Variational and Time-Distributed Methods for Real-time Model Predictive Control

by

Dominic M. Liao-McPherson

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in the University of Michigan
2020

Doctoral Committee:

Professor Ilya Kolmanovsky, Chair
Dr. Ken Butts, Toyota Motor North America
Assistant Professor Alex Gorodetsky
Professor Jing Sun

Dominic M. Liao-McPherson

dliaomcp@umich.edu

ORCID iD: 0000-0002-5675-5635

To my family and friends.

Acknowledgments

During my time at the University of Michigan I've had the opportunity to spend time with with some exceptional people. Unfortunately I only have space to thank a fraction of them here, but I'll try my best.

First and foremost, I'd like to thank my advisor, Professor Ilya Kolmanovsky, for his guidance, patience, and support. Thank you for being the best mentor and role model one could hope for.

To my friends, labmates, and colleagues. Thank you Will, Greg, and Chris for helping me get through the first year (and the ones that came after as well). Thank you AJ for wanting to talk about things other than work. Marco, thank you for teaching me to actually enjoy systems theory. Vyas, Andrew, Joey, and Jacqueline, thank you for helping to make AA home. Efe and Anil, thank you for listening, helping me blow off steam, and for all the good times. Thank you Richard, Jordan, Torbjørn, Nan, Huayi, Kyoungseok, Brad, Tam and Will for making FXB a fun place to be and thank you Mike, Tony, and Kevin at the TTC for the same reason. Doruk, Tyler, Alex, Andrew, Jin, Kevin, Jeremy, Anil, Yeliz, Ahmet, Yunis, Mike, Eric, Sam, Pierre, Ricardo, James, Patrick, Bryan, Kelly and Matt, thank you for your friendship and support over the years.

Next, to my committee members and mentors. Professors Sun and Gorodetsky, thank you for taking the time to review my work and serve on my committee. Ken, thank you both for serving on my committee and for your mentorship throughout my time working with/at Toyota. Professor Anouck Girard and Dr. Asen Dontchev thank you for your insight on everything from teaching classes to service to variational analysis.

Finally, to those closest to me. Maya, thank you for being there, I don't know where I'd be without you. To my siblings Gabby and Alex, thank you for keeping me grounded. To Grandma, thank you for everything over the years. Finally, to my parents, Dr's Mary Liao and Harry McPherson, thank you for your love, support, and confidence in me from the very beginning.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

NOx Oxides of Nitrogen

PM Particulate Matter

MPC Model Predictive Control

FBRS Fischer-Burmeister Regularized and Smoothed

TDO Time-distributed Optimization

SMPC Supervisory Model Predictive Control

OCP Optimal Control Problem

WLTC Worldwide Harmonized Light Vehicles Test Cycle

GE Generalized Equation

VI Variational Inequality

KKT Karush-Kuhn-Tucker

MIMO Multiple Input Multiple Output

DAP Diesel Engine Airpath

CQ Constraint Qualification

LICQ Linear Independence Constraint Qualification

SSOSC Strong Second Order Sufficient Condition

VGT Variable Geometry Turbocharger

EGR Exhaust Gas Recirculation

MAF Mass Air Flow

MAP Manifold Air Pressure

THC Total Hydrocarbon Concentration

NEDC New European Drive Cycle

Abstract

This dissertation concerns the theoretical, algorithmic, and practical aspects of solving optimal control problems (OCPs) in real-time. The topic is motivated by Model Predictive Control (MPC), a powerful control technique for constrained, nonlinear systems that computes control actions by solving a parameterized OCP at each sampling instant. To successfully implement MPC, these parameterized OCPs need to be solved in real-time. This is a significant challenge for systems with fast dynamics and/or limited onboard computing power and is often the largest barrier to the deployment of MPC controllers. The contributions of this dissertation are as follows.

First, I present a system theoretic analysis of Time-distributed Optimization (TDO) in Model Predictive Control. When implemented using TDO, an MPC controller distributed optimization iterates over time by maintaining a running solution estimate for the optimal control problem and updating it at each sampling instant. The resulting controller can be viewed as a dynamic compensator which is placed in closed-loop with the plant. The resulting coupled plant-optimizer system is analyzed using input-to-state stability concepts and sufficient conditions for stability and constraint satisfaction are derived. When applied to time distributed sequential quadratic programming, the framework significantly extends the existing theoretical analysis for the real-time iteration scheme. Numerical simulations are presented that demonstrate the effectiveness of the scheme.

Second, I present the Proximally Stabilized Fischer-Burmeister (FBstab) algorithm for convex quadratic programming. FBstab is a novel algorithm that synergistically combines the proximal point algorithm with a primal-dual semismooth Newton-type method. FB-

stab is numerically robust, easy to warmstart, handles degenerate primal-dual solutions, detects infeasibility/unboundedness and requires only that the Hessian matrix be positive semidefinite. The chapter outlines the algorithm, provides convergence and convergence rate proofs, and reports some numerical results from model predictive control benchmarks and from the Maros-Meszaros test set. Overall, FBstab shown to be is competitive with state of the art methods and to be especially promising for model predictive control and other parameterized problems.

Finally, I present an experimental application of some of the approaches from the first two chapters: Emissions oriented supervisory model predictive control (SMPC) of a diesel engine. The control objective is to reduce engine-out cumulative $\mathrm{NO_x}$ and total hydrocarbon (THC) emissions. This is accomplished using an MPC controller which minimizes deviation from optimal setpoints, subject to combustion quality constraints, by coordinating the fuel input and the EGR rate target provided to an inner-loop airpath controller. The SMPC controller is implemented using TDO and a variant of FBstab which allows us to achieve sub-millisecond controller execution times. We experimentally demonstrate $10-15\%$ cumulative emissions reductions over the Worldwide Harmonized Light Vehicles Test Cycle (WLTC) drivecycle.

CHAPTER 1

Introduction

All systems are constrained. For example, aircraft are subject to angle-of-attack constraints to prevent stall, chemical reactors must limit temperatures and pressures to ensure safe operation, autonomous vehicles must avoid obstacles, and electric motors have torque and power limits. In the past, it was often sufficient to handle constraints and nonlinearities through conservative control design and linearization. However, as control systems are become more complex, and as systems are pushed closer and closer to their limits in search of e.g., faster performance, lower emissions, higher efficiency, lower cost and smaller sizes, addressing constraints and nonlinearities "head on" is becoming increasingly necessary.

At the same time, many engineered systems are being developed to operate with increasing levels of autonomy. For instance, aerial drones are being used for everything from package delivery to fire fighting to surveillance, almost all major car manufacturers have active autonomous vehicle programs, and collaborative robots are being developed for applications in eldercare, warehouse logistics, manufacturing, and surgery. Successfully deploying these autonomous systems will require advances in sensing, perception, decision making, human robot interaction, learning, and control. In particular, they will need to be able to perform highly dynamic maneuvers with little to no human intervention while accounting for nonlinearities, disturbances, and constraints.

Online optimization is a powerful paradigm for approaching these challenges. It can enable engineered systems to consider complex performance metrics, constraints, and nonlinearities all while incorporating feedback to improve robustness. However, solving optimization problems reliably, in real-time, using limited onboard computing resources is a difficult undertaking. This dissertation presents recent developments in real-time optimization of dynamical systems. Specifically, it focuses on reducing the computational burden of Model Predictive Control (MPC), an optimization based control methodology, to help enable its application to systems with pronounced nonlinear dynamics, fast sampling rates, and limited available onboard computing power.

1.1 Model Predictive Control

MPC [1–3] is a control methodology wherein a feedback law is defined implicitly as the solution of a finite horizon Optimal Control Problem (OCP). In this thesis, we focus on deterministic, discrete time, but often nonlinear, systems.

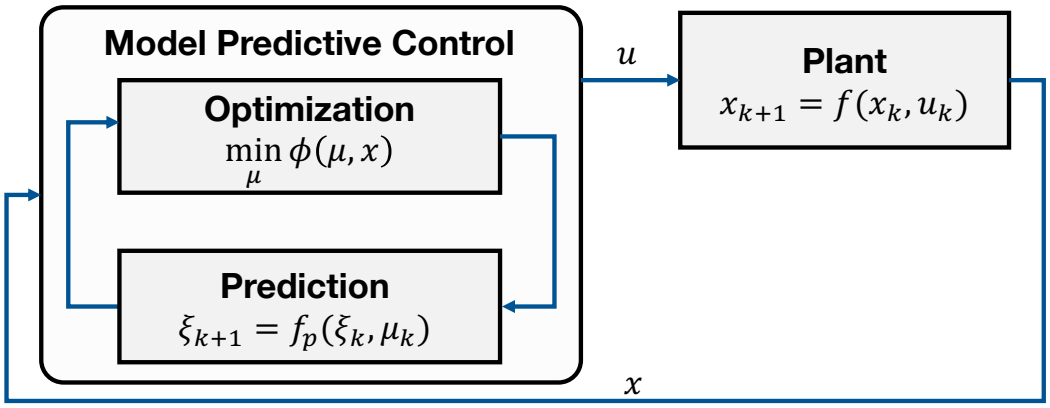

Figure 1.1: A model predictive controller in closed-loop with a plant.

The essential idea behind MPC is as follows. First, a model of the system is derived, typically of the form

$$x_{k+1} = f_p(x_k, u_k), \tag{1.1}$$

where $k \in \mathbb{Z}_+$ denotes the discrete time instant, $x \in \mathcal{X} \subseteq \mathbb{R}^n$ is the system state, $u \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input and $f_p : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the prediction model[1]. The state and control inputs are subject to the constraints $(x, u) \in \mathcal{Z} \subseteq \mathcal{X} \times \mathcal{U}$. Next, an OCP over a prediction horizon if formulated, a common form is

$$\min_{\xi,\mu} \ \phi(\xi, \mu) = V_f(\xi_N) + \sum_{i=0}^{N-1} l(\xi_i, \mu_i), \tag{1.2a}$$

$$\text{s.t.} \ \ \xi_{i+1} = f_p(\xi_i, \mu_i, 0), \quad i \in \mathbb{Z}_{[0,N-1]}, \tag{1.2b}$$

$$\xi_0 = x, \tag{1.2c}$$

$$(\xi_i, \mu_i) \in \mathcal{Z}, \qquad i \in \mathbb{Z}_{[0,N-1]}, \tag{1.2d}$$

$$\xi_N \in \mathcal{X}_f. \tag{1.2e}$$

where $N \in \mathbb{Z}_{++}$ is the horizon length, $\mathcal{X}_f \subseteq \mathcal{X}$ is the terminal state constraint which is typically used to guarantee stability, $\xi = (\xi_0, \xi_1, \ldots, \xi_N)$ is the state sequence, and $\mu =$

[1] See Chapter 2 for notation.

$(\mu_0, \mu_1, \ldots, \mu_{N-1})$ is the control sequence. Note that the OCP (1.2) is parameterized by the measured system state x. This parameterization leads to an implicitly defined state feedback law which is placed in closed-loop with the plant as illustrated in Figure 1.1.

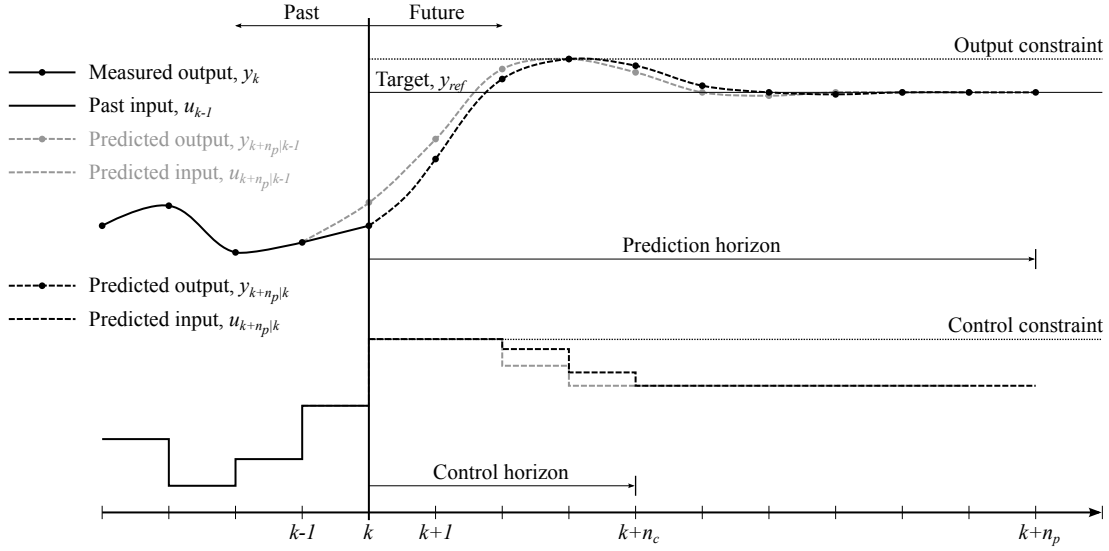

Figure 1.2: Illustration of the MPC process at time k. An optimization algorithm is used to pick a sequence of control actions over the N step prediction horizon to minimize the predicted cost. The first input of the sequence is applied to the plant and the process is repeated at time $k+1$.

Then, online, the control action at sampling instant k is computed by solving (1.2), which yields a solution $(\xi^*(x_k), \mu^*(x_k))$, where x_k is the measured/estimated state at time k, and $\mu_0^*(x_k)$ is applied to the system. The process is then repeated at the next timestep using the new measurement x_{k+1}, the incorporation of the new measurement introduces feedback. The process is illustrated graphically in Figure 1.2.

MPC is a general and powerful methodology that is able to handle input and output constraints, nonlinearities, and highly coupled Multiple Input Multiple Output (MIMO) systems. In practice, it is often the only option for achieving high-performance control in a systematic way, especially for nonlinear and/or constrained systems. This makes it attractive for industrial practitioners, who are continually under pressure to make their systems more efficient, lighter, cheaper, safer, and so on. Indeed, a recent survey by the International Federation of Automatic Control (IFAC) [4] concluded that MPC is second only to proportional-integral-derivative (PID) control in industrial adoption. Moreover, the MPC is a key enabling technology for autonomy, allowing systems to generate trajectories that satisfy constraints and optimize specified performance metrics "on the fly".

In addition to its practical applicability, MPC is supported by an extensive theoretical

literature. The control community has extensively studied the stability properties of MPC see e.g., [5–7] and the references therein. Moreover, other critical qualitative properties such as robustness [8–11], integral action [12, 13], and suboptimality [14–16] have also been extensively studied. There are a variety of textbooks and monographs available that provide a good introduction to MPC, see e.g., [1–3, 17, 18].

Despite its many virtues, MPC has two major drawbacks:

1. MPC requires a model of the system for use in prediction, and the performance of the closed-loop system depends on the quality of the model. Moreover, such models need to be simple enough to be readily incorporated into an optimization routine if they are to be used in practice.

2. Computing control actions means solving a (possibly non-convex) OCP at every sampling instance.

The difficulty of obtaining prediction models depends entirely on the application. For example, relatively high quality physics based models are available in many spacecraft and vehicle dynamics applications whereas the diesel engine controller presented in Chapter 5 extensively uses identified models. Other applications, such as very flexible aircraft or humanoid robots, require complex models such as partial differential equations (or nonlinear finite element approximations of them) or hybrid dynamics, which greatly complicates the application of MPC to these systems.

The second challenge is often the greatest hurdle to practical deployment. In some situations, it is possible to presolve the optimization problem offline using multiparametric programming, this technique is commonly known as Explicit MPC [19–23]. However, such approaches are not applicable to all types of MPC formulations and tend to scale badly with the dimension of the state space and the number of constraints. Thus, it is usually necessary to solve the OCP online using an embedded optimization routine. This dissertation focuses on accelerating these embedded optimizers, specifically it explores algorithmic ideas[2] for reducing the computational cost associated with solving the OCP online.

1.2 Contributions and Outline

The objective of this thesis is to help enable the implementation of MPC for nonlinear, constrained systems with pronounced nonlinear dynamics, fast sampling rates, and limited

[2]Other ways to reduce the computational footprint of MPC, such as bespoke hardware [24, 25], symbolic optimization [26], and code optimization [27] etc., are important but outside the scope of this thesis.

onboard computing power. The key idea is that implementing MPC leads to a *sequence* of related OCPs (rather than isolated optimization problems) and that a (possibly approximate) solution from a previous sampling instant can be passed to the real-time optimization algorithm to reduce the computational burden at the next. This practice is widespread and is commonly known as "warmstarting" (or sometimes "hotstarting") the optimization algorithm. The idea that this technique can be used to distribution computational load over time is an important part of this dissertation.

This dissertation is composed of three main chapters (Chapters 3 - 5). Each chapter is based on a manuscript which has been submitted to or accepted by a peer reviewed scientific journal. Chapter 3 explores the systems theoretic consequences of warmstarting, Chapter 4 presents a novel optimization algorithm designed with warmstarting in mind, and Chapter 5 demonstrates how the material from the first two chapters can be applied to make an impact in the real-world through the application of MPC to diesel engine emissions reduction. More specifically, the contributions of Chapters 3 - 5 are as follows[3]:

1. Chapter 3 is based on the material in [28] and contains an analysis of Time-distributed Optimization (TDO) in MPC. In TDO, a running solution estimate is incrementally improved at each iteration using a warmstarted optimization algorithm. This allows us to distribute optimization iterations over time, greatly reducing the overall computational footprint of the controller. However it also introduces the dynamics of the optimizer into the closed-loop system, greatly complicating analysis. In Chapter 3 we present a framework for studying TDO from a systems theoretic perspective. In the chapter, we analyze TDO applied to a broad class of MPC formulations and optimization algorithms and show that, under appropriate assumptions, we can recover the stability, robustness, and constraint satisfaction of the optimal MPC feedback law using a *finite* amount of computational resources. In particular, the analysis significantly extends existing literature on the topic by considering inequality constraints, explicitly studying robustness properties, and analyzing how the availability of computational resources affects closed-loop performance.

2. Chapter 4 is based on [29] and details the development of a novel algorithm for solving quadratic programs (QPs). The proximally stabilized Fischer-Burmeister method (FBstab) is a stabilized semismooth Newton-type method that can exploit sparsity and be easily warmstarted. It can also detect primal and/or dual infeasibility over the course of its iterations. In the chapter we outline the method, perform a convergence analysis, and present extensive numerical benchmarking results.

[3] A detailed literature review and introduction section is provided in each chapter.

3. Chapter 5 is based on [30] and details the design and experimental validation of a supervisory emissions oriented model predictive controller for a Diesel Engine Airpath (DAP). The DAP is a highly nonlinear, constrained MIMO system; the control objective is to reduce Oxides of Nitrogen (NOx) and Particulate Matter (PM) emissions while limiting smoke and maintaining drivability. We show that, using a inner/outer loop architecture that includes both a supervisory and nonlinear MPC controllers, it is possible to reduce emissions compared to the mass production controller and to overcome the associated computational challenges using TDO and a version of FBstab. Chapter 5 focuses on the Supervisory Model Predictive Control (SMPC) module which was primarily my work. The NMPC module is an extension of Mike Huang's dissertation research [31] and was primarily developed by Mike Huang and Shinhoon Kim at Toyota Motor North America Research & Development.

This dissertations draws heavily from a variety of mathematical disciplines, including systems theory, optimization, and numerical analysis. In particular, it makes extensive use of concepts and results from set-valued variational analysis [32]. In Chapter 3 we study the feedback interconnection of an a parameterized optimization algorithm and the physical plant arising in time-distributed optimization. Generalized equations and their associated regularity conditions and implicit function theorems [33, 34], are invaluable tools for analyzing the parameterized optimization problems that arise in MPC and allow us to characterize the behaviour of solutions as the parameter varies, i.e., as the state of the physical system evolves. Moreover, the solver in Chapter 4, FBstab, is based around two variational analysis concepts, the proximal point method for monotone inclusions and Newton's method for non-smooth equations [35] using set-valued generalized Jacobians [36]. Synergistically combining these two methods leads to a method that is effective on sparse and parameterized problems, e.g., those arising from time-distributed optimization in MPC.

CHAPTER 2

Mathematical Preliminaries

This dissertation draws from several branches of mathematics, notably dynamical systems theory, mathematical optimization, real analysis, linear algebra, and variational analysis. This chapter summarizes some concepts and notations that will be useful in subsequent chapters.

2.1 Notation

We denote by $\mathbb{Z}_{+(+)}$ the non-negative (positive) integers and by $\mathbb{R}_{+(+)}$ the non-negative (positive) reals. We use I_n to denote the identity matrix of size $n \times n$, the subscript will be omitted if the dimension is clear from context. We use $\mathrm{id} : \mathbb{R} \to \mathbb{R}$ to denote the identity function. For a vector, $||\cdot||$ denotes the Euclidean or 2-norm. If A is a matrix then A_i is its ith row and $||A||$ is understood to be the induced 2-norm. If $\mathcal{I}$ is an index set, $|\mathcal{I}|$ is its cardinality and $A_{\mathcal{I}}$ denotes the row wise concatenation of $A_i,\ \forall i \in \mathcal{I}$. For two vectors, (a, b) denotes vertical concatenation. We denote the unit ball centered at x by $\mathcal{B}(x)$, it is understood that $\mathcal{B} = \mathcal{B}(0)$. If X is a closed neighbourhood of the origin, we denote its radius by $\mathrm{rad}\ \mathrm{X}$, i.e., the largest $r > 0$ such that $\{x \mid ||x|| \leq r\} \subseteq X$. Set addition/subtraction is defined as

$$A \pm B = \{y \mid y = a \pm b,\ a \in A, b \in B\},$$

we use $\setminus$ to denote the set difference operation.

For a point $x^* \in \mathbb{R}^n$, a function $F : \mathbb{R}^n \to \mathbb{R}^m$ and a function $\phi : \mathbb{R}^n \to \mathbb{R}_+$ we say that $F(x) = \mathcal{O}(\phi(x))$ as $x \to x^*$ if there exists $c > 0$ such that $||F(x)|| \leq c\phi(x)$ for all x sufficiently close to x^* and that $F(x) = o(\phi(x))$ if for every $\varepsilon > 0$, $||F(x)|| \leq \varepsilon\phi(x)$ for all x sufficiently close to x^*. Analogously, for sequences $\{x_k\} \subset \mathbb{R}^n$ and $\{r_k\} \subset \mathbb{R}_+$ we

say that $x_k = \mathcal{O}(r_k)$ as $k \to \infty$ if $\exists c > 0$ such that $||x_k|| \leq cr_k$ for all k sufficiently large and $x_k = o(r_k)$ if $||x_k|| \leq \varepsilon r_k$ for all $\varepsilon > 0$ for all k sufficiently large.

We say a sequence $\{x_k\} \subset \mathbb{R}^n$ converges if there exists some $x^* \in \mathbb{R}^n$ such that $\lim_{k\to\infty} x_k = x^*$. The sequence converges q-linearly if there exists $\eta \in (0, 1)$ such that

$$\lim_{k\to\infty} \frac{||x_{k+1} - x^*||}{||x_k - x^*||} = \eta, \tag{2.1}$$

the sequence is said to converge q-superlinerly if

$$||x_{k+1} - x^*|| = o(||x_k - x^*||) \quad \text{or} \quad ||x_{k+1} - x^*|| = \mathcal{O}(||x_k - x^*||^q), \tag{2.2}$$

for some $q \in (1, 2)$ and q-quadratically if

$$||x_{k+1} - x^*|| = \mathcal{O}(||x_k - x^*||^2). \tag{2.3}$$

2.2 Dynamical Systems

Consider a discrete time dynamical system

$$x_{k+1} = g(x_k, u_k), \tag{2.4}$$

governed by a Lipschitz continuous vector field $g : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. Given an initial state $x_0 \in \mathbb{R}^n$, and an input sequence $\mathbf{u} : \mathbb{Z}_+ \to \mathbb{R}^m$ we denote its solution by $x(k, x_0, \mathbf{u})$. For $\mathbf{u} : \mathbb{Z}_+ \to \mathbb{R}^m$ we let $||\mathbf{u}|| = \sup\{||u_k|| : k \in \mathbb{Z}_+\}$. We use $\overline{\lim}$ as shorthand for $\limsup$.

Recall that a function $\gamma : \mathbb{R}_+ \to \mathbb{R}_+$ is said to be of class $\mathcal{K}$ if it is continuous, strictly increasing and $\gamma(0) = 0$. If it is also unbounded, then $\gamma \in \mathcal{K}_\infty$. A function $\beta : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ is said to be of class $\mathcal{KL}$ if $\beta(\cdot, s) \in \mathcal{K}$ for each fixed $s \geq 0$ and $\beta(r, s) \to 0$ as $s \to \infty$ for fixed $r \geq 0$. If $\gamma_1, \gamma_2 : \mathbb{R} \to \mathbb{R}$, we denote their composition by $\gamma_1 \circ \gamma_2$. We will also make use of the following robust stability and invariance concepts.

Definition 2.1 (Local Input-to-state Stability [37])**.** *A system* (2.4) *is said to be Locally Input-to-State Stable (LISS) if there exists* $\epsilon > 0$*,* $\beta \in \mathcal{KL}$*, and* $\gamma \in \mathcal{K}$ *such that,* $\forall k \in \mathbb{Z}_+$*,*

$$||x(k, x_0, \mathbf{u})|| \leq \max\{\beta(||x_0||, k), \gamma(||\mathbf{u}||)\}, \tag{2.5}$$

provided $||x_0|| \leq \epsilon$ *and* $||\mathbf{u}|| \leq \epsilon$*.*

Definition 2.2 (Asymptotic gain [38])**.** *Consider system* (2.4)*, we say that it has an asymp-*

totic gain if there exists some $\gamma \in \mathcal{K}$ such that

$$\overline{\lim_{k\to\infty}} ||x(k, x_0, \mathbf{u})|| \leq \gamma \left(\overline{\lim_{k\to\infty}} ||u_k|| \right), \tag{2.6}$$

for all $x_0 \in \mathbb{R}^{n_x}$.

Definition 2.3 (RPI set)**.** *Consider the system $x_{k+1} = h(x_k, u_k, d_k)$ which is subject to the constraint $(x, u) \in \mathcal{Z}$, where $h : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l \to \mathbb{R}^n$, and $d \in \mathcal{D} \subseteq \mathbb{R}^l$ is a disturbance. Suppose the system is being controlled by the feedback law $\kappa : \mathbb{R}^n \to \mathbb{R}^m$ so that*

$$x_{k+1} = h(x_k, \kappa(x_k), d_k). \tag{2.7}$$

A set $\Omega \subseteq \mathbb{R}^n$ is a Robust Positively Invariant (RPI) set for system (2.7) if $h(x, \kappa(x), d) \in \Omega$ for all $x \in \Omega$, $d \in \mathcal{D}$. In addition, if $\Omega \subseteq \{x \mid (x, \kappa(x)) \in \mathcal{Z}\}$, then Ω is called an admissible RPI set.

2.3 Variational Analysis

We use the notation $\Phi : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ to denote set-valued mappings[1], which are mathematical objects which associate each point in their domains with a set. A set-valued mapping can be associated with its graph

$$\operatorname{gph} \Phi = \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^m \mid y \in \Phi(x)\}, \tag{2.8}$$

we denote is domain by $\operatorname{dom} \Phi = \{x \in \mathbb{R}^n \mid \Phi(x) \neq \emptyset\}$ and its range by $\operatorname{rng} F = \{y \in \mathbb{R}^m \mid y \in \Phi(x) \text{ for any } x \in \mathbb{R}^n\}$. A set-valued mapping always has a well defined inverse which is given by $\Phi^{-1}(y) = \{x \mid y \in \Phi(x)\}$ and is in general set-valued. If a set-valued mapping assigns a single element at a point x then F is (locally) a function and we write $\Phi(x) = y$ which is equivalent to $\Phi(x) = \{y\}$.

In this dissertation, we make extensive use of one mapping in particular which is the normal cone mapping of a closed, convex set C defined as

$$\mathcal{N}_C(v) = \begin{cases} \{y \mid y^T(w - v) \leq 0, \forall w \in C\}, & \text{if } v \in C, \\ \emptyset & \text{else.} \end{cases} \tag{2.9}$$

This mapping is trivial in the interior of C and can be roughly conceptualized as the set of

[1] Also known as a point-to-set mapping, correspondence, relation, or multifunction

all "outwards" pointing vectors on the boundary of C. Some normal cones are illustrated in Figure 2.1.

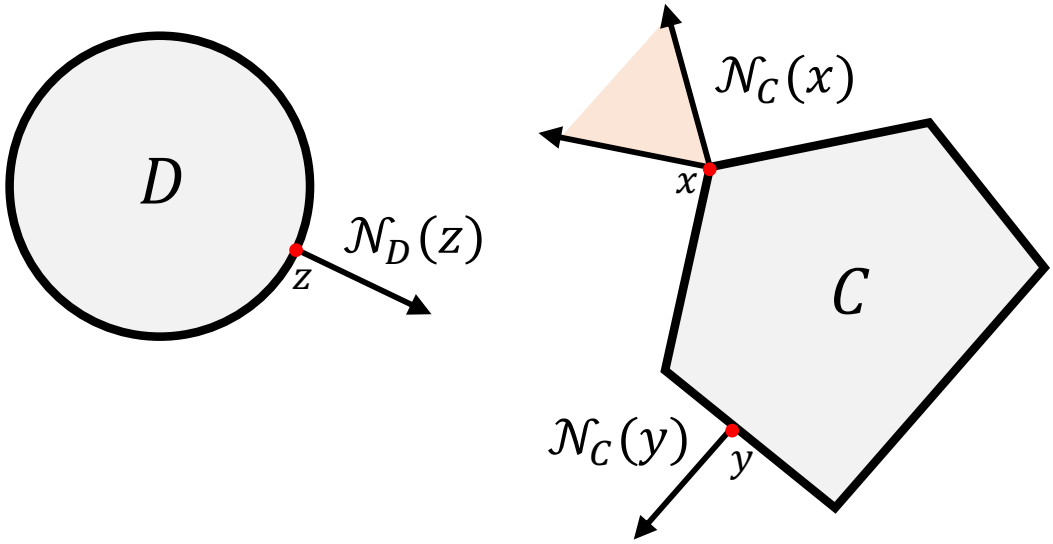

Figure 2.1: Normal cone mappings for several convex sets.

The normal cone mapping is related to the indicator function of C,

$$\mathcal{I}_C(v) = \begin{cases} 0 & \text{if } v \in C \\ \infty & \text{else,} \end{cases} \tag{2.10}$$

through the convex subdifferential operator[2], i.e., $\mathcal{N}_C = \partial \mathcal{I}_C$. We also make extensive use of

$$\mathcal{N}_+(v) = \begin{cases} \{y \mid y^T(w - v) \leq 0, \forall w \geq 0\}, & \text{if } v \geq 0, \\ \emptyset & \text{else,} \end{cases} \tag{2.11}$$

the normal cone mapping of the non-negative orthant. See e.g., [33] or [32] for more information on set-valued mappings.

2.3.1 Generalized Equations

Given $F : \mathbb{R}^n \to \mathbb{R}^m$ and $\Phi : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$, a Generalized Equation (GE) is a structured inclusion of the form

$$F(x) + \Phi(x) \ni 0, \tag{2.12}$$

[2]See e.g. [39] for more details on subdifferentials of convex functions

where F is the single-valued "base mapping" and Φ is the multi-valued "field". Solving the GE involves finding elements of the set $(F + \Phi)^{-1}(0)$. We often encounter structured GEs of the form

$$F(x) + \mathcal{N}_C(x) \ni 0. \tag{2.13}$$

This is known as a Variational Inequality (VI) and is equivalent to the problem of finding $x \in C$ such that

$$\langle F(x), x - x' \rangle \geq 0, \ \ \forall x' \in C. \tag{2.14}$$

This property links VIs with the complementarity conditions that commonly arise in constrained optimization. The following provides a useful notion of regularity for GEs:

Definition 2.4 (Strong Regularity [40])**.** *A set-valued mapping $\Phi : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is said to be strongly regular at x for y if $y \in \Phi(x)$ and there exist neighborhoods U of x and V of y such that the truncated inverse mapping $\tilde{\Phi}^{-1} : V \mapsto \Phi^{-1}(V) \cap U$ is single-valued, i.e., a function, and is Lipschitz continuous on V.*

Strong regularity reduces to non-singularity of the Jacobian matrix if Φ is a continuously differentiable function.

We also commonly encounter parameterized GEs of the form

$$F(x, p) + \Phi(x) \ni 0, \tag{2.15}$$

where $p \in \mathbb{R}^l$ is a parameter, $\Phi : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$, and $F : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}^n$, in the context of parameterized optimization. The solution mapping of (2.15) is

$$S(p) = \{x \mid F(x, p) + \Phi(x) \ni 0\}, \tag{2.16}$$

and is itself a set-valued mapping. The following implicit function type theorem connects the notion of strong regularity with the properties of the solution mapping.

Theorem 2.1. *[41, Theorem 1.24] Let $\Phi : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ and suppose $F : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}^n$ is Lipschitz continuous. Moreover assume that $\nabla_x F$ is well defined in a neighbourhood of $(\bar{x}, \bar{p})$, is continuous at $(\bar{x}, \bar{p})$, and that $\bar{x}$ is a strongly regular solution of* (2.15) *at $\bar{p}$. Then there exists neighbourhoods U of $\bar{x}$ and V of $\bar{p}$ such that there exists a restriction of the solution mapping* (2.16)*, denoted by $s : V \to U$, that is a Lipschitz continuous function at $\bar{p}$ and satisfies*

$$s(p) - \bar{x} = \mathcal{O}(||F(\bar{x}, p) - F(\bar{x}, \bar{p})||), \tag{2.17}$$

as $p \to \bar{p}$.

2.3.2 Monotone Operators and the Proximal Point Algorithm

A set-valued mapping $T : D \rightrightarrows \mathbb{R}^n$ is said to be monotone if

$$\langle x - y, u - v \rangle \geq 0, \quad \forall\, u \in T(x), v \in T(y),\ x, y \in D, \tag{2.18}$$

where $D = \operatorname{dom} T \subseteq \mathbb{R}^n$. In addition, if $\operatorname{gph} T$ is not properly contained in the graph of any other monotone operator then T is said to be maximal [42]. If

$$\langle x - y, u - v \rangle \geq m||x - y||^2, \quad \forall\, u \in T(x), v \in T(y),\ x, y \in D, \tag{2.19}$$

then T is strongly monotone with monotonicity constant $m > 0$. Inverses of strongly monotone operators are Lipschitz continuous functions with constant m^{-1}. Notable examples of monotone operators are subdifferential mappings for convex functions, projections operators for convex sets, and normal cone mappings. See [39] for a primer on monotone operators.

The proximal point algorithm (PPA) [42] can be used to find zeros of maximal monotone operators. This algorithm generates a sequence $\{x_k\}$ by the rule

$$x_{k+1} = P_k(x_k), \quad P_k = (I + \sigma_k^{-1} T)^{-1}, \tag{2.20}$$

where σ_k is a sequence of positive numbers. Evaluating P_k is equivalent to solving

$$T(x_{k+1}) + \sigma(x_{k+1} - x_k) \ni 0. \tag{2.21}$$

Since T is monotone $T + \sigma I$ is strongly monotone for any $\sigma > 0$ and thus the proximal operator P_k is single valued and well defined for all $x \in \operatorname{dom} T$. For an arbitrary maximal monotone operator the proximal point algorithm converges to an element of the set $T^{-1}(0)$ if it is nonempty. The $\{\sigma_k\}$ sequence acts as regularization parameters, the larger they are the easier it is to evaluate the proximal operator but the slower the overall algorithm converges. One of the main advantages of the PPA is that $\{\sigma_k\}$ need not go to zero. Because of this methods based on the PPA are usually numerically robust.

The proximal point algorithm also allows for approximate evaluation of P_k. It was shown in [42] that the proximal point algorithm can tolerate errors which satisfy the following condition

$$||x_{k+1} - P_k(x_k)|| \leq \delta_k \sigma_k, \quad \sum_{k=0}^{\infty} \delta_k < \infty. \tag{2.22}$$

See [43] and the references therein for more information on proximal methods.

2.3.3 Generalized Differentiation

Suppose a function $G : \mathbb{R}^n \to \mathbb{R}^m$ is locally Lipschitz on a set $U \subseteq \mathbb{R}^n$, so that G is differentiable almost everywhere on U by Rademacher's theorem [44]. Clarke's generalized Jacobian [36] is defined as

$$\partial G(x) = \text{co}\,\{J \in \mathbb{R}^{m\times n} \mid \exists\{x^k\} \subset D_G : \{x^k\} \to x,\ \{\nabla G(x_k)\} \to J\}, \tag{2.23}$$

where D_G is the dense set of points where G is differentiable, and co $(\cdot)$ denotes the convex hull. The C-subdifferential [45] is defined as

$$\bar{\partial} G = \partial G_1 \times \partial G_2 \times ... \times\ \partial G_m, \tag{2.24}$$

where ∂G_i are the generalized Jacobians of the components mappings of G. Note that each element of ∂G_i is a row vector. The C-subdifferential is used in [46, 47] and possesses many of the useful properties of the generalized Jacobian while being easier to compute and characterize.

A function $G : \mathbb{R}^n \mapsto \mathbb{R}^m$ is said to be semismooth [35] at $x \in \mathbb{R}^n$ if G is locally Lipschitz at x, directionally differentiable in every direction and the estimate

$$\sup_{J\in\partial G(x+\xi)} ||G(x+\xi) - G(x) - J\xi|| = o(||\xi||), \tag{2.25}$$

holds. If $o(||\xi||)$ is replaced with $\mathcal{O}(||\xi||^2)$ in (2.25) then G is said to be strongly semismooth at x. The generalized Jacobian and the C-subdifferential can be used to construct Newton-type methods for semismooth systems of nonlinear equations [35, 45]. The following Newton-type method,

$$x_{k+1} = x_k - V^{-1}G(x_k), \quad V \in \partial G(x_k), \tag{2.26}$$

is locally superlinearly convergent to roots of G which satisfy some regularity properties [35, 48]. Similar results are available using the C-subdifferential [45, 47].

2.4 Parameterized Optimization

Optimization problems of the following form

$$\min_{w} \; J(w,p) \tag{2.27a}$$

$$\text{s.t.} \; g(w,p) = 0, \tag{2.27b}$$

$$h(w,p) \leq 0, \tag{2.27c}$$

where $w \in \mathbb{R}^n$ and the functions $J : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}$, $g : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}^m$, and $h : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}^q$, are twice continuously differentiable in their first argument and Lipschitz continuous in the second, are commonly known and parameterized nonlinear programs (PNLPs). The Lagrangian function associated with (2.27) is

$$L(w,\lambda,v,p) = J(w,p) + \lambda^T g(w,p) + v^T h(w,p), \tag{2.28}$$

where $\lambda \in \mathbb{R}^m$ and $v \in \mathbb{R}^q$ are the dual variables (or Lagrange multipliers) associated with the equality and inequality constraints, respectively.

The Karush-Kuhn-Tucker (KKT) conditions for (2.27) are

$$\nabla_w L(w,\lambda,v,p) = 0, \tag{2.29a}$$

$$g(w,p) = 0, \tag{2.29b}$$

$$h(w,p) \leq 0, \;\; v \geq 0, \;\; h(w,p) \odot v = 0, \tag{2.29c}$$

where $\odot$ denotes the elementwise product. The KKT conditions are necessary for optimality under an appropriate Constraint Qualification (CQ). In this dissertation, we use the Linear Independence Constraint Qualification (LICQ) which is defined below.

Definition 2.5 (LICQ)**.** *The Linear Independence Constraint Qualification (LICQ) is said to hold at* $(\bar{w},\bar{p})$ *if*

$$\textit{rank} \begin{bmatrix} \nabla_w g(\bar{w},\bar{p}) \\ [\nabla_w h(\bar{w},\bar{p})]_{\mathcal{A}(\bar{w},\bar{p})} \end{bmatrix} = m + |\mathcal{A}(\bar{w},\bar{p})|,$$

where $\mathcal{A}(w,p) = \{i \in \{1,\ ...,\ q\} \mid h_i(w,p) = 0\}$ *is the set of active constraint indices and* m *is the number of equality constraints.*

We also need a suitable sufficient condition for some of the subsequent developments. For this we use the following Strong Second Order Sufficient Condition (SSOSC).

Definition 2.6 (SSOSC)**.** *Consider a point* $(\bar{w}, \bar{\lambda}, \bar{v}, \bar{p})$ *that satisfies the KKT conditions* (2.29) *and the LICQ. If, in addition, the following condition holds*

$$y^T \nabla_w^2 L(\bar{w}, \bar{\lambda}, \bar{v}, \bar{p}) y > 0, \ \forall y \in \mathcal{K}_+(\bar{w}, \bar{v}, \bar{p}) \setminus \{0\},$$

where

$$\mathcal{K}_+(w, v, p) = \{y \in \mathbb{R}^n \mid \nabla_w g(\bar{w}, \bar{p}) y = 0, \ [\nabla_w h(\bar{w})]_i y = 0 \ \forall i \in \mathcal{A}^+(w, v, p)\},$$

and $\mathcal{A}^+(w, v, p) = \mathcal{A}(w, p) \cap \{i \mid v_i > 0\}$*, then the point* $(\bar{w}, \bar{\lambda}, \bar{v}, \bar{p})$ *is said to satisfy the Strong Second Order Sufficient Condition.*

Taken together the LICQ and SSOSC imply that a point $(\bar{w}, \bar{p})$ is (locally) optimal for (2.27) and that the associated multipliers satisfying (2.29) are unique, see e.g., [41, 49].

Its possible to rewrite the KKT conditions as a GE by exploiting that

$$v \geq 0, \ h(w, p) \leq 0 \implies (h(w, p) \odot v = 0 \Leftrightarrow h(w, p)^T v = 0). \tag{2.30}$$

Using this equivalence and the definition of the normal cone mapping we have that

$$-h(w, p) + \mathcal{N}_+(v) \ni 0 \Leftrightarrow -h(w, p) \geq 0, \ v \geq 0, h(w, p)^T v = 0 \tag{2.31}$$

so we can rewrite (2.29) as

$$\begin{bmatrix} \nabla_w L(w, \lambda, v, p) \\ -g(w, p) \\ -h(w, p) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \mathcal{N}_+(v) \end{bmatrix} \ni 0 \tag{2.32a}$$

where $\mathcal{N}_+$ denotes the normal cone mapping of the nonnegative orthant. This can be written in the standard compact form

$$F(z, p) + \mathcal{N}_K(z) \ni 0, \tag{2.33}$$

by defining

$$F(z, p) = \begin{bmatrix} \nabla_w L(w, \lambda, v, p) \\ -g(w, p) \\ -h(w, p) \end{bmatrix}, \quad z = \begin{bmatrix} w \\ \lambda \\ v \end{bmatrix}, \tag{2.34}$$

and $K = \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^q_+$. Thus KKT points can be characterized using the solution mapping

$$S(p) = \{z \mid F(z,p) + \mathcal{N}_K(z) \ni 0\}, \tag{2.35}$$

i.e., all KKT points satisfy $z \in S(p)$ or equivalently $(z,p) \in \operatorname{gph} S$. Unsurprisingly, there is also a relationship between the LICQ and SSOSC and the regularity properties of (2.32) that is summarized in the following theorem.

Theorem 2.2. *[41, Prop 1.27, 1.28] Consider a parameterized nonlinear program of the form* (2.27) *and let* $S(p)$ *be the solution mapping of its KKT conditions* (2.29)*. A point* $(\bar{z}, \bar{p}) \in \operatorname{gph} S$ *is strongly regular if it satisfies the LICQ and the SSOSC. Moreover, if* $\bar{w}$ *is a local minimizer of* (2.27)*, the LICQ and SSOSC are necessary for strong regularity.*

See [49] or [41] more more information on (parameterized) optimization.

CHAPTER 3

Time-distributed Optimization for Real-time Model Predictive Control

3.1 Introduction

Model Predictive Control is a widely used constrained control technique that defines a feedback control law as the solution of a receding horizon optimal control problem (OCP). The need to repeatedly and reliably solve this constrained, potentially non-convex, OCP in real-time is often the the most significant challenge when implementing MPC in practice. While the development of robust and efficient quadratic and convex programming solvers, see e.g., [50–53] and Chapter 4, has enabled the application of linear-quadratic MPC to a wide variety of systems, the application of MPC to systems with limited onboard computing power, fast sampling rates, and/or pronounced nonlinear dynamics remains an open problem.

One approach for reducing the computational cost of MPC is time-distributed optimization (TDO). When using TDO, rather than accurately solving the OCP at each sampling instant, the controller maintains a guess of the optimal solution and improves it at each timestep using a finite number of iterations of an optimization algorithm, thus distributing iterations over time. TDO can be interpreted as a dynamic compensator that maintains a solution estimate as an internal state, the dynamics of which are defined by the optimizer iterations. As illustrated in Figure 3.1, this interpretation differs from "ideal", or "optimal" MPC which is an implicitly defined static feedback law.

There are a variety of TDO variants proposed in the literature. The stability of input constrained TDO controllers using linearly convergent optimization algorithms are studied in [54]. Unconstrained suboptimal NMPC without terminal conditions is considered in [55]. A fixed point scheme for input constrained MPC of sampled data input affine systems is proposed in [56], a gradient based dynamic programming approach is considered in [57], a proximal gradient method for linear input-constrained MPC is studied in [58], and

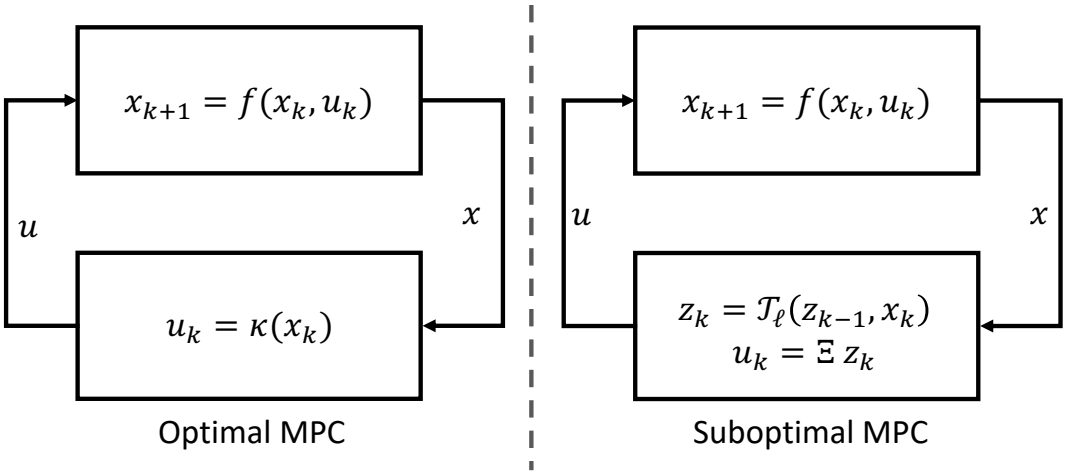

Figure 3.1: A comparison of suboptimal MPC with TDO and optimal MPC. The κ operator represents the optimal MPC feedback law, the $\mathcal{T}_\ell$ operator represents ℓ iterations of an optimization algorithm, and Ξ is a selection matrix that extracts the control action. One can roughly identify $\kappa(x) = \Xi\mathcal{T}_\infty(z_0, x)$ for any z_0 for which $\mathcal{T}$ converges.

a continuous time gradient flow based approach is described in [59]. These methods use some combination of shifting terminal control updates and first order optimization methods. In [15, 16], a generic suboptimal MPC scheme is considered and sufficient conditions on the warmstart for robust stability are derived; the optimization algorithm is not specified and its convergence is not considered. The robustness of MPC to disturbances arising from incomplete optimizations is considered in [60, 61] and conditions for complexity certification of suboptimal state constrained linear MPC are presented in [62]. However, the treatment of the optimizer itself as a dynamic system was not pursued.

An alternative to gradient based approaches are second order methods. In particular, Time Distributed Sequential Quadratic Programming (TD-SQP) methods are attractive since they can be implemented using existing Quadratic Programming (QP) solvers. The fundamental idea behind a TD-SQP based model predictive controller is to apply a finite number of SQP iterations at each sampling instant and to *warmstart* the iterations with the solution estimate from the previous sampling instant. A popular variant of TD-SQP is the Real-Time Iteration (RTI) scheme [63] which uses a Gauss-Newton Hessian approximation and performs a single SQP iteration per sampling instant. The RTI scheme has been successfully applied to a variety of applications including engines [64,65], kites [66], cranes [67], ground vehicles [68], race cars [69], distillation columns [70] and wind turbines [71]. Software for implementing the RTI scheme is provided by the `ACADO` toolkit [72] and its successor `acados` [73]. Despite its widespread success, formal stability guarantees for the RTI scheme have only been provided in the absence of inequality

constraints [74].

It should be noted that TDO is distinct from so-called suboptimal solution tracking, sensitivity, or running methods, see e.g., [60, 75–79], which are tailored numerical methods for tracking the solutions of parameterized nonlinear programs/generalized equations. These methods are typically used to accelerate or replace existing nonlinear programming solvers to reduce computation times, which is different from considering the dynamic interactions between the plant and the optimizer. Some of them, e.g., [60], consider robust stability by treating suboptimality as a bounded disturbance. This differs from our approach where we treat suboptimality as the output of a dynamic system which is coupled with the closed-loop plant.

We begin by presenting a system theoretic framework for analyzing a broad class of TDO algorithms. Specifically, the framework applies to any MPC feedback law that is Locally Input-to-State Stable (LISS) combined with any optimization algorithm whose convergence rate is at least locally q-linear. We establish the existence of a joint region of attraction for the state and solution estimate, i.e., we show that if the initial state is sufficiently close to the origin and if the initial solution estimate is sufficiently accurate, the state will converge to the origin and the estimate will converge to the optimal solution. Further, we analyze the effect of performing more iterations, establish robustness properties, and show that, if the initial solution guess is within the convergence basin of the optimization method, TDO can recover the robust region of attraction of optimal MPC with a finite number of iterations.

Next, we specialize our theoretical framework to the RTI scheme. The existing analysis of the RTI scheme [74] is limited to a nominal MPC controller with a terminal equality constraint and does not take state and input inequality constraints into account. Our analysis extends that in [74] as follows: (i) We explicitly consider inequality constraints and relax the terminal state constraint to a terminal set constraint; (ii) We explicitly consider the robustness properties of the RTI scheme by establishing LISS of the closed-loop system; (iii) We analyze the effect of the number of SQP iterations performed at each sampling instant and establish sufficient conditions for robust constraint satisfaction. We also provide a proof which extends the classical preconditioned fixed-point type analysis of Newton's method, see [80, Section 5.4.2], to the setting of generalized equations and establish conditions under which discrete time optimal control problems with polyhedral constraints are strongly regular. The latter property is important since it is a sufficient condition for Lipschitz continuity of the optimal value function and thus for robust stability.

TDO is closely related to the practice of *warmstarting* optimization algorithms. Indeed, TDO can be understood as the combination of warmstarting, i.e., using the solution of

the previous OCP as a starting point when solving the current one, and truncating, i.e., limiting the number iterations, a real-time optimization algorithm. As a result, the material in this Chapter can be interpreted as a systems theoretic analysis of the consequences of warmstarting and truncation on the closed-loop system.

The layout of the chapter is as follows. We start by describing the problem setting and the class of optimization algorithms we consider in Sections 3.2 and 3.3. Then, we establish the ISS properties of the optimization algorithms and of the coupled plant-optimizer system in Sections 3.4 and 3.5. Next, we discuss the strong regularity assumption in Section 3.6, we discuss relevant SQP methods in Section 3.7, and we illustrate how they fit into our optimization framework in Section 3.8. Finally, we present simulation results in Section 3.9.

3.2 Problem Setting and Control Strategy

Consider the following discrete time system,

$$x_{k+1} = f_d(x_k, u_k, d_k), \tag{3.1}$$

where $x_k \in \mathcal{X} \subset \mathbb{R}^{n_x}$, $u_k \in \mathcal{U} \subset \mathbb{R}^{n_u}$ and $d_k \in \mathcal{D} \subset \mathbb{R}^{n_d}$ denote the state, input, and disturbance. Throughout the chapter we assume full state feedback and that the following assumption holds.

Assumption 3.1. *The function f_d in* (3.1) *is twice continuously differentiable in its first two arguments, Lipschitz continuous in the third, and $f_d(0,0,0) = 0$. Moreover, the sets $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{D}$ are compact and contain the origin.*

We wish to control (3.1) using MPC and thus consider an OCP of the following form,

$$\min_{\xi,\mu} \ \phi(\xi,\mu) = V_f(\xi_N) + \sum_{i=0}^{N-1} l(\xi_i, \mu_i), \tag{3.2a}$$

$$\text{s.t. } \xi_0 = x, \ \ \xi_N \in \mathcal{X}_f, \tag{3.2b}$$

$$\xi_{i+1} = f_d(\xi_i, \mu_i, 0), \quad i = 0, \dots, N-1, \tag{3.2c}$$

$$(\xi_i, \mu_i) \in \mathcal{Z}, \qquad i = 0, \dots, N-1. \tag{3.2d}$$

where $N \in \mathbb{Z}_{++}$ is the horizon length, $\mathcal{Z} \subseteq \mathcal{X} \times \mathcal{U}$ represents the constraints, $\mathcal{X}_f \subseteq \mathcal{X}$ is the set defining the terminal state constraint, $\xi = (\xi_0, \xi_1, \dots, \xi_N)$ is the state sequence, and $\mu = (\mu_0, \mu_1, \dots, \mu_{N-1})$ is the control sequence. The OCP (3.2) is parameterized by

the measured system state x. We impose the following conditions on (3.2) to ensure that it is well posed and can be used to construct a stabilizing control law for (3.1).

Assumption 3.2. *All functions in* (3.2) *are twice continuously differentiable in their arguments and their second derivatives are Lipschitz continuous.*

Assumption 3.3. *The stage cost satisfies* $l(0,0) = 0$*, and there exists* $\alpha_l \in \mathcal{K}_\infty$ *such that* $\alpha_l(||x||) \leq l(x,u)$ *for all* $(x,u) \in \mathcal{Z}$*. The terminal set* $\mathcal{X}_f$ *is a subset of* $\mathcal{X}$*, contains the origin in its interior, and is an admissible control invariant set for* (3.1)*, i.e., for all* $x \in \mathcal{X}_f$ *there exists* u *such that* $f_d(x,u,0) \in \mathcal{X}_f$ *and* $(x,u) \in \mathcal{Z}$*.* V_f *is a Control Lyapunov Function for* (3.1) *with* $d = 0$*, such that,*

$$\min_u \{V_f(x^+) - V_f(x) + l(x,u) \mid (x,u) \in \mathcal{Z}, x^+ \in \mathcal{X}_f\} \leq 0,$$

for all $x \in \mathcal{X}_f$*, where* $x^+ = f_d(x,u,0)$*.*

Denote by

$$\Gamma = \{x \in \mathcal{X} \mid \text{(3.2) is feasible}\}, \tag{3.3}$$

the set of feasible parameters. Under Assumptions 3.1 and 3.2, the set Γ is compact and, for all $x \in \Gamma$, a minimum of (3.2) exists by the Weierstrass theorem. The ideal/optimal MPC feedback policy is then

$$\kappa(x) = \mu_0^*(x), \tag{3.4}$$

where $\mu^*(x)$ is a global minimizer of (3.2). To address the effects of incomplete optimization, we consider the perturbed closed loop system

$$x_{k+1} = f(x_k, \Delta u_k, d_k) := f_d(x_k, \kappa(x_k) + \Delta u_k, d_k), \tag{3.5}$$

where the control signal is corrupted by an additive disturbance that represents suboptimality caused by incomplete optimization, i.e., $u_k = \kappa(x_k) + \Delta u_k$. The following theorem summarizes the LISS properties of nominal MPC.

Theorem 3.1. *[10, Theorem 4] Let Assumptions 3.1 - 3.3 hold and suppose that* $\phi^*(x)$*, the optimal value function for* (3.2)*, is uniformly continuous. Then, the closed-loop system* (3.5) *is LISS with respect to* $(\Delta u, d)$ *on a non-empty Robust Positive Invariant (RPI) set* $\Omega \subseteq \Gamma$*. Moreover, there exist* $c_1, c_2 > 0$ *such that if* $\operatorname{rad} \Delta\mathcal{U} \leq c_1$ *and* $\operatorname{rad} \mathcal{D} \leq c_2$*, then* Ω *is an admissible RPI set for* (3.5)*.*

In this chapter, we consider the situation where not enough computational resources are available to accurately solve (3.2) at each sampling instant. Instead we will approximately

track solution trajectories of (3.2) as the measured state x in (3.2b) varies over time. To track the solution trajectories, we use an appropriate iterative optimization algorithm, e.g. SQP, which is *warmstarted* at time instance t_k with the approximate solution from t_{k-1}. In this way we construct a dynamic system,

$$z_k = \mathcal{T}_\ell(z_{k-1}, x_k), \tag{3.6}$$

where z is an estimate of the primal-dual solution of (3.2) and $\mathcal{T}_\ell$ represents a fixed number of optimizer iterations ($\mathcal{T}_\ell$ is formally defined in (3.10)). This leads to the interconnected system illustrated in Figure 3.1. The objective of this chapter is to analyze the interconnection between the plant and the dynamic controller from a systems theoretic perspective.

Remark 3.1. *We focus on a common nominal MPC formulation for the sake of clarity. However, our analysis is applicable to any MPC formulation for which it is possible to prove LISS e.g., formulations that employ exact penalty functions [81] or robust MPC formulations [10]. Moreover, note that in* (3.4) *the MPC feedback law is defined using a global optimum of* (3.2)*. This requirement is not intrinsic to our analysis but rather an artifact of the specific MPC formulation. Our analysis is performed relative to a nominal "ideal" MPC feedback law. If the nominal feedback law is input-to-state stable our analysis is applicable regardless of whether the nominal feedback law is globally optimal or not.*

3.3 Optimization Framework

In this section, we describe the class of optimization algorithms considered in this chapter. We start in an abstract setting to clarify which properties are essential to our analysis. Later in Sections 3.7 and 3.8 we will illustrate how SQP fits into this framework.

Suppose that the first order necessary conditions for (3.2) can be written as a parameterized Generalized Equation (GE) of the form

$$F(z, x) + \mathcal{N}_K(z) \ni 0, \tag{3.7}$$

where $\mathcal{N}_K : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is the normal cone mapping of a closed, convex set $K \subseteq \mathbb{R}^n$, $F : \mathbb{R}^n \times \Gamma \to \mathbb{R}^n$ is a function, $z \in \mathbb{R}^n$ is the optimization variable and $x \in \Gamma$ is the parameter. Its solution mapping is

$$S(x) = \{z \mid F(z, x) + \mathcal{N}_K(z) \ni 0\}, \tag{3.8}$$

which can be set valued. The necessary conditions (3.7) may be satisfied at saddle points,

local maxima, and local minima. To account for this we perform a local analysis that allows us to exclude saddle points and local maxima.

Many optimization algorithms are designed to "solve" necessary conditions. We thus associate (3.7) with an iterative optimization algorithm of the form

$$z_{i+1} = \mathcal{T}(z_i, x, i), \tag{3.9}$$

where $\mathcal{T} : \mathbb{R}^n \times \Gamma \times \mathbb{Z}_+ \to \mathbb{R}^n$. Multiple iterations of the algorithm can be represented by the action of the function $\mathcal{T}_\ell(z, x) : \mathbb{R}^n \times \Gamma \to \mathbb{R}^n$ which is defined recursively by the sequence

$$\mathcal{T}_\ell(z, x) = \mathcal{T}(\mathcal{T}_{\ell-1}(z, x, \ell - 1), x, \ell), \tag{3.10}$$

where $\ell \in \mathbb{Z}_{++}$ is the number of iterations per sampling instant and $\mathcal{T}_0(z, x) := z$.

Remark 3.2. *The optimality conditions for* (3.2) *can be written in multiple forms depending on the choice of* (3.7) *and* (3.9)*. For example, ξ can either be treated as a decision variable or a function of μ. As a result, the definition of z is not unique and is chosen by the designer of the optimization algorithm. Specifically, the vector z always includes the control sequence, but may also include the state sequence and/or the Lagrange multipliers associated with equality (dynamic) or inequality constraints.*

In our framework, we consider algorithms that converge at least q-linearly for a fixed parameter x. The following definition formalizes this notion.

Definition 3.1 (At least q-linear convergence)**.** *For any $x \in \Gamma$ and $z^* \in S(x)$ an optimization algorithm $\mathcal{T}$ converges to z^* if there exists $\varepsilon > 0$ such that*

$$\lim_{\ell \to \infty} \mathcal{T}_\ell(z, x) = z^* \tag{3.11}$$

for all $z \in \varepsilon\mathcal{B}(z^*)$. If there exists $\eta > 0$ and $q \geq 1$ such that

$$||\mathcal{T}_\ell(z, x) - z^*|| \leq \eta ||\mathcal{T}_{\ell-1}(z, x) - z^*||^q, \tag{3.12}$$

for all $\ell > 0$ and $\eta\varepsilon^{q-1} < 1$ then $\mathcal{T}$ is said to converge at least q-linearly over Γ.

Next we consider the regularity properties of the solution mapping (3.8). Our main regularity assumption is *pointwise strong regularity*, it establishes that any solution trajectories are Lipschitz continuous.

Assumption 3.4. *All points (z, x) satisfying $z \in S(x)$ that correspond to minimizers are strongly regular for all $x \in \Gamma$.*

We discuss conditions for strong regularity for specific instances of (3.7) in Section 3.6.

The following theorem shows that Assumption 3.4 ensures that the notion of tracking a local solution trajectory is well defined.

Theorem 3.2. *[34] Let the parameter $x \in \Gamma$ be a Lipschitz continuous function of $t \geq 0$. Then each solution trajectory $z(t) \in S(x(t))$ is isolated and Lipschitz continuous.*

A local optimization algorithm, such as SQP, can be used to track a specific "isolated branch" of the solution mapping. The branch is implicitly selected through the choice of initial guess supplied to the algorithm. Some MPC formulations require global optima while some do not, as discussed in Remark 3.1. In practice, local methods like SQP are often used regardless due to the prohibitive computational complexity of global methods.

Remark 3.3. *To summarize, an algorithm/optimality condition pair fits in our framework if:*

- *The optimality conditions can be written in the form* (3.7) *and satisfy Assumption 3.4.*
- *The algorithm can be written in the form* (3.9)*.*
- *The algorithm is at least q-linearly convergent.*

3.4 Input-to-state Stability of TDO

Consider the application of TDO to problem (3.2). In a real-time setting it is only possible to perform a finite number of iterations per sampling instant, which we denote by $\ell \in \mathbb{Z}_{++}$. If we also warmstart the optimizer with the solution from the previous sampling instant, the optimizer can be viewed as a dynamical system of the form,

$$z_k = \mathcal{T}_\ell(z_{k-1}, x_k), \tag{3.13a}$$

$$u_k = \Xi z_k, \tag{3.13b}$$

where Ξ is a surjective matrix that selects μ_0 from the solution estimate, i.e., $\kappa(x) = \Xi\bar{s}(x)$ where

$$\bar{s}(x) \in S(x), \tag{3.14}$$

is an isolated single valued restriction[1] of S (this is possible due to Theorem 3.2).

[1] Our analysis is performed relative to the ideal feedback law $\kappa(x) = \Xi\bar{s}(x)$. Any choice of the restriction $\bar{s}$ that renders the origin of the closed loop system (3.5) LISS is admissible.

In this section, we establish conditions under which (3.13) is LISS. We consider the associated error system,

$$e_{k+1} = \mathcal{G}_\ell(e_k, x_k, \Delta x_k), \tag{3.15a}$$

$$\Delta u_k = \Xi e_k, \tag{3.15b}$$

where $e_k = z_k - \bar{s}(x_k)$, and $\Delta x_k = x_{k+1} - x_k$. The error system dynamics can be explicitly constructed as follows

$$\mathcal{G}_\ell(e_k, x_k, \Delta x_k) = \mathcal{T}_\ell(e_k + \bar{s}(x_k), x_k + \Delta x_k) - \bar{s}(x_k + \Delta x_k). \tag{3.16}$$

Lemma 3.1. *Consider* (3.13) *and its error system* (3.15) *and suppose that $\mathcal{T}$ is at least q-linearly convergent. Further, let Assumption 3.4 hold. Then, there exists $a, \theta : \mathbb{Z}_{++} \to \mathbb{R}_{++}$, such that the error satisfies*

$$||e_{k+1}|| \leq a(\ell)||e_k|| + \theta(\ell)||\Delta x_k||, \tag{3.17}$$

subject to the restriction

$$||e_k|| + b||\Delta x_k|| \leq \varepsilon, \tag{3.18}$$

where ε is the convergence radius in Definition 3.1 and b is the Lipschitz constant of $\bar{s}$ over Γ. Further, $a(\ell) \in (0, 1)$ and both $a(\ell) \to 0$ and $\theta(\ell) \to 0$ monotonically as $\ell \to \infty$.

Proof. If $z_{k+1} = \mathcal{T}_\ell(z_k, x)$ for some fixed x then the error bound (3.12) implies that

$$||z_{k+1} - \bar{s}(x)|| \leq \eta^{\alpha(\ell)}||z_k - \bar{s}(x)||^{q^\ell}, \tag{3.19}$$

for all $z_k \in \varepsilon\mathcal{B}(\bar{s}(x))$, d where $\alpha(\ell) = \sum_{i=0}^{\ell-1} q^i$. Now consider any $x_{k+1}, x_k \in \Gamma$ and let $z_{k+1} = \mathcal{T}_\ell(z_k, x_{k+1})$. Then, applying (3.19) with $x = x_{k+1}$, we obtain that

$$\begin{aligned}
||z_{k+1} - \bar{s}(x_{k+1})|| &\leq \eta^{\alpha(\ell)}||z_k - \bar{s}(x_{k+1})||^{q^\ell}, \\
||e_{k+1}|| &\leq \eta^{\alpha(\ell)}||z_k - \bar{s}(x_{k+1})||^{q^\ell}, \\
&\leq \eta^{\alpha(\ell)}||[z_k - \bar{s}(x_k)] - [\bar{s}(x_{k+1}) - \bar{s}(x_k)]||^{q^\ell}, \\
&\leq \eta^{\alpha(\ell)}(||e_k|| + b||\Delta x_k||)^{q^\ell},
\end{aligned}$$

where we have used that $\bar{s}$ is Lipschitz on Γ with constant b by Assumption 3.4. Recall that ε denotes the convergence radius of $\mathcal{T}$ in Γ. A sufficient condition for the restriction

$z_k \in \varepsilon\mathcal{B}(\bar{s}(x_{k+1}))$ is then

$$\begin{aligned}
&||e_k|| + b||\Delta x_k|| \leq \varepsilon, \\
&\implies |||[z_k - \bar{s}(x_k)]|| + |||[\bar{s}(x_{k+1}) - \bar{s}(x_k)]|| \leq \varepsilon, \\
&\implies |||[z_k - \bar{s}(x_k)] - [\bar{s}(x_{k+1}) - \bar{s}(x_k)]|| \leq \varepsilon, \\
&\implies ||z_k - \bar{s}(x_{k+1})|| \leq \varepsilon.
\end{aligned}$$

Continuing and imposing $||e_k|| + b||\Delta x_k|| \leq \varepsilon$, it follows that

$$||e_{k+1}|| \leq \eta^{\alpha(\ell)}\varepsilon^{q^\ell - 1}(||e_k|| + b||\Delta x_k||). \tag{3.20}$$

If $q = 1$, then $\eta^{\alpha(\ell)}\varepsilon^{q^\ell-1} = \eta^\ell$ since $\alpha(\ell) = \sum_{i=0}^{\ell-1} q^i = \ell$ and $\varepsilon^{q^\ell-1} = \varepsilon^0 = 1$. Otherwise, note that

$$\eta^{\alpha(\ell)}\varepsilon^{q^\ell-1} = \eta^{\frac{q^\ell-1}{q-1}}\varepsilon^{q^\ell-1} = (\eta\varepsilon^{q-1})^{\frac{q^\ell-1}{q-1}}, \tag{3.21}$$

where we used that, for $q > 1$,

$$\alpha(\ell) = \sum_{i=0}^{\ell-1} q^i = \frac{q^\ell - 1}{q-1}. \tag{3.22}$$

Thus,

$$\begin{aligned}
||e_{k+1}|| &\leq \eta^{\alpha(\ell)}\varepsilon^{q^\ell-1}(||e_k|| + b||\Delta x_k||), &(3.23)\\
&\leq a(\ell)||e_k|| + \theta(\ell)||\Delta x_k||, &(3.24)
\end{aligned}$$

where

$$a(\ell) = \begin{cases} \eta^\ell & \text{if } q = 1, \\ (\eta\varepsilon^{q-1})^{\frac{q^\ell-1}{q-1}} & \text{if } q > 1, \end{cases} \tag{3.25}$$

and $\theta(\ell) = b\,a(\ell)$. Since $q \geq 1$ and $\eta\varepsilon^{q-1} < 1$ by assumption, $a(\ell) \in (0,1)$, the functions a and θ are monotonically decreasing and $a(\ell), \theta(\ell) \to 0$ as $\ell \to \infty$. □ □

The following Theorem establishes the LISS properties of the error system.

Theorem 3.3. *Consider* (3.13) *and its error system* (3.15) *and suppose that $\mathcal{T}$ is at least q-linearly convergent. Further, let Assumption 3.4 hold. Then, there exists $\tau : \mathbb{Z}_{++} \to \mathbb{R}_{++}$ such that the system is LISS if $||e_0|| \leq 0.5\varepsilon$ and $||\Delta\mathbf{x}|| \leq \tau(\ell)\varepsilon$, where $\tau(\ell) = 0.5\,(\sigma(\ell) + b)^{-1}$, $\sigma(\ell) = ba(\ell)/(1 - a(\ell))$ and ε, b and a are as defined in Lemma 3.1. Further, the*

asymptotic gain of (3.15) *is of the form*

$$\gamma_\ell(s) = 2\sigma(\ell)s_1 + 0 \cdot s_2 \tag{3.26}$$

where s_1 *and* s_2 *correspond to the* Δx *and* x *inputs, respectively, and* $\sigma(\ell) \to 0$ *monotonically as* $\ell \to \infty$.

Proof. Given Lemma 3.1, if (3.18) holds for all time instants leading up to $k-1$, it follows by direct computation (see e.g. [38, Example 3.4]) that

$$||e_k|| \leq a(\ell)^k||e_0|| + \theta(\ell)\sum_{j=0}^{k} a(\ell)^{k-j}||\Delta x_j||, \tag{3.27a}$$

$$\leq a(\ell)^k||e_0|| + \sigma(\ell)||\Delta \mathbf{x}||, \tag{3.27b}$$

where $\sigma(\ell) = \theta(\ell)/(1-a(\ell))$. To ensure that (3.18) holds, we first consider the case $k=0$ and note that[2]

$$||e_0|| + b||\Delta x_0|| \leq \max\{2||e_0||, 2b||\Delta \mathbf{x}||\}, \tag{3.28}$$

$$\leq \max\{\varepsilon, \varepsilon \cdot b/(\sigma(\ell)+b)\} = \varepsilon. \tag{3.29}$$

Next, assuming (3.18) holds for $k-1$ and recalling that $a(\ell) < 1$, we can apply (3.27) at iteration k to show that

$$\begin{aligned} ||e_k|| + b||\Delta x_k|| &\leq a(\ell)^k||e_0|| + \sigma(\ell)||\Delta \mathbf{x}|| + b||\Delta x_k||, \\ &\leq a(\ell)^k||e_0|| + (\sigma(\ell)+b)||\Delta \mathbf{x}||, \\ &\leq \max\{2a(\ell)^k||e_0||, 2(\sigma(\ell)+b)||\Delta \mathbf{x}||\}, \\ &\leq \max\{\varepsilon, \varepsilon\} = \varepsilon, \end{aligned}$$

thus ensuring that (3.18) also holds at k due to the restriction $||e_0|| \leq 0.5\varepsilon$ and $||\Delta \mathbf{x}|| \leq \tau(\ell)\varepsilon$. Since (3.27) recursively enforces its restrictions, we obtain

$$||e_k|| \leq \max\{2a(\ell)^k||e_0||, 2\sigma(\ell)||\Delta \mathbf{x}||\}, \tag{3.30}$$

which directly establishes LISS with $\beta_\ell(s,k) = 2a(\ell)^k s$ and $\gamma_\ell(||(\Delta \mathbf{x}, \mathbf{x})||) = 2\sigma(\ell)||\Delta \mathbf{x}|| + 0 \cdot ||\mathbf{x}||$. The remaining claims follow from the expression $\sigma(\ell) = \theta(\ell)/(1-a(\ell))$ since $a(\ell), \theta(\ell) \to 0$ monotonically as $\ell \to \infty$. □ □

[2]Recall that $a + b \leq \max(2a, 2b)$ for any two scalars.

3.5 LISS Properties of Suboptimal MPC

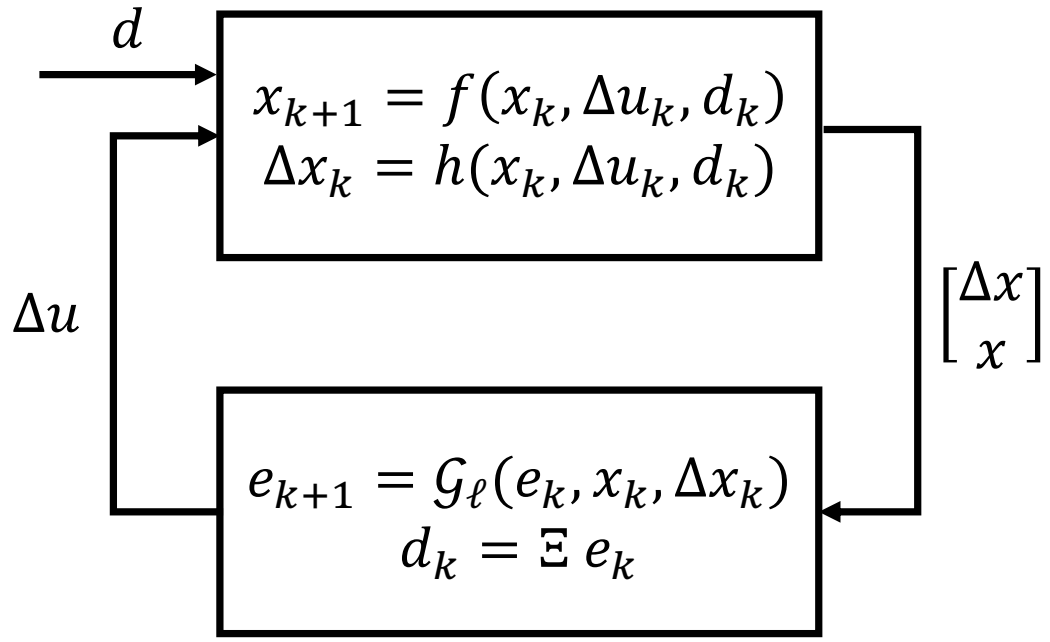

Figure 3.2: The coupled plant-optimizer error system.

Theorem 3.3 establishes sufficient conditions under which an at least q-linearly convergent optimizer, viewed as a dynamic system, is LISS. It also establishes that the asymptotic gain of (3.15) can be made arbitrarily small by increasing the number of iterations. Since the closed-loop system (3.5) is itself LISS, we can derive sufficient conditions under which the coupled system, as shown in Figure 3.2, is LISS with respect to the disturbance input d.

Theorem 3.4. *Consider the dynamical systems*

$$\Sigma_1 : \begin{cases} x_{k+1} = f(x_k, \Delta u_k, d_k), \\ \Delta x_k = h(x_k, \Delta u_k, d_k), \end{cases} \tag{3.31a}$$

$$\Sigma_2 : \begin{cases} e_{k+1} = \mathcal{G}_\ell(e_k, x_k, \Delta x_k), \\ \Delta u_k = \Xi e_k \end{cases} \tag{3.31b}$$

where $h(x, \Delta u, d) = f(x, \Delta u, d) - x$ and f is defined in (3.5)*. Let the optimization algorithm used to construct $\mathcal{G}_\ell$ satisfy* (3.12) *and let Assumptions 3.1 - 3.4 hold. Then, there*

exists $\ell^* > 0$ such that if $\ell \geq \ell^*$ the interconnected system (3.31) *is LISS with respect to the input d.*

Proof. Under Assumptions 3.1 - 3.4, Theorem 3.1 holds[3]. Thus system Σ_1 is LISS, meaning that there exist asymptotic gains $\gamma_1, \gamma_2 \in \mathcal{K}$ such that

$$\overline{\lim_{k\to\infty}} ||x_k|| \leq \gamma_1 \left(\overline{\lim_{k\to\infty}} ||\Delta u_k|| \right) + \gamma_2 \left(\overline{\lim_{k\to\infty}} ||d_k|| \right), \tag{3.32}$$

for suitably restricted $d_k \in \mathcal{D}$ and $\Delta u_k \in \Delta\mathcal{U}$. Let L denote the Lipschitz constant of h, then

$$||\Delta x_k|| \leq L||x_k|| + L||\Delta u_k|| + L||d_k||, \tag{3.33}$$

combining this with (3.32) we obtain that

$$\overline{\lim_{k\to\infty}} ||\Delta x_k|| \leq \gamma_3 \left(\overline{\lim_{k\to\infty}} ||\Delta u_k|| \right) + \gamma_4 \left(\overline{\lim_{k\to\infty}} ||d_k|| \right), \tag{3.34}$$

where $\gamma_3 = L \cdot (\gamma_1 + \mathrm{id})$, and $\gamma_4 = L \cdot (\gamma_2 + \mathrm{id})$. Similarly, due to Theorem 3.3, there exists $\varepsilon > 0$ and positive functions σ and τ such that

$$\overline{\lim_{k\to\infty}} ||e_k|| \leq \sigma(\ell) \overline{\lim_{k\to\infty}} ||\Delta x_k||, \tag{3.35}$$

provided $||\Delta \mathbf{x}|| \leq \tau(\ell)\varepsilon$. Therefore, it follows from (3.31b) that

$$\overline{\lim_{k\to\infty}} ||\Delta u_k|| \leq ||\Xi|| \overline{\lim_{k\to\infty}} ||e_k|| \leq \sigma(\ell)||\Xi|| \overline{\lim_{k\to\infty}} ||\Delta x_k||. \tag{3.36}$$

Combining (3.36) with (3.34) we obtain that

$$\overline{\lim_{k\to\infty}} ||\Delta u_k|| \leq \sigma(\ell)||\Xi||\gamma_3 \left(\overline{\lim_{k\to\infty}} ||\Delta u_k|| \right) + \sigma(\ell)||\Xi||\gamma_4 \left(\overline{\lim_{k\to\infty}} ||d_k|| \right). \tag{3.37}$$

Thus, if the contraction property

$$||\Xi||\sigma(\ell)\gamma_3(s) \leq s \tag{3.38}$$

is satisfied for all $s \in [0, \operatorname{rad} \Delta\mathcal{U}]$, (3.31) is LISS with suitable restrictions on the initial state and on the disturbance d, as detailed in [82, Theorem 2]. Note that, since $\Delta u = \Xi e$, we have $\operatorname{rad} \Delta\mathcal{U} \leq \Xi\varepsilon$ where ε is the convergence radius of the optimizer defined in Theorem 3.3. Since $\sigma(\ell) \to 0$ monotonically as $\ell \to \infty$, the existence of $\ell^* < \infty$ such that (3.38) is satisfied follows from the finiteness of γ_3, $\operatorname{rad} \Delta\mathcal{U}$, and $||\Xi||$. □ □

[3]Recall that Assumption 3.4 is sufficient for Lipschitz continuity of the optimal value function $\phi^*(x)$.

Theorem 3.4 establishes conditions under which the interconnected plant-optimizer system is LISS. However, this result does not provide any information about the set of admissible initial conditions and does not consider constraint satisfaction. By noting that the ideal MPC feedback law admits a robust positively invariant set, we can extend our result by deriving sufficient conditions for constraint satisfaction.

Theorem 3.5. *Suppose that the assumptions of Theorem 3.4 hold so the interconnected system* (3.31) *is LISS. Let* Ω *denote the admissible RPI set in Theorem 3.1, let* $\gamma_\ell(s) = 2\sigma(\ell)s$ *denote the asymptotic gain of* (3.15)*, and let* $(x_k, e_k) = (x(k, x_0, \mathbf{d}), e(k, e_0, \mathbf{d}))$ *denote the closed-loop trajectory of* (3.31) *for some initial condition* (x_0, e_0) *and disturbance sequence* $\mathbf{d}$*. Then, if the disturbances are sufficiently small, there exists* $\bar{\ell} \geq \ell^*$ *and* $\delta > 0$ *such that, if* $||e_0|| \leq \delta$ *and* $x_0 \in \Omega$*, then* $x_k \in \Omega$ *for all* $k \geq 0$*.*

Proof. Due to Theorem 3.1, given a sufficiently small disturbance set $\mathcal{D}$, there exists a neighbourhood of the origin $\Delta\mathcal{U}$ such that, if $\Delta u_k \in \Delta\mathcal{U},\ \forall k \geq 0$ and $x_0 \in \Omega$, then $x_k \in \Omega,\ \forall k \geq 0$. Since $\Delta u = \Xi e$ for a surjective matrix Ξ, there exists $\rho > 0$ such that, if $||e_k|| \leq \rho$, then $\Delta u_k \in \Delta\mathcal{U}$. Given the restriction $||\Delta\mathbf{x}|| \leq \tau(\ell)\varepsilon$ and $||e_0|| \leq 0.5\varepsilon$, where τ and ε are defined in Theorem 3.3, it follows from (3.30) that $||e_k|| \leq \rho$ can be imposed by enforcing $||e_0|| \leq 0.5\rho$ and $2\sigma(\ell)||\Delta\mathbf{x}|| \leq \rho$. To enforce $2\sigma(\ell)||\Delta\mathbf{x}|| \leq \rho$, we note that the set Ω is bounded [10, Theorem 4], thus implying that $\Delta x \in \Delta\Omega = \Omega - \Omega$ is bounded by

$$\bar{s} = \sup_{w \in \Delta\Omega} ||w|| < \infty. \tag{3.39}$$

Since $\bar{s}$ is finite and $\sigma(\ell) \to 0$ monotonically as $\ell \to \infty$, there exists ℓ_1 such that $2\sigma(\ell_1)\bar{s} \leq \rho$. Moreover, since $\bar{s}$ is finite and $\tau(\ell) \to \infty$ monotonically as $\ell \to \infty$ there exists ℓ_2 such that $\bar{s} \leq \tau(\ell_2)\varepsilon$. Thus, letting $\delta = 0.5\max\{\rho, \varepsilon\}$ and $\bar{\ell} = \max(\ell^*, \ell_1, \ell_2)$, it follows that the system is LISS with restrictions on the initial conditions $x_0 \in \Omega$ and $\|e_0\| \leq \delta$, as well as restrictions on the external disturbance $d \in \mathcal{D}$. □ □

Theorem 3.5 establishes that, if enough computational resources are available and the initial solution guess is sufficiently accurate, then TDO recovers the robustness properties of optimal MPC.

Remark 3.4. *The results presented in this section are quite general: as long as the MPC formulation is LISS, the solution mapping of the OCP is strongly regular, and the convergence rate of the iterative solver is at least q-linear, Theorems 3.4 and 3.5 prove that it is possible to achieve robust stability and constraint satisfaction by performing a limited number of solver iterations per time step. Due to the generality of the framework, however, the actual values we obtain for* ℓ^* *and* $\bar{\ell}$ *are likely to be conservative and would be ill-suited*

for, e.g., complexity certification as in [62], which, it should be noted, only considers the linear case. Despite this drawback, our results significantly extend the existing analysis of the RTI scheme [74] when our framework is applied to TD-SQP. Complexity certification is significantly more challenging in the nonlinear case due to the nonconvexity of the OCPs and is left to future work.

3.6 Conditions for Strong Regularity

The main results in this chapter are all predicated upon Assumption 4, that for each parameter value $x \in \Gamma$ the solution mapping of the OCP is strongly regular. In this section, we discuss some common settings and derive strong regularity conditions for each.

3.6.1 Closed Convex Constraint Sets

If the constraint sets $\mathcal{Z}$ and $\mathcal{X}_f$ in (3.2) are closed and convex, it is possible to write the optimality conditions without introducing dual variables for the inequality constraints. In particular, we can express (3.2) compactly as

$$\min_{w \in W} \ \phi(w), \ \text{ s.t. } \ g(w, x) = 0, \tag{3.40}$$

where $W = \mathcal{Z} \times \mathcal{Z} \ldots \times \mathcal{X}_f$ and $w = (\xi_0, \mu_0, \ldots, \xi_N) \in \mathbb{R}^p$. The Lagrangian associated with (3.40) is

$$\mathcal{L}(w, \lambda, x) = \phi(w) + \lambda^T g(w, x), \tag{3.41}$$

where $\lambda \in \mathbb{R}^l$ are dual variables associated with the equality constraints. In the context of optimal control these are sometimes referred to as "co-states". The KKT conditions for (3.40) are [33]

$$\nabla_z \mathcal{L}(z, x) + \mathcal{N}_Z(z) \ni 0, \tag{3.42}$$

where $z = (w, \lambda)$ and $Z = W \times \mathbb{R}^l$. Note that (3.42) can be reduced to (3.7) by choosing $F = \nabla_z \mathcal{L}$ and $K = Z$. Our framework requires that (3.42) be necessary for optimality. To ensure this, we impose the following constraint qualification [32, Theorem 6.14]

$$-\nabla_w g(\bar{w}, \bar{x})^T y \in \mathcal{N}_W(\bar{w}) \implies y = 0, \tag{3.43}$$

for all $(\bar{w}, \bar{\lambda}) \in S(\bar{x})$. The following lemma proves that (3.43) holds automatically in this setting.

Lemma 3.2. *The constraint qualification* (3.43) *holds at all points* $(z, x) \in \mathbb{R}^{p+l} \times \Gamma$.

Proof. The constraint qualification is implied by surjectivity of the matrix $\nabla_w g(w,x)$. Denoting $A_i = \nabla_\xi f_d(\xi_i,\mu_i,0)$, and $B_i = \nabla_\mu f_d(\xi_i,\mu_i,0)$, the surjectivity of $\nabla_w g(w,x)$ becomes the condition that for every $\xi = (\xi_0,\ldots,\xi_N)$ the system

$$x_0 = \xi_0,\ \ \zeta_{i+1} - A_i\zeta_i - B_i\nu_i = \xi_{i+1},\ \ i \in \mathbb{Z}_{[0,N-1]},$$

has a solution. This condition clearly holds: pick an arbitrary sequence $(\nu_0,\ldots,\nu_{N-1})$ and determine $(\zeta_0,\ldots,\zeta_N)$ recursively. □ □

Before stating the conditions for strong regularity, we recall the following second order condition.

Definition 3.2 (SOSC)**.** *The Second Order Sufficient Condition (SOSC) is said to hold at* $\bar{z} = (\bar{w},\bar{\lambda}) \in S(\bar{x})$ *if*

$$y^T\nabla_w^2\mathcal{L}(\bar{z},\bar{x})y > 0,\ \ \forall y \text{ s.t. } \nabla_w g(\bar{w},\bar{x})y = 0. \tag{3.44}$$

This SOSC reduces to the SSOSC discussed in Chapter 2 in the absence of inequality constraints. This condition can be monitored numerically, see e.g., [49, Section 16.2]. Doing so would allow regularization to be added when necessary to ensure the SOSC holds.

3.6.1.1 Convex Control Constraints

If only convex control constraints are present, Theorem 3.6 provides sufficient conditions for strong regularity.

Theorem 3.6. *[83, Theorem 1.2] Suppose that* $\mathcal{Z} = \mathbb{R}^{n_x} \times \mathcal{U}$*, where* $\mathcal{U}$ *is closed and convex, and consider any* $\bar{z} \in S(\bar{x})$*. If* (3.44) *holds, then* S *is strongly regular at* $(\bar{z},\bar{x})$*.*

As a result of Theorem 3.6, Assumption 3.4 reduces to the assumption that (3.44) holds at all minimizers in Γ. In this scenario, any terminal set constraints would have to be enforced through penalty functions.

3.6.1.2 Polyhedral State and Control Constraints

If the state and control constraints are convex polyhedra, the following theorem applies. The result was previously asserted without proof in [77, Section 3.2], we provide a proof for completeness.

Theorem 3.7. *Suppose that W in* (3.40) *is polyhedral with a representation $W = \{w \mid Mw \leq h\}$. Now consider a KKT point $\bar{z} = (\bar{w}, \bar{\lambda}) \in S(\bar{x})$. If* (3.44) *holds, then S is strongly regular at $(\bar{z}, \bar{x})$.*

Proof. Strong regularity of the nonlinear GE (3.42) at $(\bar{z}, \bar{x})$ follows from strong regularity of its partial linearization [40]. This can be written as

$$\begin{bmatrix} R & G^T \\ -G & 0 \end{bmatrix} \begin{bmatrix} w \\ \lambda \end{bmatrix} + \begin{bmatrix} r \\ g \end{bmatrix} + \mathcal{N}_C(z) \ni 0, \tag{3.45}$$

where $\hat{f} = \nabla_w \mathcal{L}(\bar{z}, \bar{x})$, $R = \nabla_w^2 \mathcal{L}(\bar{z}, \bar{x})$, $G = \nabla_w g(\bar{z}, \bar{x})$, $r = \hat{f} - R\bar{w} - G^T\bar{\lambda}$, $g = G\bar{w}$, $\Theta = \{w \mid Gw = g,\ Mw \leq h\}$ and $C = \Theta \times \mathbb{R}^l$. Equation (3.45) is an affine GE of the form,

$$Az + a + \mathcal{N}_C(z) \ni 0, \tag{3.46}$$

to which we apply [33, Theorem 2E.6] to establish strong regularity of the mapping $A + \mathcal{N}_C$. This requires

$$z \in \mathcal{E}^+,\ Az \perp \mathcal{E}^-,\ z^T Az \leq 0 \Rightarrow z = 0, \tag{3.47}$$

where $\mathcal{E}^+ = \mathcal{E} - \mathcal{E}$, $\mathcal{E}^- = \mathcal{E} \cap -\mathcal{E}$, and

$$\mathcal{E} = \{(w, \lambda) \mid Gw = 0, M_i w \leq 0\ i \in \mathcal{A}(\bar{w}), \hat{f}^T w = 0\},$$

is the critical cone[4] of C at $\bar{z}$. Next, note that $\mathcal{E} \subseteq \mathcal{E}^+ \subset \ker G \times \mathbb{R}^l$ thus, by the second order condition, $y^T R y = y^T \nabla_w^2 \mathcal{L}(\bar{z}, x) y > 0$ for all $y \in \ker G$. Thus

$$z^T Az = w^T Rw > 0, \forall w \in \mathcal{E}^+, \tag{3.48}$$

which implies that $z^T Az \leq 0 \implies z = 0$ for all $z \in \mathcal{E}^+$. As a result, (3.47) is satisfied and (3.42) is strongly regular. □ □

Thus, as in the case of convex input constraints, Assumption 3.4 reduces to the condition that (3.44) holds at all minimizers in Γ.

3.6.2 Nonlinear Inequality Constraints

If the constraint sets in (3.2) can be expressed in the form $\mathcal{Z} = \{(\xi, \mu) \mid c(\xi, \mu) \leq 0\}$ and $\mathcal{X}_f = \{\xi \mid c_f(\xi) \leq 0\}$ for suitable twice continuously differentiable functions c :

[4] See [33, Section 2E] for more details on critical cones. We've simplified the expression for $\mathcal{E}$ using [33, Theorem 2E.3] and (3.42).

$\mathbb{R}^{n_x+n_u} \to \mathbb{R}^{n_c}$ and $c_f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{cf}}$, then (3.2) can be written compactly as the following Nonlinear Program (NLP),

$$\min_{w} \quad \phi(w), \tag{3.49a}$$

$$\text{s.t.} \quad g(w, x) = 0, \;\; h(w) \leq 0, \tag{3.49b}$$

where $w = (\xi, \mu) \in \mathbb{R}^p$ are the decision variables. The Lagrangian associated with (3.49) is

$$L(w, \lambda, v, x) = \phi(w) + \lambda^T g(w, x) + v^T h(w), \tag{3.50}$$

where $\lambda \in \mathbb{R}^l$ and $v \in \mathbb{R}^m$ are dual variables. Its KKT conditions [41] are

$$\nabla_w L(w, \lambda, v, x) = 0, \tag{3.51a}$$

$$-g(w, x) = 0, \tag{3.51b}$$

$$-h(w) + \mathcal{N}_+(v) \ni 0, \tag{3.51c}$$

where $\mathcal{N}_+$ is the normal cone mapping of the non-negative orthant. Comparing (3.51) with (3.7) we can identify $z = (w, \lambda, v)$, $K = \mathbb{R}^p \times \mathbb{R}^l \times \mathbb{R}^m_{\geq 0}$, and

$$F(z, x) = \begin{bmatrix} \nabla_w L(w, \lambda, v, x) \\ -g(w, x) \\ -h(w) \end{bmatrix}. \tag{3.52}$$

To ensure that (3.51) are necessary for optimality, as required by our framework, we impose the LICQ (Definition 2.5) (3.49). Further, invoking Theorem 2.2, the LICQ and SSOSC (Definition 2.6) are sufficient for strong regularity of (3.51). Thus, Assumption 3.4 reduces to the assumption that the SSOSC and LICQ hold at all minima in Γ.

3.7 Sequential Quadratic Programming

Having identified under what conditions the solution mapping of the OCP is strongly regular, we investigate the convergence properties of two widely used SQP schemes to show that they can be used for TDO. To this effect, note that the OCPs (3.40) and (3.49) can both be solved using SQP. Specifically, for (3.49), given a solution estimate z_i, the next iterate

can be computed by solving the following Quadratic Program (QP)

$$\min_{\Delta w_i} \quad \frac{1}{2}\Delta w_i^T B_i \Delta w_i + \nabla_w \phi(z_i)^T \Delta w_i, \tag{3.53a}$$

$$\text{s.t.} \quad \nabla_w g(w_i, x)\Delta w_i + g(w_i, x) = 0, \tag{3.53b}$$

$$\nabla_w h(w_i)\Delta w_i + h(w_i) \leq 0, \tag{3.53c}$$

where B_i approximates the Hessian of the Lagrangian $\nabla_w^2 L$. Specifically, if we denote the Lagrange multipliers associated with the equality and the inequality constraints by π_i and η_i, respectively, the SQP update for (3.49) is $z_{i+1} = (w_i + \Delta w_i, \pi_i, \eta_i)$. Note that (3.53) is fully defined by (3.40) or (3.49) except for B_i, which will depend on the specific SQP method.

SQP applied to (3.40) is similar, the QP subproblem becomes

$$\min_{\Delta w_i} \quad \frac{1}{2}\Delta w_i^T B_i \Delta w_i + \nabla_w \phi(z_i)^T \Delta w_i, \tag{3.54a}$$

$$\text{s.t.} \quad \nabla_w g(w_i, x)\Delta w_i + g(w_i, x) = 0, \tag{3.54b}$$

$$w_i + \Delta w_i \in W. \tag{3.54c}$$

The main changes are that the SQP update becomes $z_{i+1} = (w_i + \Delta w_i, \pi_i)$, i.e., the inequality duals are removed from the iteration, and the Hessian matrix B_i must approximate $\nabla_w^2 \mathcal{L}$ instead of $\nabla_w^2 L$.

To provide a unified formulation, we exploit that SQP can be seen as a Newton-type process for solving GEs of the form

$$F(z, x) + \mathcal{N}_K(z) \ni 0, \tag{3.55}$$

where $z \in \mathbb{R}^n$, $x \in \Gamma$, $F : \mathbb{R}^n \times \Gamma \to \mathbb{R}^n$ is continuously differentiable and $\mathcal{N}_K : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is the normal cone mapping for a closed, convex set $K \subseteq \mathbb{R}^n$. Newton's method applied to (3.55) is

$$H_i(z_{i+1} - z_i) + F(z_i, x) + \mathcal{N}_K(z_{i+1}) \ni 0, \tag{3.56}$$

where the sequence $\{H_i\}$ approximates $\nabla_z F(z_i, x)$. Referring to the QP subproblem (3.53), we note that

$$H_i = \begin{bmatrix} B_i & \nabla_w^T g(w_i, x) & \nabla_w^T h(w_i) \\ -\nabla_w g(w_i, x) & 0 & 0 \\ -\nabla_w h(w_i) & 0 & 0 \end{bmatrix}, \tag{3.57}$$

for (3.49). To obtain the expression for (3.40) simply discard the third row and column in (3.57). Thus, the the sequence $\{H_i\}$ is fully determined by the Hessian approximation sequence $\{B_i\}$.

Remark 3.5. *In this chapter, we only consider "undamped" Newton methods, which are intrinsically local methods. More sophisticated implementations may include various type of regularization and/or globalization techniques such as trust regions or linesearches to enlarge the methods region of attraction. Nevertheless, undamped Newton methods are commonly used in practice, especially in the context of the RTI scheme, and the tools we develop in this chapter are applicable to locally convergent algorithms. We leave the application of our tools to globalized SQP methods to future work and refer readers to e.g., [49] or [41], for more detailed treatments of SQP methods.*

3.7.1 The Josephy-Newton (JN) method

Using the exact Hessian of the Lagrangian results in the Josephy-Newton method. The following theorem summarizes the convergence properties of the JN method applied to (3.55).

Theorem 3.8. *[41, Theorem 3.2] Let $z^* \in \bar{S}(x)$ for some fixed x and suppose that Assumption 3.2 holds and (z^*, x) is strongly regular. Let the sequence $\{z_i\}$ be generated by repeatedly solving*

$$\nabla_z F(z_i, x)(z_{i+1} - z_i) + F(z_i, x) + \mathcal{N}_K(z_{i+1}) \ni 0. \tag{3.58}$$

Then, there exist $\bar{\eta} = \bar{\eta}(x) > 0$ and $\bar{\epsilon} = \bar{\epsilon}(x) > 0$ satisfying $\bar{\eta}\bar{\epsilon} < 1$, such that, if $z_0 \in \bar{\epsilon}\mathcal{B}(z^*)$, then $\{z_i\}$ is unique and converges to z^* q-quadratically, i.e.,

$$||z_{i+1} - z^*|| \leq \bar{\eta}||z_i - z^*||^2. \tag{3.59}$$

In general, we cannot expect $\nabla_w^2 L$ to be positive semidefinite even in the vicinity of a solution. This may make solving the QP subproblems difficult and is a well known issue in the SQP literature. A detailed discussion is outside the scope of this chapter, we refer interested readers to e.g., [41, 49, 84].

3.7.2 The Gauss-Newton (GN) method

The Gauss-Newton method is applicable when the objective function has the form $\phi(w) = ||r(w)||_2^2$ for some residual function r. The Hessian of the Lagrangian is then approximated

by

$$B(w) = \nabla_w r(w) \nabla_w r(w)^T \approx \nabla_w^2 L(z, x). \tag{3.60}$$

For example, if $\phi(w) = x^T Q x + u^T R u$, the GN Hessian approximation is $B = \texttt{blkdiag}(Q, R)$. The GN method has the advantage that the Hessian approximation is guaranteed to be positive semidefinite, so the QP subproblems can be solved reliably. Because of this, the GN method is widely used in practice, see e.g., [61, 64, 67, 71, 72, 85]. The GN approximation error satisfies

$$\nabla_w^2 L(z, x) - B(w) = \mathcal{O}(||r(w)||) + \mathcal{O}(\sum_{i=1}^{m} ||\lambda_i||\, ||\nabla_w^2 g_i(w, x)||), \tag{3.61}$$

from which we see that the approximation error depends on the size of the residuals and on the second derivative g which is related to the nonlinearity of the dynamics. We show in Theorem 3.9 that it is important to approximate $\nabla_w^2 L(z^*, x)$ where $z^* \in \bar{S}(x)$.

The following theorem establishes sufficient conditions for q-linear convergence of the GN method by extending the classical fixed-point type analysis of Newton's method, see [80, Section 5.4.2]. The nearest analysis we found in the literature is [77, Theorem 3.5] which considers a path tracking problem rather than a fixed one.

Theorem 3.9. *Fix some parameter $x \in \Gamma$, let $z^* \in S(x)$ and suppose that Assumptions 3.2 and 3.4 hold. Consider a sequence $\{z_i\}$ generated by repeatedly solving* (3.56)*. Further, define $e_i = z_i - z^*$ and suppose that there exist $\bar{\delta} = \bar{\delta}(x) > 0$ such that $||H_i - \nabla F(z^*, x)|| \leq \bar{\delta}$ for all $i \geq 0$. If the mapping*

$$J_i(z) = H_i z + \mathcal{N}_K(z) \tag{3.62}$$

is strongly regular for all $i \geq 0$, i.e., J_i^{-1} is a Lipschitz continuous function with Lipschitz constant $M > 0$, and $\bar{\delta} M < 1$, then there exists $\bar{\epsilon} = \bar{\epsilon}(x) > 0$, and $L > 0$ such that if $z_0 \in \bar{\epsilon}\mathcal{B}(z^*)$, then $\{z_i\}$ is unique, converges to z^* q-linearly, and

$$||e_{i+1}|| \leq M(\bar{\delta} + L||e_i||)||e_i|| \leq \bar{\eta}||e_i||, \tag{3.63}$$

where $\bar{\eta} = \bar{\eta}(x) = M(\bar{\delta} + L\bar{\epsilon})$.

Proof. A solution, $z^* \in \bar{S}(x)$, exists for every $x \in \Gamma$ thanks to Assumption 3.4; from this point forward we will suppress the dependencies on x in the subsequent expressions. The GN method can be written as

$$z_{i+1} = J_i^{-1} \circ G_i(z_i) = T_i(z_i), \tag{3.64}$$

where $G_i(z) = H_i z - F(z)$; note that $z^* = T_i(z^*)$ for any choice of $\{H_i\}$. First consider

$$\begin{aligned} G_i(z_i) - G_i(z^*) &= H_i(z_i - z^*) - F(z_i) + F(z^*), \\ &= [\nabla F(z^*)(z_i - z^*) - F(z_i) + F(z^*)] + [(H_i - \nabla F(z^*))(z_i - z^*)]. \end{aligned}$$

Since ∇F is Lipschitz (Assumption 3.2) the fundamental theorem of calculus, see e.g., [86, Theorem 1.2.1], implies that there exist $L, \epsilon_1 > 0$ such that

$$||\nabla F(z^*)(z_i - z^*) - F(z_i) + F(z^*)|| \leq L||z_i - z^*||^2,$$

for all $z_i \in \epsilon_1 \mathcal{B}(z^*)$, so, taking norms, we obtain that

$$||G_i(z_i) - G_i(z^*)|| \leq L||z_i - z^*||^2 + \bar{\delta}||z_i - z^*||.$$

By assumption the mapping J_i^{-1} is Lipschitz continuous so $\Delta T_i^* = ||T_i(z) - T_i(z^*)||$ satisfies

$$\begin{aligned} \Delta T_i^* &= ||J_i^{-1}(G_i(z_i)) - J_i^{-1}(G_i(z^*))||, && \text{(3.65a)} \\ &\leq M||G_i(z_i) - G_i(z^*)||, && \text{(3.65b)} \\ &\leq M(\bar{\delta} + L||e_i||)||e_i||, && \text{(3.65c)} \end{aligned}$$

for all $z_i \in \epsilon_1 \mathcal{B}(z^*)$. Now consider the update equation

$$||z_{i+1} - z^*|| = ||T_i(z_i) - z^*|| = ||T_i(z_i) - T_i(z^*)||,$$

where we have used that $z^* = T_i(z^*)$. Since J_i is strongly regular, T_i is a function and $\{z_i\}$ is unique. Using (3.65) we have

$$||e_{i+1}|| \leq M(\bar{\delta} + L||e_i||)||e_i||, \;\; \forall e_i \in \epsilon_1 \mathcal{B}. \tag{3.66}$$

Since $M\bar{\delta} < 1$ by assumption, it is possible to pick $\bar{\epsilon} \in (0, \epsilon_1)$ such that $\bar{\eta} = M(\bar{\delta} + L\bar{\epsilon}) < 1$. Then $\{z_i\}$ converges q-linearly to z^* if $z_0 \in \bar{\epsilon}\mathcal{B}(z^*)$, i.e.,

$$||e_{i+1}|| \leq \bar{\eta}||e_i|| \;\; \forall e_i \in \bar{\epsilon}\mathcal{B}. \quad \square \tag{3.67}$$

$\square$

Theorem 3.9 requires that H_i be a sufficiently good approximation of $\nabla_z F(z^*)$ and that the GN subproblems be strongly regular. A sufficient condition for strong regularity is that

the QP (3.53) satisfies the LICQ and SSOSC (Theorem 2.2). In practice, strong regularity can be achieved by a judicious choice of H_i. For example, if the Hessian approximation is convex then it is possible to guarantee strong regularity of the subproblems by adding a regularization term, i.e., $H \leftarrow H+\delta I$ for some small $\delta > 0$. Then the mapping $H+\delta I+\mathcal{N}_K$ is strongly monotone, which implies strong regularity [33, Theorem 2F.6].

Remark 3.6. *Theorem 3.9 just requires the Hessian approximation be sufficiently good. One can conceive of useful approximation schemes other than the GN approximation, e.g., $B_i = \nabla_w^2 \phi(w_i)$ when ϕ is convex or $B_i = \nabla_w^2 L(\bar{z}, x)$ for some fixed $\bar{z}$ near z^*. These could be used in place of the GN Hessian approximation and would result in an algorithm with very similar theoretical properties.*

3.8 Time-distributed SQP

In this section, we demonstrate that the methods described in Sections 3.6 and 3.7 satisfy the conditions of Remark 3.3 and can therefore be used within the framework presented in Section 3.3.

TD-SQP using the GN Hessian approximation and with $\ell = 1$ corresponds to the RTI scheme [63]. As such, when specialized to the RTI scheme, Theorems 3.4 and 3.5 are a significant extension of the existing analysis [74] which does not consider inequality constraints.

Strong Regularity Assumption: As detailed in Section 3.6.1, in the presence of convex constraint sets Assumption 3.4 can be reduced to the following:

Assumption 3.5. *The second order sufficient condition* (3.44) *holds at all minimizers in* Γ.

As detailed in Section 3.6.2, in the nonlinear inequalities setting, Assumption 3.4 can instead be ensured under the following:

Assumption 3.6. *The linear independence constraint qualification (see Definition 2.5) and strong second order sufficient condition (see Theorem 2.2) hold at all minimizers in* Γ.

Algorithm Definition: Both SQP methods described in Section 3.7 are instances of the following iterative process

$$H_i(z_{i+1} - z_i) + F(z_i, x) + \mathcal{N}_K(z_{i+1}) \ni 0, \tag{3.68}$$

for specific choices of z, F, and K. Thus, in both cases the optimization mapping (3.9) can be written as

$$\mathcal{T}(z, x, i) = (H_i + \mathcal{N}_K)^{-1}(H_i z - F(z, x)). \tag{3.69}$$

Convergence Rate: If the exact Hessian is used then Theorem 3.8 applies and the method is at least q-linearly convergent with $q = 2$. If the GN Hessian approximation is used then Theorem 3.9 applies under some additional assumptions regarding the accuracy of the Hessian approximation, and the method is at least q-linearly convergent with $q = 1$. In both cases the definition of q-linear convergence requires that there be a uniform convergence constant η and convergence radius ε over Γ. Under the assumption that the functions $\bar{\eta}(x)$ and $\bar{\epsilon}(x)$ in Theorems 3.8 and 3.9 are upper and lower semicontinuous, respectively, these can be defined as $\varepsilon = \inf_{x\in\Gamma} \bar{\epsilon}(x)$ and $\eta = \sup_{x\in\Gamma} \bar{\eta}(x)$. Thus, SQP fits into the framework in Section 3.3 and can be used for time distributed optimization.

3.9 A Numerical Example

Figure 3.3 illustrates a bicycle model of a sedan. We only consider the lateral portion of the dynamics; the longitudinal velocity s is assumed constant. The states and control inputs are,

$$x = [y \;\; \psi \;\; \nu \;\; \omega \;\; \delta_f \;\; \delta_r], \;\; u = [\dot{\delta}_f \;\; \dot{\delta}_r], \tag{3.70}$$

where y is the lateral position, ν is the lateral component of velocity, ψ is the yaw angle, ω is the yaw rate, δ_f is the front steering angle, and δ_r is the rear steering angle.

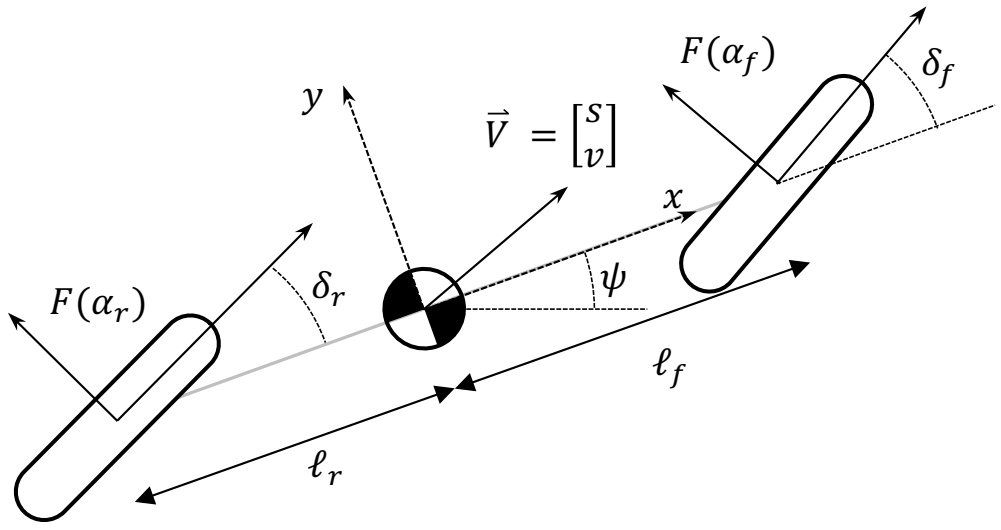

Figure 3.3: A diagram of the bicycle model

The equations of motion are

$$
\begin{aligned}
\dot{y} &= s\sin(\psi) + \nu\cos(\psi), \\
\dot{\psi} &= \omega, \\
\dot{\nu} &= -s\omega + \frac{F(\alpha_f)\cos(\delta_f) + F(\alpha_r)\cos(\delta_r) + F_w}{m}, \\
\dot{\omega} &= \frac{F(\alpha_f)\cos(\delta_f)\ell_f - F(\alpha_r)\cos(\delta_r)\ell_r}{I_{zz}}, \\
\dot{\delta_f} &= \dot{\delta}_f, \quad \dot{\delta}_r = \dot{\delta}_r,
\end{aligned}
$$

where

$$
\begin{aligned}
F(\alpha) &= \mu\ 9.81\ m \sin\left(C\arctan(B\ \alpha)\right), \\
\alpha_f &= \delta_f - \arctan\left(\frac{\nu + \ell_f\omega}{s}\right), \\
\alpha_r &= \delta_r - \arctan\left(\frac{\nu - \ell_r\omega}{s}\right), \\
\text{and } F_w &= 1/2\rho C_d A|d|d.
\end{aligned}
$$

The tire forces are described by a Pacejka [87] model, the slip angle vs. force curve is shown in Figure 3.4. This model is a modified version of the one presented in [88] and roughly represents a 2017 BMW 740i. The vehicle is disturbed by normally distributed wind gusts d with a mean velocity of $15\ m/s$ and standard deviation of $5\ m/s$. We obtain a discrete time model using a forward Euler integration scheme with a sampling period of $t_s = 0.04s$ leading to a discrete time model of the form $x_{k+1} = f_d(x_k, u_k, d_k)$. The model parameters are summarized in Table 3.1[5].

The control objective is to perform a lane change maneuver. This can be achieved by stabilizing the origin which is chosen to coincide with the center of the target lane. The vehicle begins in the neighboring lane at the initial condition $x_0 = [-3.7\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]^T$.

[5]SI units are used and all angles are in radians unless otherwise noted.

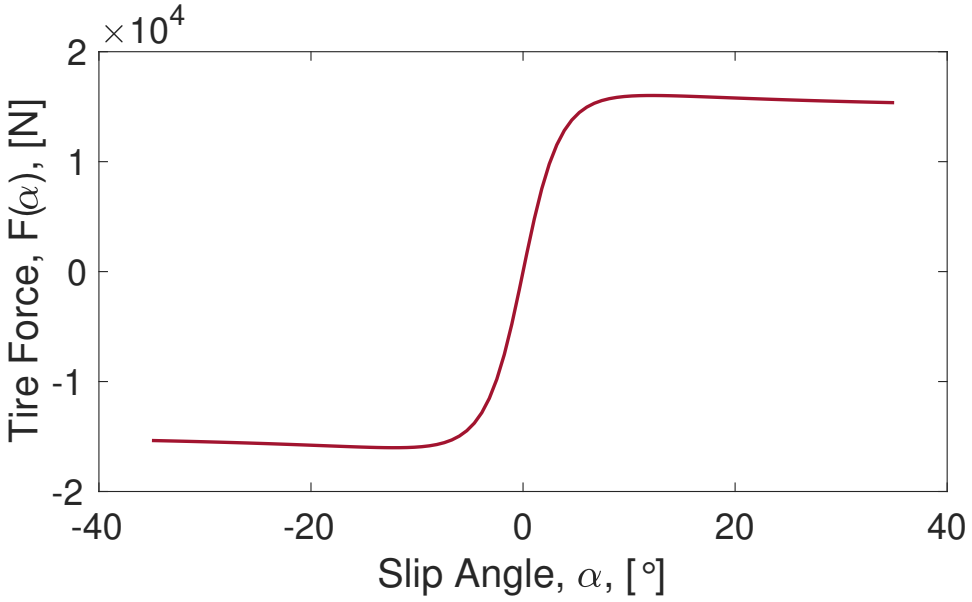

Figure 3.4: The tire force curve

The OCP is

$$\min_{\xi,\mu} \; ||\xi_{30}||^2_{Q_f} + \sum_{i=0}^{29} ||\xi_i||^2_Q + ||\mu_i||^2_R, \tag{3.71a}$$

$$\text{s.t.} \; \xi_{i+1} = f_d(\xi_i, \mu_i, 0), \quad i = 0, \ldots, 29, \tag{3.71b}$$

$$\xi_0 = x, \;\; A_f \xi_{30} \leq b_f, \tag{3.71c}$$

$$x_{lb} \leq \xi_i \leq x_{ub}, \quad i = 1, \ldots, 30, \tag{3.71d}$$

$$u_{lb} \leq \mu_i \leq u_{ub}, \quad i = 0, \ldots, 29, \tag{3.71e}$$

Table 3.1: Bicycle Model Parameters

Name	Symbol	Value
Mass	m	$2041\ kg$
Yaw Inertia	I_{zz}	$4964\ kgm^2$
Front, Rear CG distance	ℓ_f, ℓ_r	$1.56, 1.64\ m$
Coefficient of friction	μ	0.8
Tire parameters	B, C	$12, 1.285$
Lateral Area	A	$7.8\ m^2$
Air Density	ρ	$1.225\ kg/m^2$
Lateral Drag Coefficient	C_d	1.5
Longitudinal Velocity	s	$30\ m/s$

where f_d is the discrete time model of the sedan. The vehicle is subject to state constraints which keep the vehicle on the road and restrict its yaw and steering angles. The state constraints on y, ψ, v and ω are softened using L_1 exact penalty functions which are implemented using slack variables in order to satisfy our smoothness assumptions. The upper and lower bounds are

$$\begin{aligned} x_{ub} &= [0.4 \;\; 7^\circ \;\; 100 \;\; 100 \;\; 35^\circ \;\; 4^\circ], \\ x_{lb} &= -[4.7 \;\; 7^\circ \;\; 100 \;\; 100 \;\; 35^\circ \;\; 4^\circ], \\ u_{ub} &= [1.2 \;\; 0.6], \quad u_{lb} = -[1.2 \;\; 0.6], \end{aligned}$$

and the weighting matrices are $Q = I_{6\times 6}$, and $R = I_{2\times 2}$. The terminal weight is obtained by solving the discrete time algebraic Riccati equation using the linearization about the origin. The matrices encoding the terminal set, A_f and b_f, are computed using the MPT3 toolbox [89]. The natural residual

$$\pi(z, x) = ||z - \Pi_K[z - F(z, x)]||, \tag{3.72}$$

is an error bound [90], i.e., it upper and lower bounds $||z - z^*(x)||$, where $z^*(x) \in S(x)$, and is commonly used as an easily computable surrogate for the error. We use it throughout this section to measure $||z - z^*(x)||$.

Figure 3.5 compares the RTI scheme [74], i.e., a TD-SQP scheme using the GN Hessian approximation with $\ell = 1$, with an LQR controller and the optimal MPC feedback law[6]. The RTI feedback law successfully stabilizes the origin of the plant-optimizer system and outperforms the LQR controller. The state error and the optimization residual both converge to a ball about the origin, demonstrating the expected robustness due to the LISS properties of the combined system (Theorem 3.4). The closed-loop trajectories generated by the RTI controller are nearly indistinguishable from those from the optimal feedback law but are an order of magnitude cheaper to compute. The RTI scheme took $0.067s$ on average and $0.75s$ in the worst case vs. $0.65s$ and $3.2s$ for the optimal feedback law. Closed-loop responses using the RTI controller for 15 different initial position and yaw angle combinations, with all other states are initialized to zero, are shown in Figure 3.7.

Figure 3.6 compares the GN and JN methods with $\ell = 1$ and $\ell = 2$. In the bottom plot of Figure 3.6 note that if $\ell = 2$ iterations are performed, the yaw angle constraint is

[6]All simulations were carried out in MATLAB 2017b on a 2015 Macbook Pro with 16GB of RAM and a 2.8GHz i7 processor. We solved quadratic programs using `quadprog`. The optimal MPC feedback law was computed using `fmincon` with default settings. CASADI [91] was used to compute analytic derivatives which were supplied to the optimization routines.

satisfied exactly, even in the presence of disturbances, as predicted by Theorem 3.5. Also, note that the residuals of the computational subsystem converge faster, for a given number of iterations, if the JN method is used instead of the GN method. This is as expected since the convergence rate of the SQP algorithm is faster when the exact Hessian is used.

3.10 Conclusions

In this chapter, we presented a general framework for the stability analysis of model predictive controllers implemented using time-distributed optimization. When specialized to SQP, our result extends the existing stability analysis of the RTI scheme by explicitly considering inequality constraints, analyzing the effect of performing additional SQP iterations, considering a wider class of Hessian approximations, and proving local input-to-state stability of the closed-loop system. Future work includes analyzing the effect of the sampling rate, applying our framework to globalized SQP methods, and developing numerical methods for estimating the the asymptotic gain functions used in the analysis.

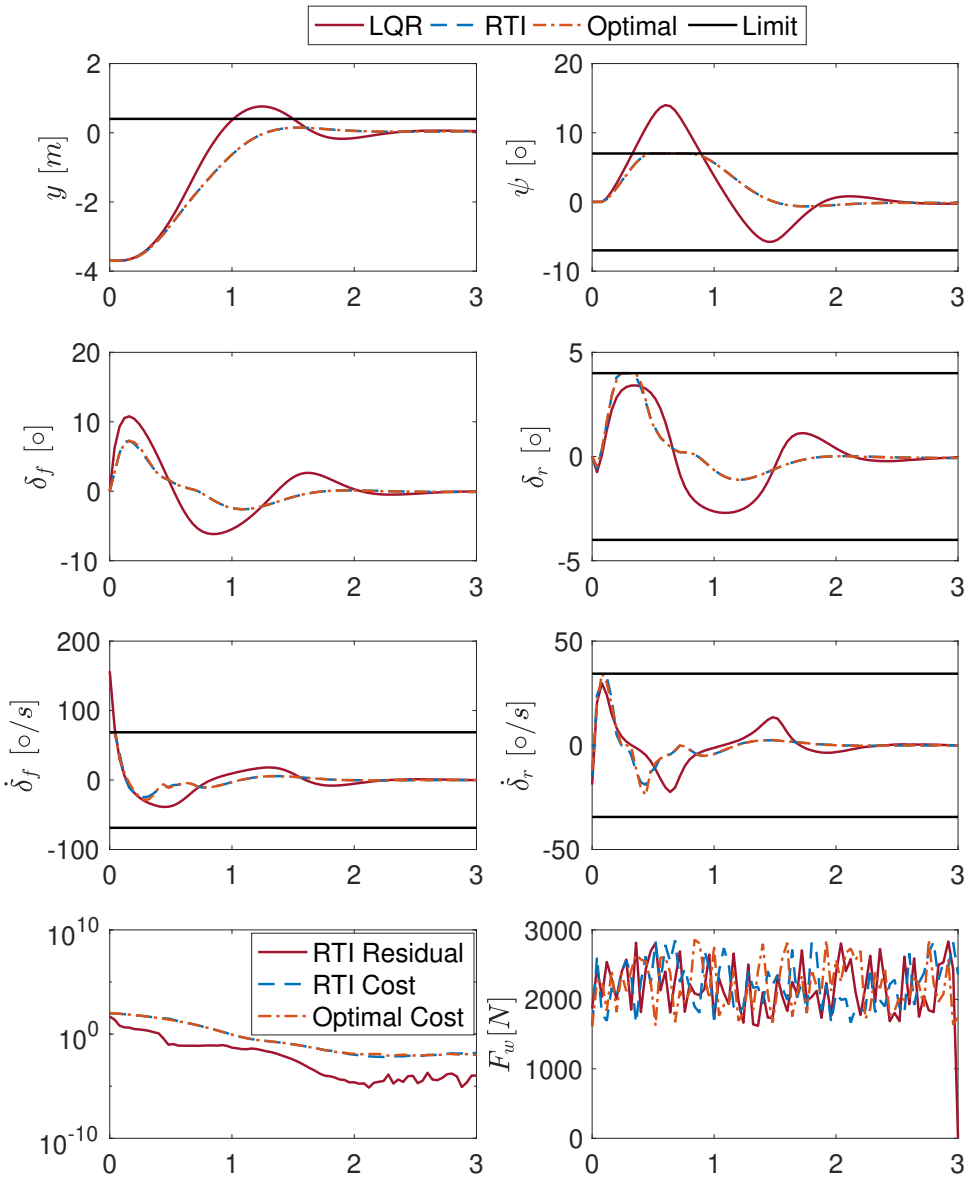

Figure 3.5: A comparison between an LQR controller, the RTI scheme, and the optimal MPC controller.

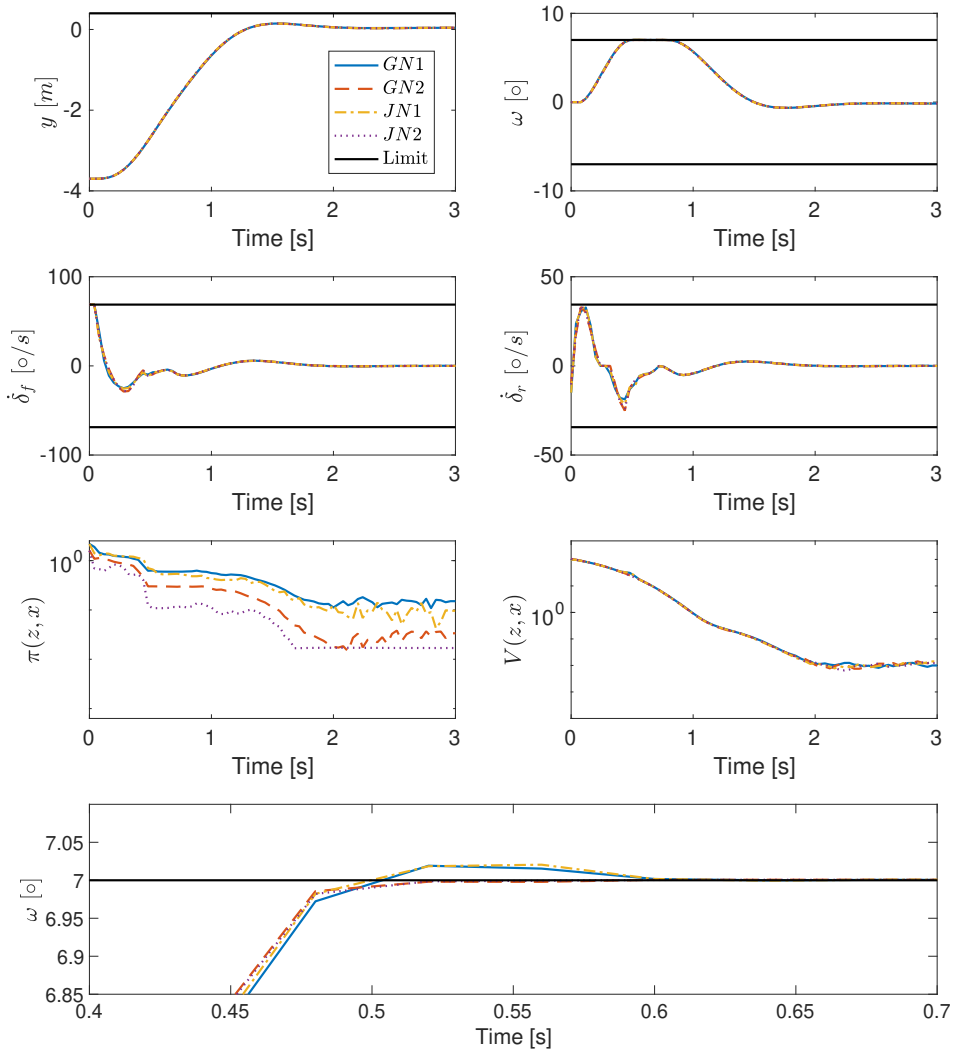

Figure 3.6: A comparison of TD-SQP controllers implemented using the JN and GN methods.

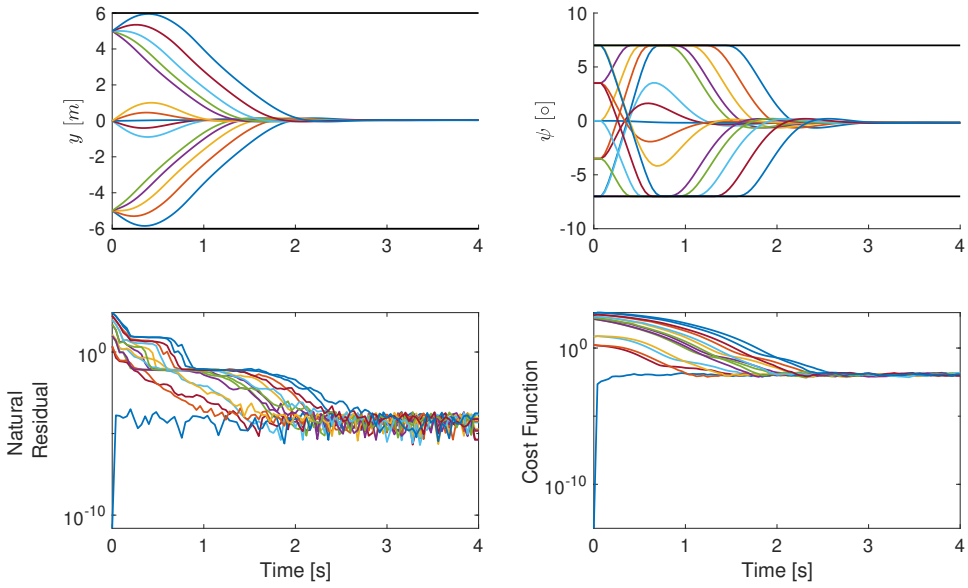

Figure 3.7: Closed-loop responses for an RTI controller for a variety of initial positions and yaw angles.

CHAPTER 4

The Proximally Stabilized Fischer-Burmeister Method for Convex Quadratic Programming

4.1 Introduction

Convex quadratic programs (QPs) arise in many fields including finance [92], control [93], estimation [18], machine learning [94, 95], and signal processing [96]. They also form the basis for sequential quadratic programming (SQP) methods [84] in nonlinear programming and branch and bound methods [97] for mixed integer optimization. Due to their wide applicability, there is a continuing need for increasingly fast and reliable QP solvers driven by emerging applications in artificial intelligence, decision making, embedded systems, and autonomy.

One of the aforementioned emerging technologies is Model Predictive Control (MPC) [1,2], a powerful optimization based control methodology for constrained and/or nonlinear systems. In the case of a linear prediction model, polyhedral constraints and quadratic costs the MPC control law is defined by the solution of a quadratic program (QP). Moreover, QPs are commonly used in methods for nonlinear MPC, e.g., in time-distributed SQP methods as described in Section 3.8. These QPs need to be solved in real-time on embedded systems with limited computing power; warmstarting, where the solver is initialized using a solution from the previous sampling instance, and sparsity exploitation are often necessary to meet real-time requirements. Moreover, reliability, exception safety, robustness to early termination, and infeasibility detection are important concerns due to the safety critical nature of many MPC controllers.

A number of useful algorithms and packages have been developed for solving convex QPs including: Active Set (AS) methods [98–101], Interior Point (IP) methods [52,53,102, 103], first order (FO) methods [51, 104–106], and Dual Newton (DN) methods [107, 108]. AS methods are typically very fast for small to medium problem sizes and can be easily warmstarted using an estimate of the active constraint set. However, they do not scale well,

since they have difficulties exploiting sparsity, and are not robust to early termination, i.e., intermediate iterates do not approximate the solution in a meaningful way. IP methods are fast, efficient, robust to early termination and can solve large, sparse problems efficiently using advanced linear algebra techniques. However, they are notoriously difficult to warmstart. FO methods can be warmstarted easily, and are attractive from a certification standpoint due to their simplicity and the availability of tight complexity bounds. However, they have slow convergence rates relative to AS, IP, and DN methods; they tend to be most effective on small, strongly convex, or simply constrained problems. Dual Newton methods can exploit sparsity and be warmstarted but require restrictive assumptions, e.g., that the QP be strongly convex and the linear independence constraint qualification (LICQ) holds. These assumptions reduce their applicability and may cause robustness issues. Finally, primal-dual Newton-type methods, e.g., FBRS [50], retain the same warmstarting and sparsity exploitation capabilities as DN methods but relax the strong convexity requirement to the weaker strong second order sufficient condition (SSOSC), however they still require the LICQ.

Recently, a hybrid method has been proposed that combines a first order and active set method. QP nonnegative least squares (QPNNLS) [109] uses the proximal point algorithm [42] to construct a sequence of regularized QP subproblems whose solutions converge to the solution of the original problem. Each regularized QP is strictly convex and is efficiently solved using a non-negative least squares based active set method [100]. Since proximal-point subproblems are expensive, QPNNLS heavily exploits warmstarting to reduce the cost of solving subsequent subproblems. Another, very recent, related method is QPALM [110], which is based on the Augmented Lagrangian Method.

In this chapter, we propose the proximally stabilized Fischer-Burmeister method (FBstab). FBstab is a hybrid method in the same vein as QPNNLS, however, we employ a primal-dual version of the proximal point algorithm (i.e., the proximal method of multipliers [111]) rather than a primal version, and solve the proximal subproblems with a primal-dual Newton-type method. Using a Newton-type method, rather than an active set method, allows us to solve the proximal subproblems inexactly in addition to warmstarting them, leading to considerable computational savings. In turn, the proximal regularization yields subproblems which automatically satisfy the regularity conditions needed to guarantee robustness and rapid convergence of the Newton-type method.

The contributions of this chapter are as follows: (i) We describe FBstab, a fast method that is numerically robust, and can handle problems with degenerate solutions. Moreover, it is easy to warmstart making it useful for SQP and parameterized problems e.g., those arising in MPC. (ii) We provide detailed convergence and convergence rate proofs under

only the assumption that the Hessian is positive semidefinite and a solution exists. (iii) When a solution does not exist, we prove that FBstab can detect and certify infeasibility and unboundedness. (iv) We illustrate the performance of FBstab through numerical examples, including experiments on embedded hardware. (v) We demonstrate that, since the linear systems at the core of FBstab are structured similarly to some IP methods, we can exploit existing linear algebra techniques for sparse problems. (vi) We have prepared an open source MATLAB implementation of FBstab[1] and are actively developing a C++ implementation[2].

FBstab significantly extends FBRS [50] by removing the SSOSC and LICQ assumptions, improving its numerical robustness, and enabling infeasibility detection, while still being easy to warmstart and retaining the ability to exploit sparsity. FBstab requires almost no assumptions aside from (non-strict) convexity making it robust and capable of solving any convex QP; only IP methods based on self-dual embedding, e.g., ECOS [52], QPNNLS [109], and the alternating direction method of multipliers (ADMM) [105,106] are as widely applicable. Finally, the use of a primal-dual proximal point method allows FBstab to detect dual infeasibility, in addition to primal infeasibility, unlike QPNNLS which can only detect primal infeasibility.

The layout of the chapter is as follows. First, we describe the algorithm in Section 4.2. Next, we analyze its behaviour when a solution of the QP exists in in Section 4.3 and when one does not in Section 4.4. Finally, we perform some numerical experiments illustrating the practical performance of FBstab in Section 4.5 before ending with some concluding remarks.

4.2 Description of the Algorithm

We consider convex QPs of the following form,

$$\min_{w} \quad \frac{1}{2} w^T H w + f^T w, \tag{4.1a}$$

$$\text{s.t} \quad Gw = h, \tag{4.1b}$$

$$Aw \leq b, \tag{4.1c}$$

where $H \in \mathbb{R}^{n \times n}$, $f \in \mathbb{R}^n$, $w \in \mathbb{R}^n$, $G \in \mathbb{R}^{m \times n}$, $h \in \mathbb{R}^m$, $A \in \mathbb{R}^{q \times n}$, and $b \in \mathbb{R}^q$. We make no assumptions about the problem data aside from the following[3].

[1] https://github.com/dliaomcp/fbstab-matlab.git

[2] https://github.com/dliaomcp/fbstab.git

[3] In particular, we require neither a constraint qualification (of any kind) nor existence of an interior point.

Assumption 4.1. *The Hessian matrix H is symmetric and positive semidefinite.*

The Lagrangian for this problem is given by

$$L(w, \lambda, v) = \frac{1}{2} w^T H w + f^T w + \lambda^T (Gw - h) + v^T (Aw - b),$$

where $\lambda \in \mathbb{R}^m$ and $v \in \mathbb{R}^q$ are dual variables, and its dual is

$$\min_{w,\lambda,v} \quad \frac{1}{2} w^T H w + b^T v + h^T \lambda, \tag{4.2a}$$

$$\text{s.t} \quad Hw + f + G^T \lambda + A^T v = 0, \tag{4.2b}$$

$$v \geq 0. \tag{4.2c}$$

We will use $z = (w, \lambda, v) \in \mathbb{R}^l$ to denote the primal-dual triple. The Karush-Kuhn-Tucker (KKT) conditions for the problem are

$$\nabla_w L(w, \lambda, v) = 0, \tag{4.3a}$$

$$Gw = h, \tag{4.3b}$$

$$Aw - b \leq 0,\ v \geq 0,\ v^T (Aw - b) = 0. \tag{4.3c}$$

Any vector satisfying (4.3) is called a critical point. If the feasible set

$$\Omega = \{w \mid Aw \leq b,\ Gw = h\}, \tag{4.4}$$

is nonempty, then the KKT conditions are necessary and sufficient for global optimality [112].

The main idea of the FBstab algorithm is to regularize the original problem, solve it using a semismooth Newton-type method, then use the proximal point algorithm to iteratively refine the solution. The regularization ensures that each proximal subproblem has a unique primal-dual solution and satisfies the regularity conditions needed to ensure fast convergence of the inner Newton-type solver. It also removes the need for any constraint qualifications by suitably regularizing the dual problem. Moreover, semismooth Newton-type methods can be warmstarted and terminated early. As a result, each proximal subproblem can be solved approximately and warmstarted with the solution of the previous one. This makes FBstab efficient, often requiring only one to two Newton iterations to solve each subproblem.

4.2.1 Outer Proximal Point Iterations

The primary iterative procedure in FBstab is an instance of the proximal point algorithm (PPA) [42, 111] which finds zeros of monotone inclusions of the form

$$T(z) \ni 0, \tag{4.5}$$

where $T : \mathcal{H} \rightrightarrows \mathcal{H}$ for some Hilbert space $\mathcal{H}$, by generating a sequence $\{z_k\}$ using the rule

$$z_{k+1} = P_k(z_k), \quad P_k = (I + \sigma_k^{-1} T)^{-1}, \tag{4.6}$$

where $\{\sigma_k\}$ are positive numbers[4]. If T is maximal monotone then the PPA converges to an element of the set $T^{-1}(0)$ if it is nonempty [42].

The KKT conditions of (4.1) can be rewritten as the following variational inequality (VI)

$$\nabla_w L(w, \lambda, v) = 0, \tag{4.7a}$$

$$h - Gw = 0, \tag{4.7b}$$

$$b - Aw + \mathcal{N}_+(v) \ni 0, \tag{4.7c}$$

where $\mathcal{N}_+$ is the normal cone mapping of the non-negative orthant. Thus (4.7) can be compactly expressed as,

$$T(z) = Kz + r + \mathcal{N}_\Gamma(z) \ni 0, \tag{4.8}$$

where $\Gamma = \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^q_{\geq 0}$,

$$K = \begin{bmatrix} H & G^T & A^T \\ -G & 0 & 0 \\ -A & 0 & 0 \end{bmatrix}, p = \begin{bmatrix} f \\ h \\ b \end{bmatrix}, \text{ and } z = \begin{bmatrix} z \\ \lambda \\ v \end{bmatrix}. \tag{4.9}$$

The outermost loop of FBstab is simply (4.6) applied to (4.8). The following proposition shows that (4.8) is indeed maximal monotone as required by the PPA.

Proposition 4.1. *The variational inequality* (4.8) *has the following properties: (i) It is maximal monotone. (ii) If nonempty, its solution set,* $T^{-1}(0)$*, is closed and convex.*

Proof. (i): The base mapping (4.9) is maximal monotone since it is single valued, affine and $K^T + K \succeq 0$, where K is defined in (4.9). Variational inequalities of the form (4.8)

[4]There is significant freedom in how $\{\sigma_k\}$ is chosen. This is discussed further in Section 4.2.5.

are maximal monotone if the single valued portion F is monotone [113]. (ii): See [114]. □

Practical algorithms need a stopping criterion. FBstab uses the condition

$$||\pi(x)|| \leq \tau_a + \tau_r(||p|| + 1) \tag{4.10}$$

where p is defined in (4.9), $\tau_a \geq 0$ and $\tau_r \geq 0$ are absolute and relative tolerances, and

$$\pi(z) = z - \Pi_\Gamma \left[z - (Kz + p) \right], \tag{4.11}$$

where Π_Γ denotes euclidean projection onto Γ. The natural residual function (4.11) is a local error bound [90, Theorem 18], i.e[5]., $\text{dist}(x, T^{-1}(0)) = \mathcal{O}(||\pi(x)||)$.[6]

4.2.2 Inner Semismooth Newton Solver

Evaluating the proximal operator is the most computationally expensive step in the PPA and must be done efficiently to produce a practical algorithm. To evaluate $P_k(x_k)$ we must find an x satisfying

$$T_\sigma(z, k) = Kz + p + \sigma_k(z - z_k) + \mathcal{N}_\Gamma(z) \ni 0. \tag{4.12}$$

This is itself a variational inequality which is in general expensive to solve. However, due to the regularization term $\sigma_k(z - z_k)$, (4.12) is guaranteed to have a unique solution and to satisfy certain useful regularity properties (Theorem 4.1). We can exploit these properties to construct a Newton-type method for the subproblems with a quadratic rate of convergence (Theorem 4.2). Moreover, the PPA allows for approximate evaluation of P_k and we warmstart the inner Newton-type solver at each iteration. Taken together, these measures allow FBstab to evaluate the proximal operator efficiently.

The subproblem solver works by applying Newton's method to a semismooth[7]reformulation of (4.12). We construct the reformulation using a so-called nonlinear complementarity problem (NCP) function [115]. An NCP function $\phi : \mathbb{R}^2 \to \mathbb{R}$ has the property that

$$\phi(a_1, a_2) = 0 \quad \Leftrightarrow \quad a_1 \geq 0,\ a_2 \geq 0,\ a_1 a_2 = 0.$$

[5]For a closed set C, $\text{dist}(x, C) = \inf_{\bar{x}}\{x - \bar{x} \mid \bar{x} \in C\}$.
[6]See [41, A.2] or [49, A.2] for more details on O notation.
[7]See Chapter 2 for more details on semismooth functions.

In this chapter, we use the penalized Fischer-Burmeister (PFB) function [47],

$$\phi(a_1, a_2) = \alpha\left(a_1 + a_2 - \sqrt{a_1^2 + a_2^2}\right) + (1-\alpha)a_1^+ a_2^+,$$

where $\alpha \in (0,1)$ is fixed and $(\cdot)^+$ denotes projection onto the nonnegative orthant. The PFB function is similar to the Fischer-Burmeister [116] function but has better theoretical and numerical properties [47]. Using this NCP function we can construct the following mapping,

$$R_k(z) = R(z, z_k, \sigma_k) = \begin{bmatrix} \nabla_w L(w, \lambda, v) + \sigma_k(w - w_k) \\ h - Gw + \sigma_k(\lambda - \lambda_k) \\ \phi(y + \sigma_k(v - v_k), v) \end{bmatrix}, \quad y = b - Aw, \tag{4.13a}$$

where the NCP function is applied elementwise.

The inner solver evaluates P_k by solving the rootfinding problem $R_k(z) = 0$ using a damped semismooth Newton's method. We use the following generalized derivative, known as the the C-subdifferential [45],

$$\bar{\partial} G = \partial G_1 \times \partial G_2 \times ... \times \partial G_M, \tag{4.14}$$

where ∂G_i are the Clarke generalized Jacobians [36] of the components mappings of G. The inner iterative scheme[8], is then

$$z_{i+1|k} = z_{i|k} - t_{i|k} V^{-1} R_k(z_{i|k}), \quad V \in \bar{\partial} R_k(z_{i|k}), \tag{4.15}$$

where $t_{i|k} \in (0,1]$ is a step length that enforces global convergence and is chosen using a backtracking linesearch on the merit function

$$\theta_k(z) = \frac{1}{2}||R_k(z)||_2^2. \tag{4.16}$$

The properties of R_k and θ_k are summarized in Proposition 4.2.

Proposition 4.2. *The function, R_k in* (4.13)*, and its associated merit function, θ_k, have the following properties:*

1) R_k is strongly semismooth on $\mathbb{R}^l$.

2) $R(z_k^*, z_k, \sigma_k) = 0$ if and only if $z_k^* = P_k(z_k)$. Further, z_k^* is unique and exists irrespective of the problem data.

[8] i and k are used for inner and outer iterations respectively.

3) $||R_k(\cdot)||$ *is a global error bound, i.e., there exists* $\tau > 0$ *such that* $||z - z_k^*|| \leq \tau ||R_k(z)||$

4) θ_k *is continuously differentiable and, for any* $V \in \bar{\partial} R_k(z)$*, its gradient is* $\nabla \theta_k(z) = V^T R_k(z)$.

Proof. 1) The PFB function is strongly semismooth [47, Proposition 2.1] and is composed with affine functions to form R. Strong semismoothness of R then follows from the composition rules for semismooth functions, see, e.g., [41, Propositions 1.73 and 1.74].

2) The VI (4.12) is defined by the sum of a monotone and strongly monotone operator and is thus itself strongly monotone. Strongly monotone operators always have a unique zero [114]. The zeros of R exactly coincide with those of (4.12) by the properties of NCP functions.

3) Let $\mathcal{K} = \{z \mid (K + \sigma I)z + \mathcal{N}_\infty(z) \ni 0\}$ denote the "kernel" of (4.12), where K is defined in (4.9), and $\mathcal{N}_\infty(x)$ is the normal cone of Γ_∞, the recession cone of Γ. Since Γ is a convex cone, $\Gamma_\infty = \Gamma$. Theorem 20 of [90] states that the norm of the natural residual function

$$\pi_k(z) = z - \Pi_\Gamma \left[z - (Kz + p + \sigma_k(x - x_k)) \right], \tag{4.17}$$

is a global error bound if $\mathcal{K} = \{0\}$. Its clear that $z = 0$ satisfies $(K + \sigma I)z + N_\infty(z) \ni 0$ and because $K + \sigma I$ is strongly monotone the solution must be unique. Thus, applying [90, Theorem 20] there exists $\tau_1 > 0$ such that $||z - z_k^*|| \leq \tau_1 ||\pi_k(z)||$. The equivalence of $||\pi_k(z)||$ and $||R_k(z)||$, when used as an error bound, can then be established using the same arguments as [47, Theorem 3.11]; we omit the details for brevity.

4) See [47, Theorem 3.2] or [50, Proposition 2]. □

The inner solver uses the following stopping condition

$$||R_k(z)|| \leq \epsilon_k \min\{1, z - z_k\}, \tag{4.18}$$

where ϵ_k is controlled in the outer layer. How ϵ_k is chosen is discussed further in Section 4.2.5. Next, we show that the matrix V is always invertible and thus the iteration (4.2.2) is well defined. In particular, all elements of $\bar{\partial} R_k(z^*)$, where z^* is the root, are non-singular which leads to quadratic convergence of (4.15).

4.2.3 The Newton Step System

The computational core of FBstab is the Newton step system

$$V\Delta z = r, \quad V \in \bar{\partial} R(z, \bar{z}, \sigma), \tag{4.19}$$

which needs to be solved for Δz for various values of $z, \bar{z}, \sigma$ and r. We begin with the following proposition, that establishes some properties of the C-subdifferential.

Proposition 4.3. *For any $z, \bar{z} \in \mathbb{R}^l$, $\sigma > 0$, all $V \in \bar{\partial} R(z, \bar{z}, \sigma)$ are of the form*

$$V = \begin{bmatrix} H_\sigma & G^T & A^T \\ -G & \sigma I & 0 \\ -CA & 0 & D \end{bmatrix}, \tag{4.20}$$

where $H_\sigma = H + \sigma I$, and $C = diag(\gamma_j)$, $D = diag(\mu_j + \sigma\gamma_j)$, are diagonal matrices with entries $(\gamma_j, \mu_j) \in \partial\phi(y_j, v_j)$.

Proof. The proof follows [47, Proposition 2.3]. By definition

$$\bar{\partial} R(z, \bar{z}, \sigma) = \partial R_1(z, \bar{z}, \sigma) \times \ ... \ \times \partial R_l(x, \bar{z}, \sigma), \tag{4.21}$$

thus we need only to characterize the generalized gradients. The first two blocks are continuously differentiable so $\partial R_i(z, \bar{z}, \sigma) = \{\nabla R_i(z, \bar{z}, \sigma)^T\}$, $i = 1, \ldots, n + m$. The last block satisfies

$$V_i = \begin{bmatrix} -\gamma_i(z) A_i & 0 & (\mu_i(z) + \sigma\gamma_i(z)) e_i \end{bmatrix}, \tag{4.22}$$

where e_i are rows of identity, by [36, Proposition 2.1 and Theorem 2.3.9]. □

Explicit expressions for the generalized gradient of $\phi(a_1, a_2)$ are given by [47, Proposition 2.1]

$$\partial\phi(a_1, a_2) = (\gamma_i, \mu_i) = \begin{cases} \alpha(1 - \frac{a_1}{r}, 1 - \frac{a_2}{r}) + (1 - \alpha)(a_2^+ \partial a_1^+, a_1^+ \partial a_2^+) & \text{if } r \neq 0 \\ \alpha(1 - \eta, 1 - \zeta) & \text{if } r = 0 \end{cases}$$

where $r = \sqrt{a_1^2 + a_2^2}$, η and ζ are arbitrary numbers satisfying $\eta^2 + \zeta^2 = 1$, and

$$\partial a^+ \in \begin{cases} \{1\} & \text{if } a > 0 \\ [0,1] & \text{if } a = 0 \\ \{0\} & \text{if } a < 0. \end{cases} \tag{4.23}$$

Next, we consider the regularity properties of V which are critical to the behaviour of (4.15). Due to the stabilizing effect of the outer proximal point algorithm all elements of $\bar{\partial} R(z, \bar{z}, \sigma)$ are nonsingular, a property we will refer to as C-regularity.

Definition 4.1. *A mapping $G : \mathbb{R}^N \to \mathbb{R}^N$ is C-regular at a point $x \in \mathbb{R}^N$ if it is semismooth at x and all $V \in \bar{\partial} G(x)$ are non-singular.*

The C-regularity of R_k guarantees that the inner iterations are globally well defined and that the iteration (4.15) will converge at a quadratic rate to the unique solution of (4.12) (Theorem 4.2).

Theorem 4.1 (Regularity of the C-subdifferential)**.** *$R(z, z_k, \sigma)$ is C-regular for any $z, \bar{z} \in \mathbb{R}^l$ and $\sigma > 0$.*

Proof. For any $V \in \bar{\partial} R(z, \bar{z}, \sigma)$, the Newton step system (4.15) has the form

$$\begin{bmatrix} H_\sigma & G^T & A^T \\ -G & \sigma I & 0 \\ -CA & 0 & D \end{bmatrix} \begin{bmatrix} \Delta w \\ \Delta \lambda \\ \Delta v \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix}, \tag{4.24}$$

where $R(z, \bar{z}, \sigma) = -[r_1^T \;\; r_2^T \;\; r_3^T]^T$ as in (4.13). For all $j \in \{1, ...q\}$ we have that $\mu_j \geq 0$, $\gamma_j \geq 0$ and $(\mu_j, \gamma_j) \neq 0$. Thus $D_{jj} = \mu_j + \sigma\gamma_j > 0$ implying $D \succ 0$. Since $D \succ 0$ we can eliminate the third row of (4.24) algebraically and negate the second leading to the following pair of linear systems of equations

$$\begin{bmatrix} E & G^T \\ G & -\sigma I \end{bmatrix} \begin{bmatrix} \Delta w \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} r_1 - A^T D^{-1} r_3 \\ -r_2 \end{bmatrix}, \tag{4.25a}$$

$$D\Delta v = r_3 + CA\Delta w. \tag{4.25b}$$

The matrices $E = H_\sigma + A^T C D^{-1} A$ and σI are positive definite so the block 2×2 matrix in (4.25) is symmetric quasidefinite and thus invertible [117]. As a result, (4.24) has a unique solution, implying that V is nonsingular. □

4.2.4 Infeasibility Detection

When either (4.1) or (4.2) is infeasible FBstab is able to detect it, allowing for a graceful exit. This feature is also important in e.g., in branch and bound algorithms for mixed integer QPs [97]. Recall the following infeasibility conditions.

Proposition 4.4. *(Infeasibility conditions)*
Dual infeasibility: Suppose there exists a vector $w \in \mathbb{R}^n$ satisfying $Hw = 0$, $Aw \leq 0$, $Gw = 0$, and $f^T w < 0$. Then (4.2) is infeasible.
Primal infeasibility: Suppose there exits a vector (λ, v) such that $G^T\lambda + A^T v = 0$ and $\lambda^T h + v_+^T b < 0$, where v_+ is the projection of v onto the nonnegative orthant. Then the feasible set of (4.1) is empty.

Proof. See e.g., [118, Proposition 1]. □

Any vector satisfying the conditions of Proposition 4.4 is a certificate of primal or dual infeasibility. When the primal is feasible, dual infeasibility is the same as the primal problem being unbounded below.

For infeasibility detection we use the proximal increment

$$\Delta z = P_k(z) - z = (\Delta w, \Delta\lambda, \Delta v). \tag{4.26}$$

FBstab detects dual infeasibility using a relative tolerance $\tau_{inf} > 0$ and the following criteria

$$||H\Delta w||_\infty \leq \tau_{\text{inf}}||\Delta w||_\infty, \tag{4.27}$$
$$f^T\Delta w < 0, \tag{4.28}$$
$$\max(A\Delta w) \leq 0, \tag{4.29}$$
$$\text{and } ||G\Delta w||_\infty \leq \tau_{inf}||\Delta w||_\infty. \tag{4.30}$$

For primal infeasibility the criteria are

$$||A^T v + G^T\lambda||_\infty \leq \tau_{inf}(||\Delta v|| + ||\Delta\lambda||), \text{ and } \Delta v^T b + \lambda^T h < 0. \tag{4.31}$$

In Section 4.4 we prove that Δz converges to suitable certificates when either (4.1) or (4.2) are infeasible.

4.2.5 Summary of the Algorithm

The FBstab algorithm is summarized in Algorithms 1-4. Algorithm 1 implements the proximal point algorithm as discussed in Section 4.2.1. Algorithm 3 evaluates the proximal operator as discussed in Section 4.2.2 and Algorithm 4 checks for infeasibility as discussed in Section 4.2.4.

There is significant freedom in how to choose the regularization sequence $\{\sigma_k\}$ and the subproblem tolerance sequence $\{\epsilon_k\}$. Our choices are summarized in Algorithm 2. In a practical implementation, it is sensible to have bounds on the subproblem solver tolerance ϵ to ensure the desired tolerance is achievable and on the regularization strength σ to control the conditioning of (4.19). These are incorporated into FBstab through the choices of $\epsilon_{max}, \epsilon_{min}, \sigma_{max}$, and σ_{min}. In exact arithmetic, Algorithm 3 will always succeed (Theorem 4.2). However, in practice it can fail, the typical cause is inability to decrease the merit function due to ill-conditioning in (4.19). In this case, FBstab increases the regularization strength, which improves the conditioning of (4.2.3), and tries again. On the other hand, if the `EvalProx` routine is successful, then both σ and ϵ are reduced to accelerate convergence.

Algorithm 1 The FBstab algorithm

Inputs: $\sigma_0, \tau_r, \tau_a > 0$, Initial guess $z_0 = (w_0, \lambda_0, v_0)$
Outputs: Primal-dual solution, z^*, or infeasibility status and certificate Δz^*

procedure FBSTAB
  $z \leftarrow z_0$, $k \leftarrow 0$, $\epsilon \leftarrow \min(||\pi(z_0)||, 1)$
  repeat
    $(z^+, \texttt{flag}) \leftarrow$ EVALPROX(z, ϵ, σ)
    $(\sigma, \epsilon) \leftarrow$ UPDATE$(\sigma, \epsilon, ||\pi(z^+)||, \texttt{flag})$
    CHECKFEASIBILITY$(z^+ - z)$
    if `flag == success` **then**
      $z \leftarrow z^+$
    end if
    $\epsilon \leftarrow ||\pi(z)||$, $k \leftarrow k + 1$
  until $\epsilon \leq (||p|| + 1)\tau_r + \tau_a$ ▷ p defined in (4.9)
  Stop: $z^* \leftarrow z$ is optimal
end procedure

Remark 4.1. *Both semismooth Newton methods [116] and proximal point algorithm [42] have been extensively studied in the literature. Our main contribution is the novel synergistic combination of the two methods with infeasibility detection techniques originally derived for ADMM [118] to produce a method that is very general, theoretically justified, and effective in practice.*

Algorithm 2 Tolerance and regularization updating

Inputs: $\sigma_{max}, \sigma_{min}, \epsilon_{max}, \epsilon_{min} > 0$, $\kappa_\sigma, \kappa_\epsilon \in (0,1)$

procedure UPDATE($\sigma, \epsilon, \epsilon_0$, `flag`)
  if `flag == success` **then**
    $\sigma \leftarrow \sigma\kappa_\sigma, \;\; \epsilon \leftarrow \min(\epsilon\kappa_\epsilon, \epsilon_0)$
  else
    $\sigma \leftarrow \sigma/\kappa_\sigma, \;\; \epsilon \leftarrow \epsilon/\kappa_\epsilon$
  end if
  $\sigma \leftarrow \Pi_{[\sigma_{min},\sigma_{max}]}(\sigma)$
  $\epsilon \leftarrow \Pi_{[\epsilon_{min},\epsilon_{max}]}(\epsilon)$
  return (σ, ϵ)
end procedure

Algorithm 3 Evaluate the proximal operator

Inputs: $\beta \in (0,1), \eta \in (0, 0.5), k_{max} > 0$

procedure EVALPROX($\bar{z}, \epsilon, \sigma$)
  $z \leftarrow \bar{z}, k \leftarrow 0$
  repeat
    Compute $V \in \bar{\partial} R(z, \bar{z}, \sigma)$, see Section 4.2.3
    Solve $V\Delta z = -R(z, \bar{z}, \sigma)$ for Δz
    $t \leftarrow 1, \; k \leftarrow k + 1$
    while $\theta(z + t\Delta z) \geq \theta(z) + \eta t \nabla\theta(z)$ **do**
      $t \leftarrow \beta t$
    end while
    $z \leftarrow z + t\Delta z$
  until $||R(z, \bar{z}, \sigma)|| \leq \epsilon \min\{1, z - \bar{z}\}$ or $k \geq k_{max}$
  `flag` $\leftarrow (k \leq k_{max}$ `? success : failure`$)$
  return (z, `flag`)
end procedure

Remark 4.2. *This method could theoretically be extended to more general smooth convex programs. However, then the curvature of the constraint Hessians would enter into the subproblems and we would need to maintain positivity of the dual variables during the Newton iterations to ensure non-singularity of the generalized Jacobians. This would make warmstarting the algorithm difficult; as a result we have elected to focus on QPs. Moreover, due to the polyhedrality of the solution set, we are able to establish a stronger convergence rate result for QPs than for more general convex programs (see Theorem 4.3).*

Algorithm 4 Check for infeasibility

Inputs: $\tau_{inf} > 0$

procedure CHECKFEASIBILITY(Δz)
  $(\Delta w, \Delta\lambda, \Delta v) \leftarrow \Delta z$
  if $||H\Delta w||_\infty \leq \tau_{\text{inf}}||\Delta w||_\infty$, $f^T\Delta w < 0$,
   $\max(A\Delta w) \leq 0$, $||G\Delta w||_\infty \leq \tau||\Delta w||_\infty$ **then**
    $\Delta z^* \leftarrow (\Delta w, \Delta\lambda, \Delta v)$
    Stop: Δw^* certifies dual infeasibility
  end if
  if $||A^T v + G^T\lambda||_\infty \leq \tau(||\Delta v|| + ||\Delta\lambda||)$, $\Delta v^T b + \lambda^T h < 0$ **then**
    $\Delta z^* \leftarrow (\Delta w, \Delta\lambda, \Delta v)$
    Stop: $(\Delta\lambda^*, \Delta v^*)$ certifies primal infeasibility
  end if
end procedure

4.3 Convergence Analysis

In this section, we analyze the convergence properties of FBstab when (4.1) has a primal-dual solution. We address the case when $T^{-1}(0) = \emptyset$ in Section 4.4. First, we address the convergence of the inner Newton-type solver.

Theorem 4.2 (Inner solver convergence)**.** *Consider an arbitrary but fixed iteration k of the outer proximal point algorithm. Suppose $z_0 = z_{0|k} = z_k \in \mathbb{R}^l$ and let the sequence $\{z_i\}$ be generated by the* `EvalProx` *procedure in Algorithm 3. Then:*

i. The sequence $\{z_i\}$ is well defined and converges to the unique point z^* satisfying $x^* = P_k(z_0)$.

ii. The asymptotic rate of convergence is quadratic i.e.,

$$||z^* - z_{i+1}|| = \mathcal{O}(||z^* - z_i||^2) \text{ as } i \to \infty.$$

Proof. See [45, 47, 119] or [50, Section VII]. □

Having established the convergence of the inner Newton-type solver, we can prove convergence of the outer proximal loop and thus of FBstab.

Theorem 4.3 (Convergence of FBstab)**.** *Let $z_0 \in \mathbb{R}^l$ be arbitrary, suppose $T^{-1}(0) = (Kz + p + N_\Gamma)^{-1}(0)$ is nonempty, and let $\{z_k\}$ be generated by FBstab. Moreover, let either $\kappa_\epsilon < \kappa_\delta$ or $\sigma_{min} > 0$ and $\epsilon_{min} = 0$. Then $\{z_k\} \to z^* \in T^{-1}(0)$ as $z \to \infty$.*

Furthermore, the convergence rate is at least linear and if, in addition, $\sigma_k \to 0$ as $k \to \infty$ then the convergence rate is superlinear.

Proof. FBstab is an instance of the proximal point algorithm so we can employ [120, Theorem 2.1] to establish both convergence and the convergence rate. The error bound condition needed by [120, Theorem 2.1] ((A_r') in [120]) is

$$\operatorname{dist}(0, T_\sigma(z_{k+1}, k)) \leq \epsilon_k \min\{1, ||z_{k+1} - z_k||\} \tag{4.32}$$

where T_σ is defined in (4.12), and $\sum_{k=0}^{\infty} \epsilon_k/\sigma_k < \infty$. The inequality (4.32) is enforced by construction (Line 11 of Algorithm 3) since $||R_k(z)||$ is an error bound for the subproblems by Proposition 4.2. Note that in exact arithmetic the inner solver always succeeds. Thus, in the case where $\kappa_\epsilon < \kappa_\sigma$, we have that $\delta_k = \epsilon_k/\sigma_k$ satisfies, $\delta_k \leq (\epsilon_0/\sigma_0)(\kappa_\epsilon/\kappa_\sigma)^k$, thus $\kappa_\delta = \kappa_\epsilon/\kappa_\sigma < 1$ which implies that $\sum_0^\infty \delta_k < \infty$. In the case where $\sigma_{min} > 0$ and $\epsilon_{min} = 0$, as $k \to \infty$ we have that $\delta_k \leq \epsilon_0(\kappa_\epsilon)^k/\sigma_{min}$. Since $\kappa_\epsilon < 1$ this implies that $\sum_0^\infty \delta_k < \infty$. It remains to show that there exists $a, \epsilon > 0$ such that for all $s \in \epsilon\mathbb{B}$

$$\operatorname{dist}(z, T^{-1}(0)) \leq a||s||, \ \forall z \in T^{-1}(s), \tag{4.33}$$

or equivalently (see e.g., [33, Section 3D])

$$T^{-1}(s) \subseteq T^{-1}(0) + a||s||\mathbb{B}, \tag{4.34}$$

where $\mathbb{B}$ is the unit ball. This property actually holds globally because $T(z) \ni 0$ is a polyhedral variational inequality, see [33, Section 3D]. Thus we can invoke [120, Theorem 2.1] to complete the proof. □

Finally, we state the following theorem which provides rigorous justification for the subproblem warmstarting strategy employed in FBstab.

Theorem 4.4 (Lipschitz continuity of subproblems)**.** *Let the sequence $\{z_k\}$ be generated by FBstab with z_0 arbitrary. Then the proximal operator is Lipschitz continuous, i.e., at any iteration k the proximal operator P_k satisfies*

$$||P_k(z_k) - P_{k-1}(z_{k-1})|| \leq \eta||z_k - z_{k-1}||, \tag{4.35}$$

$$\eta^{-1} = \lambda_{min}\left(\frac{K^T + K + 2\sigma_k I}{2\sigma_k}\right), \tag{4.36}$$

where $\lambda_{min}(\cdot)$ designates the smallest eigenvalue of a symmetric matrix and K is defined in (4.9).

Proof. The variational inequality (4.12) can be written as $F(z)/\sigma_k + z + N_\Gamma(z) \ni z_k$ which is a parameterized variational inequality with z_k as the parameter. Its strong monotonicity constant is $\eta > 0$. The result then follows from [33, Theorem 2F.6]. □

Each proximal subproblem computes $P_k(z_k)$ by solving $R_k(z) = 0$ starting from z_k as an initial guess. Theorem 4.4 implies that eventually $||P_k(z_k) - z_k||$ will become sufficiently small so that the quadratic convergence rate of Theorem 4.2 holds immediately and the semismooth method converges rapidly. This happens because,

$$\begin{aligned} ||P_k(z_k) - z_k|| &= |||[P_k(z_k) - P_{k-1}(z_{k-1})] - [z_k - P_{k-1}(z_{k-1})]||, \\ &\leq ||z_k - P_{k-1}(z_{k-1})|| + ||P_k(z_k) - P_{k-1}(z_{k-1})||, \\ &\leq ||z_k - P_{k-1}(z_{k-1})|| + \eta||z_{k-1} - z_{k-2}||, \end{aligned}$$

and, since the algorithm is converging, $||z_{k-1} - z_{k-2}|| \to 0$ and $||z_k - P_{k-1}(z_{k-1})|| \to 0$ as $k \to \infty$. We observe this behaviour in practice, typically after the first or second proximal iteration each subsequent proximal subproblem takes only one or two Newton iterations to converge.

4.4 Infeasibility Detection Analysis

In this section, following [118] we the behaviour of FBstab in the case where either the QP (4.1) or its dual (4.2) are infeasible. These results are not specific to FBstab and hold whenever the proximal point algorithm is used to solve (4.7). First we review the limiting behaviour of the proximal point algorithm. The following lemma summarizes some results for averaged nonexpansive operators, a class which includes the proximal operator.

Lemma 4.1. *Let $T : \mathcal{H} \mapsto \mathcal{H}$ be an averaged nonexpansive operator over a Hilbert space $\mathcal{H}$. In addition, suppose z_k is generated by $z_k = T^k(z_0)$, $z_0 \in \mathcal{H}$, define $\delta z_k = z_{k+1} - z_k$, and let δz be the projection of 0 onto $\mathrm{cl\ range}\ (T - I_d)$ where cl denotes the closure of a set and I_d denotes the identity operator. Then as $k \to \infty$ we have that:*

i. $\frac{1}{k} z_k \to \delta z$

ii. $\delta z_k \to \delta z$

iii. If $\text{Fix}\, T \neq \emptyset$ *then* $z_k \to z^* \in \text{Fix}\, T$*, where* $\text{Fix}\, T$ *denotes the fixed points of* T*.*

Proof. (i): [121, Corrolary 2]. (ii), (iii): [122, Fact 3.2] . □

An immediate corollary of this is that δz in Lemma 4.1 satisfies $\delta z = 0$ if $\text{Fix}\, T \neq \emptyset$. The following proposition applies Lemma 4.1 to our specific situation.

Proposition 4.5. *Let the sequence* $\{z_k\} = \{(w_k, \lambda_k, v_k)\}$ *be generated by the proximal point algorithm, suppose that* $\sigma_k \to \sigma > 0$ *as* $k \to \infty$*,* $\epsilon_{min} = 0$*, and define* $\delta z_k = z_k - z_{k-1}$*. Then there exists* $\delta z = (\delta w, \delta\lambda, \delta v) \in \mathbb{R}^l$ *such that* $(\delta w_k, \delta\lambda_k, \delta v_k) \to (\delta w, \delta\lambda, \delta v)$ *as* $k \to \infty$ *and also satisfies the following properties:*

(i) $H\delta w = 0$*,*

(ii) $A\delta w \leq 0$*,*

(iii) $G\delta w = 0$*,*

(iv) $\delta v \geq 0$*,*

(v) $f^T\delta w = -\sigma||\delta w|| \leq 0$*,*

(vi) $\delta\lambda^T h + \delta v^T b \leq 0$*,*

(vii) $G^T\delta\lambda + A^T\delta v = 0$*.*

Proof. The proximal operator is firmly non-expansive [42] and thus averaged, see e.g., [123, rmk 4.34]. The convergence of δz_k to δz as $k \to \infty$ then follows from Lemma 4.1. Note that $\lim_{k\to\infty} \frac{1}{k}\delta z_k = 0$ and $\lim_{k\to\infty} \frac{1}{k} z_k = \delta z$ which we will use often in the sequel. We begin by rewriting (4.12) in the following form:

$$Hw_k + f + G^T\lambda_k + A^T v_k + \sigma\delta w_k = 0, \tag{4.37a}$$

$$h - Gw_k + \sigma\delta\lambda_k = 0, \tag{4.37b}$$

$$\langle b - Aw_k + \sigma\delta v_k, v_k\rangle = 0, \tag{4.37c}$$

$$v_k \geq 0,\ b - Aw_k + \sigma\delta v_k \geq 0. \tag{4.37d}$$

Further, (4.37) is satisfied exactly in the limit since the condition $\epsilon_k \to 0$ as $k \to \infty$ is enforced by construction in Algorithm 1. We now proceed point by point.

(i): Taking inner products, multiplying (4.37c) and (4.37b) by $1/k$, taking the limit, and applying Lemma 4.1 yields

$$\lim_{k\to\infty} \frac{1}{k}\langle b - Aw_k + \sigma\delta v_k, v_k\rangle = \langle -A\delta w, \delta v\rangle = 0, \tag{4.38a}$$

$$\Rightarrow \delta v^T A\delta w = 0, \tag{4.38b}$$

$$\lim_{k\to\infty} \frac{1}{k}\langle h - Gw_k + \sigma\delta\lambda_k, \lambda_k\rangle = \langle -G\delta w, \delta\lambda\rangle = 0. \tag{4.38c}$$

$$\Rightarrow \delta\lambda^T G\delta w = 0. \tag{4.38d}$$

The same procedure applied to (4.37a) yields

$$\lim_{k\to\infty} \frac{1}{k}\langle Hw_k + f + G^T\lambda_k + A^T v_k + \sigma\delta w_k, \delta w\rangle$$
$$= \delta w^T H\delta w + \delta\lambda^T G\delta w + \delta v^T A\delta w = 0, \tag{4.39}$$

combining this with (4.38) we obtain that, since $H \succeq 0$,

$$\delta w^T H\delta w = 0 \Rightarrow H\delta w = 0. \tag{4.40}$$

(ii): Multiplying the second inequality in (4.37d) by $1/k$ and taking the limit yields

$$\lim_{k\to\infty} \frac{1}{k}(b - Aw_k + \sigma\delta v_k) = -A\delta w \geq 0 \Rightarrow A\delta w \leq 0. \tag{4.41}$$

(iii): Multiplying (4.37b) by $1/k$ and taking the limit yields

$$\lim_{k\to\infty} \frac{1}{k}(h - Gw_k + \sigma\delta\lambda_k) = -G\delta w = 0 \Rightarrow G\delta w = 0. \tag{4.42}$$

(iv): Multiplying (4.37d) by $1/k$ and taking the limit yields

$$\lim_{k\to\infty} \frac{1}{k}v_k = \delta v \geq 0. \tag{4.43}$$

(v): Taking the inner product of (4.37a) with δz_k then taking the limit and applying (4.38) and (i) yields:

$$\lim_{k\to\infty} \langle Hz_k + f + G^T\lambda_k + A^T v_k + \sigma\delta z_k, \delta z_k\rangle$$
$$= \delta z^T H\delta z + \delta\lambda^T G\delta z + \delta v^T A\delta z + f^T\delta z + \sigma||\delta z||_2^2$$
$$= f^T\delta z + \sigma||\delta z||_2^2 = 0 \ \Rightarrow f^T\delta z = -\sigma||\delta z||_2^2 \leq 0. \tag{4.44}$$

(vi): Taking the inner product of (4.37c) and δv_k, and taking the limit yields

$$\lim_{k\to\infty} \langle b - Aw_k + \sigma\delta w_k, \delta v_k\rangle = b^T\delta v - \delta v^T A\delta w + \sigma||\delta v||_2^2 = 0, \tag{4.45}$$

since $\delta v^T A\delta w = 0$ we have that

$$b^T\delta v = -\sigma||\delta v||_2^2 \leq 0. \tag{4.46}$$

Applying the same procedure to (4.37b) yields,

$$\lim_{k\to\infty} \langle h - Gw_k + \sigma\delta\lambda_k, \delta\lambda_k\rangle = h^T\delta\lambda - \delta\lambda G\delta w + \sigma||\delta\lambda||_2^2 = 0, \tag{4.47}$$

since $\delta\lambda^T G\delta w = 0$ this implies $h^T\delta\lambda = -\sigma||\delta\lambda||_2^2$, combining this with (4.46) yields

$$h^T\delta\lambda + b^T\delta v = -\sigma(||\delta\lambda||_2^2 + ||\delta v||_2^2) \leq 0. \tag{4.48}$$

(vii): Dividing (4.37a) by k and taking the limit yields

$$\lim_{k\to\infty} \frac{1}{k}(Hw_k + f + G^T\lambda_k + A^T v_k + \sigma\delta w_k) = H\delta w + G^T\delta\lambda + A^T\delta v = 0, \tag{4.49}$$

applying (i) we have that $H\delta w = 0$ so we obtain

$$G^T\delta\lambda + A^T\delta v = 0, \tag{4.50}$$

which completes the proof. □

Armed with Proposition 4.5 and the infeasibility conditions in Proposition 4.4. we can prove the following theorem summarizing the behaviour of FBstab when (4.1) or (4.2) is infeasible.

Theorem 4.5 (Infeasibility Detection)**.** *Suppose that* (4.1) *is primal or dual infeasible, i.e.,* $T^{-1}(0) = \emptyset$*. Suppose* $x_0 \in \mathbb{R}^l$ *is arbitrary, let the sequence of iterates* $\{z_k\} = \{w_k, \lambda_k, v_k\}$ *be generated by FBstab with* $\sigma_{min} > 0, \epsilon_{min} = 0$*, and define* $\delta z_k = z_{k+1} - z_k$*. Then* $\delta z_k \to \delta z$ *as* $k \to \infty$ *where* $\delta z = (\delta w, \delta\lambda, \delta v)$ *satisfies the following properties:*

(i) If $\delta w \neq 0$ *then the dual QP* (4.2) *is infeasible and* δw *satisfies the dual infeasibility conditions in Proposition 4.4.*

(ii) If $(\delta\lambda, \delta v) \neq 0$ *then the primal QP* (4.1) *is infeasible and* $(\delta\lambda, \delta v)$ *satisfies the primal infeasibility conditions in Proposition 4.4.*

(iii) If $\delta z \neq 0$ *and* $(\delta\lambda, \delta v) \neq 0$ *then* (4.1) *and* (4.2) *are infeasible.*

Proof. (i): Follows by comparing points (i), (ii), (iii), and (v), of Proposition 4.5 with Proposition 4.4. Note that if $\delta w \neq 0$ then $f^T \delta w = -\sigma ||\delta w|| < 0$.

(ii): Follows by comparing points (vi), (vii), and (iv) of Proposition 4.5 with Proposition 4.4. Note that since $\delta v > 0$ due to point (iv) of Proposition 4.5, the condition $b^T \delta v_+ + h^T \delta\lambda < 0$ simplifies to $b^T \delta v + h^T \delta\lambda$.

(iii): Follows from points (i) and (ii) above. □

4.5 Numerical Experiments

In this section we illustrate the performance of FBstab with some numerical experiments. In addition to general QPs of the form (4.1), we solve instances of the following optimal control problem (OCP),

$$\min_{\xi,\mu} \sum_{i=0}^{N} \frac{1}{2} \begin{bmatrix} \xi_i \\ \mu_i \end{bmatrix}^T \begin{bmatrix} Q_i & S_i^T \\ S_i & R_i \end{bmatrix} \begin{bmatrix} \xi_i \\ \mu_i \end{bmatrix} + \begin{bmatrix} q_i \\ r_i \end{bmatrix}^T \begin{bmatrix} \xi_i \\ \mu_i \end{bmatrix}, \tag{4.51a}$$

$$\text{s.t. } \xi_0 = x, \tag{4.51b}$$

$$x_{i+1} = A_i \xi_i + B_i \mu_i + c_i, \ i \in \mathbb{Z}_{[0,N-1]}, \tag{4.51c}$$

$$E_i \xi_i + L_i \mu_i + d_i \leq 0, \ i \in \mathbb{Z}_{[0,N]}, \tag{4.51d}$$

where $A_i, Q_i \in \mathbb{R}^{n_x \times n_x}$, $B_i, S_i \in \mathbb{R}^{n_x \times n_u}$, $R_i \in \mathbb{R}^{n_u \times n_u}$, $q_i, c_i \in \mathbb{R}^{n_x}$, $r_i \in \mathbb{R}^{n_u}$, $E_i \in \mathbb{R}^{n_c \times n_x}$, $L_i \in \mathbb{R}^{n_c \times n_u}$, $d_i \in \mathbb{R}^{n_c}$, $\xi_i, x \in \mathbb{R}^{n_x}$, $u_i \in \mathbb{R}^{n_u}$, $\xi = (\xi_0, ..., \xi_N)$, and $\mu = (\mu_0, \ldots, \mu_N)$. We require that

$$\begin{bmatrix} Q_i & S_i^T \\ S_i & R_i \end{bmatrix} \succeq 0 \ \ \forall i \in \mathbb{Z}_{[0,N]}, \tag{4.52}$$

so the problem is convex. This QP is large but sparse and is often called the simultaneous or multiple shooting form of the MPC problem [93]. The QP is also often solved in the so-called condensed form,

$$\min_{\mu} \quad \frac{1}{2} \mu^T H \mu + f(x)^T \mu, \tag{4.53a}$$

$$\text{s.t} \quad A\mu \leq b(x), \tag{4.53b}$$

which is in the control variables only and can be derived by eliminating the state variables ξ in (4.51) using the dynamic equations see e.g., [17, Section 2.3] or [2]. We consider three linear MPC benchmark problems; their properties are summarized in Table 4.1.

Control of a Servo Motor [124]: The objective is to drive the motor position y_1 to a desired angular position $r = 30^\circ$ while respecting the constraint $|y_{2,k}| \leq 78.5\ Nm$ on the shaft torque and the constraint $|u| \leq 220\ V$ on the motor input voltage. The continuous time model is

$$\frac{d}{dt}x(t) = \begin{bmatrix} 0 & 1 & 0 & 1 \\ -128 & -2.5 & 6.4 & 0 \\ 0 & 0 & 0 & 1 \\ 128 & 0 & -6.4 & -10.2 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t),$$
$$y(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1282 & 0 & -64.0 & 0 \end{bmatrix} x(t),$$

which is discretized at 0.05 s using a zero-order hold. The tuning matrices and initial condition are $Q_i = diag([10^3, 0, 0, 0])$, $R_i = 10^{-4}$, and $x_0 = 0$. The traces of this model in closed-loop with an MPC controller are shown in Figure 4.1, the shaft angular position is driven to the reference while respecting the constraints on the shaft torque and input voltage.

Control of Spacecraft Relative Motion [125]: These equations describe the radial, along track, and across track positions $\zeta = [x_1\ x_2\ x_3]^T$ and velocities $\dot{\zeta}$ of a spacecraft relative to a nominal circular orbit. The control objective is to drive the spacecraft to the origin from $\zeta_0 = -[2.8\ 0.01\ 1]\ km$ and $\dot{\zeta}_0 = 0$. The system dynamics are given by the Hill-Clohessy-Wiltshire (HCW) equations,

$$\ddot{x}_1 - 3\omega^2 x_1 - 2\omega\dot{x}_2 = 0, \tag{4.54}$$
$$\ddot{x}_2 + 2\omega\dot{x}_1 = 0, \tag{4.55}$$
$$\ddot{x}_3 + \omega^2 x_3 = 0, \tag{4.56}$$

where $\omega = 0.0011\ s^{-1}$ is the mean motion of the reference orbit. The dynamics of $x = (\zeta, \dot{\zeta})$ can be compactly written as $\dot{x} = A_c x$. The control inputs are modelled as impulsive thrusts which instantaneously change the velocity of the spacecraft, see [125], so that the discrete time model is

$$x_{k+1} = A\left(\begin{bmatrix} \zeta_k \\ \dot{\zeta}_k \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} \Delta v_k\right) = Ax_k + Bu_k, \tag{4.57}$$

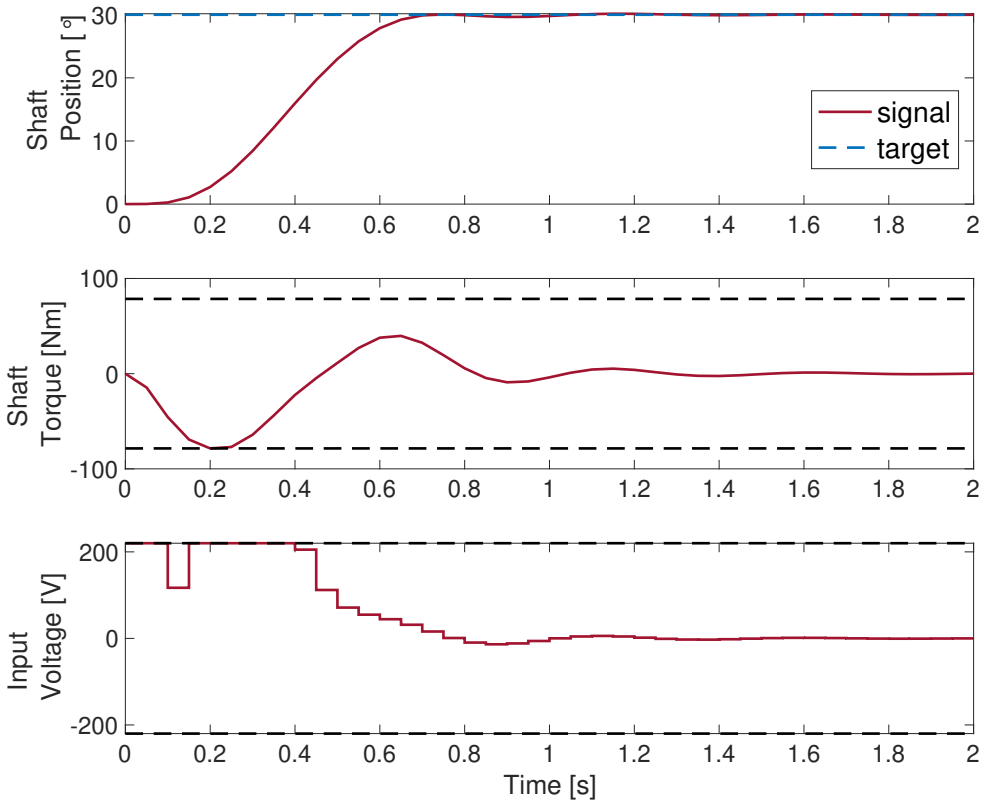

Figure 4.1: Closed-loop response in the servo motor example.

where $u = \Delta v$ is the instantaneous change in velocity due to the impulsive thrusters, $A = e^{A_c \tau}$, and $\tau = 30\ s$. The control inputs must satisfy $||u_k||_\infty \leq 1\ m/s$. The spacecraft velocity is constrained to satisfy $||\dot{\zeta}_k||_\infty \leq 1\ m/s$ and the tuning matrices are $Q = diag([1\ 1\ 1\ 0.001\ 0.001\ 0.001])$ and $R = I_{3\times 3}$. The closed-loop response of the system is shown in Figure 4.2.

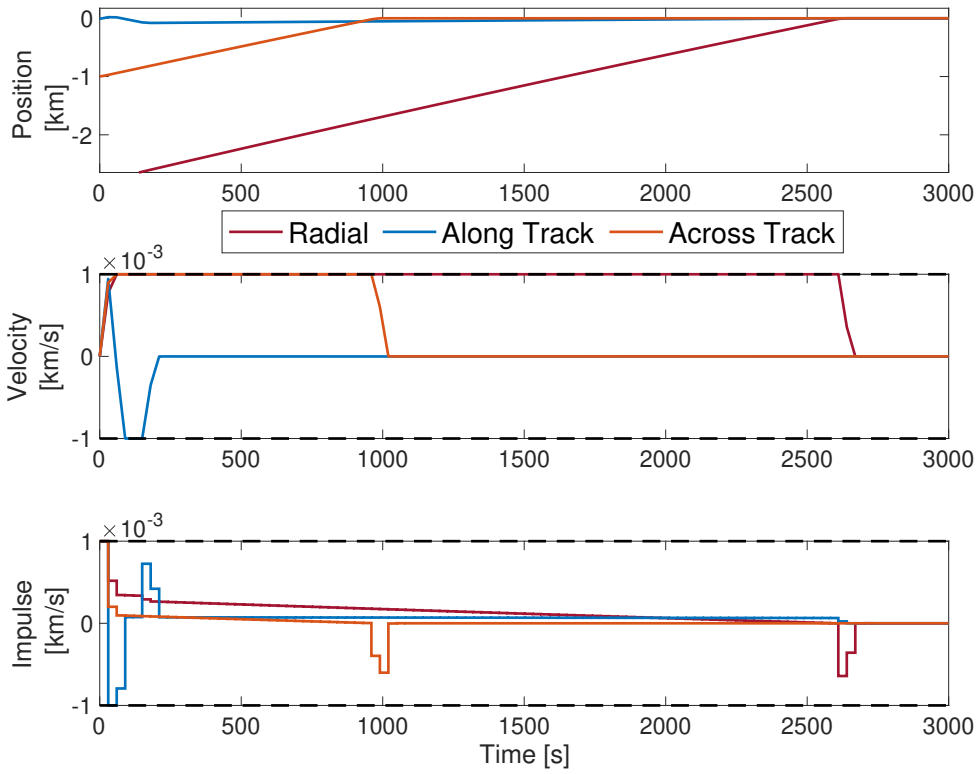

Figure 4.2: Closed-loop response for the spacecraft relative motion example.

Control of a Copolymerization Reactor [126]: The following normalized transfer function models the copolymerization of methyl methacrylate (MMA) and vinyl acetate (VA) in a continuous stirred tank reactor:

$$\begin{bmatrix} \frac{0.34}{0.85s+1} & \frac{0.21}{0.42s+1} & \frac{0.25s+0.5}{12s^2+0.4s+1} & 0 & \frac{6.46(0.9s+1)}{0.07s^2+0.3s+1} \\ \frac{-0.41}{2.41s+1} & \frac{0.66}{1.51s+1} & \frac{-0.3}{1.45s+1} & 0 & \frac{-3.72}{0.8s+1} \\ \frac{0.3}{2.54s+1} & \frac{0.49}{1.542+1} & \frac{-0.71}{1.35s+1} & \frac{-0.20}{2.71s+1} & \frac{-4.71}{0.008s^2+0.41s+1} \\ 0 & 0 & 0 & 0 & \frac{1.02}{0.07s^2+0.31s+1} \end{bmatrix}.$$

The normalized inputs are flows of monomer MMA (u_1), monomer VA (u_2), initiator (u_3), transfer agent (u_4), and the reactor jacket temperature (u_5). The normalized outputs are the polymer production rate (y_1), the mole fraction of MMA in the polymer (y_2), the molecular weight of the polymer (y_3), and the reactor temperature (y_4). All inputs and outputs are relative to nominal operating conditions [126]. The model was realized in modal form using the `ss` command in MATLAB and discretized using a zero-order hold with a normalized sampling period of 0.5 (corresponding to three hours in physical time). The resulting model has 18 states, 5 inputs and 4 outputs. The states are initially disturbed as $\xi_{0,i} = \sin(i)$ for $i = 1, \ ..., \ 18$; the control objective is to drive the outputs to the origin. The inputs are constrained as $||u_k||_\infty \leq 0.05$, i.e., 5% deviation from nominal. The horizon length is $N = 70$, and the weighting matrices are chosen as $Q = C^T C$, where C is the output matrix from the realization process, and $R = 0.1 I_{5\times 5}$. Closed-loop traces are shown in Figure 4.3.

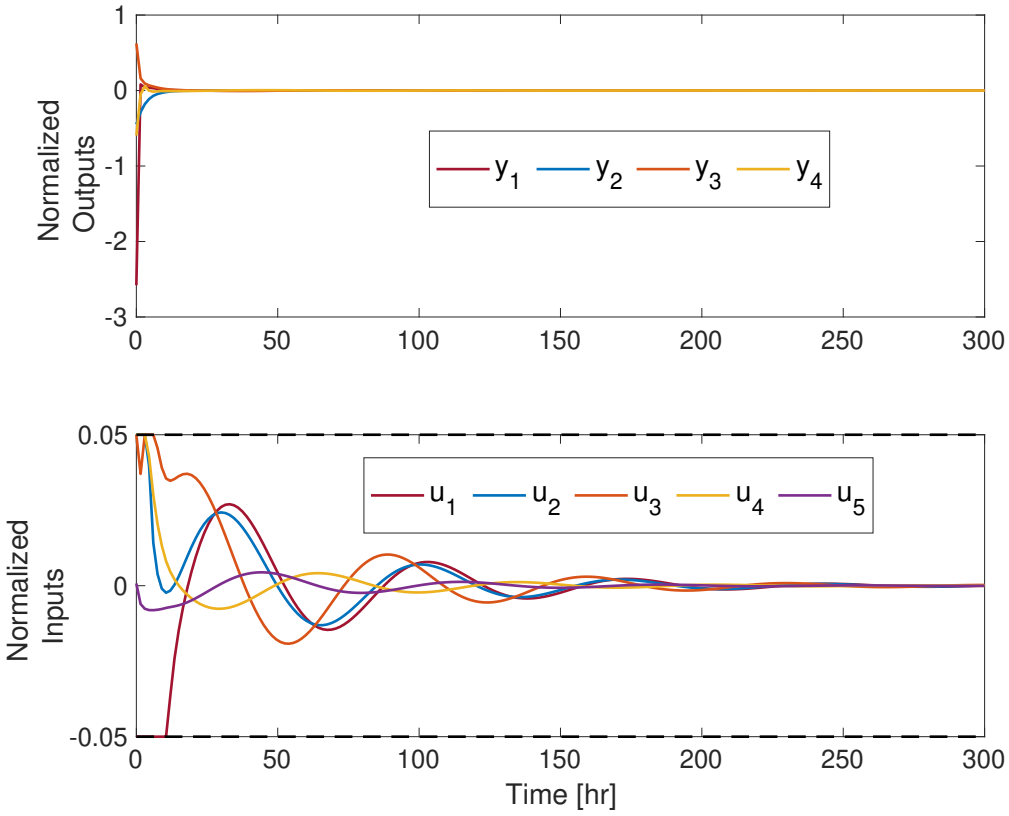

Figure 4.3: Closed-loop response for the copolymerization example.

Table 4.1: Problem data for the QPs.

	Servo	HCW	Copoly
Number of States	4	6	18
Number of Controls	1	3	5
Number of timesteps	40	100	200
Horizon Length	30	40	80
Inequality constraints	124	492	810
Sparse Problem			
Variables	155	369	1863
Equality constraints	124	246	1458
Condensed Problem			
Variables	31	123	405
Hessian condition number	189	3.19×10^8	1.4×10^3

4.5.1 Implementation Details

We have implemented three variants of FBstab in MATLAB.

`fbstab_dense` solves dense problems of the form (4.1) and is implemented using MATLABs built-in dense linear algebra routines. It solves the Newton-step systems (4.25) using a Cholesky factorization.

`fbstab_sparse` solves sparse problems of the form (4.1) and is implemented using MATLABs built-in sparse linear algebra routines. It solves the Newton-step systems (4.25) using MA57 [127].

`fbstab_mpc` solves problems of the form (4.51). It exploits the sparsity structure present in (4.51) when computing products with H, G, or A and uses a Ricatti-like recursion, similar to the one in [53], to solve (4.25) efficiently.

The default parameters are $\sigma_0 = \sigma_{max} = \sigma_{min} = \sqrt{\epsilon_m}$, $\zeta = 10^{-14}$, $\tau_a = 10^{-4}$, $\tau_r = 0$, $\tau_{inf} = 10^{-8}$, $\alpha = 0.95$, $\beta = 0.7$, $\kappa_\epsilon = 0.2$, $\kappa_\sigma = 0.1$ $\epsilon_{max} = 0.1$, $\epsilon_{min} = 10^{-12}$, $\eta = 10^{-8}$, where ϵ_m is machine precision; $\epsilon_m \approx 10^{-16}$ for our implementation. We also use a non-monotone linesearch technique [128] to improve performance without jeopardizing convergence. We recommend this set of parameters for most problems and when warmstarts are available e.g., when solving sequences of parameterized problems. The choice of the parameter $\sigma > 0$ is important for the practical performance of the algorithm. Based on our

numerical experience, for most problems we found a fixed value $\sqrt{\epsilon_m}$ to be large enough to ensure the linear systems are sufficiently well conditioned without slowing convergence of the outer loop. An investigation of the sensitivity of a related PPA based method to the σ parameter in [109] indicates the existence of a wide range of σ values that offer good performance. Our numerical experience indicates that this is the case for FBstab as well.

We used the following more conservative parameters for the hard Maros-Meszaros (MM) test set: $\sigma_0 = 1000\sqrt{\epsilon_m}$, $\sigma_{max} = \epsilon_m^{1/4}$, $\sigma_{min} = 10^{-10}$, $\tau_a = 10^{-5}$, $\tau_r = 10^{-8}$, $\beta = 0.9$ and used a monotone linesearch. Unlisted parameters are the same as in the default case. We recommend this parameter set for very difficult problems when no warmstart is available.

Remark 4.3. *In practice, we do not expect convergence to the exact solution, even in the limit, due to the limitations of finite precision arithmetic. Instead we pick parameters that yield fast and robust convergence to a neighbourhood of the solution set even if they only approximately meet the requirements of Theorem 4.3. Notably, we pick $\epsilon_{min} \approx 0$ and $\sigma_{min} > 0$.*

4.5.2 Solver Scaling

To demonstrate that FBstab can efficiently solve sparse problems, we compared `fbstab_mpc` and `fbstab_sparse` with the external solvers: (1) `quadprog` (MATLAB 2017b, *interior-point-convex*), (2) ECOS (Embedded Conic Solver) [52] and (3) qpOASES [98]. We also implemented the following in MATLAB: (4) the dual active set (DAS) method [99], including factorization updating, (5) QPNNLS (QP Nonnegative Lest Squares) [109], (6) GPAD [51] and (7) OSQP (Operator Splitting QP) [105]. The DAS method, QPNNLS, and qpOASES solve (4.53) and included the cost of condensing in the analysis, while all other methods solve (4.51) directly. The MATLAB routines were converted into `C` code using the `mex` command. We found that GPAD was not competetive; it has been omitted from Figures 4.4 and 4.5 for clarity.

We solved the first QP, i.e., at $t = 0$, in the servo motor and copolymerization examples[9] and measured wall clock times as the horizon was varied from $N = 10$ to $N = 1000$. All methods were cold started at the origin, the experiments were performed on a 2015 Macbook Pro with a 2.8 GHz i7 processor and 16 GB of RAM running MATLAB 2017b. Recorded execution times were averaged as necessary to obtain consistent timings. Figures 4.4 and 4.5 display the results. ECOS, `quadprog`, OSQP, and FBstab scale[10] like

[9]The spacecraft example dynamics are unstable; the resulting condensed problems are ill-conditioned enough at large N to make most of the methods fail.

[10]$\mathcal{O}$ scalings are determined using least-squares power fits.

$\mathcal{O}(N)$. The active set methods qpOASES, QPNNLS, and DAS are efficient for small problems but scale like $\mathcal{O}(N^3)$. The DAS method is the quickest method for the servo example for short horizons before being overtaken by `fbstab_mpc`. OSQP is the fastest method for the copolymerization example but struggles with the servo example (it always reaches its iteration limit) showcasing the conditioning dependency typical of first order methods. As expected, active set methods are very effective for small problems but are quickly overtaken. FBstab is shown to be faster than several interior point methods at all horizon lengths, `fbstab_mpc` is faster than `fbstab_sparse` since it is converted into `C` code and compiled instead of running directly in MATLAB. Overall, FBstab scales well as is competitive with and often superior to several established solvers.

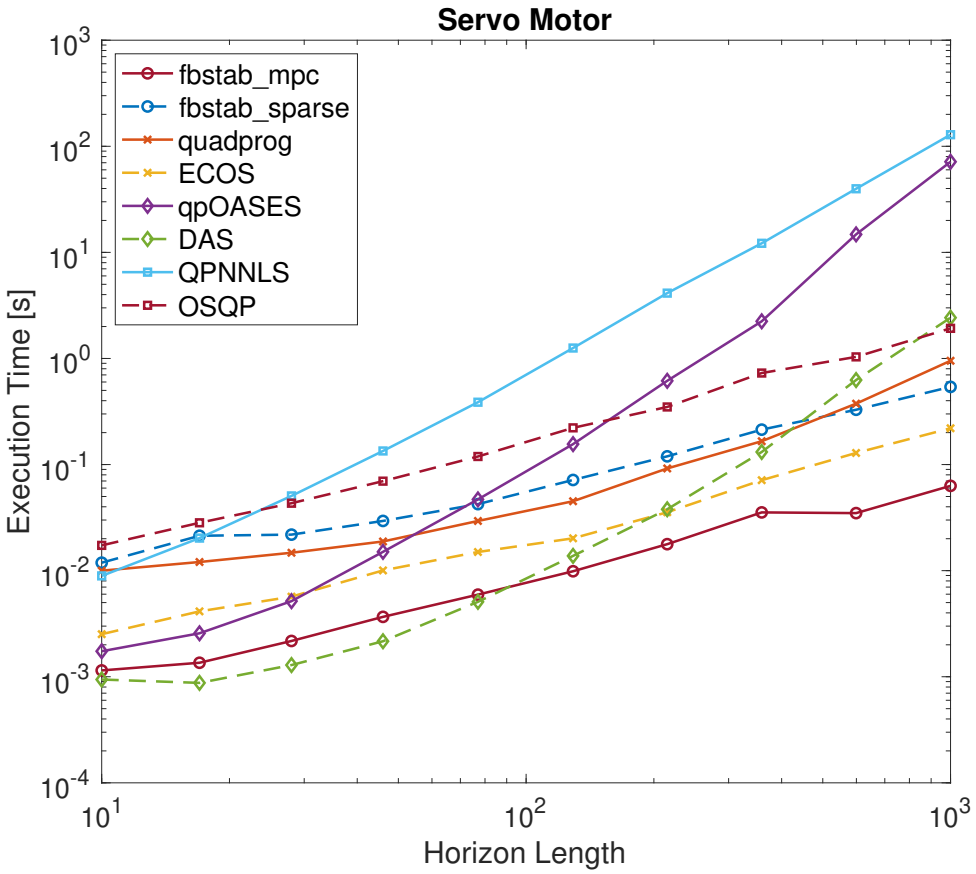

Figure 4.4: Solver scaling for the servo motor example.

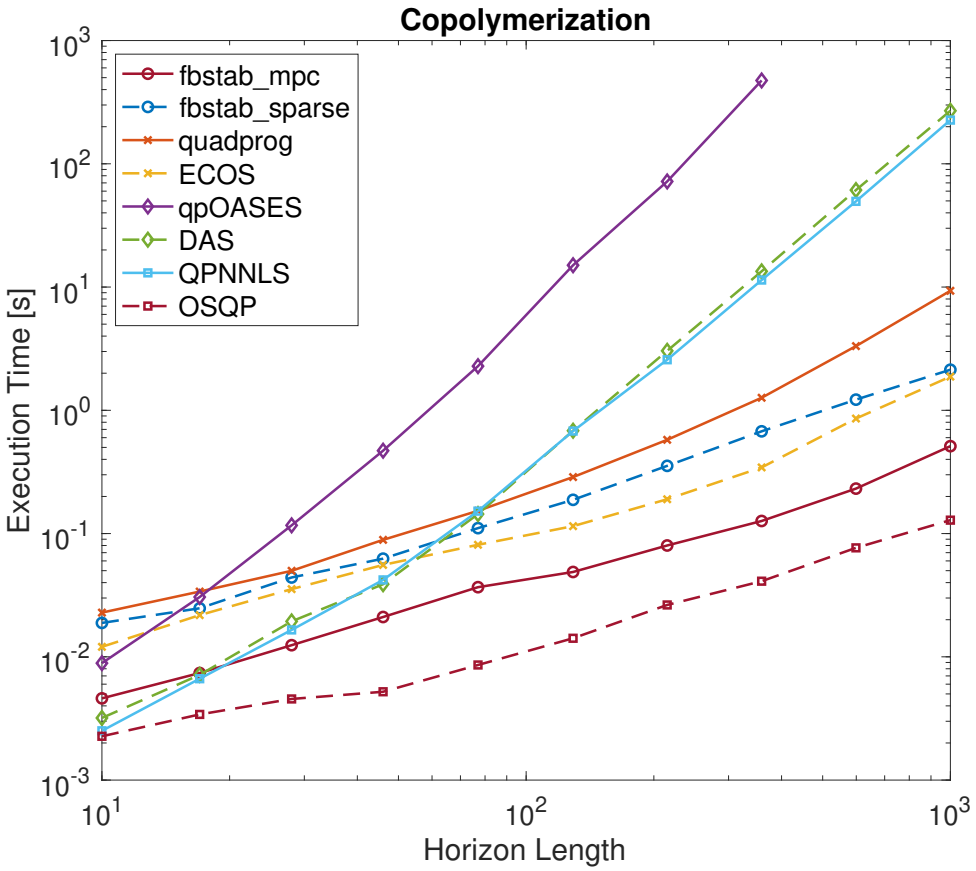

Figure 4.5: Solver scaling for the copolymerization example.

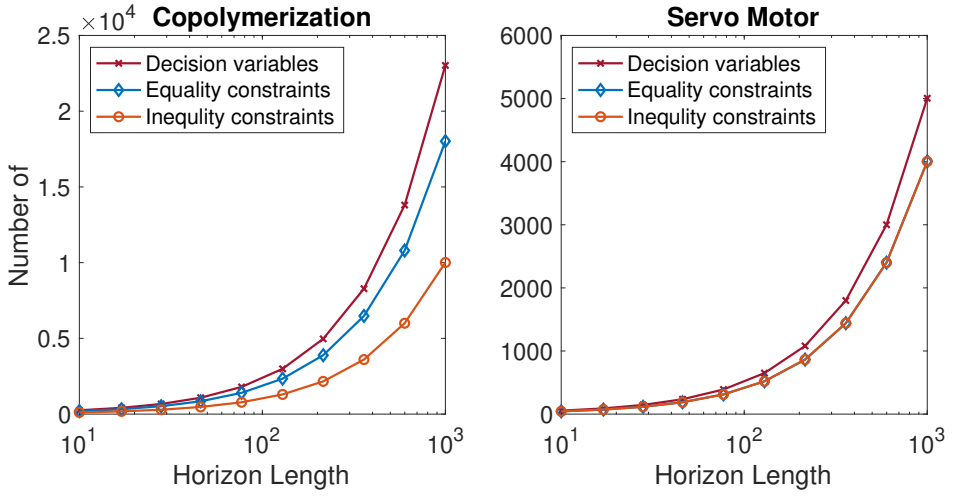

Figure 4.6: Problem size scaling as a function of N.

4.5.3 Maros-Meszaros Test Set

We also consider the Maros-Meszaros (MM) test set of hard QPs. We solved the entire test set with `fbstab_sparse`, and the open source solvers OSQP [105], and ECOS [52]. We also solved the 69 problems with less than 1000 decision variables and less than 1000 constraints with qpOASES [98]. Larger problems were prohibitively slow because qpOASES uses dense factorization routines.

We use FBstab's stopping criterion to decide if a problem is solved or not, i.e., we consider a problem solved if $||\pi(z^*)|| \leq 10^{-4} + 10^{-8}(||p|| + 1)$, where p is defined in (4.9) and z^* is the solver output. We used default settings except for tolerances and iteration limits. We adjusted the tolerances for OSQP and ECOS to ensure good agreement between our stopping criterion and each solvers internal one. The absolute and relative tolerances for OSQP were set at 10^{-5} and 10^{-8}. The absolute and relative tolerances for ECOS were set at 10^{-6} and 10^{-8}. FBstab and ECOS were allowed up to 500 Newton iterations, OSQP was allowed 8000 (the default is 3000) iterations, and qpOASES was allowed 3000 active set changes. FBstab was initialized from the origin, all other solvers used their default starting points.

The results are shown in Figures 4.7 and 4.8 in the form of performance profiles [129]. Overall, `fbstab_sparse` is extremely reliable, solving 84.7% of the problems. The active set solver qpOASES was nearly as reliable for the small problems but slightly slower than FBstab. OSQP and ECOS were fast when they were successful but not as reliable. The relative slowness of `fbstab_sparse` is expected, it is implemented in MATLAB (MATLAB sparse matrix algebra cannot yet be automatically converted to `C` code) while the other methods are implemented in `C/C++`. We originally designed FBstab for parameterized problems, such as those arising from model predictive control, and are encouraged by its performance when exercised using the MM test set despite the lack of warmstarting information.

Overall, the most difficult problems for FBstab tend to be those with poor scaling, particularly when the optimal primal and dual variables are of very different magnitudes. Investigating preprocessing or preconditioning techniques could help with this in the future. We didn't observe any particular difficulties with semidefinite problems. We found that for some of the ill-conditioned problems in the MM test set that solving the first subproblem could be slow. Using a monotone linesearch and increasing the initial regularization parameter improved the efficiency of the Newton-type subproblem solver during its initial "damped" phase for the first subproblem after which good warmstarts for each subsequent subproblem become available. Improving the performance of Algorithm 3 during the "damped" Newton phase is a topic for future research.

We note that ECOS had some numerical difficulties, this is consistent with [105]. Specifically, it has a tendency to declare ill-conditioned but feasible problems infeasible. The authors of OSQP observed a higher success rate for OSQP [105] than we did, but used significantly more relaxed tolerances; the default relative and absolute tolerances for OSQP are 10^{-3} and it uses a different, more generous, relative stopping criterion. These tolerances are not entirely unreasonable, given that OSQP is a first order method, but since ECOS, FBstab, and qpOASES are second order methods we opted for a more stringent criterion. Performing fair and general comparisons between first and second order methods remains challenging.

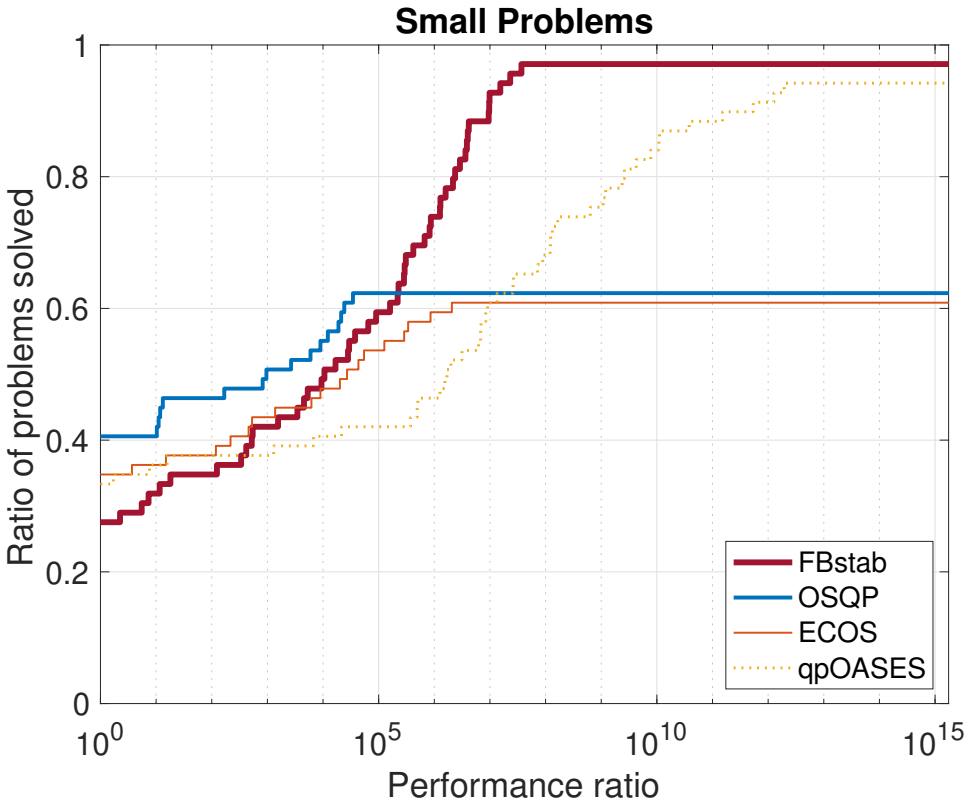

Figure 4.7: Performance profiles over all problems in the Maros-Meszaros test set with $n \leq 1000$ and $m + q \leq 1000$.

4.5.4 Benchmarking on Real-time Hardware

To investigate the performance of FBstab on embedded hardware we performed some benchmarking on a Speedgoat Baseline Real-time Target Machine (SGTM). The SGTM (2.0 GHz Celeron CPU, 4 GB RAM) is a rapid prototyping platform which runs a real-

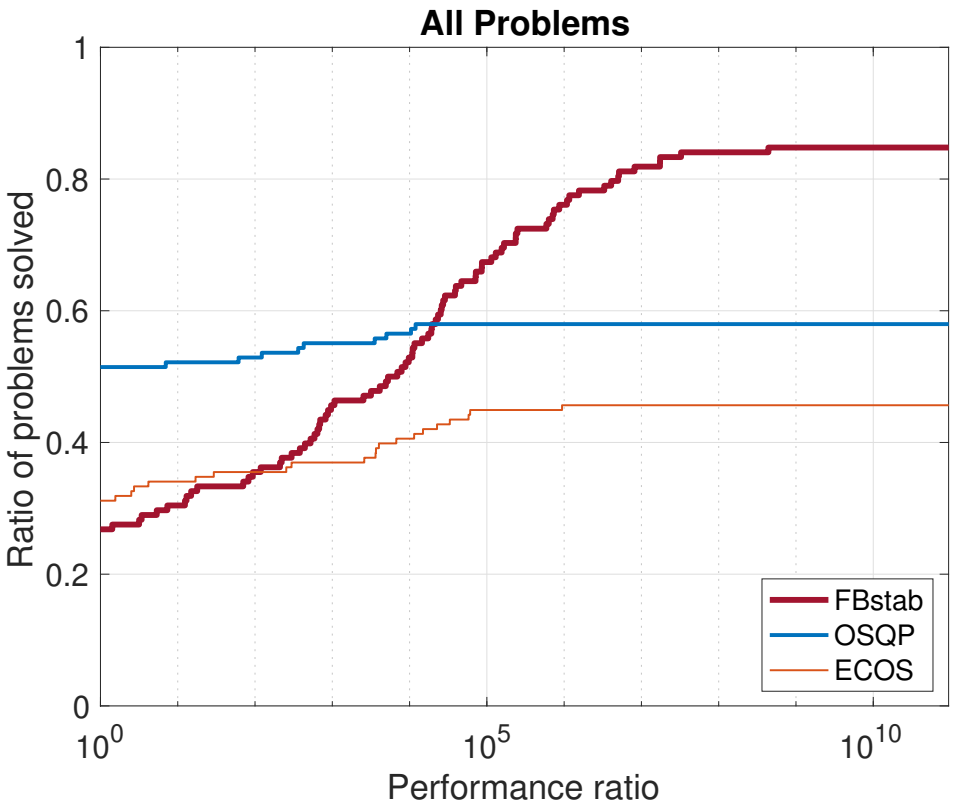

Figure 4.8: Performance profiles for the complete Maros-Meszaros test set.

time operating system (RTOS) and is representative of an embedded computing environment. Using the RTOS allows us to obtain deterministic execution time measurements. Moreover, we did not use any advanced factorization routines, e.g., LAPACK, so solving ill-conditioned problems is more difficult.

We deployed `fbstab_mpc` and `fbstab_dense` on the SGTM. For comparison we implemented the following in MATLAB: (1) FBRS (Fischer-Burmeister Regularized and Smoothed) [50], (2) DAS (Dual Active Set) [99], (3) QPNNLS [109], (4) PDIP (primal-dual IP using Mehrotra's predictor-corrector) [49, Algorithm 14.3], (5) GPAD (accelerated dual gradient projection) [51], and (6) accelerated ADMM [106]. Since `quadprog` cannot codegenerate, ECOS and OSQP do not have Simulink interfaces, and qpOASES was outperformed by DAS during the scaling trials all three were omitted from testing; GPAD and ADMM were not competitive and were omitted as well. The methods were converted into C code using Simulink Real-time (2017b). A method is deemed to have failed if it is stalls and is unable to solve any of the QPs in the sequence to the desired precision (10^{-4}). Note that we have implemented LDL^T and QR factorization updating for QPNNLS and DAS to ensure a competitive comparison.

The results are shown in Table 4.2. When warmstarting is enabled, `fbstab_mpc` is the fastest method in the worst case for all three examples and `fbstab_dense` is competitive with the IP and AS methods, especially on the larger copolymerization example. In terms of average execution times[11] `fbstab_mpc` and `fbstab_dense` are dominant. When warmstarting is disabled, the PDIP and active set methods become more competitive, however `fbstab_mpc` is still more efficient.

Overall, when the cost of condensing is considered, see Remark 4.4, `fbstab_mpc` is shown to outperform the other methods tested in terms of both maximum and average execution time. FBstab derives significant benefit from warmstarting, this is especially noticeable for the HCW example, and is significantly faster than `fbstab_dense`, showcasing the importance of specialized linear algebra routines. Further, both DAS and FBRS fail on the ill-conditioned HCW example while their regularized versions, QPNNLS and FBstab respectively, succeed, demonstrating the expected improved robustness due to proximal regularization. FBstab is often faster than FBRS, demonstrating that the addition of proximal regularization makes the methods more robust without a significant reduction in speed.

Remark 4.4. *The cost of condensing, i.e., of converting* (4.51) *to* (4.53) *is included in the results reported in Table 4.2. This simulates solving e.g., trajectory tracking problems or*

[11] Average execution times are an indicator of power consumption. This is an important metric in aerospace applications where reduced power consumptions leads to e.g., extended range for drones.

real-time iteration [63] subproblems. These computations can sometimes be moved offline, in this situation the normalized cost of condensing listed in Table 4.2 should be subtracted from each row of the last five columns. In this scenario, the DAS method is the best method for the servo motor example.

Table 4.2: Summary of normalized Speedgoat benchmarking reporting the maximum and average QP solutions times for each sequence. Warm and cold starting are indicated by W and C respectively.

	FBstab MPC	FBstab Dense	FBRS	PDIP	NNLS	DAS
Servo Motor, Normalization $= 4.5\ ms$ Normalized Cost of Condensing $= 1.1$						
MAXW	1.00	2.9	2.4	3.2	3.3	1.7
AVEW	0.2	1.4	1.4	2.1	1.8	1.3
MAXC	1.5	5.0	5.0	3.0	2.9	2.1
AVEC	0.2	1.6	1.6	2.1	1.8	1.3
Spacecraft, Normalization $= 63.9\ ms$ Normalized Cost of Condensing $= 1.5$						
MAXW	1.00	14.8	F	11.7	15.6	F
AVEW	0.1	3.3	F	8.3	6.5	F
MAXC	3.4	73.2	F	29.3	7.6	F
AVEC	2.2	62.8	F	25.5	3.7	F
Copolymerization, Normalization $= 97.1\ ms$ Normalized Cost of Condensing $= 76.3$						
MAXW	1.00	96.6	102.9	238.7	94.4	149.2
AVEW	0.4	82.5	82.6	204.2	85.8	96.2
MAXC	1.5	113.5	112.9	238.2	88.4	293.3
AVEC	0.3	83.0	82.9	205.3	85.9	101.7

4.5.5 Degenerate and Infeasible Problems

FBstab is capable of detecting both primal and dual infeasibility and handling high degenerate problems. Consider a double integrator, its state is $\xi = (\xi_1, \xi_2)$, where ξ_1 is position, ξ_2 is velocity and the control input μ is acceleration. Next consider the following optimal

control problem

$$\max_{\xi,\mu} \sum_{i=0}^{N} \begin{bmatrix} 1 & 1 \end{bmatrix} \xi_i, \tag{4.58a}$$

$$\text{s.t. } \xi_0 = [0,0]^T, \ \xi_N \in \mathcal{X}_f \tag{4.58b}$$

$$\tag{4.58c}$$

$$\xi_{i+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \xi_i + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mu_i, \ i \in \mathbb{Z}_{[0,N-1]}, \tag{4.58d}$$

$$\mu_i \in \mathcal{U}, \ i \in \mathbb{Z}_{[0,N]}. \tag{4.58e}$$

Letting $N = 4$, $\mathcal{X}_f = \{(\xi_1, \xi_2) \mid \xi_1 = 4\}$, and

$$\mathcal{U} = \left\{ \mu \mid \begin{bmatrix} 1 & -1 & 0 \end{bmatrix}^T \mu + \begin{bmatrix} -1 & -1 & 0 \end{bmatrix}^T \leq 0 \right\} \tag{4.59}$$

makes the dual of (4.58) degenerate, the constraint $0\mu \leq 0$ contradicts the LICQ. When `fbstab_mpc` is applied to this problem, starting from the origin, it signals optimality after 7 Newton iterations and 3 proximal iterations.

Letting $N = 3$, $\mathcal{X}_f = \{(\xi_1, \xi_2) \mid \xi_1 = 4\}$, and $\mathcal{U} = \{u \mid |u| \leq 1\}$ renders (4.58) primal infeasible. Intuitively, the system is unable to reach its position goal in $N = 3$ steps due to the acceleration limit ($N = 4$ is feasible). When `fbstab_mpc` is applied to this problem, starting from the origin, it signals primal infeasibility after 11 Newton iterations and 2 proximal iterations.

Letting $N = 3$, $\mathcal{X}_f = \mathbb{R}^2$, and $\mathcal{U} = \{u \mid u \geq 0\}$ makes (4.58) dual infeasible, i.e., unbounded above. Intuitively, this occurs because the objective of (4.58) is to maximize velocity and, since μ can be increased without bound, the cost can be made arbitrarily large by making μ arbitrarily large. When `fbstab_mpc` is applied to this problem, starting from the origin, it signals dual infeasibility after 11 Newton iterations and 1 proximal iteration.

4.6 Conclusions

This chapter presents FBstab, a proximally stabilized Fischer-Burmeister method for convex quadratic programming. FBstab is attractive for real-time optimization because it is easy to code, numerically robust, easy to warmstart, can exploit sparsity, and converges or detects infeasibility under only the assumption that the Hessian of the quadratic program is positive semidefinite. An open source MATLAB implementation of FBstab is available

online. Future work includes exploring the application of stabilized semismooth Newton-type methods to nonlinear problems and continued development of an open source `C++` implementation of FBstab.

Acknowledgments

We would like to thank Dr. Chris Petersen of the U.S. Air Force Research Laboratories for encouraging us to develop methods for optimization problems with degenerate solutions.

CHAPTER 5

Model Predictive Emissions Control of a Diesel Engine Airpath

5.1 Introduction

Increasingly stringent emissions and fuel economy regulations have created a need for more complex engine systems and sophisticated engine control strategies. Diesel engines offer superior fuel economy compared to their gasoline counterparts; however sophisticated strategies are needed to manage oxides of nitrogen ($\mathrm{NO_x}$) and particulate matter (PM) emissions to meet regulatory standards. Many modern diesels incorporate a Variable Geometry Turbocharger (VGT) to provide variable boost, and an external Exhaust Gas Recirculation (EGR) system to reduce emissions. However, the addition of both the turbocharger and EGR systems into the engine design introduces strong nonlinearities and interactions, complicating control development.

Emissions control in automotive diesels is especially challenging as, in contrast to e.g., marine or generator applications, load and engine speed are highly transient, leading to frequent emissions spikes which must be managed. In particular, smoke control is strongly coupled to torque response (drivability) and there is a strong tradeoff between transient $\mathrm{NO_x}$ and smoke production. Model predictive control (MPC) provides a useful framework for managing these tradeoffs using real-time constraint enforcement. The application of MPC to diesel engine control has been considered in references [130–136]. Commercial software for automotive MPC development is also available [137]. In this chapter, we present an MPC strategy for emissions management in turbocharged diesels with external EGR and apply it to a 2.8L diesel engine.

The contributions of this chapter are as follows: We develop a novel SMPC controller for the DAP that coordinates the EGR rate setpoint and fuel input to enforce combustion quality constraints. The SMPC controller uses move blocking, symbolic tools, time-distributed optimization, and a version of FBstab to achieve real-time feasibility despite, the

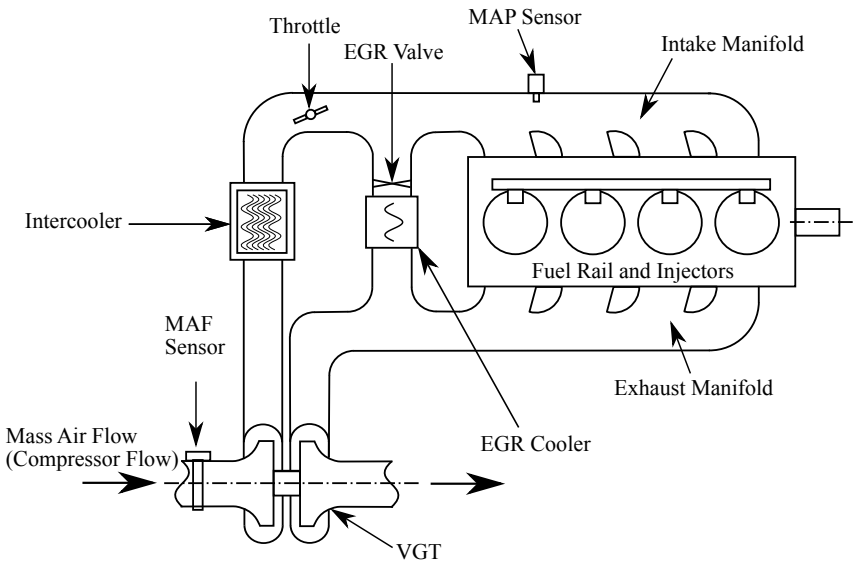

Figure 5.1: A diagram of the diesel engine airpath.

fast sampling rate required by the engine. In addition, we demonstrate on an experimental engine that our controller is able to outperform the current state-of-the-art industrial benchmark controller and, in particular, reduce engine out NOx and hydrocarbon emissions by $10 - 15\%$ relative to the benchmark. Moreover, we accomplish this using only production sensors, meaning that our controller can be deployed on existing engines without hardware modifications. Taken together, the material in this chapter illustrates how the algorithmic and theoretical tools developed in Chapters 3 and 4 can be used to enable the application of MPC to physical systems to make an impact in the real-world.

The material in this chapter is distinct from existing application domain literature; it directly considers emissions management for a high speed automotive diesel engine using only sensors that are available in production vehicles. Many publications only consider the tracking of intake pressure and EGR (or mass airflow) setpoints [130–136] rather than high level objectives such as fuel efficiency or emissions. Other works which directly consider high level (economic) objectives either use sensors which are not available in production vehicles [138] [139] or consider heavy duty [140], off-highway [141], or marine/generator [139] diesel engines which are not subject to the same transient conditions as an automotive diesel. In particular, we demonstrate visible smoke control during fast transients which is both challenging and specific to automotive diesel applications.

The results in this chapter are the culmination of a multi-year collaborative project between Toyota Motor North America, Toyota Motor Corporation, and the University of Michigan. The SMPC controller is primarily my work and is an extension of the economic

MPC controller presented in [142]. This chapter presents an updated formulation, which removes the need for a $\mathrm{NO_x}$ sensor, and experimental data, [142] contains only simulation results. The controller architecture, see Figure 5.5, also includes a NMPC based inner-loop airpath controller, which is an evolution of the work presented in [134] and [31]. A partial history of the project and an overview of the NMPC controller development can be found in [143], more details on the final NMPC controller can be found in [30].

Remark 5.1. *It should be emphasized that, while some promising experimental results have been reported [136], most literature on NMPC for the DAP involves only simulation studies [134, 144, 145] due to the high computational complexity of nonlinear optimization and the limited computational power available in engine control units. These computational considerations also limit the complexity of OCP formulations, leading to reduced controller performance. The theoretical and algorithmic developments presented in Chapters 3 and 4 are essential for not only allowing experimental testing to proceed, but also enabling the use of more sophisticated OCP formulations that allowed us to design controllers that perform well in experimental testing.*

5.1.1 Layout

A model predictive controller is fundamentally based on three constituents: (i) Prediction models used to estimate the response of the system to a control action and any associated estimators, (ii) An optimal control problem (OCP) formulation, and (iii) a method for solving the aforementioned OCP in real-time. Section 5.2 provides some background on engine control and describes the engine system, Section 5.3 describes the prediction models used in the supervisory controller. Section 5.4 describes the control objectives, controller architecture, and the optimal control problem. Section 5.5 describes how the OCP is solved in real-time. Section 5.6 contains experimental results and analysis demonstrating the performance and reliability of the proposed strategy. Finally, Section 5.7 contains some concluding remarks.

5.2 Engine Control Background

Figure 5.2 is a block diagram of a typical diesel powertrain. The driver inputs a pedal angle command that is transformed into a fueling rate command q^{trg}. The engine then produces torque which is applied to the transmission input shaft, this torque is transmitted to the wheels after passing through the transmission and the differential. The wheel torque interacts with the road and accelerates the vehicle. The engine also produces emissions which

are heavily regulated; in modern vehicles engine exhaust gases pass through an aftertreatment system, see e.g., [146] for more details, before being emitted into the atmosphere. From the perspective of the engine there are two inputs, the fueling rate command (q^{trg}) and the engine speed (N_e). The fueling rate command represents a driver acceleration demand, the engine speed is determined by the powertrain; neglecting driveshaft compliance and wheel slip, it is a function of the vehicle speed, the transmission gear ratio, and the differential gear ratio. The outputs are the engine torque applied to the transmission input shaft (τ_q), the $\mathrm{NO_x}$ (ψ_{NO_x}), and total hydrocarbon (THC) (ψ_{THC}) concentrations, and the exhaust gas opacity (ψ_{OP}). More details on powertrain modelling and control can be found in e.g., [147] or [148].

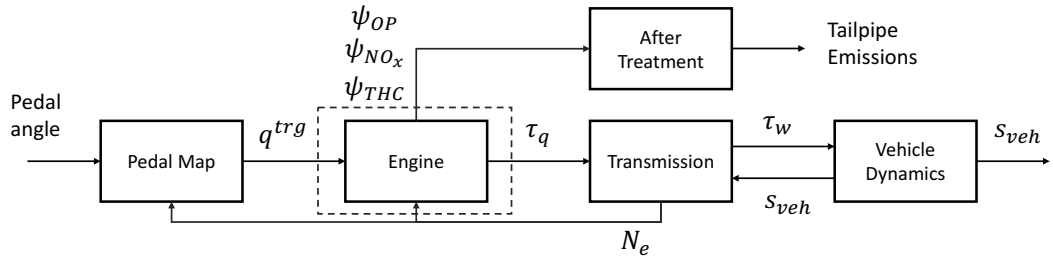

Figure 5.2: A high level diagram illustrating the architecture of a typical diesel powertrain.

A schematic of the diesel engine is shown in Figure 5.1. The engine consists of a cylinder block, intake and exhaust manifolds, an external EGR system, and a turbocharger. The Manifold Air Pressure (MAP) and Mass Air Flow (MAF) sensors are used to measure the intake manifold pressure and compressor flow respectively. The exogenous inputs to the system are the fuel target and the engine speed. Fluid flows through the engine are controlled by the EGR throttle, EGR valve and VGT. The EGR valve controls the amount of exhaust gas allowed to flow into the intake manifold; adding exhaust gas into the intake manifold reduces peak combustion temperature in the cylinders, lowering $\mathrm{NO_x}$ concentrations, but can cause poor quality combustion, leading to vibration, roughness, and smoke. The EGR throttle is used to reduce airflow into the intake manifold allowing for a higher proportion of exhaust gases when it is beneficial. The VGT influences the pressure in the intake manifold by controlling how much energy is extracted from the exhaust gases. Higher intake manifold pressure increases air flow into the cylinders, increasing engine power. In addition, in this work we consider the total fueling rate as a control input; an external injection strategy is used for the rest of the fuel path control.

5.3 Engine Modelling

This section describes the prediction models used SMPC controller. The engine operating condition consists of the fueling rate target q^{trg} and the engine speed N_e, i.e.,

$$\rho = [q^{trg} \;\; N_e]^T. \tag{5.1}$$

The state vector for the supervisory controller consists of the intake pressure p_{im}, exhaust pressure p_{ex}, compressor mass flow rate $\dot{m}_c$, intake manifold burnt gas fraction F_1, and exhuast manifold burnt gas fraction F_2, the control vector consists of the EGR rate target passed to the airpath controller χ^{trg} and the fueling rate command applied to the engine q i.e.,

$$x = [p_{im}\; p_{ex}\; \dot{m}_c\; F_1\; F_2]^T, \text{ and } u = \left[\chi^{trg}\; q\right]^T. \tag{5.2}$$

The fueling rate q is proportional to the fuel mass flow $\dot{m}_f$ over the engine speed, i.e., to $\dot{m}_f/N_e$, and has units of volume per stroke. The EGR rate χ is defined as the mass flow through the EGR valve into the intake manifold $\dot{m}_{egr}$ over the total mass flow into the cyinders $\dot{m}_{cyl}$, i.e., $\chi = \dot{m}_{egr}/\dot{m}_{cyl}$. The fueling rate applied to the engine q may be different than the target q^{trg} in order to enforce combustion quality constraints. The state vector of the supervisory controller can be partitioned into airpath and EGR loop variables, denoted by

$$\zeta = [p_{im}\; p_{ex}\; \dot{m}_c]^T, \text{ and } \Upsilon = [F_1\; F_2]^T, \tag{5.3}$$

respectively so that $x = [\zeta^T \;\; \Upsilon^T]^T$. Throughout the chapter we will make use of a 2 dimensional grid of operating conditions,

$$\begin{bmatrix} \rho_1 & \rho_{15} & \cdots & \rho_{141} \\ \rho_2 & \rho_{16} & \cdots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \rho_{14} & \cdots & \cdots & \rho_{154} \end{bmatrix} = \begin{bmatrix} (q_1^{trg}, N_{e1}) & (q_1^{trg}, N_{e2}) & \cdots & (q_1^{trg}, N_{e11}) \\ (q_2^{trg}, N_{e1}) & (q_2^{trg}, N_{e2}) & \cdots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ (q_{14}^{trg}, N_{e1}) & \cdots & \cdots & (q_{14}^{trg}, N_{e11}) \end{bmatrix} \tag{5.4}$$

which are contained within the box $[q_1^{trg}, q_{14}^{trg}] \times [N_{e,1}, N_{e,11}]$ and are used for scheduling gains, targets, and model parameters. This grid was chosen so as to contain the test cycles used for experimental validation. If the controller described in this chapter was to be used in a production vehicle the grid would be expanded to cover the entire engine operating region. We use a dense uniform grid for simplicity, it would be more efficient to adapt the spacing of the gridpoints so the density is higher where the dynamics change more rapidly. However, because we are able to gather data quickly, we found the effort required

to implement an adaptive scheme was unnecessary. Note that having additional gridpoints does not noticeably affect online complexity.

5.3.1 Closed-loop Airpath Modelling

The closed-loop airpath prediction model is used by the SMPC controller to estimate the response of the inner-loop to EGR target and fueling rate commands. The model is data driven and is identified from experimental data. We use a linear parameter varying (LPV) model, identified using the local LPV modelling approach [149]. At each operating point $\rho_i,\ i = 1, \ldots, 154$, we identify a model of the form

$$\zeta^+ = F_i + A_i\zeta + B_i u, \tag{5.5}$$

where $F_i \in \mathbb{R}^3$, $A_i \in \mathbb{R}^{3\times3}$, and $B_i \in \mathbb{R}^{3\times2}$, using linear least squares. The final model is of the form

$$\zeta^+ = F(\rho) + A(\rho)\zeta + B(\rho)u, \tag{5.6}$$

where $A : \mathbb{R}^2 \to \mathbb{R}^{3\times3}$, $B : \mathbb{R}^2 \to \mathbb{R}^{3\times2}$, and $F : \mathbb{R}^2 \to \mathbb{R}^3$ are operating condition dependent matrices. To construct A, B and F, we apply linear interpolation to the model coefficients A_i, B_i, F_i for $i = 1, ..., 154$. Figure 5.3 illustrates a typical outcome of the fitting process. The model quality is good over most of the operating range since the inner loop controller tends to "linearize" the closed loop plant from the perspective of the SMPC controller. However, care must be taken to ensure that stability of the model is enforced during the fitting process. Instability can occur when the exhaust pressure is insensitive to the perturbations, causing the identification procedure to attempt to capture unstable noise. We handle this in an ad hoc manner by filtering the exhaust pressure signal and adjusting the weighting factors used in the least squares problem. More systematic procedures will be developed in future work. Further, to ensure correct behaviour when used online, the DC gain of the identified EGR rate target to EGR rate transfer function must have the correct sign. If this is not true then the identification experiment should be repeated with smaller EGR rate target perturbations.

Remark 5.2. *We have observed that the closed-loop performance of the SMPC controller is not very sensitive to the accuracy of the closed-loop airpath model, likely because the presence of the inner loop controller makes the closed-loop airpath dynamics relatively benign. We were able to obtain good performance as long as the time constants of the modelled states were of the correct order of magnitude.*

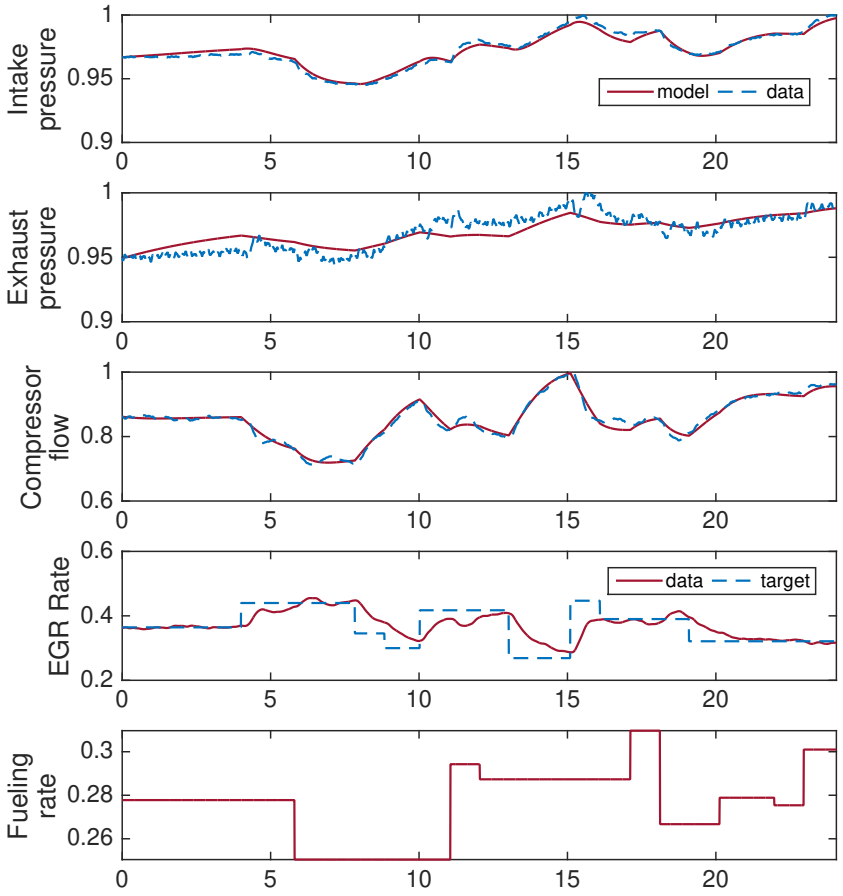

Figure 5.3: Fitting result for a local LTI model of the closed-loop airpath at $N_e = 1400$ rpm

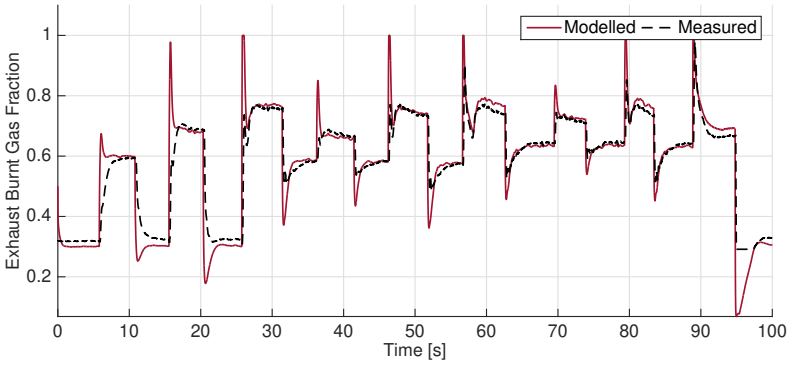

Figure 5.4: Validation of the burnt gas fraction model against experimental data.

5.3.2 Burnt Gas Fraction Modelling

The burnt gas fraction (BGF) model is used by the SMPC to predict the emissions response of the system by tracking the time evolution of the burnt gas fractions in the intake manifold, F_1, and exhaust manifold, F_2. We use the model from [150], expressed in the form $\dot{\Upsilon} = G(\zeta)\Upsilon + b(\zeta)q$, the BGF equations can be written as

$$\begin{bmatrix} \dot{F}_1 \\ \dot{F}_2 \end{bmatrix} = \begin{bmatrix} \frac{-(\dot{m}_{egr}+\dot{m}_{th})}{m_1} & \frac{\dot{m}_{egr}}{m_1} \\ \frac{\dot{m}_{cyl}}{m_2} & \frac{-(\dot{m}_{cyl}+\dot{m}_f)}{m_2} \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1+(A/F)_E}{m_2} \end{bmatrix} c_q N_e q, \tag{5.7a}$$

where

$$m_1 = \frac{p_{im}V_{im}}{R_{air}T_{im}}, \quad m_2 = \frac{p_{ex}V_{ex}}{R_{ex}T_{ex}}, \tag{5.8}$$

c_q is the constant such that $\dot{m}_f = c_q N_e q$, R_{air} is the gas constant of air, R_{ex} is the gas constant of the exhaust gas, and V_{im}, V_{ex}, T_{im}, and T_{ex} are the volumes and gas temperatures of the of the intake and exhaust manifolds. The effective air-fuel ratio, denoted by $(A/F)_E$, quantifies the mass of oxygen consumed per unit fuel and is calibrated as a function of operating condition using exhaust analyzer data. The cylinder flow is estimated as a linear, operating condition dependent function of intake pressure, i.e.,

$$\dot{m}_{cyl} = a(\rho)p_{im} + b(\rho), \tag{5.9}$$

the EGR flow is estimated as $\dot{m}_{egr} = \dot{m}_{cyl} - \dot{m}_c$, and the throttle flow, $\dot{m}_{th}$, is assumed to be equal to the compressor flow. In steady state the BGF equations reduce to the following relationship between the gas fractions and EGR rate: $\chi = F_1/F_2 = \dot{m}_{egr}/\dot{m}_{cyl}$. The BGF equations are stiff so we discretize them using the implicit Euler integration scheme [151]. Since the equations are linear in Υ the update equation can be determined analytically as

$$\Upsilon_{i+1} = (I_{2\times 2} - \Delta\tau_i G(\zeta_i, \rho_i))^{-1}(\Upsilon_i + \Delta\tau_i b(\zeta_i)\, q_i), \tag{5.10}$$

where $\Delta\tau_i$ is the integration step size. In Figure 5.4 we compare the BGF model, under the temperature, EGR flow, and throttle flow assumptions, against measurements from a wideband oxygen concentration sensor, situated immediately downstream of the VGT. The model is in good agreement with the measurements; the placement of the sensor induces a filtering effect which accounts for the error in transients.

5.4 Control Design

The objective of diesel airpath control is to promptly supply the torque requested by the driver, while maximizing fuel economy, respecting regulatory constraints on $\mathrm{NO_x}$ and particulate matter (PM), and limiting visible smoke. The inputs to the system are the engine speed, and the fueling rate target. The system outputs are split into vectors of measurements, and performance variables, defined respectively as,

$$y_m = \begin{bmatrix} p_{im} & \dot{m}_c & N_e \end{bmatrix}^T \text{ and } y_p = \begin{bmatrix} \psi_{OP} & \psi_{NO_x} & \psi_{THC} & \tau_q \end{bmatrix}^T. \tag{5.11}$$

The measurements, y_m, are available for feedback. The performance outputs: torque, exhaust opacity (smoke), $\mathrm{NO_x}$ concentration, and Total Hydrocarbon Concentration (THC), are measured for evaluation purposes but are not available for feedback. Note that THC is strongly correlated with particulate matter.

We concern ourselves primarily with engine-out emissions. Regulatory constraints are on tailpipe emissions, however, reducing engine-out emissions is an important intermediate step in meeting regulatory requirements. In addition, reduced engine-out $\mathrm{NO_x}$ emissions means a smaller urea selective catalytic reduction module can be used to meet the same tailpipe emissions targets. This could allow a manufacturer to use a lighter, cheaper aftertreatment system which can have positive effects on overall vehicle cost and fuel economy.

5.4.1 Architecture and Control Strategy

The overall architecture of the controller is shown in Figure 5.5. The outer-loop supervisory controller generates an EGR rate target and a fueling rate input. The fueling rate input is applied directly to the engine while the EGR rate target is passed to the inner-loop airpath controller. The airpath controller tracks EGR rate and intake pressure commands. The supervisory controller does not control p_{im}^{trg} which is instead obtained from a map. We chose this architecture because the SMPC controller is focused on combustion quality control and combustion quality is most sensitive to χ and q.

In typical engine control strategies, the EGR rate and intake pressure targets are static functions of the operating condition, i.e., $\bar{\chi}_{egr}(\rho) : \mathbb{R}^2 \to \mathbb{R}$ and $\bar{p}_{im}(\rho) : \mathbb{R}^2 \to \mathbb{R}$, and are implemented using lookup tables. These targets are obtained during engine development and are chosen to be "optimal" in steady state. The steady state maps can be chosen based on a variety of objectives e.g., maximizing fuel economy subject to emissions limits as well as other constraints, e.g., maximum temperature, pressure, etc. There is a complicated

tradeoff between fuel economy, $\mathrm{NO_x}$, and PM/THC emissions so determining the steady state maps is nontrivial. A variety of methods have been investigated in the literature for this purpose including constrained optimization techniques [152, 153], extremum seeking [154], and model based calibration [155]; engine manufacturers and suppliers also employ sophisticated proprietary methodologies.

We assume that these maps are provided and we focus on transient optimization, i.e., using MPC to shape the transient response of the system as it transitions between operating points. The steady state maps usually neglect dynamic effects, e.g., the intake and exhaust manifold filling dynamics, so transient shaping can have a significant impact on performance. For example, a significant portion of cumulative emissions production occurs in transients, see e.g., Figure 5.12, and drivability depends on the response speed of the system to fuel commands. Moreover, we have observed that remaining as close as possible to the steady-state targets is a very effective strategy, this is reflected in the SMPC formulation in Section 5.4.3 which minimizes deviation of χ^{trg} from $\bar{\chi}_{egr}$ and q from q^{trg} in a manner similar to a reference governor [156]. We hypothesize that this is because emissions formation occurs in the cylinders at a fast timescale and is essentially a quasi-static function of the airpath states. Thus remaining near the optimal steady state targets during transient operation is nearly optimal and provides good performance in practice (as we demonstrate in Section 5.6). In a previous paper [142], we explored simultaneous setpoint determination and constraint enforcement using Economic MPC. We found that, in practice, determining optimal setpoints online was slow and lead to poor performance. This motivated our current approach of "precomputing" the setpoints, i.e., using the predetermined setpoint maps $\bar{\chi}_{egr}(\rho)$ and $\bar{p}_{im}(\rho)$.

Most inner-outer loop architectures operate at different rates to minimize interactions between the loops. The supervisory and airpath controllers operate at the same update rate to allow the supervisory controller to respond to driver fuel requests as quickly as supported by the hardware. This minimizes delays between driver commands and the system response leading to a more responsive vehicle. We manage the possibility of interference between the loops by using closed-loop models that account for the inner loop controller as the SMPC prediction models. This technique is well established in the reference governor literature [156]. Further, it allows us to decompose constraint handling and nonlinearity compensation/integral action and makes the architecture modular; the inner loop airpath controller can be of any type, including e.g., PID or MPC etc. However, choice of inner loop controller will affect the performance of the overall system so using a high performance inner loop controller is advantageous. The results presented in this chapter all use a nonlinear MPC controller for the airpath controller. More details on this controller can be

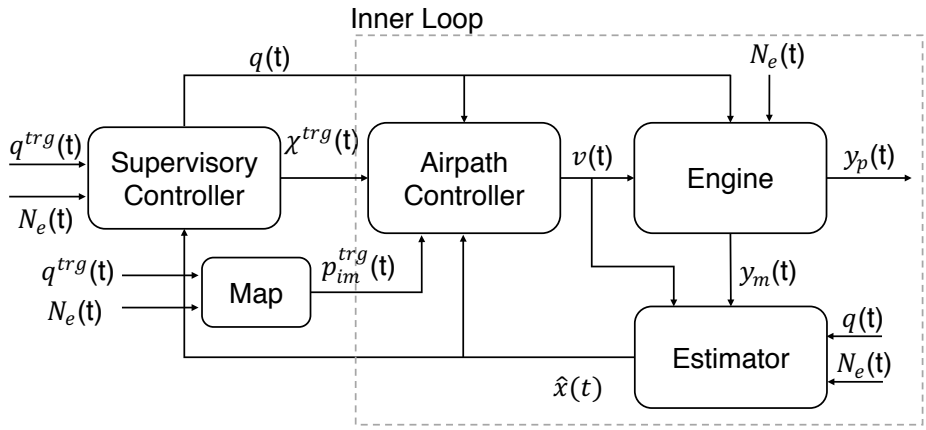

Figure 5.5: A schematic of the MPC control architecture; see Section 5.3 for notation. In this work, the supervisory controller is a supervisory MPC (SMPC) controller and the airpath or inner-loop controller is a nonlinear MPC (NMPC) controller.

found in [31, 143] and [30].

Remark 5.3. *The controller presented in this chapter does not use cylinder pressure,* $\mathrm{NO_x}$ *, opacity, or oxygen concentration sensors for feedback. As a result, it can be implemented using only sensors available in a standard production vehicle.*

5.4.2 Estimator Design

The EGR flow is estimated using a steady state mass balance equation, $\dot{m}_{egr} \approx \dot{m}_{cyl} - \dot{m}_c$, and the cylinder flow is estimated as a function of the operating condition and intake pressure using a static regression map. The EGR rate is calculated as $\chi = max(0, \frac{\dot{m}_{egr}}{\dot{m}_{cyl}})$. The burnt gas fractions are obtained by propagating (5.7). The normalized fuel-air ratio is estimated as

$$\psi(x, u, \rho) = \frac{\dot{m}_f(u, \rho)}{\dot{m}_{cyl}(p_{im}, \rho)(1 - F_1)} \left(\frac{A}{F}\right)_s, \tag{5.12}$$

where $(A/F)_s$ is the stoichiometric air-fuel ratio of the fuel. The cylinder flow is a function of p_{im} and ρ as described in (5.9) and $\dot{m}_f = c_q N_e q$ where $c_q > 0$ is a fixed constant that depends on the engine. Thus the fuel-air ratio is a function of the current operating condition, and the SMPC state and control vectors i.e., $\phi = \phi(x, u, \rho)$. The normalized fuel-air ratio is strongly correlated with smoke production as shown in Figure 5.6.

The intake temperature, exhaust temperature and the exhaust pressure estimates are obtained from the engine control unit (ECU). Some literature on constructing these estimators includes [157–159].

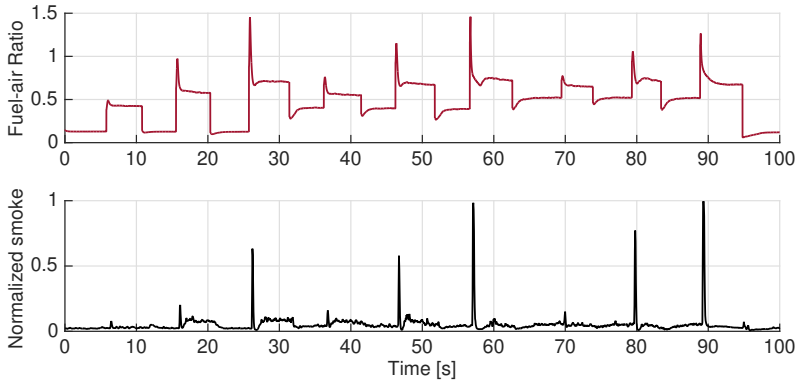

Figure 5.6: Correlation of the normalized fuel-air ratio with smoke.

5.4.3 Optimal Control Problem Definition

The objective of the supervisory layer is to enforce safety, and fuel-air ratio constraints to limit smoke during transients as unobtrusively as possible. In this context, safety is ensured by an upper bound on the fuel input and EGR rate to prevent damage to the engine. The following optimal control problem is solved at each sampling instance for the fueling rate and EGR target. The index i runs over the prediction horizon while the index k indicates the sampling instance; the notation $\xi_{i|k}$ indicates the predicted value of ξ at the ith step in the prediction horizon at time t_k.

$$\min_{s,\mu,\xi} \quad J(s,\xi,\rho_k) = \sum_{i=0}^{N} l(\mu_{i|k}, \mu_{i-1|k}, \rho_k, s), \tag{5.13a}$$

$$\text{s.t} \quad \xi_{i+1|k} = f_s(\xi_{i|k}, \mu_{i|k}, \rho_k, \Delta\tau_i),\ i = 0,\ \dots, N_s - 1, \tag{5.13b}$$

$$\xi_{0|k} = x_k, \tag{5.13c}$$

$$\phi(\xi_{i|k}, \mu_{i|k}, \rho_k) - \phi_l(N_{e,k}, \dot{m}_{c,k}) \leq s,\ i = 0,\ \dots, N_s - 1, \tag{5.13d}$$

$$0 \leq \chi^{trg}_{i|k} \leq \bar{\chi}_{egr}(\rho_k),\ i = 0,\ \dots, N_s - 1, \tag{5.13e}$$

$$0 \leq q_{i|k} \leq q^{trg}_k,\ i = 0,\ \dots, N_s - 1, \tag{5.13f}$$

$$s \geq 0, \tag{5.13g}$$

where $\mu = \big(\mu_{0|k},\ \dots, \mu_{N-1|k}\big)$, and $\xi = \big(\xi_{1|k},\ \dots,\ \xi_{N|k}\big)$. The OCP is parameterized by the current state estimate x_k and the current operating condition ρ_k. The quantity $\phi_l(N_e, \dot{m}_c)$ is the fuel-air ratio limit, it corresponds to when the engine begins to produce visible smoke and is a characteristic of the engine. It is determined experimentally during engine characterization, it increases with compressor flow and decreases with engine speed. The fuel-air ratio ϕ is computed using (5.12).

5.4.3.1 Stage Cost

The stage cost function is given by

$$l(\mu, \bar{\mu}, \rho, s) = \gamma_s(\chi^{trg} - \bar{\chi}_{egr}(\rho))^2 + \alpha_s(q^{trg} - q) + \beta_s s + ||\mu - \bar{\mu}||^2_{R_s}, \tag{5.14}$$

where $\alpha_s, \beta_s, \gamma_s > 0$, and $R_s \succ 0$ are weighting parameters, and reflects tracking objectives for the EGR rate target and fueling rate, a penalty to soften the fuel-air ratio (FAR) constraint to guarantee feasibility, and a damping term[1]. Since the cost function has no dependence on the system outputs, the fuel command and/or EGR rate target are only modified in response to predicted constraint violation, and, as a result, the SMPC controller can considered a hybrid between an MPC controller and a reference governor [156]. Together with (5.13d) and (5.13g) the slack penalty term, $\beta_s s$, defines an ℓ_1 softened constraint on the fuel-air ratio, which is used to limit smoke. The fuel tracking term, $\alpha(q^{trg} - q)$, which is equivalent to $\alpha_s|q^{trg} - q|$ due to (5.13f), promotes drivability. The remaining constraints, a lower bound on χ^{trg} and a fueling rate nonnegativity constraint in (5.13f), make the control constraint set compact. We use linear or 1-norm penalties for both the fuel tracking and fuel-air ratio constraints because they are more robust to ill-conditioning compared to quadratic penalties, ensuring that we are able to reliably solve (5.13) numerically.

5.4.3.2 Prediction Model

Achieving a sufficiently long prediction horizon to capture the dynamics of interest using a uniform prediction horizon discretization requires a large number of discrete timesteps. Unfortunately, additional timesteps introduce additional decision variables which increases computational complexity. As a countermeasure, we implemented a non-uniform integration timestep in the prediction model. Figure 5.7 illustrated the idea. This technique is related to move-blocking [160] and adaptive numerical integration [1]. It was introduced in [161], which also provides stability conditions for the case of continuous time LTI systems. We use the following function to determine the timestep sizes over the prediction horizon:

$$\Delta\tau_i = \begin{cases} t_s & i \leq 2, \\ 6 \cdot t_s & 2 < i \leq 4, \\ 40 \cdot t_s & 4 < i \leq 8. \end{cases} \tag{5.15}$$

The total length of the prediction horizon is approximately 1.1 sec and corresponds to $N_s = 8$ steps. The sampling period of the system is approximately 8 msec. The short steps

[1] $||x||^2_R = x^T R x$ for $R = R^T \succ 0$

ensure consistency between the model and the system, the medium steps capture emissions peaks, and the long steps capture the intake and exhaust pressure responses.

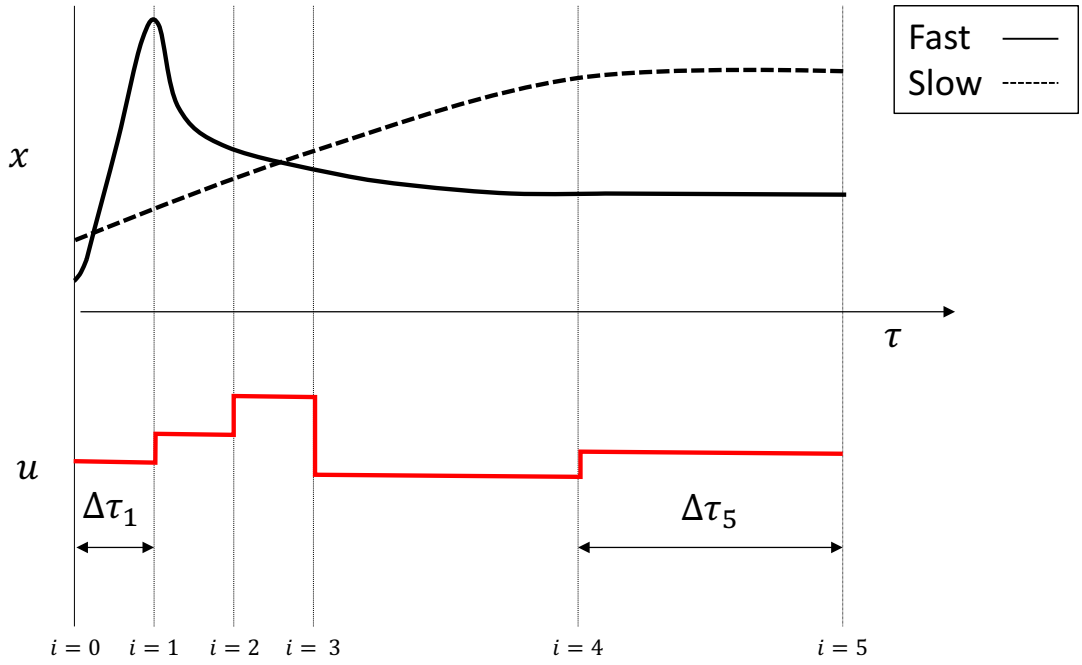

Figure 5.7: A diagram illustrating a non-uniform prediction horizon.

The prediction dynamics (5.13b) of the supervisory MPC controller are formed by combining the closed-loop airpath and EGR loop models described in Sections 5.3.1 and 5.3.2. The prediction model is thus given by,

$$\xi_{i+1} = f_s(\xi_i, \mu_i, \rho, \Delta\tau_i) = \begin{bmatrix} \zeta \\ \Upsilon \end{bmatrix}_{i+1} = \begin{bmatrix} F(\rho) + \bar{A}(\rho, \Delta\tau_i)\zeta_i + \bar{B}(\rho, \Delta\tau_i)\mu_i \\ [I_{2\times 2} - \Delta\tau_i G(\zeta_i, \rho, T_k)]^{-1}(\Upsilon_i + \Delta\tau_i bq_i) \end{bmatrix}, \tag{5.16}$$

which is the concatenation of a downsampled version of (5.6) and (5.10). The temperatures $T = (T_{im}, T_{ex})$ and operating condition ρ are assumed constant over the prediction horizon. The downsampled linear model matrices, $\bar{A}(\rho, \Delta\tau)$ and $\bar{B}(\rho, \Delta\tau)$ are computed as follows,

$$\bar{A}(\rho, \Delta\tau) = (A(\rho))^{\ell}, \quad \bar{B}(\rho, \Delta\tau) = \left(\sum_{j=0}^{\ell-1} (A(\rho))^j \right) B(\rho), \tag{5.17}$$

where $\ell = \Delta\tau / t_s$ is the downsampling factor.

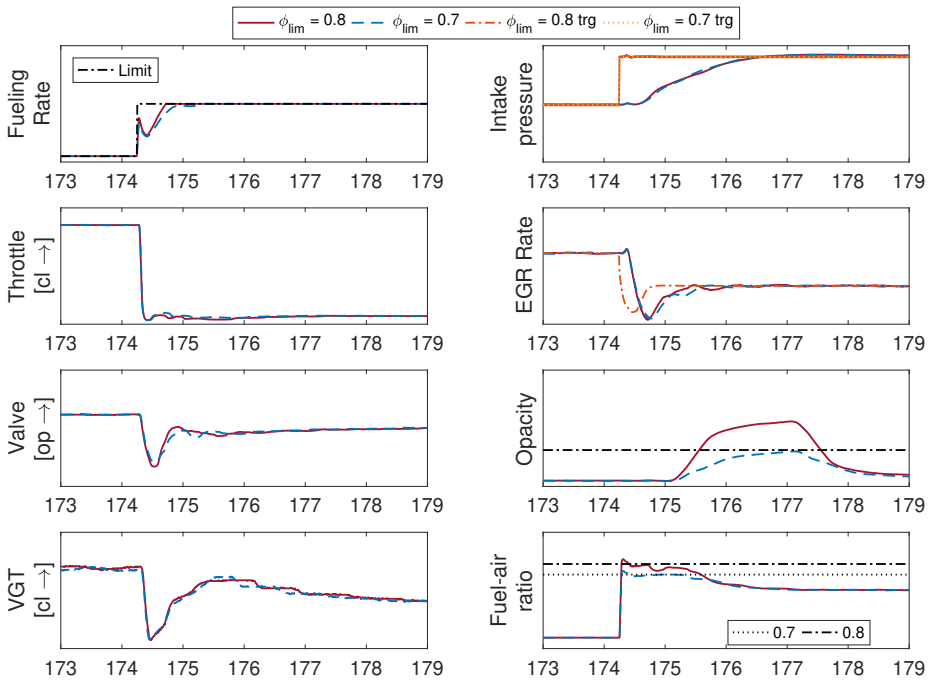

Figure 5.8: Experimental results of two tip ins (fuel step ups) at 2400 rpm using different fuel-air ratio limits. Recall that the SMPC controller computes the fueling rate and EGR rate target and the NMPC controller manipulates the valve, throttle, and VGT to track the intake pressure and EGR rate targets. When we set $\phi_{lim} = 0.8$ the opacity constraint is violated, reducing ϕ_{lim} to 0.7 eliminates the problem. The fuel-air ratio constraint is slightly violated in both cases due to an unmodelled 3 step delay between the MPC controller and the actuators. The y-axes scales have been removed to protect confidential data.

5.4.4 Illustrative Closed-loop Responses

Typical closed-loop responses under the SMPC controller are illustrated in Figure 5.8. During a tip-in (fuel step up) the fuel-air ratio rapidly increases as fuel is added to the system more quickly than the airflow can be raised to compensate. This causes the controller to predict a fuel-air ratio (FAR) constraint violation. In the responses shown, the fueling rate is filtered and the EGR rate target undershoots its steady state value to reduce F_1 and the fuel-air ratio as quickly as possible. Figure 5.9 illustrates the importance of the supervisory controller. Without the supervisory controller large amounts of visible smoke are produced while the SMPC controller successfully manipulates the fueling rate and EGR rate target to limit smoke production.

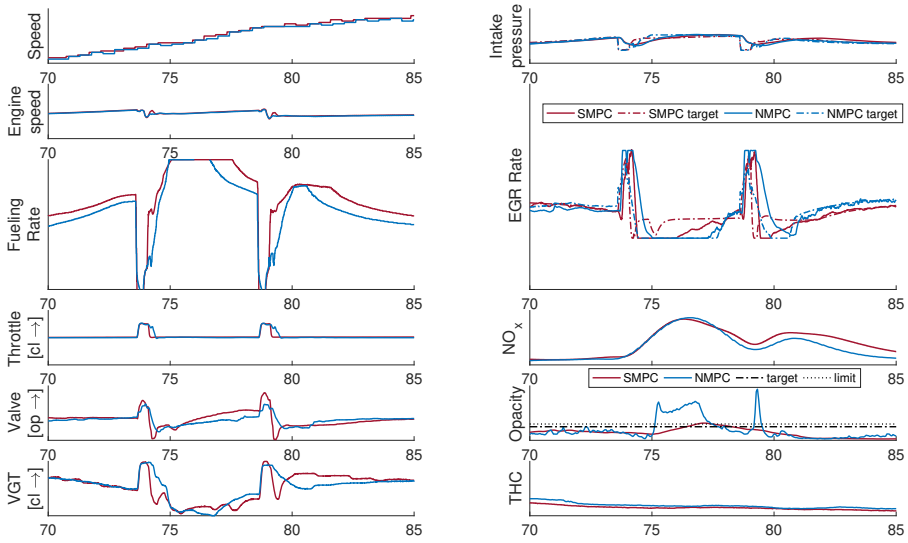

Figure 5.9: A comparison between the MPC controller with and without the supervisory controller. Without the supervisor massive amounts of visible smoke is produced; the opacity limit is where the smoke becomes visible. Time is measured in seconds. The y-axes scales have been removed to protect confidential data.

5.4.5 Stability and Feasibility

The SMPC controller uses some nonstandard techniques to achieve the performance required by the application. As a result, a complete a-priori stability analysis is beyond the scope of this dissertation. However, in lieu of an a-priori stability analysis on idealized models, we performed exhaustive simulations and hundreds of hours of bench testing to verify stability in practice. No stability issues arose during bench testing. This is a standard approach in engine control applications, see e.g., [64, 65], likely because engines are usually open-loop stable. However, existing theory was used to guide the design of the controllers.

We will prove recursive feasibility and provide a stability proof for a nominal case where $\Delta\tau_i = t_s$, the horizon is sufficiently long, the OCP is solved exactly, and in the absence of model mismatch. Since SMPC uses a closed loop model of the airpath controller and engine, this nominal analysis addresses loop interactions. In this section we will use $\mathbb{P}$ to denote the speed and fuel operating range of the engine. We will also denote the set of "safe", i.e., no misfires, surge, overpressure, etc., states by $\mathbb{X}(\rho)$. The set of admissible

pairs is

$$\mathbb{Y}(\rho) = \{(x,u) \mid \phi(x,u,\rho) \leq \phi_l(x,\rho),\ 0 \leq \chi^{trg} \leq \bar{\chi}_{egr},\ 0 \leq q \leq q^{trg}\} \tag{5.18}$$

and we define the set of admissible controls as $\mathbb{U}(x,\rho) = \{u \mid (x,u) \in \mathbb{Y}(\rho)\}$. Note that the constraint $x \in \mathbb{X}(\mathbb{P})$ is handled implicitly through careful selection of $\bar{\chi}_{egr}$ and $\bar{p}_{im}$, i.e., through good calibration.

Theorem 5.1. *(Feasibility) For any $\rho \in \mathbb{P}$, $x \in \mathbb{X}(\rho)$ the set $\mathbb{U}(x,p)$ is nonempty, implying that* (5.13) *is always feasible.*

Proof. Algebraic manipulation of (5.13d) yields,

$$q \leq q_{max}(\rho,x) = \phi_l(\rho,x)\frac{\dot{m}_{cyl}(1-F_1)}{N_e c}\left(\frac{A}{F}\right)_s^{-1}, \tag{5.19}$$

an explicit bound on the fuel. For any $x \in \mathbb{X}(\rho)$ the right hand side is positive thus $q_{max} \geq 0$ and the constraint $0 \leq q \leq \min(q_{max}, q^{trg})$ is feasible. The constraint $0 \leq \chi^{trg} \leq \bar{\chi}_{egr}(\rho)$ is feasible by construction which completes the proof. □

Theorem 5.2. *(Nominal stability of SMPC) Let the following assumptions hold:*

(A1) *(Stability of the inner loop) For all steady state admissible $\bar{u} = [\chi^{trg}\ q]$ and $\rho \in \mathbb{P}$, the inner-loop, with dynamics given by $x^+ = f(x,u,\rho)$, is asymptotically stable about the corresponding equilibrium point $\bar{x}(\bar{u},\rho)$ with region of attraction $\mathbb{X}(\rho)$.*

(A2) *(Asymptotic controllability) The inner-loop system is asymptotically controllable with respect to* (5.14) *in the sense of [1, Assumption 6.5].*

Then there exists a horizon length $N^* \in \mathbb{N}$ and a set $\mathbb{X}_s \subseteq \mathbb{X}$ such that if $N \geq N^*$ and $(x_0,u_0) \in \mathbb{X}_s \times \mathbb{R}^2$ then the sequence $\{(x_k,u_k)\} \to (\bar{x},\bar{u})$ as $k \to \infty$ with all $(x_k,u_k) \in \mathbb{Y}(\rho)$.

Proof. See appendix. □

Note that Theorem 5.2 only proves the existence of a sufficiently large N^*; we have to assume that the horizon length chosen is long enough. We employ move blocking related techniques to increase N, see Section 5.4.3. In practice, we observed no stability issues, lending some credence to our assumption.

The assumption (A1) is not particularly stringent, it just requires that the inner-loop controller be stabilizing and have integral action to provide zero offset tracking. Moreover, the engine is open-loop stable so achieving closed-loop stability isn't challenging.

It's difficult to verify (A2) a-priori since obtaining a high accuracy nonlinear model for the engine is extremely challenging. However, to support our assumption, we linearized (5.16) at each operating condition in the grid defined in (5.4) and observed that each linearized model was controllable. That is, we checked the controllability of $A_i = \nabla_x f_s(\bar{x}_i, \bar{u}_i)$ and $B_i = \nabla_u f_s(\bar{x}_i, \bar{u}_i)$ for all $i = 1, \dots 154$ where ρ_i is defined in (5.4), $\bar{u}_i = [\bar{\chi}_{egr}(\rho_i) \;\; q_i^{trg}]$ and $\bar{x}_i$ is the steady state associated with $\bar{u}_i$. This implies that (A2) holds in the vicinity of each operating point see e.g., [162]. This does not imply that (A2) holds globally but it does support that it holds in our region of interest. Further, in Section 5.6 we present experimental results that demonstrate the robustness of the SMPC controller a-posteriori.

5.5 Controller Implementation

Implementation of a model predictive controller requires an algorithm for (approximately) solving the optimal control problems online. This is challenging for the DAP due to its fast (8 msec) sampling rate and pronounced nonlinear dynamics. To overcome these challenges, we use a combination of time-distributed sequential quadratic programming, commonly known as the Real-time Iteration scheme, an early version of the FBstab solver known as Fischer-Burmeister Regularized and Smoothed (FBRS) [50], and symbolic differentiation, code optimization, and code generation techniques.

The supervisory controller OCP, (5.13), can be compactly represented as

$$\min_{\xi,\mu,s} \quad J(\mu, s, \rho), \tag{5.20a}$$

$$\text{s.t} \quad \xi = g(\mu, x, \rho), \tag{5.20b}$$

$$h(\xi, \mu, s, x, \rho) \leq 0, \tag{5.20c}$$

where ξ, μ and s are the decision variables, x is the SMPC state estimate, ρ is the operating condition, J is the cost function, g collects (5.13b) and (5.13c), and h collects the inequality constraints (5.13d) - (5.13g). These functions are implemented in symbolic form using the MAPLE computer algebra system using a custom set of symbolic tools developed as part of the project [26].

To reduce the size of the OCP and consequently computation time, we eliminate the equality constraints to for the so-called "condensed" OCP. Due to limitations in the symbolic tools, the formation of the function $\xi = g(\mu, x, \rho)$ was impractical. The algebraic expressions became too complicated and could not be easily manipulated due to

recursive composition of (5.13b). To bypass these issues, we instead linearize (5.13b) about the points x, ρ, and $\bar{\mu} = (0, q_k^{trg})$, leading to linearized set of equality constraints, $\xi = \nabla g(x, \bar{\mu}, \rho)\mu + g(x, \bar{\mu}, \rho)$ which are more easily formed. These were then eliminated via substitution, using the symbolic tools, resulting in the following condensed OCP

$$\min_{w} \quad J(w, \rho), \tag{5.21a}$$

$$\text{s.t} \quad c(w, x, \rho) \leq 0, \tag{5.21b}$$

where $w = (\mu, s)$ and $c(w, x, \rho) = h(\nabla g(x, \bar{\mu}, \rho) + g(x, \bar{\mu}, \rho), s, x, \rho)$.

We solve this in real-time using time-distributed SQP, which was discussed at length in Chapter 3, using the Gauss-Newton Hessian approximation[2] and with $\ell = 1$, i.e., one SQP iterations per timestep. Thus, to evaluate the optimization operator

$$z_k = \mathcal{T}_{\ell=1}(z_{k-1}, x_k, \rho_k) \tag{5.22}$$

where $z = (w, \lambda)$ and λ is the dual variable associated with (5.21b), we solve the following linear variational inequality (VI)

$$H_k \Delta w + f_k + G_k^T \lambda = 0, \tag{5.23a}$$

$$d_k - G_k \Delta w + \mathcal{N}_+(\lambda) \ni 0 \tag{5.23b}$$

where $H_k = \nabla_w^2 J(w_{k-1}, x_k, \rho_k) \succeq 0$, $f_k = \nabla_w J(w_{k-1}, x_k, \rho_k)$, $G_k = \nabla_w c(w_{k-1}, x_k, \rho_k)$, $d = -c(w_{k-1}, x_k, \rho_k)$, and $\mathcal{N}_+$ is the normal cone mapping of the non-negative orthant, for $(\Delta w^*, \lambda^*)$ and set $z_k = (w_{k-1} + \Delta w^*, \lambda^*)$. The VI (5.23) are the optimality conditions for the following QP:

$$\min_{\Delta w} \quad \frac{1}{2} \Delta w^T H_k \Delta w + f_k^T \Delta w, \tag{5.24a}$$

$$\text{s.t} \quad G_k \Delta w \leq d_k, \tag{5.24b}$$

we solve (5.24) using the FBRS method [50], which is a precursor to the FBstab method presented in Chapter 4.

Due to the presence of slack variables in the original OCP, each QP is guaranteed to be feasible and, due to the penalty on $\mu_i - \mu_{i-1}$ in the cost function, (5.14), the strong second order sufficient conditions always hold so that (5.24) has a unique solution. Fur-

[2]While the exact Hessian was available to us through the symbolic tools the Gauss-Newton Hessian approximation is guranteed to be positive semidefinite, which allows us to solve the resulting QPs reliably.

ther, the inequality constraints (5.13d)-(5.13g) in the SMPC OCP reduced to a set of (state dependant) box constraints, see Theorem 5.1, so the LICQ holds as well. Thus we expect its solution mapping to be strongly regular so that Assumption 3.4 holds. Moreover, we expect the ideal feedback law to be stabilizing by Theorem 5.2 and input-to-state stability follows from nominal stability under strong regularity of the solution mapping [10]. Thus we expect that Theorem 3.4 is applicable since we are using Gauss-Newton SQP, see Section 3.8. This conclusion is borne out by our experimental results, we observed that the OCP solution error was quite stable, see e.g., the residual plots in Figure 5.11.

Remark 5.4. *To guarantee finite-time execution, which is necessary for real-time implementation, the FBRS algorithm is limited to 5 Newton iterations per sampling period. FBRS is amenable to warmstarting so each QP is warmstarted with the previous solution. This was sufficient for convergence in almost every case, we observed no degradation of control performance due to the iteration limit.*

5.5.1 Executable Generation and Execution Time

The MPC controller was implemented on a dSPACE DS1006 rapid prototyping unit (2.6 GHz CPU clock speed) using Simulink 2010b SP2 real-time workshop. All necessary derivatives for (5.13) were calculated using symbolic methods. The tools, collectively referred to as the symbolic control design environment (SCDE) [26], translate symbolic expressions into highly optimized C code (and corresponding S-function templates) which meets the Motor Industry Software Reliability Association (MISRA) standard for embedded computations [163]. Expressions for the cost, constraints, and dynamics were written in the Maple symbolic language then processed and differentiated symbolically. The SCDE was then used to generate S-functions for all necessary derivatives. The FBRS QP solver was implemented in an embedded MATLAB block within Simulink.

Table 5.1: Summary problem sizes and execution times

	Average execution time [μs]	Maximum execution time [μs]	Variables	Constraints
SMPC	530	550	17	41

Execution times on the DS1006 rapid prototyping unit are shown in Table 5.1. The entire MPC controller takes approximately 550 μs to execute in the worst case, well below the sampling period of 8 ms. The total size of the real-time executable file was 5547

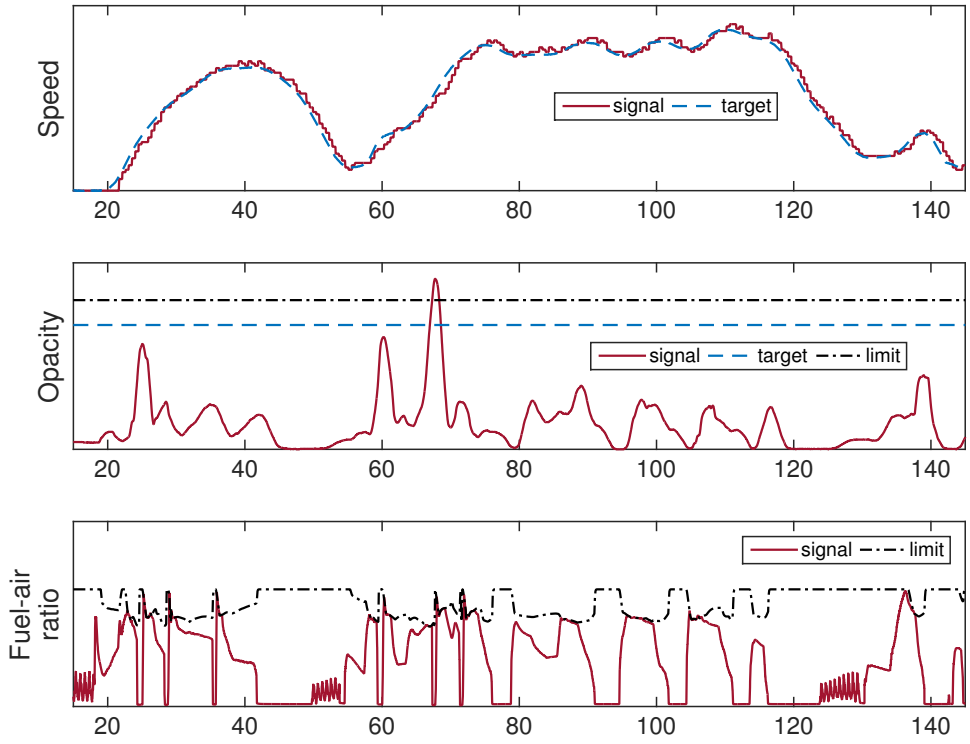

Figure 5.10: A portion of the WLTC with the MPC controller in closed loop with the engine. The vehicle was able to successfully complete the drivecycle despite some fuel limiting. The smoke is well controlled with only a few spikes reaching the edge of the visible range. Time is measured in seconds. The y-axes scales have been removed to protect confidential data.

KB. Note that since the number of Newton iterations performed by the SMPC QP solver was fixed at 5 the convergence check was disabled; as a result, the average and maximum execution times are similar.

5.5.2 Calibration of the Supervisory Controller

The supervisory controller has 4 tuning parameters, $\alpha_s, \beta_s, \gamma_s > 0$ and $R_s \succ 0$, which is a 2 by 2 matrix, and is straightforward to tune. Tunning is performed during fuel step experiments at fixed engine speeds. The fuel deviation parameter α_s is fixed to a sufficiently large value to ensure fuel tracking, we used $\alpha_s = 1$. The constraint softening parameter was set to $\beta_s = 1000$, γ_s was set to 0.05 and $R_s = diag(5, 0.1)$ was used.

5.6 Results

The MPC controller was placed in closed-loop with a 2.8L diesel engine on a transient engine dynomometer. In this section, we showcase the performance of the MPC controller; it was run over the Worldwide Harmonized Light Vehicles Test Cycle (WLTC) and New European Drive Cycle (NEDC) using a simulated vehicle and driver. The results are shown in Figures 5.10 to 5.14.

A summary of the results, using the best tunings obtained during testing, is shown in Table 5.2. Emissions performance was evaluated by estimating the cumulative mass of $\mathrm{NO_x}$ and THC emitted during the drivecycle, smoke was evaluated by integrating the exhaust opacity whenever it was above the visible limit[3]. Drivability was evaluated using the root mean square velocity tracking error (RMSE) between the longitudinal speed of the simulated vehicle and its target. Relative difference is defined as

$$\%\ \text{difference} = \frac{\text{DAP MPC} - \text{Benchmark}}{\text{Benchmark}} * 100. \tag{5.25}$$

Over the WLTC the DAP MPC controller was able to significantly reduce cumulative $\mathrm{NO_x}$ and THC, compared to a state of the art benchmark controller. The aggressive tuning resulted in slightly worse drivability compared to the benchmark and yielded significant $\mathrm{NO_x}$ and THC reductions at the cost of a small increase in smoke production. A more conservative tuning reduced smoke emissions compared to the benchmark but led to smaller $\mathrm{NO_x}$ and THC improvements; drivability was adversely affected as well.

Over the NEDC the DAP MPC controller slightly increased $\mathrm{NO_x}$ and slightly decreased THC. The NEDC cycle is not aggressive enough to trigger fuel limiting or cause visible smoke. The increase in fuel consumption is not large enough to be considered significant as the results are estimated by integrating the commanded fuel signal rather than measured using a fuel meter. The explicit incorporation of fuel-economy into the DAP MPC controller is a topic of future work.

[3]The visibility threshold used here is consistent with current calibration guidelines.

Table 5.2: Summary of the results obtained using the DAP MPC controller.

	WLTC aggressive tuning	WLTC conservative tuning	NEDC aggressive tuning
	[% difference]	[% difference]	[% difference]
NO_x	-16	-11	1.4
THC	-14	-4	-2
Fuel	0.67	0.54	0.9
V_{RMSE}	1	4	2
Smoke	24	-49	0

Figure 5.10 shows the responses when DAP MPC controller is in closed-loop with the engine during the high load portion of the WLTC. The smoke is well controlled, only a handful of spikes approach the boundary of the visible range.

Figure 5.11 shows a high speed portion of the WLTC. Between 165 and 170 s, the SMPC controller predicts fuel-air ratio constraint violation and, in response, lowers the EGR rate target and limits the fuel to enforce the constraint. As a result, the exhaust opacity remains below the visible limit. It also shows the QP (5.24) and nonlinear OCP (5.13) residuals. The QP residual is low showing that, thanks to warmstarting, FBRS is capable of producing a sufficiently accurate approximate solution of the QP despite the hard 5 iteration limit. Moreover, the stability of the OCP residual illustrates the efficacy of TD-SQP as predicted by Theorem 3.4.

Figure 5.12 shows the response of the DAP MPC controller during an acceleration event. As the vehicle speed increases, three gearshifts occur, each consisting of a tip out (fuel step down) followed by a tip in (fuel step up). In response to each tip in the SMPC controller predicts a fuel-air ratio constraint violation and reduces the EGR rate target, in order to empty the intake manifold of burnt gas, and limits the fuel to enforce the constraint and prevent visible smoke. The target then returns to the steady state EGR rate target as quickly as possible to reduce NO_x . The airpath controller is able to effectively track the EGR target signal, this fast EGR rate tracking ensures that the fuel-air ratio constraint satisfaction is accomplished primarily with valve and VGT actuation, rather than with fuel limiting, which is important for drivability.

Figure 5.13 shows an input-output (most benchmark signals cannot be shown for confidentiality reasons) comparison between the DAP MPC controller and a benchmark. The DAP MPC controller reduces transient NO_x by shrinking the spikes that occur after gearshifts. This is possible because the SMPC controller brings the EGR rate target back to its steady

state target quickly after a tip in by accurately calculating the fuel-air ratio response using its prediction model.

Figure 5.14 illustrates some issues we observed with the DAP MPC controller. Firstly, between 160 and 165 s the fuel is limited quite severely, despite the exhaust opacity being nowhere near the visible limit. This indicates there is room for improvement in the fuel-air ratio limit map (recall that $\phi_l = \phi_l(N_e, \dot{m}_c)$). Secondly, between 170 and 180 s the closed-loop system becomes oscillatory which is undesirable in view of increased wear on the actuators and oscillations in the engine torque. This occurs because the controller "bounces" off the fuel-air ratio constraint due to mismatch between the EGR rate tracking error predicted by the SMPC controller and the true tracking error. Future work will focus on appropriately softening constraints and improving the quality of the EGR rate model to alleviate bouncing, and adding a monotonicity constraint, similar to a reference governor [156], to the fueling rate command to prevent oscillations in the engine torque.

Remark 5.5. *The opacity target is the maximum opacity value allowed during steady state calibration. The opacity limit is when smoke becomes visible, it is desirable to minimize violation of this constraint. The smoke metric used in Table 5.2 is computed by integrating the opacity signal whenever its higher than the limit.*

5.7 Discussion

While our proposed MPC controller is promising, more development effort is necessary before it can be considered "production ready". Here we discuss the challenges we observed leading up to and during our experimental campaign and outline directions for continued controller development.

5.7.1 Computational Footprint

One of the largest drawbacks of MPC is the computational footprint. Our controller uses the real-time iteration scheme [63], a kind of time-distributed sequential quadratic programming [28], a new QP solver [50], and symbolic code generation tools [26] to reduce the worst case total execution time to around $715\mu s$ on a 2.6 GHz rapid prototyping unit. Assuming a 256 MHz ECU and estimating the computation time using a clock speed scaling analysis [4] we arrive at an estimate of $0.715ms \cdot \frac{2.6\ GHz}{256\ MHz} = 7.26\ ms$ which is slightly

[4] We have found clock scaling analyses to be sufficiently accurate for first order computation time estimates due to the relative simplicity of typical ECU computational architectures.

below the current $8\ ms$ sampling period. This indicates that our strategy is likely to be implementable in real-time on an appropriate dedicated ECU. However, ECUs must run other modules besides the engine controller so further reductions in the CPU usage are necessary before the controller is production ready.

Since completing the experiments, we have performed preliminary investigations into the use of Kylov methods, specifically the conjugate gradient method [164], for inexactly solving the linear systems arising in FBstab/FBRS [165]. We have found that significant improvements are possible without noticeably degrading controller performance. We believe this approach could be promising for solving (5.24). Another possible avenue for improvement is the use of optimized mathematical subroutines. Our current implementation uses symbolically generated or handwritten linear algebra and factorization routines. It is well known that specialized routines (e.g., those provided by high performance BLAS or LAPACK libraries) can offer much better performance; they are often optimized at an assembly code level. Further, embedded BLAS libraries have recently begun to appear [27]. We believe that routines of this type for ECUs could help further reduce CPU usage. Finally, we expect ECU hardware to continue to improve over time. In particular, the incorporation of dedicated signal processing units into future ECUs could further reduce computation times.

5.7.2 Performance Issues

We observed some performance issues during experimental testing. Specifically, the MPC controller was not able to perfectly control smoke see e.g., Figure 5.10, and we observed some undesirable oscillations in the fueling rate input, see e.g., Figure 5.14.

Typically, when using MPC, the best way to improve constraint handling is to improve the prediction models. The least accurate portion of our prediction models is the intake manifold burnt gas fraction F_1 in (5.7). Since our experimental setup did not have an oxygen sensor in the intake manifold, we were unable to directly validate our F_1 predictions and/or improve the model using data. As our smoke control approach depends directly on F_1 inaccurate predictions of F_1 are the most likely cause of the performance degradation we observed. We suspect that installing the sensor on the testbed (for modelling and validation purposes only, i.e., not for use as feedback) and an associated data collection and modelling effort could significantly improve smoke control. Moreover, as this modelling error forced us to adopt a more conservative smoke limit, we hypothesize that this caused the underperformance we observed on the less aggressive NEDC cycle where the conservatism was not warranted. Note that the benchmark controller's smoke control is also not

perfect.

Beyond improving the quality of the models, various modifications to the SMPC controller formulation could be considered. For example, re-parameterizing the fueling rate input in (5.13) as

$$q = k(q^{trg} - q^{-}) + q^{-} \tag{5.26}$$

where $k \in [0, 1]$ is the new decision variable, q^{-} is the input applied at the previous sampling instant and q^{trg} is the target, will force the fueling rate input to approach the target monotonically and thus remove oscillations. Such a parameterization is often used in reference governors [156].

5.7.3 Future Perspectives

Our proposed MPC controller offers several advantages over a typical industrial PID based engine control strategy:

- Statement of the control objective is intuitive and can be easily summarized through the OCP formulation;
- The use of prediction models reduces the amount of conservatism needed to handle constraints, leading to performance improvements;
- It straightforwardly handles multiple input and outputs, coupling between them, and nonlinearities;
- It can be calibrated quickly.

It also suffers from some drawbacks:

- It requires an understanding of more advanced control engineering concepts to troubleshoot issues;
- It has a higher computational footprint;
- Construction of the prediction models can be labour intensive.

Overall, we demonstrated that significant $(10 - 15\%)$ emissions reductions are possible without hardware changes[5], e.g., without adding additional sensors or actuators, and that the challenges associated with the computational burden of MPC are not insurmountable,

[5]This can help manufacturers meet emissions targets without major cost increases.

even when using nonlinear MPC. We also demonstrated various techniques such as non-uniform prediction horizons and inner-outer loop MPC control architectures that may be useful in a broader context.

We believe that, as emissions regulations become tighter, MPC has the potential to play a significant role in emissions reduction efforts. However, progressing from demonstrations to mass production would require significant amounts of effort both in terms of modelling, controller improvement and computational footprint reduction, as outlined in the previous two subsections. It would also require creating standardized tools for design, calibration, simulation, modelling and deployment, a significant experimental validation campaign, and a significantly larger team of engineers and researchers. See e.g., the acknowledgments section in [166] which details the deployment of linear MPC in production by a major automotive manufacturer [166].

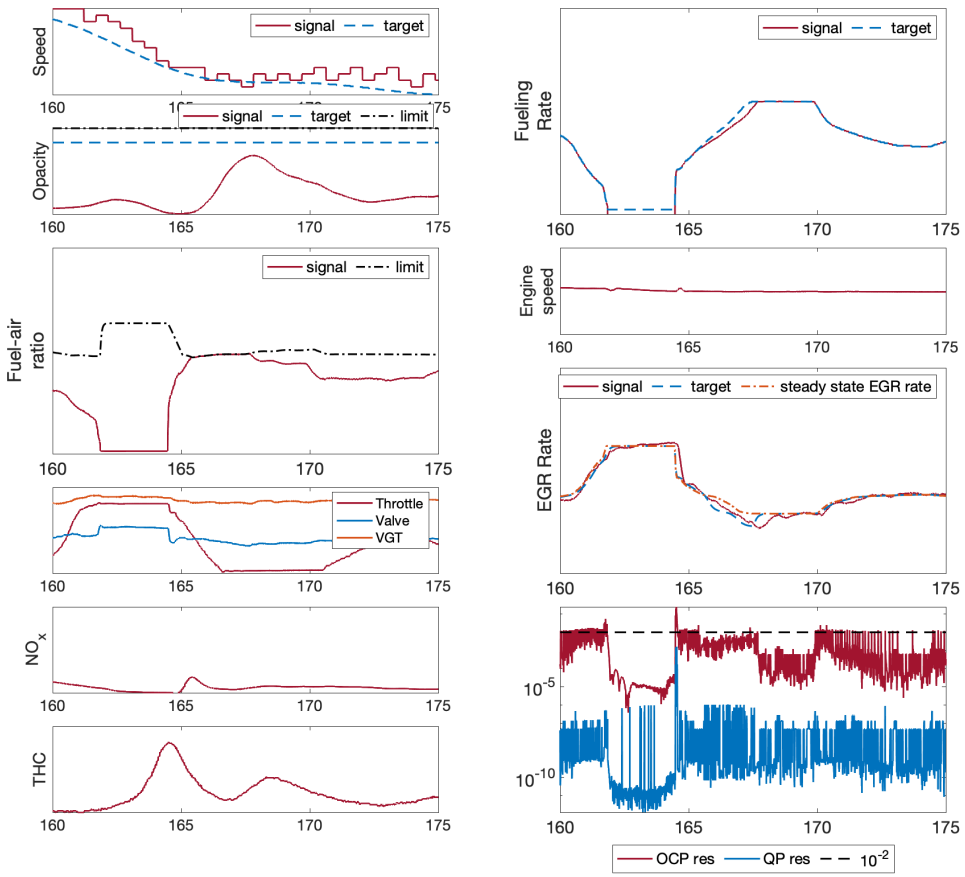

Figure 5.11: A close-up of a high speed potion of the WLTC with the MPC controller in closed-loop with the engine. The supervisory layer prevents visible smoke by coordinating the fuel input and the EGR rate. It causes the EGR rate target to undershoot the steady state optimum, this command is tracked by the airpath controller, and limits the fueling rate to satisfy the fuel-air ratio constraint and prevent visible smoke. The residual of the the QP (5.24), seen as a function of time, is well controlled showcasing that FBRS is able to exploit warmstarting to effectively solve the QP despite a tight iteration limit. Moreover, the residual of the nonlinear OCP (5.13) is robstly stable (indeed we observe oscillations that are caused by the measurement noise) as predicted by Theorem 3.4. Time is measured in seconds. The y-axes scales have been removed to protect confidential data.

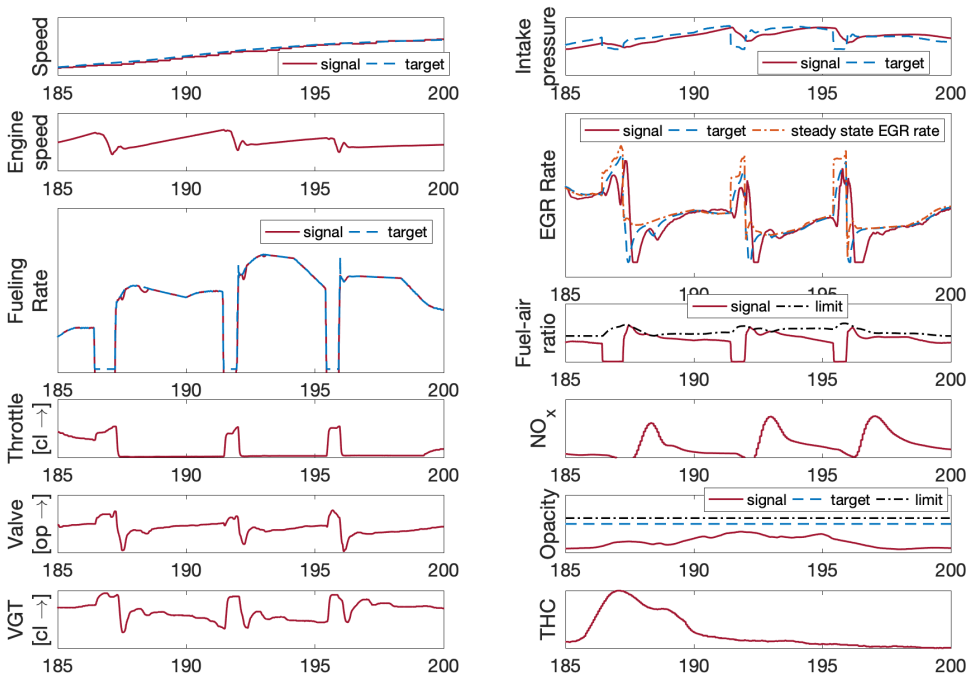

Figure 5.12: A close-up of an acceleration event with the MPC controller in closed-loop with the engine during the WLTC. During the tip ins following the gearshifts the supervisory layer commands the airpath controller to quickly reduce the EGR rate and limits the fuel to satisfy the fuel-air ratio constraint, preventing visible smoke. Time is measured in seconds. The y-axes scales have been removed to protect confidential data.

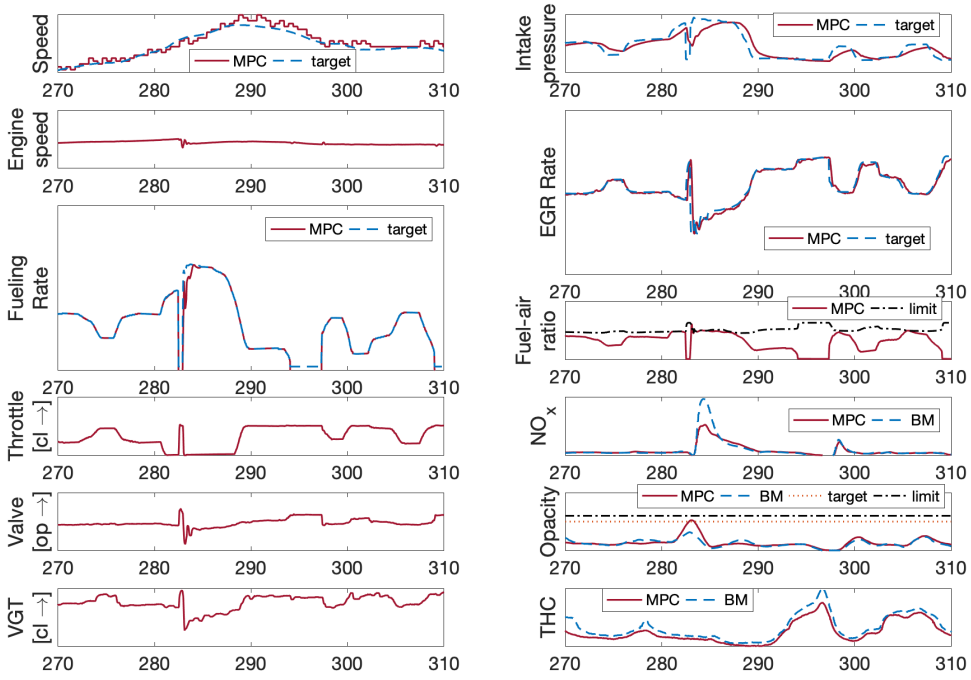

Figure 5.13: A comparison between the MPC and a benchmark (BM) strategy running in closed-loop with the engine during the WLTC. The SMPC controller is able to reduce NO_x emissions compared to the benchmark strategy without causing visible smoke or increasing hydrocarbon output. All signals are generated by the MPC controller unless otherwise noted. Time is measured in seconds. The y-axes scales have been removed to protect confidential data.

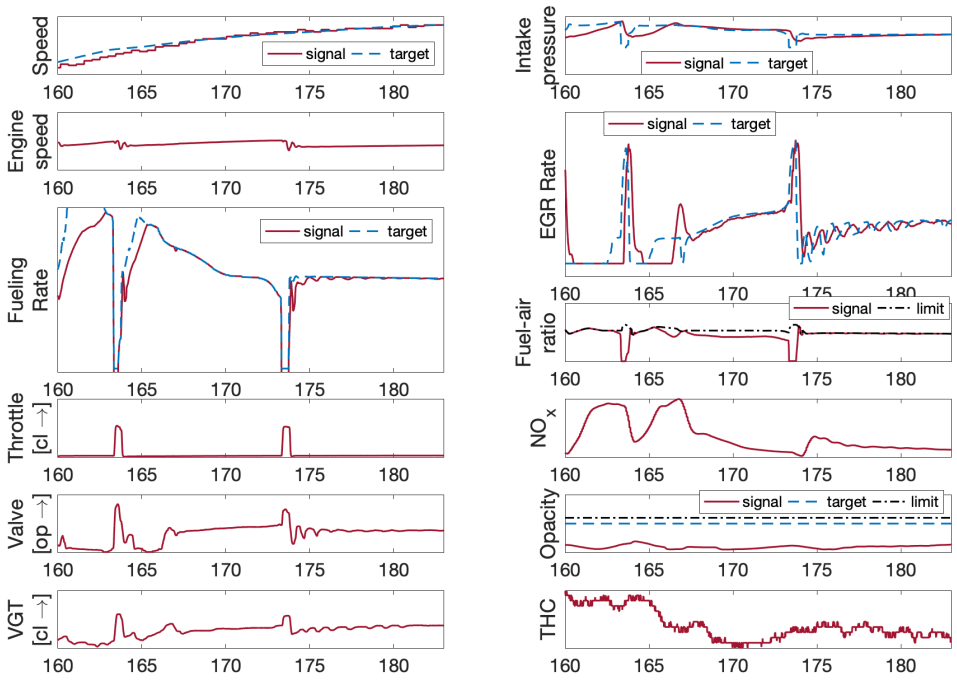

Figure 5.14: A close-up for a hard acceleration event with the MPC controller in closed-loop with the engine during the WLTC. The SMPC controller aggressively limits the fuel despite the low opacity of the exhaust. Oscillation against the fuel-air ratio constraint boundary also occurs which leads to undesirable EGR rate and torque fluctuations. Time is measured in seconds. The y-axes scales have been removed to protect confidential data.

CHAPTER 6

Conclusions and Future Work

6.1 Conclusions

This dissertation has focused on improving numerical methods for model predictive control (MPC) with the goal of helping to enable the deployment of MPC on systems with pronounced nonlinear dynamics, fast sampling rates, and limited onboard computational resources. Specifically, it focused on the common strategy of warmstarting optimization algorithms and the consequences of warmstarting in the context of real-time optimization and MPC.

First, we analyzed the effect of warmstarting and truncation, i.e., time-distributed optimization, on the closed-loop properties of MPC. We presented a general framework for systems theoretic analyses of model predictive controllers implemented using TDO that covers a broad range of MPC formulations and optimization algorithms. Using this framework, we showed it is possible to recover the desirable qualitative properties, namely stability, robustness, and constraint satisfaction, of the optimal MPC feedback law using a finite amount of computational effort. When specialized to time-distributed sequential quadratic programming, our result significantly extends the existing stability analysis of the real-time iteration scheme by explicitly considering inequality constraints, analyzing the effect of performing additional SQP iterations, considering a wider class of Hessian approximations, and proving local input-to-state stability of the closed-loop system.

Second, we proposed FBstab, the proximally stabilized Fischer-Burmeister method for convex quadratic programming. FBstab is attractive for real-time optimization because it is easy to code, numerically robust, easy to warmstart, can exploit sparsity, and converges or detects infeasibility under only the assumption that the Hessian of the quadratic program is positive semidefinite. We provided convergence, rate of convergence, and infeasibility detection analyses for the method in addition to numerical experiments benchmarking the solver. Moreover, we have released our MATLAB implementation of FBstab online

at https://github.com/dliaomcp/fbstab-matlab.git under an permissive open source license.

Finally, we applied some of the theoretical and algorithmic ideas developed in this dissertation to the problem of diesel engine emissions control. The control objective is to reduce $\mathrm{NO_x}$ and PM emissions and enforce smoke constraints while maintaining drivability and fuel economy. We developed a supervisory emissions-oriented model predictive controller which, when used in tandem with previously developed nonlinear model predictive airpath controller, was able to significantly reduce emissions over the WLTC drive cycle with only small impacts on drivability. Moreover, we demonstrated that, using a combination of TDO, a version of FBstab, and advanced symbolic differentiation/codegeneration tools, it is possible to reliably implement nonlinear MPC at a sampling rate of more than 100 Hz on a rapid prototyping system.

6.2 Future Work

Real-time Estimation and Machine Learning

The material in this dissertation focused on control, however real-time optimization is also broadly applicable in perception and estimation. In particular, investigating TDO applied to estimation/perception algorithms, such as receding horizon estimation, or online machine learning methods, and analyzing the effect of suboptimality on closed-loop performance is a promising direction for future research. Moreover, designing specialized numerical algorithms for these optimization problems might also prove to be impactful.

Time-distributed Optimization in Sampled Data Systems

The analysis of TDO presented in Chapter 3 is entirely in discrete time. However, many physical systems of interest are continuous time systems; in these situations MPC is typically implemented as a sampled data feedback law. Extending our analysis of TDO to the sampled data case would allow us to consider a broader class of systems and understand how parameters such as the sampling time, that are only present in sampled data systems, affect the systems theoretic properties of the closed-loop system.

A C++ implementation of FBstab

We are current in the process of developing an open source (and liberally licensed) C++ implementation of FBstab. The goal is to provide a useful tool for implementing real-time optimization on robotic and industrial platforms. The solver is under active development and is available at: https://github.com/dliaomcp/fbstab.git. We are also

continuously improving the solver by adding heuristics and improving the efficiency of the linear algebra.

Improving the Diesel Engine Controller

While successful as a technology demonstration, more work is required before the DAP MPC controller is ready to be deployed on production vehicles. The next steps include further reducing computation time, improving the gas fraction and airpath models, re-parameterizing the inputs to prevent fuel input oscillations, and in-vehicle testing. Moreover, the application of the same techniques to e.g., gasoline engines or heavy-duty diesel engines (e.g., semi trucks) would also be interesting research directions.

Bibliography

[1] L. Grüne and J. Pannek, "Nonlinear model predictive control," in *Nonlinear Model Predictive Control*, pp. 45–69, Springer, 2017.

[2] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[3] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: Theory, Computation and Design*. Nob Hill Pub., 2018.

[4] T. Samad, "A survey on industry impact and challenges thereof [technical activities]," *IEEE Control Systems Magazine*, vol. 37, no. 1, pp. 17–18, 2017.

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

[6] G. Grimm, M. J. Messina, S. E. Tuna, and A. R. Teel, "Model predictive control: for want of a local control lyapunov function, all is not lost," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 546–558, 2005.

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

[8] L. Magni, D. M. Raimondo, and R. Scattolini, "Regional input-to-state stability for nonlinear model predictive control," *IEEE Transactions on automatic control*, vol. 51, no. 9, pp. 1548–1553, 2006.

[9] D. L. Marruedo, T. Alamo, and E. Camacho, "Input-to-state stable MPC for constrained discrete-time nonlinear systems with bounded additive uncertainties," in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 4, pp. 4619–4624, IEEE, 2002.

[10] D. Limon, T. Alamo, D. Raimondo, D. M. De La Peña, J. Bravo, A. Ferramosca, and E. Camacho, "Input-to-state stability: a unifying framework for robust model predictive control," in *Nonlinear model predictive control*, pp. 1–26, Springer, 2009.

[11] G. Grimm, M. J. Messina, S. E. Tuna, and A. R. Teel, "Examples when nonlinear model predictive control is nonrobust," *Automatica*, vol. 40, no. 10, pp. 1729–1738, 2004.

[12] G. Pannocchia, M. Gabiccini, and A. Artoni, "Offset-free mpc explained: novelties, subtleties, and applications," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 342–351, 2015.

[13] M. Morari and U. Maeder, "Nonlinear offset-free model predictive control," *Automatica*, vol. 48, no. 9, pp. 2059–2067, 2012.

[14] P. O. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Transactions on Automatic Control*, vol. 44, no. 3, pp. 648–654, 1999.

[15] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Conditions under which suboptimal nonlinear MPC is inherently robust," *Systems & Control Letters*, vol. 60, no. 9, pp. 747–755, 2011.

[16] D. A. Allan, C. N. Bates, M. J. Risbeck, and J. B. Rawlings, "On the inherent robustness of optimal and suboptimal nonlinear MPC," *Systems and Control Letters*, vol. 106, pp. 68 – 78, 2017.

[17] B. Kouvaritakis and M. Cannon, *Model predictive control: Classical, Robust and Stochastic*. Springer, 2016.

[18] F. Allgöwer, T. A. Badgwell, J. S. Qin, J. B. Rawlings, and S. J. Wright, "Nonlinear predictive control and moving horizon estimation-an introductory overview," in *Advances in control*, pp. 391–449, Springer, 1999.

[19] P. TøNdel, T. A. Johansen, and A. Bemporad, "An algorithm for multi-parametric quadratic programming and explicit mpc solutions," *Automatica*, vol. 39, no. 3, pp. 489–497, 2003.

[20] D. Axehill, T. Besselmann, D. M. Raimondo, and M. Morari, "A parametric branch and bound approach to suboptimal explicit hybrid mpc," *Automatica*, vol. 50, no. 1, pp. 240–246, 2014.

[21] K. I. Kouramas, N. P. Faísca, C. Panos, and E. N. Pistikopoulos, "Explicit/multi-parametric model predictive control (mpc) of linear discrete-time systems by dynamic and multi-parametric programming," *Automatica*, vol. 47, no. 8, pp. 1638–1645, 2011.

[22] T. A. Johansen, "On multi-parametric nonlinear programming and explicit nonlinear model predictive control," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, vol. 3, pp. 2768–2773, IEEE, 2002.

[23] E. Pistikopoulos, "Perspectives in multiparametric programming and explicit model predictive control," *AIChE journal*, vol. 55, no. 8, pp. 1918–1925, 2009.

[24] K. Ling, S. Yue, and J. Maciejowski, "A fpga implementation of model predictive control," in *2006 American Control Conference*, pp. 6–pp, IEEE, 2006.

[25] 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, 2011.

[26] K. Walker, B. Samadi, M. Huang, J. Gerhard, K. Butts, and I. Kolmanovsky, "Design environment for nonlinear model predictive control," tech. rep., SAE Technical Paper, 2016.

[27] G. Frison, D. Kouzoupis, T. Sartor, A. Zanelli, and M. Diehl, "Blasfeo: Basic linear algebra subroutines for embedded optimization," *ACM Transactions on Mathematical Software (TOMS)*, vol. 44, no. 4, pp. 1–30, 2018.

[28] D. Liao-McPherson, M. Nicotra, and I. Kolmanovsky, "Time-distributed optimization for real-time model predictive control: Stability, robustness, and constraint satisfaction," *Automatica*, vol. 117, p. 108973, 2020.

[29] D. Liao-McPherson and I. Kolmanovsky, "FBstab: A proximally stabilized semismooth algorithm for convex quadratic programming," *Automatica*, vol. 113, p. 108801, 2020.

[30] D. Liao-McPherson, K. Shinhoon, M. Huang, M. Shimada, K. Butts, and I. Kolmanovsky, "Model predictive emissions control of a diesel engine airpath: Design and experimental evaluation," *Conditionally accepted by the International Journal of Robust and Nonlinear Control*, September 2019.

[31] M. Huang, "Low complexity model predictive control of a diesel engine airpath.," 2016.

[32] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*, vol. 317. Springer Science & Business Media, 2009.

[33] A. L. Dontchev and R. T. Rockafellar, "Implicit functions and solution mappings," *Springer Monogr. Math.*, 2009.

[34] A. L. Dontchev, M. Krastanov, R. T. Rockafellar, and V. M. Veliov, "An Euler–Newton continuation method for tracking solution trajectories of parametric variational inequalities," *SIAM Journal on Control and Optimization*, vol. 51, no. 3, pp. 1823–1840, 2013.

[35] L. Qi and J. Sun, "A nonsmooth version of Newton's method," *Mathematical programming*, vol. 58, no. 1, pp. 353–367, 1993.

[36] F. H. Clarke, *Optimization and nonsmooth analysis*. SIAM, 1990.

[37] Z.-P. Jiang, Y. Lin, and Y. Wang, "Nonlinear small-gain theorems for discrete-time feedback systems and applications," *Automatica*, vol. 40, no. 12, pp. 2129–2136, 2004.

[38] Z.-P. Jiang and Y. Wang, "Input-to-state stability for discrete-time nonlinear systems," *Automatica*, vol. 37, no. 6, pp. 857–869, 2001.

[39] E. K. Ryu and S. Boyd, "Primer on monotone operator methods," *Appl. Comput. Math*, vol. 15, no. 1, pp. 3–43, 2016.

[40] S. M. Robinson, "Strongly regular generalized equations," *Mathematics of Operations Research*, vol. 5, no. 1, pp. 43–62, 1980.

[41] A. F. Izmailov and M. V. Solodov, *Newton-type methods for optimization and variational problems*. Springer, 2014.

[42] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM journal on control and optimization*, vol. 14, no. 5, pp. 877–898, 1976.

[43] N. Parikh, S. Boyd, *et al.*, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[44] H. Rademacher, "Über partielle und totale differenzierbarkeit von funktionen mehrerer variabeln und über die transformation der doppelintegrale," *Mathematische Annalen*, vol. 79, no. 4, pp. 340–359, 1919.

[45] L. Qi, "C-differentiability, c-differential operators and generalized newton methods," *Applied Mathematics Report AMR96/5, University of New South Wales, Sydney, Australia*, 1996.

[46] X. Chen, L. Qi, and D. Sun, "Global and superlinear convergence of the smoothing newton method and its application to general box constrained variational inequalities," *Mathematics of Computation of the American Mathematical Society*, vol. 67, no. 222, pp. 519–540, 1998.

[47] B. Chen, X. Chen, and C. Kanzow, "A penalized Fischer-Burmeister NCP-function," *Mathematical Programming*, vol. 88, no. 1, pp. 211–216, 2000.

[48] L. Qi, "Convergence analysis of some algorithms for solving nonsmooth equations," *Mathematics of operations research*, vol. 18, no. 1, pp. 227–244, 1993.

[49] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[50] D. Liao-McPherson, M. Huang, and I. Kolmanovsky, "A regularized and smoothed fischer-burmeister method for quadratic programming with applications to model predictive control," *IEEE Transactions on Automatic Control*, pp. 1–1, 2018.

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

[52] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *Control Conference (ECC), 2013 European*, pp. 3071–3076, IEEE, 2013.

[53] C. V. Rao, S. J. Wright, and J. B. Rawlings, "Application of interior-point methods to model predictive control," *Journal of optimization theory and applications*, vol. 99, no. 3, pp. 723–757, 1998.

[54] K. Graichen and A. Kugi, "Stability and incremental improvement of suboptimal MPC without terminal constraints," *IEEE Transactions on Automatic Control*, vol. 55, no. 11, pp. 2576–2580, 2010.

[55] L. Grüne and J. Pannek, "Analysis of unconstrained NMPC schemes with incomplete optimization," in *Proceedings of the 8th IFAC Symposium on Nonlinear Control Systems–NOLCOS*, pp. 238–243, 2010.

[56] K. Graichen, "A fixed-point iteration scheme for real-time model predictive control," *Automatica*, vol. 48, no. 7, pp. 1300–1305, 2012.

[57] A. Steinboeck, M. Guay, and A. Kugi, "A design technique for fast sampled-data nonlinear model predictive control with convergence and stability results," *International Journal of Control*, pp. 1–17, 2017.

[58] R. Van Parys, M. Verbandt, J. Swevers, and G. Pipeleers, "Real-time proximal gradient method for embedded linear mpc," *Mechatronics*, vol. 59, pp. 1–9, 2019.

[59] M. M. Nicotra, D. Liao-McPherson, and I. V. Kolmanovsky, "Embedding constrained model predictive control in a continuous-time dynamic feedback," *IEEE Transactions on Automatic Control*, vol. 64, pp. 1932–1946, May 2019.

[60] V. M. Zavala and L. T. Biegler, "The advanced-step NMPC controller: Optimality, stability and robustness," *Automatica*, vol. 45, no. 1, pp. 86–93, 2009.

[61] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear MPC: bridging the gap via the real-time iteration," *International Journal of Control*, pp. 1–19, 2016.

[62] 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, 2014.

[63] M. Diehl, H. G. Bock, and J. P. Schlöder, "A real-time iteration scheme for nonlinear optimization in optimal feedback control," *SIAM Journal on control and optimization*, vol. 43, no. 5, pp. 1714–1736, 2005.

[64] T. Albin, D. Ritter, N. Liberda, R. Quirynen, and M. Diehl, "In-vehicle realization of nonlinear MPC for gasoline two-stage turbocharging airpath control," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1606–1618, 2018.

[65] Q. Zhu, S. Onori, and R. Prucka, "An economic nonlinear model predictive control strategy for si engines: Model-based design and real-time experimental validation," *IEEE Transactions on Control Systems Technology*, no. 99, pp. 1–15, 2017.

[66] A. Ilzhöfer, B. Houska, and M. Diehl, "Nonlinear MPC of kites under varying wind conditions for a new class of large-scale wind power generators," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 17, no. 17, pp. 1590–1599, 2007.

[67] M. Vukov, W. Van Loock, B. Houska, H. J. Ferreau, J. Swevers, and M. Diehl, "Experimental validation of nonlinear MPC on an overhead crane using automatic code generation," in *American Control Conference (ACC), 2012*, pp. 6264–6269, IEEE, 2012.

[68] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," in *Control Conference (ECC), 2013 European*, pp. 4136–4141, IEEE, 2013.

[69] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[70] M. Diehl, I. Uslu, R. Findeisen, S. Schwarzkopf, F. Allgöwer, H. G. Bock, T. Bürner, E. D. Gilles, A. Kienle, J. P. Schlöder, *et al.*, "Real-time optimization for large scale processes: Nonlinear model predictive control of a high purity distillation column," in *Online Optimization of Large Scale Systems*, pp. 363–383, Springer, 2001.

[71] S. Gros, "An economic NMPC formulation for wind turbine control," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pp. 1001–1006, IEEE, 2013.

[72] B. Houska, H. J. 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.

[73] R. Verschueren, G. Frison, D. Kouzoupis, N. van Duijkeren, A. Zanelli, R. Quirynen, and M. Diehl, "Towards a modular software package for embedded optimization," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 374–380, 2018.

[74] M. Diehl, R. Findeisen, F. Allgöwer, H. G. Bock, and J. P. Schlöder, "Nominal stability of real-time iteration scheme for nonlinear model predictive control," *IEE Proceedings-Control Theory and Applications*, vol. 152, no. 3, pp. 296–308, 2005.

[75] V. M. Zavala and M. Anitescu, "Real-time nonlinear optimization as a generalized equation," *SIAM Journal on Control and Optimization*, vol. 48, no. 8, pp. 5444–5467, 2010.

[76] D. Liao-McPherson, M. Nicotra, and I. Kolmanovsky, "A semismooth predictor corrector method for real-time constrained parametric optimization with applications in model predictive control," in *2018 IEEE Conference on Decision and Control (CDC)*, Dec 2018.

[77] Q. T. Dinh, C. Savorgnan, and M. Diehl, "Adjoint-based predictor-corrector sequential convex programming for parametric nonlinear optimization," *SIAM Journal on Optimization*, vol. 22, no. 4, pp. 1258–1284, 2012.

[78] R. Ghaemi, J. Sun, and I. V. Kolmanovsky, "An integrated perturbation analysis and sequential quadratic programming approach for model predictive control," *Automatica*, vol. 45, no. 10, pp. 2412–2418, 2009.

[79] J. Jäschke, X. Yang, and L. T. Biegler, "Fast economic model predictive control based on NLP-sensitivities," *Journal of Process Control*, vol. 24, no. 8, pp. 1260–1272, 2014.

[80] C. Kelley, "Iterative methods for linear and nonlinear equations," *Frontiers in applied mathematics*, vol. 16, pp. 575–601, 1995.

[81] G. Di Pillo and L. Grippo, "Exact penalty functions in constrained optimization," *SIAM Journal on control and optimization*, vol. 27, no. 6, pp. 1333–1360, 1989.

[82] A. Teel, "A nonlinear small gain theorem for the analysis of control systems with saturation," *IEEE Transactions on Automatic Control*, vol. 41, no. 9, pp. 1256–1270, 1996.

[83] A. L. Dontchev, I. Kolmanovsky, M. I. Krastanov, M. Nicotra, and V. M. Veliov, "Lipschitz stability in discretized optimal control with application to sqp," *SIAM Journal on Control and Optimization*, vol. 57, no. 1, pp. 468–489, 2019.

[84] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[85] M. Diehl, R. Findeisen, and F. Allgöwer, "A stabilizing real-time implementation of nonlinear model predictive control," in *Real-Time PDE-Constrained Optimization*, pp. 25–52, SIAM, 2007.

[86] C. T. Kelley, *Iterative methods for optimization*. SIAM, 1999.

[87] H. B. Pacejka and E. Bakker, "The magic formula tyre model," *Vehicle system dynamics*, vol. 21, no. S1, pp. 1–18, 1992.

[88] J. Wurts, J. L. Stein, and T. Ersal, "Collision imminent steering using nonlinear model predictive control," in *2018 Annual American Control Conference (ACC)*, pp. 4772–4777, IEEE, 2018.

[89] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *Control Conference (ECC), 2013 European*, pp. 502–510, IEEE, 2013.

[90] J.-S. Pang, "Error bounds in mathematical programming," *Mathematical Programming*, vol. 79, no. 1-3, pp. 299–332, 1997.

[91] J. Andersson, J. Åkesson, and M. Diehl, "Casadi: A symbolic package for automatic differentiation and optimal control," in *Recent advances in algorithmic differentiation*, pp. 297–307, Springer, 2012.

[92] H. Markowitz, "Portfolio selection," *The journal of finance*, vol. 7, no. 1, pp. 77–91, 1952.

[93] J. B. Rawlings and D. Q. Mayne, *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[94] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[95] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[96] J. Mattingley and S. Boyd, "Real-time convex optimization in signal processing," *IEEE Signal processing magazine*, vol. 27, no. 3, pp. 50–61, 2010.

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

[98] 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.

[99] 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.

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

[101] G. Cimini and A. Bemporad, "Complexity and convergence certification of a block principal pivoting method for box-constrained quadratic programs," *Automatica*, vol. 100, pp. 29–37, 2019.

[102] 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, 2010.

[103] J. Mattingley and S. Boyd, "Cvxgen: A code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[104] 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.

[105] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," in *2018 UKACC 12th International Conference on Control (CONTROL)*, pp. 339–339, IEEE, 2018.

[106] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods," *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1588–1623, 2014.

[107] P. Patrinos, P. Sopasakis, and H. Sarimveis, "A global piecewise smooth newton method for fast large-scale model predictive control," *Automatica*, vol. 47, no. 9, pp. 2016–2022, 2011.

[108] J. V. Frasch, S. Sager, and M. Diehl, "A parallel quadratic programming method for dynamic optimization problems," *Mathematical Programming Computation*, vol. 7, no. 3, pp. 289–329, 2015.

[109] A. Bemporad, "A numerically stable solver for positive semidefinite quadratic programs based on nonnegative least squares," *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 525–531, 2018.

[110] B. Hermans, A. Themelis, and P. Patrinos, "QPALM: A newton-type proximal augmented lagrangian method for quadratic programs," in *2019 Conference on Decision and Control (CDC)*, pp. 4325–4330, IEEE, 2019.

[111] R. T. Rockafellar, "Augmented lagrangians and applications of the proximal point algorithm in convex programming," *Mathematics of operations research*, vol. 1, no. 2, pp. 97–116, 1976.

[112] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[113] R. Rockafellar, "On the maximality of sums of nonlinear monotone operators," *Transactions of the American Mathematical Society*, vol. 149, no. 1, pp. 75–88, 1970.

[114] G. J. Minty *et al.*, "Monotone (nonlinear) operators in hilbert space," *Duke Mathematical Journal*, vol. 29, no. 3, pp. 341–346, 1962.

[115] D. Sun and L. Qi, "On NCP-functions," *Computational Optimization and Applications*, vol. 13, no. 1-3, pp. 201–220, 1999.

[116] A. Fischer, "A special Newton-type optimization method," *Optimization*, vol. 24, no. 3-4, pp. 269–284, 1992.

[117] R. J. Vanderbei, "Symmetric quasidefinite matrices," *SIAM Journal on Optimization*, vol. 5, no. 1, pp. 100–113, 1995.

[118] G. Banjac, P. Goulart, B. Stellato, and S. Boyd, "Infeasibility detection in the alternating direction method of multipliers for convex optimization," *Journal of Optimization Theory and Applications*, vol. 183, no. 2, pp. 490–519, 2019.

[119] F. Facchinei and J. Soares, "A new merit function for nonlinear complementarity problems and a related algorithm," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 225–247, 1997.

[120] F. J. Luque, "Asymptotic convergence analysis of the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 22, no. 2, pp. 277–293, 1984.

[121] A. Pazy, "Asymptotic behavior of contractions in hilbert space," *Israel Journal of Mathematics*, vol. 9, no. 2, pp. 235–240, 1971.

[122] H. H. Bauschke, P. L. Combettes, and D. R. Luke, "Finding best approximation pairs relative to two closed convex sets in hilbert spaces," *Journal of Approximation Theory*, vol. 127, no. 2, pp. 178–192, 2004.

[123] H. H. Bauschke, P. L. Combettes, *et al.*, *Convex analysis and monotone operator theory in Hilbert spaces*, vol. 2011. Springer, 2017.

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

[125] A. Weiss, I. Kolmanovsky, M. Baldwin, and R. S. Erwin, "Model predictive control of three dimensional spacecraft relative motion," in *American Control Conference (ACC), 2012*, pp. 173–178, IEEE, 2012.

[126] J. P. Congalidis, J. R. Richards, and W. H. Ray, "Modeling and control of a copolymerization reactor," in *American Control Conference, 1986*, pp. 1779–1793, IEEE, 1986.

[127] I. S. Duff, "Ma57—a code for the solution of sparse symmetric definite and indefinite systems," *ACM Transactions on Mathematical Software (TOMS)*, vol. 30, no. 2, pp. 118–144, 2004.

[128] L. Grippo, F. Lampariello, and S. Lucidi, "A nonmonotone line search technique for newtons method," *SIAM Journal on Numerical Analysis*, vol. 23, no. 4, pp. 707–716, 1986.

[129] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical programming*, vol. 91, no. 2, pp. 201–213, 2002.

[130] M. Huang, K. Zaseck, K. Butts, and I. Kolmanovsky, "Rate-based model predictive controller for diesel engine air path: Design and experimental evaluation," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 6, pp. 1922–1935, 2016.

[131] P. Ortner and L. del Re, "Predictive control of a diesel engine air path," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 449–456, 2007.

[132] D. Zhao, C. Liu, R. Stobart, J. Deng, E. Winward, and G. Dong, "An explicit model predictive control framework for turbocharged diesel engines," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 7, pp. 3540–3552, 2014.

[133] G. Stewart and F. Borrelli, "A model predictive control framework for industrial turbodiesel engine control," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pp. 5704–5711, IEEE, 2008.

[134] M. Huang, H. Nakada, K. Butts, and I. Kolmanovsky, "Nonlinear model predictive control of a diesel engine air path: A comparison of constraint handling and computational strategies," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 372–379, 2015.

[135] T. Maruyama, T. Shimura, A. Ejiri, Y. Ikai, and K. Shimotani, "Model predictive control applied to a diesel engine air-path system with dead time," in *SICE Annual Conference (SICE), 2011 Proceedings of*, pp. 2628–2633, IEEE, 2011.

[136] A. Murilo, M. Alamir, and D. Alberer, "A general nmpc framework for a diesel engine air path," *International Journal of Control*, vol. 87, no. 10, pp. 2194–2207, 2014.

[137] H. I. Inc., "Honeywell onramp." https://www.honeywell.com/industries/vehicles/automotive-software/onramp-powertrain-controls-and-virtual-sensing, 2017.

[138] M. Karlsson, K. Ekholm, P. Strandh, R. Johansson, and P. Tunestål, "Multiple-input multiple-output model predictive control of a diesel engine," *IFAC Proceedings Volumes*, vol. 43, no. 7, pp. 131–136, 2010.

[139] T. Broomhead, C. Manzie, P. Hield, R. Shekhar, and M. Brear, "Economic model predictive control and applications for diesel generators," *IEEE Transactions on Control Systems Technology*, 2016.

[140] K. Harder, M. Buchholz, J. Niemeyer, J. Remele, and K. Graichen, "Nonlinear mpc with emission control for a real-world off-highway diesel engine," in *Advanced Intelligent Mechatronics (AIM), 2017 IEEE International Conference on*, pp. 1768–1773, IEEE, 2017.

[141] K. Harder, M. Buchholz, J. Niemeyer, J. Remele, and K. Graichen, "A real-time nonlinear mpc scheme with emission constraints for heavy-duty diesel engines," in *American Control Conference (ACC), 2017*, pp. 240–245, IEEE, 2017.

[142] D. Liao-McPherson, S. Kim, K. Butts, and I. Kolmanovsky, "A cascaded economic model predictive control strategy for a diesel engine using a non-uniform prediction horizon discretization," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 979–986, Aug 2017.

[143] M. Huang, D. Liao-McPherson, S. Kim, K. Butts, and I. Kolmanovsky, "Toward real-time automotive model predictive control: A perspective from a diesel air path control development," in *2018 Annual American Control Conference (ACC)*, pp. 2425–2430, June 2018.

[144] P. Ortner, R. Bergmann, H. J. Ferreau, and L. Del Re, "Nonlinear model predictive control of a diesel engine airpath," *IFAC Proceedings Volumes*, vol. 42, no. 2, pp. 91–96, 2009.

[145] M. Herceg, T. Raff, R. Findeisen, and F. Allgowe, "Nonlinear model predictive control of a turbocharged diesel engine," in *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, pp. 2766–2771, IEEE, 2006.

[146] İ. A. Reşitoğlu, K. Altinişik, and A. Keskin, "The pollutant emissions from diesel-engine vehicles and exhaust aftertreatment systems," *Clean Technologies and Environmental Policy*, vol. 17, no. 1, pp. 15–27, 2015.

[147] L. Guzzella and C. Onder, *Introduction to modeling and control of internal combustion engine systems*. Springer Science & Business Media, 2009.

[148] L. Eriksson and L. Nielsen, *Modeling and control of engines and drivelines*. John Wiley & Sons, 2014.

[149] R. Tóth, "Modeling and identification of linear parameter-varying systems," 2010.

[150] I. Kolmanovsky, P. Moral, M. Van Nieuwstadt, and A. Stefanopoulou, "Issues in modelling and control of intake flow in variable geometry turbocharged engines," *Chapman and Hall CRC research notes in mathematics*, pp. 436–445, 1999.

[151] P. J. Davis and P. Rabinowitz, *Methods of numerical integration*. Courier Corporation, 2007.

[152] E. Rishavy, S. Hamilton, J. Ayers, and M. Keane, "Engine control optimization for best fuel economy with emission constraints," tech. rep., SAE Technical Paper, 1977.

[153] H. Langouët, L. Métivier, D. Sinoquet, and Q.-H. Tran, "Optimization for engine calibration," in *ENGOPT International conference on engineering optimization, Rio de Janeiro, Brazil*, pp. 1–5, Citeseer, 2008.

[154] D. Popovic, M. Jankovic, S. Magner, and A. R. Teel, "Extremum seeking methods for optimization of variable cam timing engine operation," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 3, pp. 398–407, 2006.

[155] C. Atkinson and G. Mott, "Dynamic model-based calibration optimization: An introduction and application to diesel engines," tech. rep., SAE Technical Paper, 2005.

[156] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.

[157] Z. Gao, J. Conklin, C. S. Daw, and V. K. Chakravarthy, "A proposed methodology for estimating transient engine-out temperature and emissions from steady-state maps," *International Journal of Engine Research*, vol. 11, no. 2, pp. 137–151, 2010.

[158] J. Fredriksson and B. Egardt, "Estimating exhaust manifold pressure in a turbocharged diesel engine," in *Control Applications, 2002. Proceedings of the 2002 International Conference on*, vol. 2, pp. 701–706, IEEE, 2002.

[159] P. M. Olin, "A mean-value model for estimating exhaust manifold pressure in production engine applications," tech. rep., SAE Technical Paper, 2008.

[160] R. Cagienard, P. Grieder, E. C. Kerrigan, and M. Morari, "Move blocking strategies in receding horizon control," *Journal of Process Control*, vol. 17, no. 6, pp. 563–570, 2007.

[161] M. J. Tippett, C. K. Tan, and J. Bao, "Non-constant prediction-step mpc for processes with multi-scale dynamics," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 3068–3073, 2014.

[162] D. Mozyrska, E. Pawłuszewicz, and M. Wyrwas, "Local observability and controllability of nonlinear discrete-time fractional order systems based on their linearisation," *International Journal of Systems Science*, vol. 48, no. 4, pp. 788–794, 2017.

[163] D. D. Ward, "Misra standards for automotive software," in *2006 2nd IEE Conference on Automotive Electronics*, pp. 5–18, March 2006.

[164] C. Kelley, "Iterative methods for linear and nonlinear equations, vol. 16 of frontiers in applied mathematics, society for industrial and applied mathematics (siam), philadelphia, pa, 1995," *With separately available software*.

[165] D. Liao-McPherson and I. Kolmanovsky, "The FBstab algorithm for model predictive control: An implicit condensing approach," in *2019 Conference on Decision and Control (CDC)*, pp. 3370–3376, IEEE, 2019.

[166] A. Bemporad, D. Bernardini, R. Long, and J. Verdejo, "Model predictive control of turbocharged gasoline engines for mass production," tech. rep., SAE Technical Paper, 2018.

[167] Z.-P. Jiang and Y. Wang, "A converse lyapunov theorem for discrete-time systems with disturbances," *Systems & control letters*, vol. 45, no. 1, pp. 49–58, 2002.

APPENDIX A

Proof of Theorem 5.2

Our goal is to apply [6, Theorem 1] to infer the existence of a Lyapunov function for the closed-loop system. To show this we verify the conditions of the theorem, referred to as (SA1-SA4) in [6]. Since the SMPC OCP has a penalty on Δu, and is thus nonstandard, we work with an equivalent augmented system by defining

$$y = \begin{bmatrix} x \\ u \end{bmatrix}, \text{ and } y^+ = g(y, \Delta u) = \begin{bmatrix} f(x,u) \\ u + \Delta u \end{bmatrix}, \tag{A.1}$$

which is the so-called input velocity form of the original system. We seek a Lyapunov function which proves the stability of (A.1). Note that recursive feasibility, i.e., $y_k \in \mathbb{Y}(\rho)$, is automatic thanks to Theorem 5.1. We consider each condition in turn:

(SA1) Trivially satisfied since the stage cost (5.14) is semidefinite and continuous, the terminal cost is null.

(SA2) The OCP (5.13) is guaranteed to be feasible by construction and has a strongly convex objective function, thus (5.13) has a solution and the infimum is achieved.

(SA3) Consider a candidate detection function $W(y) = V(x) + k(u)$ where $V(x)$ is a Lyapunov function which proves asymptotic stability of the inner-loop and $k(u) = l(u, 0)$. Since the inner-loop is stable by assumption, a converse Lyapunov theorem, e.g., [167], can be invoked to show the existence of V. Since V is a Lyapunov function it is bounded above and below by class $\mathcal{K}_\infty$ functions. In addition, $k(u)$ is positive definite, strictly convex with $k(\bar{u}) = 0$, and is Lipschitz continuous on the set, $\{u \mid 0 \leq \chi^{trg} \leq \max_{\mathbb{P}} \bar{\chi}_{egr}(\rho),\ 0 \leq q \leq \max_{\mathbb{P}} q^{trg}\}$, with Lipschitz constant L and thus can be bounded from above by a class $\mathcal{K}_\infty$ function. Following the notation of [6] we define our tracking measure as $\sigma(y) = V(x) + k(u)$. Let $\beta_1(\sigma(y)) = \eta_1(||x - \bar{x}||) + \eta_2(||u - \bar{u}||)$, where $\eta_1 \in \mathcal{K}_\infty$ upperbounds $V(x)$ and $\eta_2 \in \mathcal{K}_\infty$ upperbounds $k(u)$. Then $W(y) \leq \beta_1(\sigma(y))$ and $\beta_1 \in \mathcal{K}_\infty$. Now consider

$$\begin{aligned} &W(g(y, \Delta u)) - W(y) \\ &= V(f(x,u)) - V(x) + k(u + \Delta u) - k(u) \\ &\leq V(x^+) - V(x) + L||\Delta u||_R, \end{aligned}$$

and define

$$\alpha_1(x) = \begin{cases} (V(x) - V(x^+))/V(x), & x \neq \bar{x}, \\ 0 & x = \bar{x}, \end{cases} \tag{A.2}$$

Since V is a Lyapunov function $\forall x \in \mathbb{X} \setminus \{\bar{x}\}$, $V(x^+) - V(x) < 0,\ V(x) > 0$ and thus $\alpha_1(x) \in [0, 1)$ and $V(x^+) - V(x) + \alpha_1(x)V(x) = 0$. Continuing, using that $V(x^+) - V(x) = -\alpha_1(x)V(x) \leq 0$, we have that

$$\begin{aligned} W(g(y, \Delta u)) - W(y) &\leq V(x^+) - V(x) + L||\Delta u||_R, \\ &\leq -\alpha_1(x)V(x) + (\gamma_1 - \alpha_1(x))k(u) + L||\Delta u||_R, \\ &\leq -\alpha_1(x)(V(x) + k(u)) + \gamma_1 k(u) + L||\Delta u||_R, \\ &\leq -\alpha(x)\beta_1(\sigma(y)) + \gamma_1 k(u) + L||\Delta u||_R, \end{aligned}$$

for any $\gamma_1 > 1 > \alpha_1(x)$. Since $\alpha_1(x) < 1$, for any $\gamma_2 > 1$, $\alpha_1(x)(V(x) + k(u)) \leq \gamma_2\beta_1(\sigma(y))$. Thus $\beta_2(\cdot) = \gamma_2\beta_1(\cdot) \in \mathcal{K}_\infty$ is an appropriate comparison function. The remaining terms can be upper bounded by the function $\gamma_1 l(u, \Delta u) + \gamma_3\sqrt{l(u, \Delta u)}$ for sufficiently large γ_3. Thus we can take $\beta_3(\cdot) = \gamma_1(\cdot) + \gamma_3\sqrt{(\cdot)} \in \mathcal{K}_\infty$ as the final required comparison function. The existence of β_1, β_2, and β_3 verifies the detectability condition [6, Definition 1].

(SA4) Boundedness of the value function can be established using Assumption 2 (Asymptotic controllability) combined with [1, Lemma 6.6]. Verifying (SA1-SA4) completes the proof.