berlin Heidelberg: Springer-Verlag, 2009, pp. 345–369.

[43] G. Pannocchia, S. J. Wright, and J. B. Rawlings, "Partial enumeration MPC: Robust stability results and application to an unstable

CSTR," *Journal of Process Control*, vol. 21, no. 10, pp. 1459–1466, 2011.

[44] M. Zeilinger, C. Jones, and M. Morari, "Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization," *IEEE Trans. Automatic Control*, vol. 56, no. 7, pp. 1524–1534, July 2011.

[45] M. Jost and M. Monnigmann, "Accelerating model predictive control by online constraint removal," in *IEEE Conference on Decision and Control*, Dec 2013, pp. 5764–5769.

[46] P. Cortes, M. Kazmierkowski, R. Kennel, D. Quevedo, and J. Rodriguez, "Predictive control in power electronics and drives," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 12, pp. 4312–4324, Dec 2008.

[47] J. Rodriguez and P. Cortes, *Predictive control of power converters and electrical drives*. John Wiley & Sons, 2012, vol. 40.

[48] Z. Mynar, L. Vesely, and P. Vaclavek, "PMSM Model Predictive Control With Field-Weakening Implementation," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 8, pp. 5156–5166, Aug 2016.

[49] J. Rodriguez, M. P. Kazmierkowski, J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. A. Young, and C. A. Rojas, "State of the art of finite control set model predictive control in power electronics," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 1003–1016, May 2013.

[50] M. Preindl and S. Bolognani, "Model Predictive Direct Torque Control With Finite Control Set for PMSM Drive Systems, Part 1: Maximum Torque Per Ampere Operation," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 1912–1921, Nov 2013.

[51] ——, "Model Predictive Direct Torque Control With Finite Control Set for PMSM Drive Systems, Part 2: Field Weakening Operation," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 648–657, May 2013.

[52] P. Karamanakos, T. Geyer, N. Oikonomou, F. D. Kieferndorf, and S. Manias, "Direct model predictive control: A review of strategies that achieve long prediction intervals for power electronics," *IEEE Industrial Electronics Magazine*, vol. 8, no. 1, pp. 32–43, March 2014.

[53] T. Geyer, "Computationally efficient model predictive direct torque control," *IEEE Transactions on Power Electronics*, vol. 26, no. 10, pp. 2804–2816, Oct 2011.

[54] T. Geyer, G. Papafotiou, and M. Morari, "Model Predictive Direct Torque Control−Part I: Concept, Algorithm, and Analysis," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 1894–1905, June 2009.

[55] M. Preindl and S. Bolognani, "Model Predictive Direct Speed Control with Finite Control Set of PMSM Drive Systems," *IEEE Transactions on Power Electronics*, vol. 28, no. 2, pp. 1007–1015, Feb 2013.

[56] S. Bolognani, S. Bolognani, L. Peretti, and M. Zigliotto, "Design and implementation of model predictive control for electrical motor drives," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 1925–1936, June 2009.

[57] J. Scoltock, T. Geyer, and U. K. Madawala, "A comparison of model predictive control schemes for mv induction motor drives," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 909–919, May 2013.

[58] S. Mariethoz, S. Almer, M. Baja, A. G. Beccuti, D. Patino, A. Wernrud, J. Buisson, H. Cormerais, T. Geyer, H. Fujioka, U. T. Jonsson, C. Y. Kao, M. Morari, G. Papafotiou, A. Rantzer, and P. Riedinger, "Comparison of Hybrid Control Techniques for Buck and Boost DC-DC Converters," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 5, pp. 1126–1145, Sept 2010.

[59] A. G. Beccuti, S. Mariethoz, S. Cliquennois, S. Wang, and M. Morari, "Explicit Model Predictive Control of DC−DC

Switched-Mode Power Supplies With Extended Kalman Filtering," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 1864–1874, June 2009.

[60] A. Suardi, S. Longo, E. C. Kerrigan, and G. A. Constantinides, "Energy-aware mpc co-design for dc-dc converters," in *Control Conference (ECC), 2013 European*, July 2013, pp. 3608–3613.

[61] S. K. Kim, C. R. Park, J. S. Kim, and Y. I. Lee, "A Stabilizing Model Predictive Controller for Voltage Regulation of a DC/DC Boost Converter," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 5, pp. 2016–2023, Sept 2014.

[62] I. Necoara, "Computational complexity certification for dual gradient method: Application to embedded MPC," *Systems & Control Letters*, vol. 81, pp. 49 – 56, 2015.

[63] I. Necoara, A. Patrascu, and A. Nedić, "Complexity certifications of first order inexact lagrangian methods for general convex programming," *arXiv preprint arXiv:1506.05328*, 2015.

[64] D. A. Spielman and S.-H. Teng, "Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time," *Journal of the ACM*, vol. 51, no. 3, pp. 385–463, May 2004.

[65] V. Klee and G. J. Minty, "How good is the simplex method," *Inequalities-III*, pp. 159–175, 1972.

[66] K. Borgwardt, "Probabilistic analysis of the simplex method," in *DGOR/NSOR*, ser. Operations Research Proceedings, H. Schellhaas, P. van Beek, H. Isermann, R. Schmidt, and M. Zijlstra, Eds. Springer Berlin Heidelberg, 1988, vol. 1987, pp. 564–575.

[67] D. Goldfarb, "On the complexity of the simplex method," in *Advances in Optimization and Numerical Analysis*, ser. Mathematics and Its Applications, S. Gomez and J.-P. Hennart, Eds. Springer Netherlands, 1994, vol. 275, pp. 25–38.

[68] M. Jost, G. Pannocchia, and M. Mönnigmann, "Online constraint removal: Accelerating MPC with a Lyapunov function," *Automatica*, vol. 57, no. 0, pp. 164 – 169, 2015.

[69] L. Cavanini, G. Cimini, and G. Ippoliti, "Model predictive control for the reference regulation of current mode controlled DC-DC converters," in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, July 2016, pp. 74–79.

[70] C. S. Lim, E. Levi, M. Jones, N. A. Rahim, and W. P. Hew, "Fcs-mpc-based current control of a five-phase induction motor and its comparison with pi-pwm control," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 149–163, Jan 2014.

[71] L. Rovere, A. Formentini, A. Gaeta, P. Zanchetta, and M. Marchesoni, "Sensorless finite-control set model predictive control for ipmsm drives," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 9, pp. 5921–5931, Sept 2016.

[72] A. Bemporad, "Reference governor for constrained nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 3, pp. 415–419, Mar 1998.

[73] E. Garone and M. M. Nicotra, "Explicit reference governor for constrained nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 5, pp. 1379–1384, May 2016.

[74] S. R. Oh and S. K. Agrawal, "A reference governor-based controller for a cable robot under input constraints," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 4, pp. 639–645, Jul 2005.

[75] S. Jade, E. Hellström, J. Larimore, A. G. Stefanopoulou, and L. Jiang, "Reference governor for load control in a multicylinder recompression hcci engine," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1408–1421, July 2014.

[76] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, 2016.

[77] L. Ciabattoni, G. Cimini, F. Ferracuti, G. Ippoliti, S. Longhi, R. Miceli, and G. Orlando, "On the design of observers robust

to load variations for synchronous converters," in *2015 International Conference on Renewable Energy Research and Applications (ICRERA)*, Nov 2015, pp. 1609–1614.

[78] G. Cimini, G. Ippoliti, G. Orlando, S. Longhi, and R. Miceli, "Robust current observer design for DC-DC converters," in *Renewable Energy Research and Application (ICRERA), 2014 International Conference on*, Oct 2014, pp. 958–963.

[79] C. Chakraborty, H. H. C. Iu, and D. D.-C. Lu, "Power converters, control, and energy management for distributed generation," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 7, pp. 4466–4470, July 2015.

[80] M. Kumar and R. Gupta, "Stability and sensitivity analysis of uniformly sampled dc-dc converter with circuit parasitics," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 2086–2097, Nov 2016.

[81] S. Smithson and S. Williamson, "A unified state-space model of constant-frequency current-mode controlled power converters in continuous conduction mode," *IEEE Transactions on Industrial Electronics*, vol. PP, no. 99, pp. 1–1, 2015.

[82] G. Herbst, "Comment on "A Unified State-Space Model of Constant-Frequency Current-Mode Controlled Power Converters in Continuous Conduction Mode "," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 5, pp. 3198–3200, May 2016.

[83] H.-C. Chen, "Interleaved Current Sensorless Control for Multiphase Boost-Type Switch-Mode Rectifier With Phase-Shedding Operation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 2, pp. 766–775, Feb 2014.

[84] Q. Zhang, R. Min, Q. Tong, X. Zou, Z. Liu, and A. Shen, "Sensorless Predictive Current Controlled DC-DC Converter with a Self-correction Differential Current Observer," *IEEE Transactions on Industrial Electronics*, vol. PP, no. 99, pp. 1–1, 2014.

[85] G.-S. Seo, J.-W. Shin, B.-H. Cho, and K.-C. Lee, "Digitally Controlled Current Sensorless Photovoltaic Micro-Converter for

DC Distribution," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 117–126, Feb 2014.

[86] G. Cimini, G. Ippoliti, G. Orlando, S. Longhi, and R. Miceli, "A unified observer for robust sensorless control of DC–DC converters," *Control Engineering Practice*, vol. 61, pp. 21 – 27, Apr 2017.

[87] H. Sira-Ramirez and R. Silva-Ortigoza, *Control Design Techniques in Power Electronics Devices*, ser. Power Systems. Springer London, 2006.

[88] R. Foley, R. Kavanagh, and M. Egan, "Sensorless Current Estimation and Sharing in Multiphase Buck Converters," *IEEE Transactions on Power Electronics*, vol. 27, no. 6, pp. 2936–2946, June 2012.

[89] S. Mahdi Alavi, M. Saif, and B. Shafai, "Accurate state estimation in DC-DC converters using a Proportional-Integral Observer (PIO)," in *IEEE 23rd International Symposium on Industrial Electronics (ISIE), 2014*, June 2014, pp. 1304–1309.

[90] R. Rockafellar, *Convex Analysis*, ser. Princeton landmarks in mathematics and physics. Princeton University Press, 1997.

[91] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 1999.

[92] S. Boyd and L. Vandenberghe, "Convex optimization," 2004.

[93] S. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733 – 764, 2003.

[94] L. Cavanini, G. Cimini, and G. Ippoliti, "Computationally efficient model predictive control for a class of linear parameter-varying systems," *Automatica*, p. *Under Review*, 2016.

[95] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789 – 814, 2000.

[96] P. Vouzis, M. Kothare, L. Bleris, and M. Arnold, "A system-on-a-chip implementation for embedded real-time model predictive control," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1006–1017, Sept 2009.

[97] S. Richter, T. Geyer, and M. Morari, "Resource-Efficient Gradient Methods for Model Predictive Pulse Pattern Control on an FPGA," *IEEE Transactions on Control Systems Technology*, vol. PP, no. 99, pp. 1–14, 2016.

[98] A. G. Wills, G. Knagge, and B. Ninness, "Fast Linear Model Predictive Control Via Custom Integrated Circuit Architecture," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 1, pp. 59–71, Jan 2012.

[99] P. Gill and E. Wong, "Methods for convex and general quadratic programming," *Mathematical Programming Computation*, vol. 7, no. 1, pp. 71–112, 2015.

[100] B. Houska, H. Ferreau, and M. Diehl, "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[101] G. Cimini, A. Bemporad, G. Ippoliti, and S. Longhi, "FRAM evaluation as unified memory for convex optimization algorithms," in *Education and Research Conference (EDERC), 2014 6th European Embedded Design in*, Sept 2014, pp. 187–191.

[102] R. A. Bartlett, A. Wachter, and L. T. Biegler, "Active set vs. interior point strategies for model predictive control," in *American Control Conference, 2000. Proceedings of the 2000*, vol. 6, 2000, pp. 4229–4233 vol.6.

[103] R. Milman and E. Davison, "A fast MPC algorithm using nonfeasible active set methods," *Journal of Optimization Theory and Applications*, vol. 139, no. 3, pp. 591–616, 2008.

[104] M. Herceg, C. N. Jones, and M. Morari, "Dominant speed factors of active set methods for fast mpc," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 608–627, 2015.

[105] C. E. Lemke, "A method of solution for quadratic programs," *Management Science*, vol. 8, no. 4, pp. 442–453, 1962.

[106] P. Gill and E. Wong, "Sequential quadratic programming methods," in *Mixed Integer Nonlinear Programming*, ser. The IMA Volumes in Mathematics and its Applications, J. Lee and S. Leyffer, Eds. Springer New York, 2012, vol. 154, pp. 147–224.

[107] R. Fletcher and S. Leyffer, "Numerical experience with lower bounds for MIQP branch-and-bound," *SIAM J. on Optimization*, vol. 8, no. 2, pp. 604–616, Feb. 1998.

[108] C. Schmid and L. Biegler, "Quadratic programming methods for reduced Hessian SQP," *Computers & Chemical Engineering*, vol. 18, no. 9, pp. 817–832, 1994.

[109] C. Buchheim, M. D. Santis, S. Lucidi, F. Rinaldi, and L. Trieu, "A feasible active set method with reoptimization for convex quadratic mixed-integer programming," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1695–1714, 2016.

[110] A. Bemporad, "Solving mixed-integer quadratic programs via nonnegative least squares," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 73 – 79, 2015.

[111] G. Cimini, D. Bernardini, and A. Bemporad, "An Efficient Constraint Selection Strategy in Dual Active-Set Methods for Model Predictive Control," *Systems & Control Letters*.

[112] A. Patrascu and I. Necoara, "Complexity certifications of first order inexact lagrangian and penalty methods for conic convex programming," *arXiv preprint arXiv:1506.05320*, 2015.

[113] ——, "Complexity certifications of inexact projection primal gradient method for convex problems: Application to embedded mpc," in *2016 24th Mediterranean Conference on Control and Automation (MED)*, June 2016, pp. 125–130.

[114] M. Preindl, S. Bolognani, and C. Danielson, "Model Predictive Torque Control with PWM using fast gradient method," in *Applied Power Electronics Conference and Exposition (APEC), 2013 Twenty-Eighth Annual IEEE*, March 2013, pp. 2590–2597.

[115] G. Cimini and A. Bemporad, "Exact Complexity Certification of Active-Set Methods for Quadratic Programming," *IEEE Transactions of Automatic Control*, 2016, submitted for publication.

[116] C. Rao, S. Wright, and J. Rawlings, "Application of interior-point methods to model predictive control," *Journal of Optimization Theory and Applications*, vol. 99, no. 3, pp. 723–757, 1998.

[117] T. Gal and J. Nedoma, "Multiparametric linear programming," *Management Science*, vol. 18, pp. 406–442, 1972.

[118] M. J. Best and N. Chakravarti, "Active set algorithms for isotonic regression; a unifying framework," *Mathematical Programming*, vol. 47, no. 1-3, pp. 425–439, 1990.

[119] A. Bemporad and E. Mosca, "Fulfilling hard constraints in uncertain linear systems by reference managing," *Automatica*, vol. 34, no. 4, pp. 451–461, 1998.

[120] A. Bemporad, A. Casavola, and E. Mosca, "Nonlinear control of constrained linear systems via predictive reference management," *IEEE Trans. Automatic Control*, vol. AC-42, no. 3, pp. 340–349, 1997.

[121] M. Baotic, "Optimal control of piecewise affine systems – a multiparametric approach," Ph.D. dissertation, ETH Zürich, Switzerland, 2005.

[122] Y. Zhang, H. Yang, and B. Xia, "Model-predictive control of induction motor drives: Torque control versus flux control," *IEEE Transactions on Industry Applications*, vol. 52, no. 5, pp. 4050–4060, Sept 2016.

[123] X. Li and P. Shamsi, "Model predictive current control of switched reluctance motors with inductance auto-calibration," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 6, pp. 3934–3941, June 2016.

[124] M. Rivera, C. Rojas, A. Wilson, J. Rodriguez, J. Espinoza, C. Baier, and J. Muñoz, "Review of predictive control methods to improve the input current of an indirect matrix converter," *IET Power Electronics*, vol. 7, no. 4, pp. 886–894, April 2014.

[125] L. Ciabattoni, G. Cimini, M. Grisostomi, G. Ippoliti, S. Longhi, and E. Mainardi, "Supervisory control of PV-battery systems by online tuned neural networks," in *2013 IEEE International Conference on Mechatronics (ICM)*, Feb 2013, pp. 99–104.

[126] A. D. Alexandrou, N. K. Adamopoulos, and A. G. Kladas, "Development of a constant switching frequency deadbeat predictive control technique for field-oriented synchronous permanent-magnet motor drive," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 8, pp. 5167–5175, Aug 2016.

[127] L. Ciabattoni, G. Cimini, F. Ferracuti, A. Freddi, G. Ippoliti, and A. Monteriú, "A novel LDA-based approach for motor bearing fault detection," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, July 2015, pp. 771–776.

[128] G. Cimini, M. L. Corradini, G. Ippoliti, G. Orlando, and M. Pirro, "A Rapid Prototyping Scenario for Power Factor Control in Permanent Magnet Synchronous Motor Drives: Control Solutions for Interleaved Boost Converters," *Electric Power Components and Systems*, vol. 42, no. 6, pp. 639–649, 2014.

[129] L. Ciabattoni, G. Cimini, F. Ferracuti, M. Grisostomi, G. Ippoliti, and M. Pirro, "Bayes error based feature selection: An electric motors fault detection case study," in *Industrial Electronics Society, IECON 2015 - 41st Annual Conference of the IEEE*, Nov 2015, pp. 003 893–003 898.

[130] G. Cimini, M. L. Corradini, G. Ippoliti, N. Malerba, and G. Orlando, "Control of variable speed wind energy conversion systems by a discrete-time sliding mode approach," in *Mechatronics (ICM), 2013 IEEE International Conference on*, Feb 2013, pp. 736–741.

[131] P. Walde and C. Brunner, IEA (International Energy Agency), "Energy-efficiency policy opportunities for electric motor-driven systems."

[132] R. P. Aguilera, P. Lezana, and D. E. Quevedo, "Switched model predictive control for improved transient and steady-state perfor-

mance," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 4, pp. 968–977, Aug 2015.

[133] C. S. Lim, E. Levi, M. Jones, N. A. Rahim, and W. P. Hew, "A comparative study of synchronous current control schemes based on fcs-mpc and pi-pwm for a two-motor three-phase drive," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 3867–3878, Aug 2014.

[134] G. Cimini, V. Fossi, G. Ippoliti, S. Mencarelli, G. Orlando, and M. Pirro, "Model predictive control solution for Permanent Magnet Synchronous Motors," in *Industrial Electronics Society, IECON 2013 - 39th Annual Conference of the IEEE*, Nov 2013, pp. 5824–5829.

[135] V. I. Utkin, J. Guldner, and J. Shi, *Sliding mode control in electromechanical systems*, ser. Systems and Control. Florida, USA: CRC Press LLC, 1999.

[136] M. L. Corradini, G. Ippoliti, S. Longhi, and G. Orlando, "A quasi-sliding mode approach for robust control and speed estimation of pm synchronous motors," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 2, pp. 1096–1104, Feb 2012.

[137] G. Cimini, G. Ippoliti, G. Orlando, and M. Pirro, "PMSM control with power factor correction: Rapid prototyping scenario," in *Power Engineering, Energy and Electrical Drives (POWERENG), 2013 Fourth International Conference on*, May 2013, pp. 688–693.

[138] M. Preindl and S. Bolognani, "Comparison of direct and PWM model predictive control for power electronic and drive systems," in *2013 Twenty-Eighth Annual IEEE Applied Power Electronics Conference and Exposition (APEC)*, Mar. 2013, pp. 2526–2533.

[139] A. Damiano, G. Gatto, I. Marongiu, A. Perfetto, and A. Serpi, "Operating constraints management of a surface-mounted PM synchronous machine by means of an FPGA-based model predictive control algorithm," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 243–255, Feb. 2014.

[140] U. Maeder, F. Borrelli, and M. Morari, "Linear offset-free model predictive control," *Automatica*, vol. 45, no. 10, pp. 2214 – 2222, 2009.

[141] C. Bordons and C. Montero, "Basic principles of mpc for power converters: Bridging the gap between theory and practice," *IEEE Industrial Electronics Magazine*, vol. 9, no. 3, pp. 31–43, Sept 2015.

[142] M. P. Akter, S. Mekhilef, N. M. L. Tan, and H. Akagi, "Modified Model Predictive Control of a Bidirectional AC-DC Converter Based on Lyapunov Function for Energy Storage Systems," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 704–715, Feb 2016.

[143] J. Scoltock, T. Geyer, and U. K. Madawala, "Model predictive direct power control for grid-connected npc converters," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 9, pp. 5319–5328, Sept 2015.

[144] G. Cimini, "Sensorless power factor control for mixed conduction mode boost converter using passivity-based control," *IET Power Electronics*, vol. 7, pp. 2988–2995(7), December 2014.

[145] G. Cimini, A. Freddi, G. Ippoliti, A. Monteriù, and M. Pirro, "A Smart Lighting System for Visual Comfort and Energy Savings in Industrial and Domestic Use," *Electric Power Components and Systems*, vol. 43, no. 15, pp. 1696–1706, 2015.

[146] G. Cimini, G. Ippoliti, G. Orlando, and M. Pirro, "Current sensorless solutions for PFC of boost converters with passivity-based and sliding mode control," in *Power Engineering, Energy and Electrical Drives (POWERENG), 2013 Fourth International Conference on*, May 2013, pp. 1175–1180.

[147] Z. Zhang, F. Wang, T. Sun, J. Rodríguez, and R. Kennel, "Fpga-based experimental investigation of a quasi-centralized model predictive control for back-to-back converters," *IEEE Transactions on Power Electronics*, vol. 31, no. 1, pp. 662–674, Jan 2016.

[148] G. Cimini, G. Ippoliti, G. Orlando, and M. Pirro, "Explicit sensorless model predictive control of synchronous buck converter," in *Renewable Energy Research and Applications (ICRERA), 2013 International Conference on*, Oct 2013, pp. 1200–1205.

[149] G. Cimini, M. L. Corradini, G. Ippoliti, G. Orlando, and M. Pirro, "Passivity-Based PFC for Interleaved Boost Converter of PMSM Drives," in *11th International Workshop on Adaptation and Learning in Control and Signal Processing*, 2013, pp. 128–133.

[150] E. Alidori, G. Cimini, G. Ippoliti, G. Orlando, and M. Pirro, "A passivity-based solution for CCM-DCM boost converter Power Factor Control," in *39th Annual Conference of the IEEE Industrial Electronics Society, IECON*, Nov 2013, pp. 7752–7757.

[151] L. Cavanini, G. Cimini, G. Ippoliti, and A. Bemporad, "Model predictive control for pre-compensated voltage mode controlled DC-DC converters," *IET Control Theory & Applications*, p. *Accepted for Publication*, 2016.

[152] F. Kurokawa, J. Sakemi, A. Yamanishi, and H. Osuga, "A new quick transient response digital control dc-dc converter with smart bias function," in *2011 IEEE 33rd International Telecommunications Energy Conference (INTELEC)*, Oct 2011, pp. 1–7.

[153] F. Kurokawa, A. Yamanishi, and S. Hirotaki, "A Reference Modification Model Digitally Controlled DC-DC Converter for Improvement of Transient Response," *IEEE Transactions on Power Electronics*, vol. 31, no. 1, pp. 871–883, Jan 2016.

[154] K. Kogiso and K. Hirata, "Reference governor for constrained systems with time-varying references," *Robotics and Autonomous Systems*, vol. 57, no. 3, pp. 289 – 295, 2009, selected papers from 2006 {IEEE} International Conference on Multisensor Fusion and Integration (MFI 2006)2006 {IEEE} International Conference on Multisensor Fusion and Integration.

[155] I. Kolmanovsky, E. Garone, and S. D. Cairano, "Reference and command governors: A tutorial on their theory and automotive applications," in *2014 American Control Conference*, June 2014, pp. 226–241.

[156] Y. Yan, F. Lee, and P. Mattavelli, "Analysis and Design of Average Current Mode Control Using a Describing-Function-Based Equivalent Circuit Model," *IEEE Transactions on Power Electronics*, vol. 28, no. 10, pp. 4732–4741, Oct 2013.

[157] G. Zhou, J. Xu, and J. Wang, "Constant-Frequency Peak-Ripple-Based Control of Buck Converter in CCM: Review, Unification, and Duality," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 3, pp. 1280–1291, March 2014.

[158] M. Khazraei and M. Ferdowsi, "Modeling and analysis of projected cross point control- a new current-mode-control approach," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 8, pp. 3272–3282, Aug 2013.

[159] S.-K. Kim, C. R. Park, J.-S. Kim, and Y. I. Lee, "A stabilizing model predictive controller for voltage regulation of a DC/DC boost converter," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 5, pp. 2016–2023, Sept 2014.

[160] G. Cimini, G. Ippoli, G. Orlando, and M. Pirro, "Sensorless passivity-based control for Mixed Conduction Mode boost converter with power factor correction," in *Renewable Energy Research and Applications (ICRERA), 2013 International Conference on*, Oct 2013, pp. 1194–1199.

[161] G. Cimini, G. Ippoliti, G. Orlando, and M. Pirro, "Current sensorless solution for PFC boost converter operating both in DCM and CCM," in *Control Automation (MED), 2013 21st Mediterranean Conference on*, June 2013, pp. 137–142.

[162] Y. Qiu, H. Liu, and X. Chen, "Digital Average Current-Mode Control of PWM DC-DC Converters Without Current Sensors," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 5, pp. 1670–1677, May 2010.

[163] J. Abu Qahouq and V. Arikatla, "Power converter with digital sensorless adaptive voltage positioning control scheme," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 9, pp. 4105–4116, Sept 2011.

[164] H. Sira-Ramirez, R. Marquez-Contreras, and M. Fliess, "Sliding mode control of dc-to-dc power converters using integral reconstructors," *International Journal of Robust and Nonlinear Control*, vol. 12, no. 13, pp. 1173–1186, 2002.

[165] R. Erickson and D. Maksimovic, *Fundamentals of Power Electronics*. Springer, 2001.

[166] S. Saggini, D. Trevisan, P. Mattavelli, and M. Ghioni, "Synchronous-Asynchronous Digital Voltage-Mode Control for DC-DC Converters," *IEEE Transactions on Power Electronics*, vol. 22, no. 4, pp. 1261–1268, July 2007.

[167] "Synchronous Rectification in High-Performance Power Converter Design." [Online]. Available: http://www.ti.com/lit/an/snva595/snva595.pdf

[168] "Efficiency of synchronous versus nonsynchronous buck converters." [Online]. Available: http://www.ti.com/lit/an/slyt358/slyt358.pdf

[169] G. Cimini, G. Ippoliti, S. Longhi, G. Orlando, and M. Pirro, "Synchronous buck converter control via robust periodic pole assignment," in *Control Conference (ECC), 2014 European*, June 2014, pp. 1921–1926.

[170] P. Pardalos and V. Yatsenko, *Optimization and Control of Bilinear Systems: Theory, Algorithms, and Applications*, ser. Springer Optimization and Its Applications. Springer US, 2010.

[171] G. Cimini, M. L. Corradini, G. Ippoliti, G. Orlando, and M. Pirro, "A sliding mode observer for the load resistance estimation in a boost converter," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, July 2015, pp. 1556–1560.

[172] J. Sha, J. Xu, S. Zhong, S. Liu, and L. Xu, "Control Pulse Combination-Based Analysis of Pulse Train Controlled DCM Switching DC-DC Converters," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 1, pp. 246–255, Jan 2015.

[173] R. Ridley, "A new, continuous-time model for current-mode control power convertors," *IEEE Transactions on Power Electronics*, vol. 6, no. 2, pp. 271–280, Apr 1991.

[174] G. Calisse, G. Cimini, L. Colombo, A. Freddi, G. Ippoliti, A. Monteriù, and M. Pirro, "Development of a smart LED lighting system: Rapid prototyping scenario," in *Systems, Signals Devices (SSD), 2014 11th International Multi-Conference on*, Feb 2014, pp. 1–6.