Distributed model predictive control of compressor systems

Kathleen Jones

Master's Thesis
Ecole polytechnique fédérale de Lausanne
Department of Mechanical Engineering

Supervisors:	Alireza Karimi (EPFL)
	Andrea Cortinovis (ABB)
	Mehmet Mercangöz (ABB)

Lausanne, 29 July 2016

Abstract

The performance and computational cost of distributed MPC for the control of compressor networks is investigated in simulation. Both cooperative and non-cooperative approaches are considered and compared to the performance achieved with centralized control in the presence of a discharge-side disturbance. Two systems, each with two compressors, are studied: one is arranged in parallel configuration and the other in series. Due to the high degree of non-linearity of both systems, the models are re-linearized at each time step and a linear, delta MPC formulation is used. Hard input constraints are imposed, however state constraints are not considered and the cost functions used are dependent only on the inputs and outputs, not on the states themselves.

The controller is implemented in `Simulink` and in C++ using the quadratic program solver `qpOASES`, and the C++ implementation is used to evaluate the computational cost of each control approach. For the parallel case, the cooperative and non-cooperative controllers achieved near-identical performance to the centralized controller. The cooperative approach had a computation time 2% higher than that of the centralized approach, while the non-cooperative controller reduced the required computation time by 25%. For the serial case, the cooperative controller achieves performance very close to that of the centralized approach, however with only a 10% reduction in computation time. Again, the non-cooperative controller has a computation time 40% lower than the centralized controller, but in this case sacrificed performance: it showed a 9% decrease in the minimum surge control distance reached in the downstream compressor.

Acknowledgements

I would like to thank my supervisors, Andi, Mehmet and Mr. Karimi for their advice and support in writing this thesis. I'm also grateful to my colleagues and fellow students at ABB who made these past several months so enjoyable. Thank you as well to the friends whose help with derivations, writing and proofreading was invaluable.

Contents

List of Tables

List of Figures

List of Symbols

Latin Symbols

A	Cross-sectional piping area after compressor.	14
a	Speed of sound (340 m/s).	14
A^{delay}	Component of the augmented A matrix multiplied by the delayed states.	24
$A_k^a,\ B_k^a,\ C_k^a$	Linearized, discretized, augmented model matrices.	24
$A_k,\ B_k,\ C_k$	Linearized, discretized model matrices.	23
B^{aug}	Component of the augmented B matrix that outputs the delayed states.	24
B_k^{delay}	Columns of B_k corresponding to delayed inputs.	24
B_k^{nodelay}	Columns of B_k corresponding to non-delayed inputs.	24
$\Delta \boldsymbol{u}_k^{\text{a}}$	Change in input applied to the system at time step k.	24
$\boldsymbol{e}_k$	Integrator states of MPC controller.	23
$\boldsymbol{f}_{\text{d},k}$	Discretized system derivative at time step k.	23
$\boldsymbol{f}_{\text{n}}$	Nominal state derivative vector for a single compressor.	14
$\boldsymbol{f}_{\text{p}}$	Parallel system state derivative vector.	17
$\boldsymbol{f}_{\text{s}}$	Serial system state derivative vector.	19
J_{comp}	Compressor inertia.	14
$J_{\text{p,k}}^{\text{cen}}$	Centralized cost function of parallel system.	31
l_{c}	Duct length after compressor.	14
m	Move horizon of controller.	26
p	Prediction horizon.	47
p_{a}	Atmospheric pressure.	21
p_{d}	Compressor discharge pressure.	14
p_{in}	Pressure upstream of compressor.	15
p_{out}	Pressure downstream of compressor.	15
p_{s}	Compressor suction pressure.	14
p_{t}	Common tank pressure (parallel system).	17

$Q,\ R$	Weighting terms applied to inputs and outputs, respectively.	25
q_c	Mass flow rate through compressor.	14
q_d	Mass flow rate through discharge valve.	14
$q_\mathrm{d,t}$	Mass flow rate through common tank discharge valve (parallel system).	17
q_r	Mass flow rate through recycle valve.	14
q_s	Mass flow rate through suction valve.	14
SCD	Surge control distance.	4
SD	Surge distance.	4
S_{U_k}, S_{x_k}, S_{f_k}	Prediction matrices for ΔY_k.	25
$S_{U_k^\mathrm{ot}}$	Prediction matrix giving effect of the other sub-controller's inputs on ΔY_k.	28
T_c	Torque on compressor resulting from air compression.	14
T_d	Torque input to compressor.	14
t_s	Sample rate.	22
u_d	Position of discharge valve.	15
$\boldsymbol{u}_k^\mathrm{delay}$	Delayed recycle input states at time step k.	23
U_k	Optimal input sequence, stacked over the move horizon, at time step k.	25
$\boldsymbol{u}_k$	Input applied at time step k.	23
$\boldsymbol{u}_\mathrm{n}$	Nominal input vector for a single compressor.	16
U_k^ot	Input sequence from other distributed controller, stacked over the move horizon, at time step k.	28
$\boldsymbol{u}_\mathrm{p}$	Parallel system input vector.	19
u_r	Recycle valve opening.	14
$u_\mathrm{r,SP}$	Set point of recycle valve.	14
u_s	Position of suction valve.	15
$\boldsymbol{u}_\mathrm{s}$	Serial system input vector.	20
u_t	Position of common tank discharge valve (parallel system).	17
V_d	Compressor discharge tank volume.	14
V_s	Compressor suction tank volume.	14
V_t	Common tank volume (parallel system.	17
$\Delta\hat{\boldsymbol{x}}_k^a$	Augmented state estimate at time step k.	23
$\hat{\boldsymbol{x}}_k$	State estimate at time step k.	23
$\boldsymbol{x}_\mathrm{n}$	Nominal state vector for a single compressor.	14
$\boldsymbol{x}_\mathrm{p}$	Parallel system state vector.	17
$\boldsymbol{x}_\mathrm{s}$	Serial system state vector.	19

Y_k	Predicted output sequence, stacked over the prediction horizon, at time step k.	25
$\boldsymbol{y}_k$	Output at time step k.	23
$\boldsymbol{y}_\mathrm{n}$	Nominal output vector for a single compressor.	17
$\boldsymbol{y}_\mathrm{p}$	Parallel system output vector.	19
Y_k^ref	Reference output sequence, stacked over the prediction horizon, at time step k.	25
$\boldsymbol{y}_\mathrm{s}$	Serial system output vector.	20

Greek Symbols

$\boldsymbol{\alpha}$	Coefficients mapping compressor pressure ratio to compressor speed and mass flow.	14
$\boldsymbol{\beta}$	Coefficients mapping steady-state driver torque to rotational speed and mass flow of compressor.	14
$\Delta\left(\cdot\right)$	Difference in $(\cdot)$ relative to its value at the linearization point.	23
$\boldsymbol{\delta}$	Coefficients mapping mass flow through recycle valve to its pressure differential.	15
$\boldsymbol{\varepsilon}$	Coefficients mapping mass flow through discharge valve to its pressure differential.	15
$\boldsymbol{\gamma}$	Coefficients mapping mass flow through suction valve to its pressure differential.	15
ω_c	Rotational speed of the compressor.	14
Π	Compressor pressure ratio $(\frac{p_\mathrm{d}}{p_\mathrm{s}})$.	14
$\boldsymbol{\rho}$	Coefficients mapping mass flow across common tank discharge valve to its pressure differential (parallel system).	17
τ_r	Time constant of recycle valve.	14

1

Introduction

1.1 Motivation

Centrifugal gas compressors are employed in a wide range of industrial applications, particularly for gas transportation, extraction and processing. Compression is an inherently energy-intensive process, with well over 90% of operating costs spent on energy; small improvements in efficiency can therefore have a significant impact on the operating costs. At the same time, compressors are critical components in natural gas installations, meaning small downtimes can also have a large economic impact. It is therefore important that any gains in efficiency are not achieved at the expense of robustness to disturbances, since the latter could push the compressor out of its safe operating limits and cause significant downtimes. These factors make compressor controller development both challenging and crucial for improving their overall efficiency.

Compressors and compressor systems have long been an active area of research in the control community. The field has transitioned from fixed-speed drivers powering the compressor, to variable-speed gas turbines, to the variable-speed electric drivers used today. With this transition comes more possibilities in controller development. Fixed-speed drivers required throttling or adjustable guide vanes for process control, in order to achieve a discharge pressure setpoint. The advent of variable-speed drivers enabled process control to be achieved by manipulating the driver speed, however the slow dynamics of gas turbines relative to those of the compressor meant that process control and the control loops responsible for safe operation of the system had to be distinct. The variable-speed electric drivers that have appeared more recently have much faster dynamics, opening the door for new, multivariable control algorithms that combine process control and limitations on the operating regime into a single controller.

Compressor control presents unique challenges: compressors are inherently highly non-linear, multivariable systems with a high degree of coupling, and their operating regime is limited both by the presence of unstable regimes and by physical constraints on the temperature and speeds that can be supported for safe operation. The coupling between control loops and constraints, in particular, are difficult to treat using traditional frequency-domain control approaches. Current industrial practice is to implement the controllers using simple PID controllers, with added loop decoupling and hand-tuned open-loop control responses near boundaries to address these issues. Constraints are generally treated using clipping and anti-windup logic, which require further tuning. In recent years, with the adoption of electric drivers, model predictive control (MPC) has been proposed as an alternative to frequency-domain approaches as it can explicitly consider both the coupling and physical constraints that make compressor control so challenging.

The major disadvantage of MPC compared to conventional control approaches is the computational complexity inherent in the approach, and the resulting difficulty of executing the controllers at a sampling rate fast enough to handle the relatively fast dynamics observed in compressors. In particular, as the number of states and inputs increase – as for systems of multiple, coupled compressors – so does the required computation time of an MPC controller. This limitation can be overcome using a distributed MPC approach, whereby the optimization problem posed by a multi-compressor system can be divided into sub-problems to be solved at the individual compressor level, with some information exchange to converge to a globally optimal solution.

The scope of this thesis is to evaluate the controller performance and computational cost of a distributed MPC control approach compared to centralized MPC when applied to a parallel and to a serial compressor network, each containing two compressors.

1.2 Compressor Control

Compressor control generally consists of two separate, sometimes competing tasks: process control and anti-surge control. Process control seeks primarily to regulate an output variable of the compressor – in this report, the output pressure is used, but mass flow or other variables could equally be chosen. The manipulated variable used in process control varies depending on the type of compressor studied. Gas turbine-driven compressors, for example, may have a valve regulating the fuel flow to the turbine, which can be adjusted by the control system to increase or decrease the compressor speed. For the electric, variable-speed drivers considered in this thesis, the manipulated variable is the torque applied by the driver to the compressor, which is adjusted to maintain the output pressure at the desired setpoint. In conventional control systems, process control is implemented using cascaded PID controllers for the discharge pressure, driver

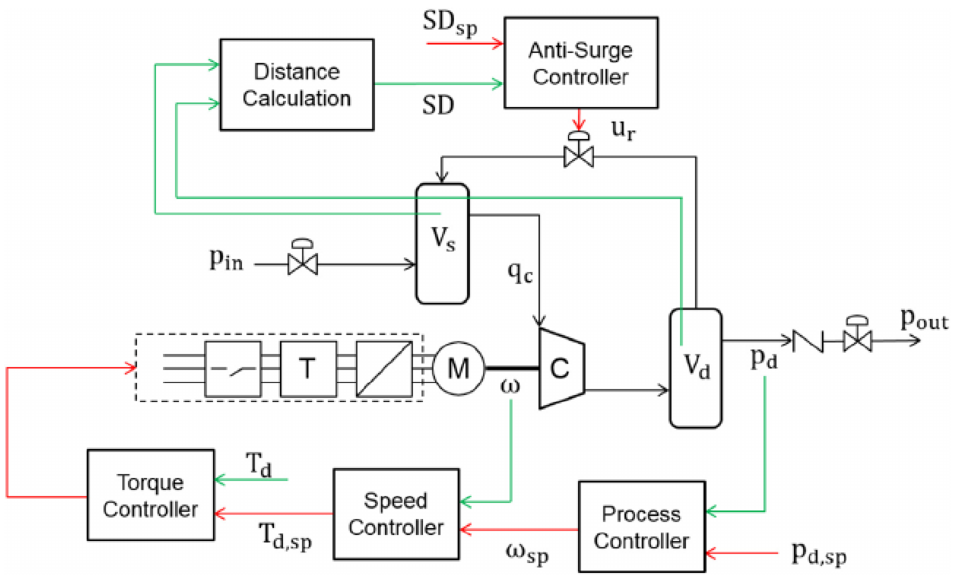


Figure 1.1: Diagram of compressor with conventional control system [1].

speed and driver torque, which operate at different sampling rates to ensure time-scale separation.

A diagram of a compressor with a typical, convention control system is shown in Figure 1.1. The cascaded process, speed and torque PID controllers and the independent anti-surge controller are depicted.

Anti-surge control keeps the compressor away from an unstable regime known as surge, which is characterized by oscillations in the mass flow rate and pressures, as well as increased temperatures and vibrations. Surge occurs when the system resistance on the compressor is too high, leading to backflow until the resistance is reduced. At this point forward flow is restored, and the compressor enters a limit cycle. Surge typically occurs at low mass flow conditions. Operating in the surge regime can cause serious damage to the compressor and the surrounding piping system.

Surge is avoided primarily through the use of a recycle valve, which can be opened to allow flow from the discharge to the suction tank of the compressor. This simultaneously reduces the discharge pressure and increases the mass flow through the compressor, moving the operating point down and to the right on the compressor map – away from surge. For compressors with electric drivers, the torque input to the driver can also be used to rapidly increase the mass flow through the compressor, thereby temporarily increasing the surge distance.

The conditions leading to surge are determined experimentally and plotted as a line on the compressor map, known as the surge line (SL). [1] This plot is then used to define a surge distance (SD) for the compressor, which is defined as the horizontal distance between the current operating point and the surge line. [2] To avoid entering the surge regime, a surge control line (SCL) that is offset from the surge line, and a corresponding surge control distance (SCD) are defined to add a safety margin for the controller. The controller should maintain the operating point to the right of the surge control line. A compressor map with the surge line and surge distance labeled is shown in Figure 1.2. From this map, it can be seen that the effect of the recycle valve, which is to increase the mass flow and decrease the pressure ratio, moves the compressor away from surge.

Although the surge regime should be avoided during compressor operation, often the most efficient operating points (and thus the operating points used in industrial applications) are on or near the surge control line. If the compressor enters surge, however, it often triggers a costly shutdown of the system to prevent damaging the machinery. The anti-surge controller is a crucial safety feature that permits the safe operation of the plant and protects the equipment from damage. Furthermore, it is crucial in allowing the compressor to maximize efficiency by operating near the surge line, while still rejecting disturbances quickly enough to avoid entering the surge regime.

1.3 Advanced Compressor Control

The biggest challenges in compressor control systems are handling the interactions between the process variable and surge distance, as well as the input constraints. Conventional control systems based on PID controllers use complex loop decoupling and integrator anti-windup terms to correct for these effects, since they cannot be directly considered by a frequency-domain control approach.

MPC is thus a promising approach for compressor control, as it permits both the coupling and the input constraints to be treated explicitly in an optimization problem. One major obstacle that prevents MPC from being implemented, particularly on systems with multiple compressors, is the computation time required to arrive at an optimal solution. Compressor controllers have to run at a sampling rate on the order of tens of milliseconds as the dynamics leading to surge are at this time scale. As the number of compressors is increased, the size of the optimization problem solved by the MPC controller increases as well, putting a practical limit on the size of the system that can

[1]The transition to surge only collapses onto a single line in the compressor map if (quasi-)invariant coordinates are used; the transition to surge is then invariant to the inlet conditions. For a detailed discussion of compressor maps and invariant coordinate systems, see [2]. For the purposes of this work, the pressure ratio and mass flow rate are used and are assumed to be quasi-invariant for the cases studied.

[2]The surge distance is sometimes also defined as the angular distance between the operating point and the surge line.

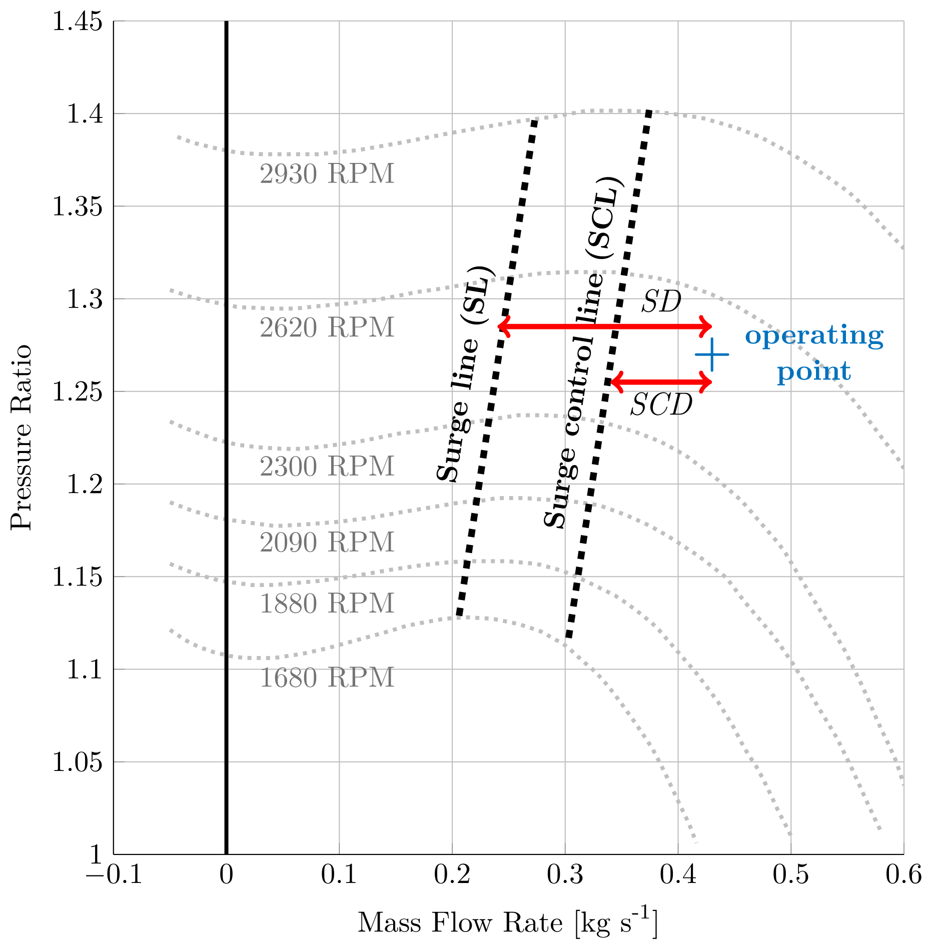


Figure 1.2: Generic compressor map with labeled surge distance (*SD*) and surge control distance (*SCD*). Curved, dotted lines are lines of constant compressor speed.

be controlled with the required sampling rate using MPC.

Distributed MPC differs from standard (centralized) MPC in that the optimization problem is split into smaller sub-problems that can be solved in parallel by several distributed controllers. It is particularly intuitive when dealing with large systems that combine several distinct but coupled subsystems, as is the case for a compressor network. The sub-problems are then often associated to a particular subsystem. Distributed MPC, unlike the centralized version, has the potential to be scaled up to arbitrarily large systems without a significant increase in computation time, making it the natural approach for compressor networks.

Because all of the sub-problems are coupled, and the solutions to each affect the other sub-problems, distributed MPC generally takes an iterative approach whereby each sub-controller assumes a value for the other solutions, solves its optimization problem, and communicates its updated solution to the other sub-controllers. The process can continue for a fixed number of steps, or until a convergence criterion is satisfied. Depending on the degree of coupling, the iterations may or may not converge. For a discussion of convergence in distributed MPC controllers, see [3].

The primary advantage of distributed MPC over centralized is the potential for reducing the computation time. Although each of the distributed sub-controllers must solve multiple optimizations at each time step, their reduced size means that in many case they still execute faster than the centralized controller. Additionally, its implementation would be closer to what is currently done, and to what is feasible in natural gas installations. A single, centralized controller acting on multiple compressors would be difficult to achieve in a large plant. With distributed MPC, however, each compressor would have its own hardware, with communication between each of these sub-controllers. This is similar to the current practice where process and anti-surge control for each compressor are implemented on separate hardware, and communication between them is used for loop decoupling terms.

1.4 Thesis Overview

This Master's thesis is organized as follows:

- An overview of modelling approaches for centrifugal compressors, current state-of-the-art control schemes and recent research in the field is given in Section 2.

- The dynamic compressor model used both for simulation and by the MPC controller is detailed in Section 3.

- Section 4 gives a brief development of the underlying optimization performed by the MPC controller, and the steps that are taken at each iteration to obtain the next input to apply to the system.

- The results of the simulation tests as well as a comparison between the distributed and centralized model predictive controllers are shown and discussed in Section 5.

2

Literature Review

Compressor control using advanced techniques such as MPC is an active field of research. A non-exhaustive summary of the work that has been done is presented here. It is divided into five areas: the development of compressor models used for control and simulation, MPC theory and stability analysis, MPC controller development for individual compressors, anti-surge and performance control approaches for networks of compressors and implementation of MPC controllers on embedded systems.

2.1 Compressor Models

Two different modelling approaches are used in industry: black box and gray box. Black box models fit experimental date to a pre-defined model type (e.g. ARX, state space, neural network), whose order is then chosen to give a good fit. Grey-box modelling uses a first principles approach, assuming a model structure based on the underlying physics of the system. The parameters of the model are then determined for the specific structure and order given. Here, grey-box models are considered, however black-box models are also commonly used in industry.

The underlying grey-box model used as the basis for most compressor control applications was developed by Greitzer [4] and expanded by Moore and Greitzer [5]. They model a system consisting of a compressor, plenum and throttle for an inviscid, incompressible flow with constant time-averaged flow around the annulus. The equations are then developed to predict both surge and rotating stall. The models are based on the use of compressor maps to characterize the behavior of each individual compressor, as well as parameters describing the up- and downstream processes. The Greitzer model was developed for axial compressors, however it has been shown to be also applicable to centrifugal compressors by Hansen et al. [6]

The Moore-Greitzer model was expanded by Gravdahl and Egeland to include both a close-coupled valve and a non-constant compressor speed for axial and centrifugal compressors [7]. This updated model was used to study surge during compressor acceleration for a variable-speed compressor.

Aaslid developed the Gravdahl and Greitzer model into a Simulink model, which was then validated against test data from a gas processing plant for use in anti-surge control (ASC) [8].

Galindo et al. developed a compressor map extending into the surge region and experimentally validated it. The model obtained was found to be accurate even during operation in the surge regime [9].

The invariant and "nearly invariant" coordinates often used to describe the surge line of a compressor with little or no dependence on inlet conditions were developed by Batson et al. [2] Using these coordinate systems allows the surge conditions to be represented as a single line (for fixed-geometry compressors) irrespective of input conditions, greatly simplifying the calculation of distance to surge – an important quantity for anti-surge control.

2.2 MPC for compressors

MPC has been used in several cases to combine process and anti-surge control (ASC) – traditionally performed by two independent controllers – resulting in increased performance and better surge avoidance during disturbances compared to traditional PID controllers [1, 10].

Cortinovis et al. showed that combined process and ASC using MPC led to a reduced distance to surge and a reduced settling time for process control in a centrifugal compressor in experiments compared to the current state-of-the-art. Both a discharge valve disturbance and a combined disturbance on both the discharge and the suction valve were considered [1]. The controller was implemented using linear MPC by linearizing the system at each time step.

Similarly, Budinis and Thornhill found that a combined process/anti-surge MPC controller better handled disturbances on the inlet and outlet valves as well as changes in load pattern compared to a PID controller in simulation [10].

Bentaleb et al. also demonstrated that an MPC controller manipulating both driver speed and the inlet guide vanes could achieve better process performance than a PID controller manipulating only the driver speed [11]. This result was extended by Bentaleb to include the use of a recycle valve to combine process and anti-surge control [12].

2.3 Control of Compressor Networks

An MPC controller for two compressors arranged in parallel was developed by Smeulers and Bouman who considered a single controller combining load-sharing for the two compressors as well as performance and anti-surge criteria for each compressor individually [13]. The controller was implemented using linear MPC, where the non-linear model was linearized at each time step to characterize the compressors' non-linear behavior more accurately. Two objectives for load-balancing were considered: equal distance of each compressor to the surge line and minimum power consumption of the system. The constraints used included surge constraints for each compressor, a maximum temperature constraint on the output plenum, and actuator constraints. The controller was implemented in simulation using a relatively high sampling time on the order of one second due to computational limitations. For this reason, dedicated anti-surge controllers with a smaller sampling time was necessary to open the recycle valves much quicker in the case where the compressor moves toward surge during the sampling period of the MPC controller. In simulation, however, these backup controllers were unused as the MPC controller successfully kept both compressors away from the surge line.

Øvervåg also considered load sharing for two parallel compressors using an MPC controller [14]. He approximated the efficiency of each compressor using continuous functions and used the minimization of the fuel consumption for the turbines driving the compressors as the optimization criterion for MPC. As in [13], linear MPC is implemented by re-linearizing the compressor models at each time step. The focus of this work was on the load-sharing algorithm and generating set-points for each compressor; it did not consider the process and surge control of the individual compressors, though constraints were used in the load-sharing formulation to keep both compressors away from the surge line.

2.4 Distributed MPC

An overview of the types of decentralized, distributed and hierarchical control schemes that have been presented in the literature, with a particular focus on MPC approaches, is given by Scattolini [15]. Venkat et al. proposed one such scheme: an iterative, cooperative distributed MPC (DMPC) approach for linear systems that achieves optimality at convergence, and for which the intermediate iterates are all feasible and result in closed-loop stability of the overall system [16].

Richards and How present similar results using DMPC for a linear system, with the addition of bounded disturbances to establish a robustness guarantee [17]. They found significant improvements in computation time compared with the centralized MPC approach, with minimal decrease in performance.

Stewart et al. studied cooperative DMPC for large systems, and established that the cooperative scheme – unlike a non-cooperative or decentralized approach – can converge to an optimal solution for arbitrary coupling strengths between the subsystems, and provides a suboptimal but stable and feasible solution at each iteration [3]. These results were extended to nonlinear systems by Stewart, Wright and Rawling, who proposed a distributed, nonconvex optimizer for solving the resulting optimization problem [18]. Their implementation did not require the use of centralized coordination between the sub-controllers, an important criteria for many industrial applications.

With the exception of the last work, these results, like much of the literature on distributed MPC, focus only on linear systems. They are therefore not directly applicable to the work done here, as the current model is not linear but rather re-linearized at each time step, however they provide valuable insight into the potential of distributed MPC controllers.

2.5 Solvers & Implementation

Ferreau et al. present a quadratic programming solver using an active-set algorithm that can be used to efficiently solve QP problems, for example of the type solved at each timestep of an MPC controller [19]. This solver is used for example by Cortinovis et al. to implement an MPC compressor controller on an embedded system [1].

Peyrl et al. also found that QPs generated by load-sharing MPC controllers have certain separability properties that can be exploited to more efficiently solve them using the alternating method of multipliers (ADMM) rather than a standard QP solver [20]. For the compression station studied, ADMM was 10 times faster than a state-of-the-art QP solver, despite being implemented using code generation by Matlab Coder.

2.6 Current State of the Art

A comprehensive overview of the current state of the art for combined compressor process and anti-surge control is given by Fausel et al. in [21] where they describe the control approach taken by the Compressor Controls Corporation. Control is performed in the frequency domain using two dedicated process and anti-surge control loops, whose interactions are offset by loop decoupling elements. An overview of the approach is shown in Figure 1.1.

The main characteristics of the approach are as follows:

1. Dedicated process and anti-surge controllers, acting on the driver torque (or gas turbine speed, etc.) and on the recycle valve opening, respectively.
2. Loop-decoupling components to offset interactions between the process and anti-surge controllers, and between coupled compressors (e.g. connected in series or in parallel).
3. An anti-surge controller with a PI response to maintain operation on the surge control line.
4. An additional, open-loop term in the anti-surge controller activated when the surge distance drops below a certain threshold. This term is modulated by the rate of change in the surge distance – a larger disturbance causing a faster move towards surge triggers a larger open-loop response. The open-loop response then decays exponentially until full control is returned to the PI controller. [1]

The disadvantage with respect to the multi-variable MPC approach considered here is that the two control loops are independent, so in compressors with an electric driver, for example, the driver torque cannot be used to regulate the surge distance and pull the compressor away from surge on a much faster time scale than that achieved using the recycle valve alone.

[1]This open-loop response can also be triggered stepwise. In this case, the open-loop term is applied for a pre-defined length of time after which the controller checks if the surge distance has increased above the safety threshold. If it has, the term decays exponentially as usual. If the surge distance is still below the threshold, the magnitude of the open-loop term is increased and the process is repeated until the compressor is pulled away from surge.

3

Modelling & Simulation

3.1 Introduction

The controllers presented in this work were evaluated in simulation, using a model based on a two-compressor system at an ABB Research Facility. The physical setup is described in [1] for a single compressor. The simulations were performed using the Dormand-Prince adaptive step-size algorithm for numerical integration of the state equations and implemented in both `MATLAB/Simulink` and C++ (see Section 4.8). In order to isolate the effects of changing controller types and parameters, the model was kept as simple as possible and the two compressors were assumed to be identical. Two different configurations were tested: a parallel configuration and a serial configuration, described in further detail below.

3.2 Compressor Model

The single compressor unit used in this work, depicted in Figure 3.1, refers to a centrifugal compressor driven by a variable-speed electric driver, with a tank at both the inlet (suction) and outlet (discharge). Both the suction and discharge tanks are connected to atmospheric pressure by a valve. These suction and discharge valves represent the flow conditions upstream and downstream of the compressor, respectively, such that various flow conditions and disturbances can be simulated by simply changing the position of the valves. The artificial disturbances applied using these valves are designed to mimic situations frequently encountered in industrial applications (e.g. shutdown or startup of a downstream compressor). A third, recycle (or anti-surge) valve connects the outlet tank to the inlet tank. Its primary purpose is to open when the compressor approaches surge in order to decrease the compressor load and increase the mass flow through the

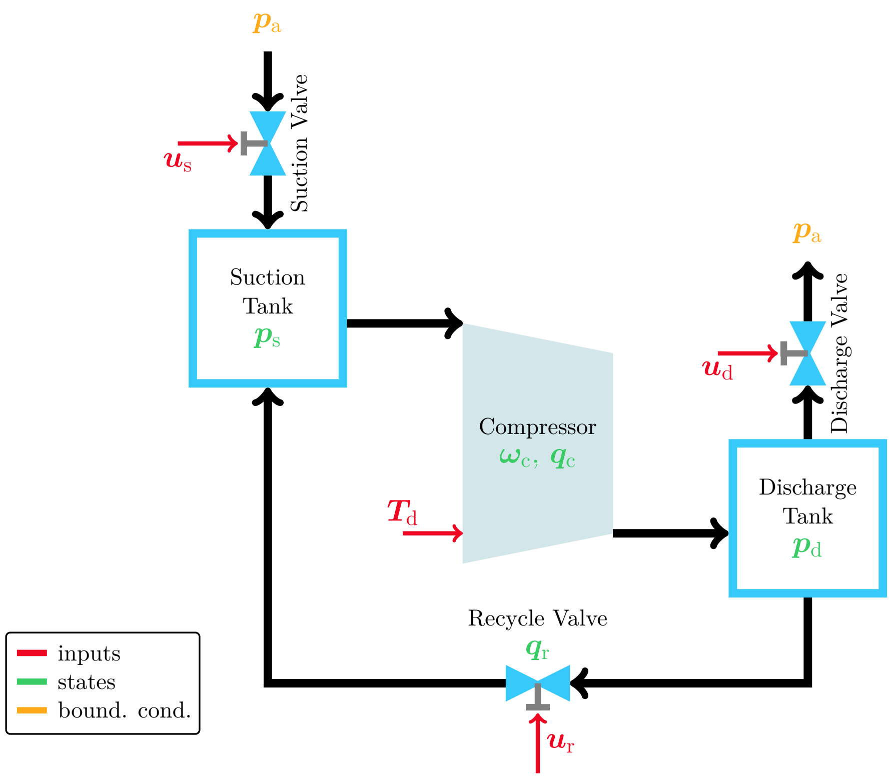


Figure 3.1: Compressor model used in simulation.

compressor, thus pulling it away from the surge line.

3.2.1 States

The compressor model used is based on the Gravdahl-Greitzer model described in [7]. It is defined by five states: the suction (p_s) and discharge (p_d) pressures, the mass flow through the compressor (q_c), the rotational speed of the compressor (ω_c), and the mass flow rate through the recycle valve (q_r). The differential equations governing these states are given by:

$$\frac{d}{dt}\underbrace{\begin{bmatrix} p_s \\ p_d \\ q_c \\ \omega_c \\ q_r \end{bmatrix}}_{\boldsymbol{x}_n} = \underbrace{\begin{bmatrix} \frac{a^2}{V_s}\left(q_s + q_r - q_c\right) \\ \frac{a^2}{V_d}\left(q_c - q_r - q_d\right) \\ \frac{A}{l_c}\left(\Pi_{ss}\left(\boldsymbol{\alpha},\ \omega_c,\ q_c\right)p_s - p_d\right) \\ \frac{1}{J_{comp}}\left(T_d - T_c\left(\boldsymbol{\beta},\ \omega_c,\ q_c\right)\right) \\ \frac{1}{\tau_r}\left(u_r - u_{r,SP}\right) \end{bmatrix}}_{\boldsymbol{f}_n} \tag{3.1}$$

$$\begin{aligned} q_{\mathrm{s}} &= q_{\mathrm{s}}(\boldsymbol{\gamma},\ u_{\mathrm{s}},\ p_{\mathrm{s}},\ p_{\mathrm{in}}) \\ q_{\mathrm{r}} &= q_{\mathrm{r}}(\boldsymbol{\delta},\ u_{\mathrm{r}},\ p_{\mathrm{s}},\ p_{\mathrm{d}}) \\ q_{\mathrm{d}} &= q_{\mathrm{d}}(\boldsymbol{\varepsilon},\ u_{\mathrm{d}},\ p_{\mathrm{out}},\ p_{\mathrm{d}}), \end{aligned} \tag{3.2}$$

where:

- a is the speed of sound;
- V_{s} and V_{d} are the suction and discharge tank volumes, respectively;
- q_{s}, q_{r}, and q_{d} are the mass flows across the suction valve, recycle valve, and discharge valve, respectively;
- A is the cross-sectional area of the piping after the compressor and l_{c} the duct length;
- $\Pi_{\mathrm{ss}}(\boldsymbol{\alpha}, \omega_{\mathrm{c}}, q_{\mathrm{c}})$ is the steady-state pressure ratio across the compressor, and $\boldsymbol{\alpha}$ contains the coefficients mapping this ratio to the compressor speed and mass flow;
- J_{comp} is the inertia of the system;
- $\boldsymbol{\beta}$ contains the parameters mapping the steady-state torque results from air compression to the mass flow and rotational speed of the compressor;
- τ_{r} and $u_{\mathrm{r,SP}}$ are the time constant of the recycle valve and its setpoint, respectively;
- $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$ and $\boldsymbol{\varepsilon}$ are the coefficients mapping mass flow through the suction, recycle and discharge valves, respectively, to the pressure differential across the valve;
- u_{s} and u_{d} are the positions of the suction and discharge valves, respectively;
- p_{in} and p_{out} are the pressures upstream and downstream of the compressor, respectively; and
- $\boldsymbol{x}_{\mathrm{n}}$ and $\boldsymbol{f}_{\mathrm{n}}$ are vectors representing the nominal state and state derivative of a single compressor, respectively.

The interested reader may refer to [1] for further information on the identification and validation of this model.

3.2.2 Inputs

The inputs to the compressor system are the positions of the suction, discharge and recycle valves, as well as the torque setpoint of the electric driver powering the compressor. It is assumed that a cascaded regulator maintains the torque at the desired setpoint with a dynamic response fast enough to be neglected at the time scales considered (on the order of tens of ms). Torque is chosen as the input variable as it allows a faster response time compared to driver speed, which is generally slower as it requires the use of cascaded torque and speed controllers. It should be noted that the torque can be adjusted in time scales on the order of tens of milliseconds while the valves move on the order of hundreds of milliseconds.

The specific recycle valve being modelled has a dead time between when the input signal is applied and when the valve moves. This characteristic is also typical of valves used in industrial applications. In general, this dead time varies as a function of the valve position, with the worst case occurring when the valve is opened from a fully-closed position. For simplification, however, it was assumed constant and equal to the worst-case dead-time, identified in [1] to be 2 s. This dead time was modelled using additional, delayed states as described in Section 4.2.

For control purposes, the torque input to the driver and the recycle valve position were used as manipulated variables, while the suction and discharge valves were used to apply disturbances to the system. The system input vector is therefore given by:

$$\boldsymbol{u}_{\mathrm{n}} = \begin{bmatrix} T_{\mathrm{d}} \\ u_{\mathrm{r}} \end{bmatrix} \tag{3.3}$$

3.2.3 Outputs

Industrial compressor installations, as well as the system being modelled, typically have sensors measuring the compressor states described above, as well as several other temperatures, pressures and mass flow rates. The choice of outputs used in the model for control purposes is thus not unique, and can be adjusted depending on the requirements of each installation. In this case, the variable used for process control is the discharge pressure of the compressor.

For anti-surge control, however, the standard controlled output used is the surge distance or surge control distance (equivalent up to a constant offset), defined in Section 1.2. It cannot be measured but is instead estimated based on the current operating point and an experimentally-determined estimate of the surge line position on the compressor map.

The surge line is estimated to be a straight line in the compressor map using the pressure ratio (Π) and mass flow rate (q_{c}) as coordinates, defined by the equation:

$$\Pi_{\mathrm{SL}} = a_1 q_{\mathrm{c,SL}} + a_2, \tag{3.4}$$

where Π_{SL} and $q_{\mathrm{c,SL}}$ are the pressure ratio and mass flow rate on the surge line, respectively, and a_1 and a_2 are experimentally-determined coefficients. The surge distance is given by:

$$SD = \left(\underbrace{\frac{p_\mathrm{d}}{p_\mathrm{s}} \frac{1}{a_1} - \frac{a_2}{a_1}}_{q_\mathrm{c,SL}} \right) - q_\mathrm{c}. \tag{3.5}$$

The output vector used for control purposes is thus:

$$\boldsymbol{y}_\mathrm{n} = \begin{bmatrix} p_\mathrm{d} \\ SD. \end{bmatrix} \tag{3.6}$$

3.3 Parallel System

The parallel compressor system, shown in Figure 3.2 consists of two compressors whose discharge tanks both feed into a common tank, which has a separate discharge valve and a downstream atmospheric pressure boundary condition. For simplicity, the two compressors are assumed to have identical parameters and behavior, and the discharge valve of the common tank is assumed to have the same characteristic as the discharge valves of the compressors.

The model used has 11 states in total: 5 states for each compressor, as described above, and another state for the pressure in the common tank. The overall system dynamics are given by:

$$\frac{d}{dt} \underbrace{\begin{bmatrix} \boldsymbol{x}_\mathrm{n,1} \\ \boldsymbol{x}_\mathrm{n,2} \\ p_\mathrm{t} \end{bmatrix}}_{\boldsymbol{x}_\mathrm{p}} = \underbrace{\begin{bmatrix} \boldsymbol{f}_\mathrm{n,1} \\ \boldsymbol{f}_\mathrm{n,2} \\ \frac{a^2}{V_\mathrm{t}} \left(q_\mathrm{d,1} + q_\mathrm{d,2} - q_\mathrm{d,t} \right) \end{bmatrix}}_{\boldsymbol{f}_\mathrm{p}}, \tag{3.7}$$

$$\begin{aligned} q_\mathrm{d,1} &= q_\mathrm{d,1}(\boldsymbol{\varepsilon}_1,\ u_\mathrm{d,1},\ p_\mathrm{t},\ p_\mathrm{d,1}) \\ q_\mathrm{d,2} &= q_\mathrm{d,2}(\boldsymbol{\varepsilon}_2,\ u_\mathrm{d,2},\ p_\mathrm{t},\ p_\mathrm{d,2}) \\ q_\mathrm{d,t} &= q_\mathrm{d,t}(\boldsymbol{\rho},\ u_\mathrm{t},\ p_\mathrm{t},\ p_\mathrm{out}) \end{aligned} \tag{3.8}$$

where:

- p_t and V_t are the pressure and the volume of the common tank, respectively;
- $q_\mathrm{d,t}$ is the mass flow across the common tank discharge valve;
- $\boldsymbol{\rho}$ contains the coefficients mapping the mass flow through the common tank discharge valve to the pressure differential across the valve;

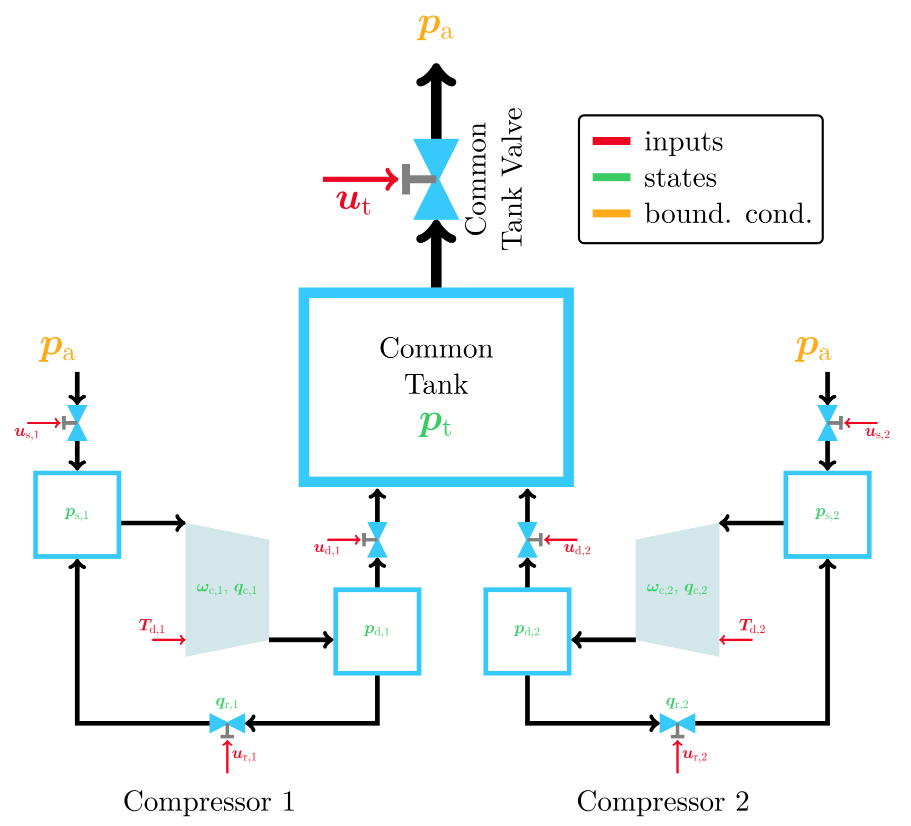


Figure 3.2: Diagram of parallel compressor system with inputs and states labelled.

- u_t is the discharge valve opening of the common tank; and
- the subscripts 1 and 2 refer to values for the first and second compressor, respectively.

The inputs to the system are thus the suction, discharge and recycle valve positions for each compressor, the torque input for each compressor, and the common tank discharge valve position. The discharge valves linking each compressor to the common tank ($u_{\mathrm{d},1}$ and $u_{\mathrm{d},2}$) are, however, kept in a fully open position in order to reduce the pressure losses in the system. The input vector used for control ($\boldsymbol{u}_\mathrm{p}$) contains the torque and recycle valve inputs for each of the two compressors:

$$\boldsymbol{u}_\mathrm{p} = \begin{bmatrix} \boldsymbol{u}_{\mathrm{n},1} \\ \boldsymbol{u}_{\mathrm{n},2} \end{bmatrix}. \tag{3.9}$$

The system outputs are given by:

$$\boldsymbol{y}_\mathrm{p} = \begin{bmatrix} \boldsymbol{y}_{\mathrm{n},1} \\ \boldsymbol{y}_{\mathrm{n},2} \\ p_\mathrm{t} \end{bmatrix} \tag{3.10}$$

where $\boldsymbol{y}_\mathrm{p}$ gives the output vector for the parallel system, and $\boldsymbol{y}_{\mathrm{n},1}$ and $\boldsymbol{y}_{\mathrm{n},2}$ are the output vectors of the first and second compressors, respectively.

3.4 Serial System

In the serial compressor system, the two compressors are arranged such that the discharge tank of the first compressor feeds into the suction tank of the second compressor, as shown in Figure 3.3. The pressure upstream of Compressor 1 as well as the pressure downstream of Compressor 2 are constrained to be at atmospheric pressure. Furthermore, the inlet valve to Compressor 2 has been replaced by the outlet valve of Compressor 1. The state equations for this system are thus given by:

$$\frac{d}{dt}\underbrace{\begin{bmatrix} \boldsymbol{x}_{\mathrm{n},1} \\ \boldsymbol{x}_{\mathrm{n},2} \end{bmatrix}}_{\boldsymbol{x}_\mathrm{s}} = \underbrace{\begin{bmatrix} \boldsymbol{f}_{\mathrm{n},1} \\ \boldsymbol{f}_{\mathrm{n},2} \end{bmatrix}}_{\boldsymbol{f}_\mathrm{s}}, \tag{3.11}$$

where $\boldsymbol{x}_\mathrm{s}$ and $\boldsymbol{f}_\mathrm{s}$ are the state vector and state derivative of the serial system, respectively, and the subscripts 1 and 2 refer to values for the first and second compressor, respectively, and the following constraints are applied representing the interconnection of the two compressors:

$$\begin{aligned} p_{\text{out},1} &= p_{\text{s},2} \\ p_{\text{in},2} &= p_{\text{d},1} \\ \boldsymbol{\gamma}_{,\mathbf{2}} &= \varepsilon_1 \\ u_{\text{s},2} &= u_{\text{d},1}. \end{aligned} \tag{3.12}$$

The valve connecting the two compressors (controlled by $u_{\text{d},1}$) is fixed at a fully open position in order to minimize pressure losses in the system. The input vector used for control ($\boldsymbol{u}_\text{s}$) contains the torque and recycle valve inputs for each of the two compressors:

$$\boldsymbol{u}_\text{s} = \begin{bmatrix} \boldsymbol{u}_{\text{n},1} \\ \boldsymbol{u}_{\text{n},2} \end{bmatrix}. \tag{3.13}$$

The system outputs are given by:

$$\boldsymbol{y}_\text{s} = \begin{bmatrix} \boldsymbol{y}_{\text{n},1} \\ \boldsymbol{y}_{\text{n},2} \end{bmatrix} \tag{3.14}$$

where $\boldsymbol{y}_\text{s}$ gives the output vector for the serial system, and $\boldsymbol{y}_{\text{n},1}$ and $\boldsymbol{y}_{\text{n},2}$ are the output vectors of the upstream and downstream compressors, respectively.

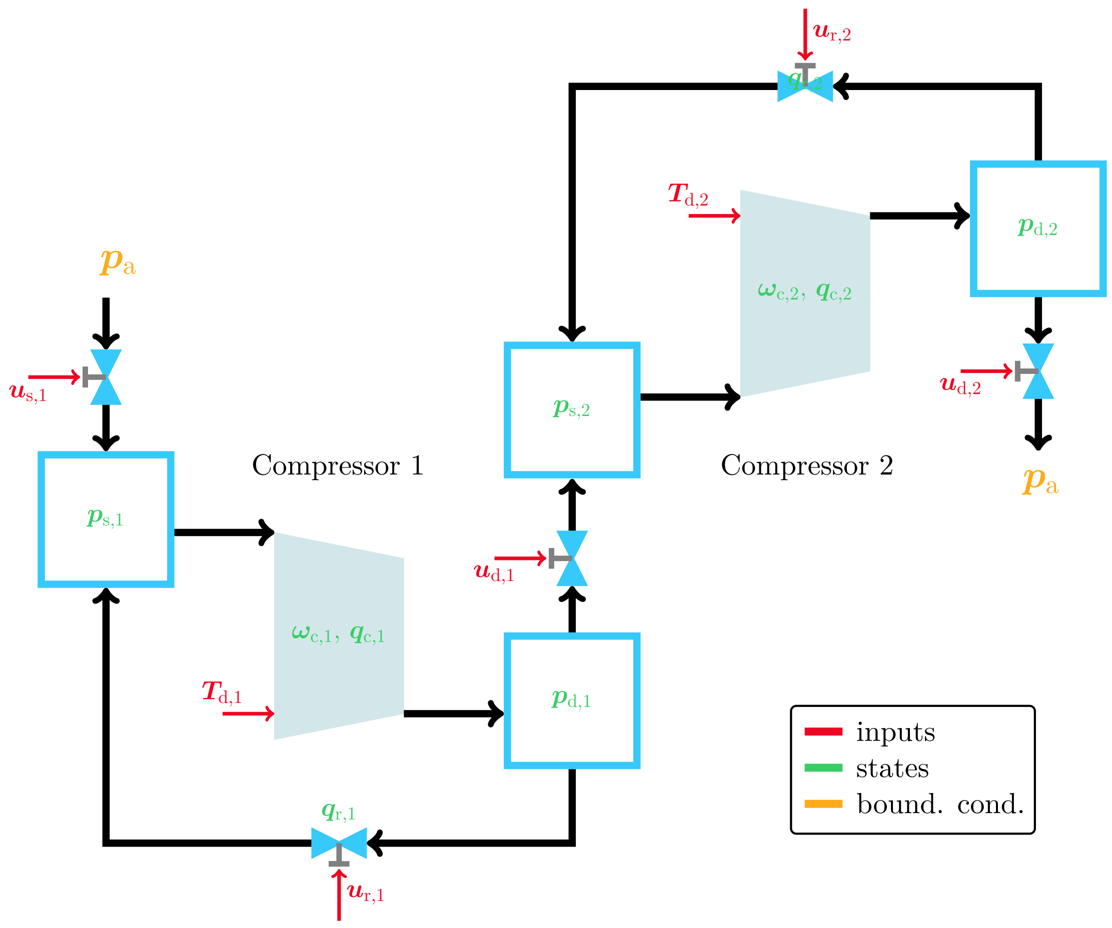


Figure 3.3: Diagram of serial compressor system with inputs and states labelled. p_a represents atmospheric pressure.

4

Model Predictive Control Formulation

4.1 Overview

Two variants of MPC are considered in this work: centralized and distributed control, described in the following sections. Centralized control is considered too computationally expensive to implement in practice, necessitating the development of more efficient distributed controllers, but it is used as a benchmark to evaluate the performance of the distributed controllers.

The model of the compressor systems used in this work is non-linear, however the optimization problem that results from the MPC formulation is in the form of an efficiently-solvable quadratic program (QP) only if a linear model is used. In order to take advantage of efficient QP solvers, a linearized approach is taken in this work whereby the non-linear system is re-linearized at each time step and the resulting linear model used to generate the optimization.

The following sections describe the process used to obtain a linearized model of the system, the formulation of the resulting optimization problem, the details of both the centralized and distributed controllers, and finally the procedure used for state estimation.

The controllers presented in this work combine both anti-surge and process control, with a primary focus on the former. For this reason, they operate at a sampling rate of $t_\mathrm{s} = 50\,\mathrm{ms}$ in order to control the relatively fast dynamics leading to compressor surge.

4.2 Linearization & Discretization

The model used for the MPC controller formulation is the same as the one used in Section 3 to simulate the compressor. No model mismatch was considered in order to compare the controllers' performance in an ideal scenario. In order to generate an optimization in the form of a quadratic program, the system was linearized at each time step about the current states ($\hat{\boldsymbol{x}}_k$) and inputs ($\boldsymbol{u}_{k-1}$), and discretized using the 4th order Runge-Kutta method, a fixed step-size integration scheme. The sampling time (50 ms) was used as the step size. The model at sampling instant k is thus given by:

$$\begin{aligned} \Delta\hat{\boldsymbol{x}}_{k+i+1} &= A_k \Delta\hat{\boldsymbol{x}}_{k+i} + B_k \Delta\boldsymbol{u}_{k+i} + \boldsymbol{f}_{\mathrm{d},k}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right) \\ \Delta\boldsymbol{y}_{k+i} &= C_k \Delta\hat{\boldsymbol{x}}_{k+i}, \qquad i \geq 0 \end{aligned} \tag{4.1}$$

where $\boldsymbol{f}_{\mathrm{d},k}$ is the derivative of the system evaluated at the linearization point, $\hat{\boldsymbol{x}}_k$ is the state estimate at time instant k, $A_k = A_k\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right)$, $B_k = B_k\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right)$ and $C_k = C_k\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right)$ give the model linearized about the current operating point, and $\Delta\left(\cdot\right)$ refers to the difference in $(\cdot)$ relative to the linearization point.

The resulting discrete-time, linear system is then augmented to include both error states as integrators for offset-free control, and delayed input states for the recycle valve. One integrator for each output is added, as well as 40 delayed states per compressor. As discussed in Section 3.2.3, the recycle valve has an input dead time, which is included in the model by adding delayed input states (a total delay of 2 s with a sampling rate of 50 ms leads to 40 delayed states per compressor). With these additions, the augmented state of the system ($\Delta\hat{\boldsymbol{x}}_k^a$) is given by:

$$\Delta\hat{\boldsymbol{x}}_k^a = \begin{bmatrix} \Delta\hat{\boldsymbol{x}}_k \\ \boldsymbol{u}_k^{\mathrm{delay}} \\ \boldsymbol{e}_k \end{bmatrix}, \qquad \boldsymbol{u}_k^{\mathrm{delay}} = \begin{bmatrix} u_{\mathrm{r},k-n_{\mathrm{del}}+1} & \cdots & u_{\mathrm{r},k-1} & u_{\mathrm{r},k} \end{bmatrix}^{\mathsf{T}} \tag{4.2}$$

where $\hat{\boldsymbol{x}}_k$ contains the original states of the system, $\boldsymbol{u}_k^{\mathrm{delay}}$ contains the delayed recycle valve inputs from the earliest to most recent, n_{del} is the total number of delayed states and $\boldsymbol{e}_k$ contains the integrator states.

The augmented system dynamics are then as follows:

$$\Delta\hat{\boldsymbol{x}}^a_{k+i+1} = \underbrace{\begin{bmatrix} A_k & \begin{bmatrix} B_k^{\text{delay}} & 0 \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} 0 & A^{\text{delay}} \end{bmatrix} & 0 \\ 0 & 0 & I_{n_\text{e}\times n_\text{e}} \end{bmatrix}}_{A_k^a} \left(\Delta\hat{\boldsymbol{x}}^a_{k+i} - \begin{bmatrix} 0 \\ \begin{bmatrix} u_{\text{r},k-1} \\ 0 \end{bmatrix} \\ 0 \end{bmatrix} \right) + \underbrace{\begin{bmatrix} B_k^{\text{nodelay}} \\ B^{\text{aug}} \\ 0 \end{bmatrix}}_{B_k^a} \underbrace{\begin{bmatrix} u_{\text{r},k+i} \\ \Delta T_{\text{d},k+i} \end{bmatrix}}_{\Delta\boldsymbol{u}^{\text{a}}_{k+i}} + \begin{bmatrix} \boldsymbol{f}_{\text{d},k} \\ 0 \\ 0 \end{bmatrix} \tag{4.3}$$

$$\Delta\boldsymbol{y}_k = \underbrace{\begin{bmatrix} C_k & 0 & I_{n_\text{e}\times n_\text{e}} \end{bmatrix}}_{C_k^a} \Delta\hat{\boldsymbol{x}}^a_k \tag{4.4}$$

where n_e gives the number of integrator states, and $A_k^a = A_k^a\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right)$, $B_k^a = B_k^a\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right)$ and $C_k^a = C_k^a\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_{k-1}\right)$ give the augmented, linearized model of the system.

The first component of $\boldsymbol{u}^{\text{delay}}_{k+i}$ is multiplied by B_k^{delay}, and must therefore be corrected by a factor $u_{\text{r},k-1}$, the value about which the system was linearized. The other components of $\boldsymbol{u}^{\text{delay}}_{k+i}$ are not corrected because they are only multiplied by A^{delay} which simply shifts them up a row.

The specific format of the augmented matrices depends on the system being modelled. For a single compressor, for example, the augmented matrices are as follows:

$$A_k^a = \begin{bmatrix} A_k & \begin{bmatrix} B_{k,u_\text{r}} & 0 \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} 0 & I_{39\times 39} \\ 0 & 0 \end{bmatrix} & 0 \\ 0 & 0 & I_{2\times 2} \end{bmatrix} \qquad B_k^a = \begin{bmatrix} B_{k,T_\text{d}} & 0 \\ 0 & \begin{bmatrix} 0_{39\times 1} \\ 1 \end{bmatrix} \\ 0 & 0 \end{bmatrix}$$

$$C_k^a = \begin{bmatrix} C_k & 0 & I_{n_\text{e}\times n_\text{e}} \end{bmatrix},$$

where B_{k,u_r} and B_{k,T_d} are the columns of B_k corresponding to the recycle valve and torque inputs, respectively. The augmented matrices for the parallel and serial systems are constructed in an equivalent manner.

4.3 Formulation of Optimization Problem

The optimization problem solved at each time step by the MPC controller is defined as follows:

$$\begin{aligned} U_k = \underset{U}{\arg\min}\ & \Delta U^\intercal\ Q\ \Delta U + \left(\Delta Y - \Delta Y_k^{\text{ref}}\right)^\intercal R \left(\Delta Y - \Delta Y_k^{\text{ref}}\right) \\ \text{s.t. } & \Delta U_{\min} \leq \Delta U \leq \Delta U_{\max} \\ & \Delta U_{\text{r,min}} \leq A_{\text{rate}} \Delta U \leq \Delta U_{\text{r,max}} \\ & \Delta Y_k = S_{U_k} \Delta U_k + S_{x_k} \Delta \hat{\boldsymbol{x}}_k^a + S_{f_k} \boldsymbol{f}_{\text{d},k}, \end{aligned} \tag{4.5}$$

where:

- U_k contains the optimal input sequence at time step k, stacked along the move horizon;
- Y_k contains the output sequence resulting from the input sequence U_k at time step k, stacked along the prediction horizon;
- Y_k^{ref} is the reference output sequence;
- Q and R are the weighting matrices used for the inputs and outputs, respectively;
- S_{U_k}, S_{x_k} and S_{f_k} are the prediction matrices giving the contribution to ΔY_k of ΔU_k, $\Delta \hat{\boldsymbol{x}}_k^a$ and $\boldsymbol{f}_{\text{d},k}$ respectively, calculated using (4.3) (see Appendix A);
- $\Delta U_{\min}$, $\Delta U_{\max}$, $\Delta U_{\text{r,min}}$ $\Delta U_{\text{r,max}}$ contain the input bounds and rate constraints; and
- A_{rate} is a matrix that is multiplied by ΔU_k to obtain the changes in inputs between two successive time steps in the sequence (see below for definition).

The optimization is then converted to a dense formulation by eliminating the dependence on ΔY_k through the equality constraint, yielding the following QP problem:

$$\begin{aligned} \underset{U}{\arg\min}\ & \frac{1}{2} \Delta U^\intercal\ H\ \Delta U + g^\intercal \Delta U \\ \text{s.t. } & \Delta U_{\min} \leq \Delta U \leq \Delta U_{\max}, \end{aligned} \tag{4.6}$$

with the QP Hessian matrix and linear term given by:

$$
\begin{aligned}
H &= 2\left(Q + S_{U_k}^{\intercal}\ R\ S_{U_k}\right) \\
g &= 2\left(\Delta\hat{\boldsymbol{x}}_k^a S_{x_k}^{\intercal} + \boldsymbol{f}_{\mathrm{d},k} S_{f_k}^{\intercal} - \Delta Y_k^{\mathrm{ref}}\right) R S_{U_k}.
\end{aligned} \tag{4.7}
$$

The input constraints are determined by combining limits on both the range of the inputs and on their rate of change. The recycle valve has a range of 0–1 with rate constraints (maximum possible change over a single sampling period) of +1/-0.1. The rate is more constrained in the negative direction (i.e. when closing) to prevent a transient re-entry into surge. The torque input has a normalized range of ±0.3 compared to its steady-state value, with rate constraints of ±0.1.

Since the solution to the QP problem is the change in inputs relative to the previous iteration that should be applied, the constraints are implemented as follows:

$$
\Delta U_{\mathrm{min},k} = \begin{bmatrix} \begin{pmatrix} -0.3 \\ 0 \\ -0.3 \\ 0 \end{pmatrix} - \begin{pmatrix} T_{\mathrm{d},1,k} \\ u_{\mathrm{r},1,k} \\ T_{\mathrm{d},2,k} \\ u_{\mathrm{r},2,k} \end{pmatrix} \\ \vdots \end{bmatrix}
\qquad
\Delta U_{\mathrm{max},k} = \begin{bmatrix} \begin{pmatrix} 0.3 \\ 1 \\ 0.3 \\ 1 \end{pmatrix} - \boldsymbol{u}_k \\ \vdots \end{bmatrix}
$$

$$
\Delta U_{\mathrm{r,min},k} = \begin{bmatrix} \begin{pmatrix} -0.1 \\ -0.1 \\ -0.1 \\ -0.1 \end{pmatrix} \\ \vdots \end{bmatrix}
\qquad
\Delta U_{\mathrm{r,max},k} = \begin{bmatrix} \begin{pmatrix} 0.1 \\ 1 \\ 0.1 \\ 1 \end{pmatrix} \\ \vdots \end{bmatrix}
$$

where the constraints are repeated m times for each of the inputs in the sequence ΔU_k. The rate constraints are implemented as follows:

$$A_{\text{rate}} \in \mathbb{R}^{8m\times 4m}, \quad A_{\text{rate}} = \begin{bmatrix} 1 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 1 & 0 & \\ 0 & 0 & 0 & 1 & \\ -1 & 0 & 0 & 0 & \ddots \\ 0 & -1 & 0 & 0 & \\ 0 & 0 & -1 & 0 & \\ 0 & 0 & 0 & -1 & \\ & & & & \ddots \end{bmatrix}.$$

4.4 Centralized Control Approach

In the centralized control approach, a single MPC controller is used to optimize for all system inputs using a single cost function, defined in Section 4.6. At each sampling period, the optimal input is computed using the following algorithm:

1. perform estimation to obtain the current estimate of the augmented state vector (see Section 4.7);
2. linearize, discretize and augment non-linear model about the current state estimate and previous inputs, as described in Section 4.2;
3. generate the prediction matrices using (4.3);
4. set up the QP problem according to (4.6) and solve using the `qpOASES` solver;
5. apply the optimal input at the first prediction interval to the system.

The algorithm is similar to that depicted in Figure 4.1 for a distributed controller, although the centralized QP solver does not have an iterative solving process.

For both the parallel and serial compressor systems, the centralized controller solves for 4 inputs (2 per compressor).

4.5 Distributed Control Approach

4.5.1 Optimization Problem

In the distributed MPC approach, the control problem is split into two smaller optimizations, each giving the optimal input for a single compressor, and each solved by a sub-controller. The two sub-controllers are assumed to have full state information and the optimizations are set up as QPs, as in (4.6). The Hessian (H) term of the QP is also identical. A slight modification to the linear term – given for the centralized case in (4.7) – is necessary to account for the fact that the QP is solved for only part of the optimal input vector:

$$\begin{aligned} H_{\text{distributed}} &= 2\left(Q + S_{U_k}^{\intercal}\ R\ S_{U_k}\right) \\ &= H_{\text{centralized}} \\ g_{\text{distributed}} &= 2\left(\Delta\hat{\boldsymbol{x}}_k^a S_{x_k}^{\intercal} + \boldsymbol{f}_{\text{d},k} S_{f_k}^{\intercal} - \Delta Y_k^{\text{ref}} + \Delta U_k^{\text{ot}}\ S_{U_k^{\text{ot}}}^{\intercal}\right) R\ S_{U_k} \\ &= g_{\text{centralized}} + \Delta U_k^{\text{ot}}\ S_{U_k^{\text{ot}}}^{\intercal} R\ S_{U_k} \end{aligned} \tag{4.8}$$

where U_k^{ot} is the optimal input at time step k calculated by the other sub-controller, and $S_{U_k^{\text{ot}}}$ is the prediction matrix giving the effect of the other sub-controller's inputs on the current outputs ΔY_k.

As stated in (4.8) the QP is unsolvable since U_k^{ot} in turn depends on the solution. To break this circular dependency, the problem is initially solved with an estimate for U_k^{ot} obtained from the previous QP solution. The sub-controllers then exchange information about their obtained solution, updating the value of U_k^{ot} and re-solving the optimization. This procedure is repeated for a fixed number of iterations, after which the optimal solution is sent to the plant. The iteration process may not converge for systems that have a high degree of coupling; this effect is discussed in further detail in [3].

The algorithm used to obtain the optimal input at each time step for a distributed controller is thus as follows:

1. perform estimation to obtain the current estimate of the augmented state vector (see Section 4.7);
2. linearize, discretize and augment non-linear model about the current state estimate and previous inputs, as described in Section 4.2;
3. generate the prediction matrices using (4.3);
4. set up the QP problems according to (4.6);
5. approximate the solution from each sub-controller using the solution from the previous iteration and use it to correct the linear QP terms;
6. iteratively solve each QP, updating the approximation of U_k^{ot} after each iteration;

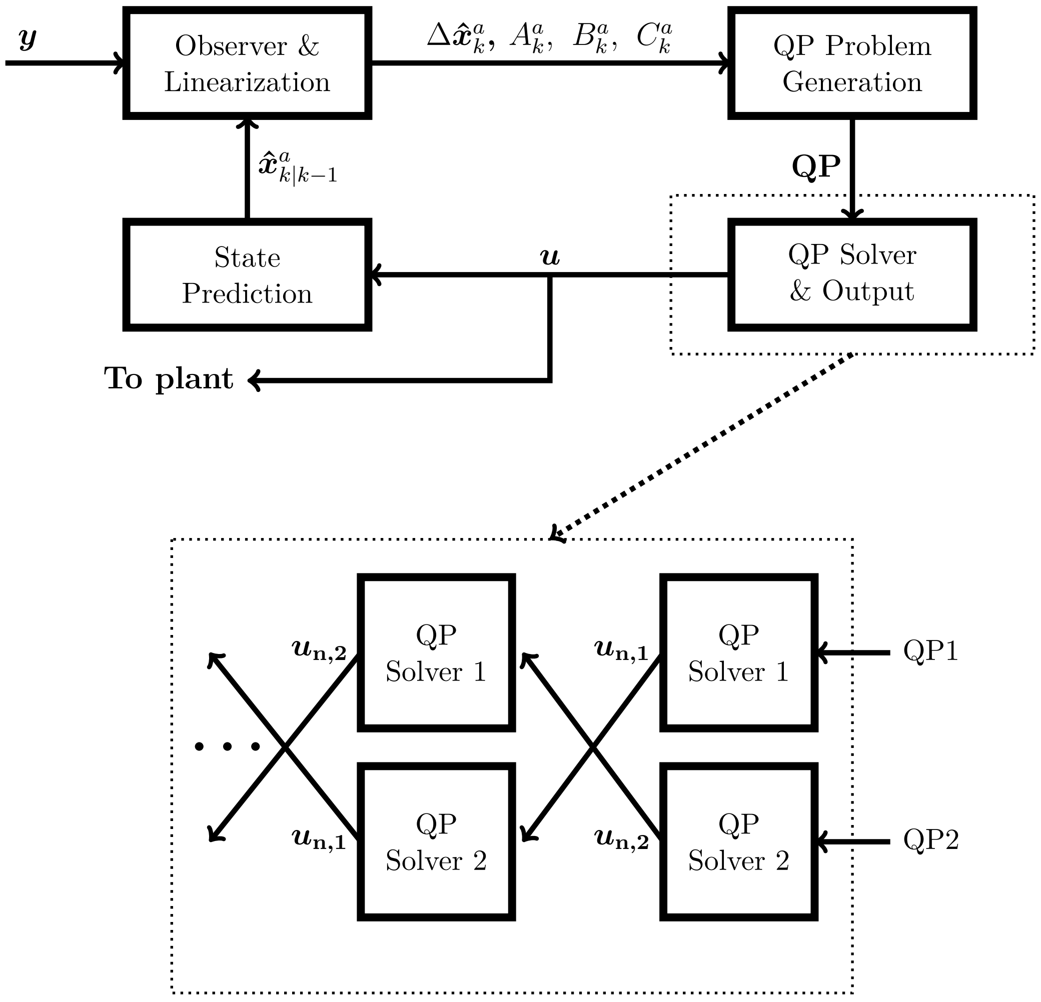

$\boldsymbol{y}$
Observer & Linearization
$\Delta\hat{\boldsymbol{x}}_k^a,\ A_k^a,\ B_k^a,\ C_k^a$
QP Problem Generation
$\hat{\boldsymbol{x}}_{k|k-1}^a$
QP
State Prediction
$\boldsymbol{u}$
QP Solver & Output
To plant
QP Solver 1
QP Solver 2
QP Solver 1
QP Solver 2
$\boldsymbol{u}_{\mathbf{n},\mathbf{2}}$
$\boldsymbol{u}_{\mathbf{n},\mathbf{1}}$
$\boldsymbol{u}_{\mathbf{n},\mathbf{1}}$
$\boldsymbol{u}_{\mathbf{n},\mathbf{2}}$
QP1
QP2

Figure 4.1: Diagram of controller algorithm. Dashed rectangle shows procedure of distributed QP solver.

7. after a fixed number of iterations, apply the optimal input from each sub-controller at the first prediction interval to the system.

The algorithm and the iterative solving procedure is depicted in Figure 4.1.

4.5.2 Stopping Criterion

The distributed optimization problem is iterative and a stopping criterion must therefore be chosen. To maintain a relatively constant computation time, the best approach is to choose a fixed number of iterations for which the tradeoff between performance and computational complexity is optimal. An alternate approach is to use the change in cost function as a criterion for convergence, however this leads to a variable number of iterations (and thus greater variation in the computation times) and it requires the cost

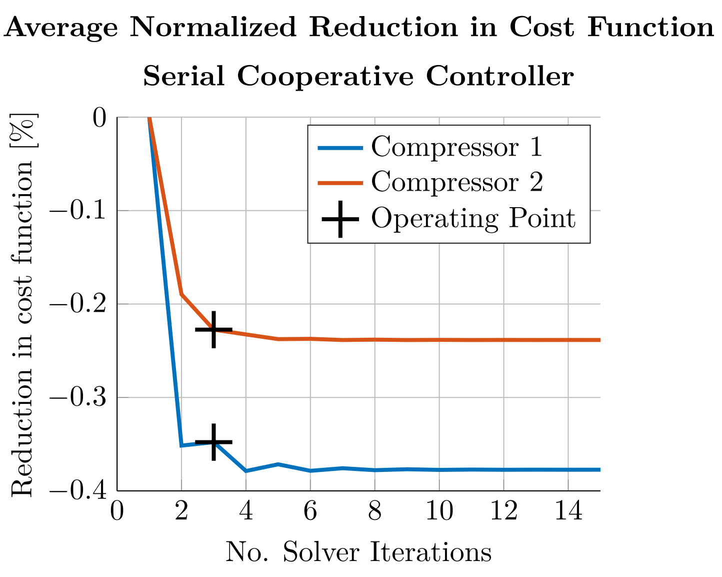


Figure 4.2: Average reduction in cost function as a function of the number of solver iterations for the serial, cooperative controller. Results are similar for other distributed controllers. 3 iterations were determined to be sufficient for convergence since further gains in the cost function are limited.

function to be computed at each solver iteration – adding a non-negligible computational cost. [1]

For controllers whose performance is significantly dependent on the number of iterations, this additional cost could be justified, however the present systems' cost functions converge relatively quickly, as shown in Figure 4.2. The average improvement between its value at 3 iterations and the optimal value reached is just 0.03% for the serial, cooperative controller, for example. The other distributed controllers show similar behavior. Furthermore, the controllers exhibited no measurable change in performance for solver iterations greater than 3. The number of iterations was therefore fixed at 3 for all distributed controllers.

4.5.3 Computational Cost

The primary advantage of distributed control over centralized control is that it allows the computational cost of the MPC controller to be reduced. The QPs can be solved in parallel using a separate thread or even a separate device for each sub-controller. Although the method still requires solving several QPs per sub-controller at each time

[1]The cost function used by the QP solver (defined in (4.6)) is not equal to the actual cost function of the control problem (defined in (4.9)) as terms that do not depend on the optimization variable are excluded. Evaluating the cost function therefore requires (4.9) to be computed. This computation is relatively expensive since ΔY_k must first be evaluated using the prediction matrices.

step, they are smaller than the single QP solved by the centralized controller and can often be solved more efficiently. In particular, the "hotstart" method implemented in the `qpOASES` QP solver can solve a QP using a previous solution as a starting point, efficiently solving series of QPs in which the Hessian and linear terms change only slightly, further reducing the computational cost of the extra iterations.

Finally, generating the prediction matrices for the sub-controllers' cost functions can be less computationally expensive than for the centralized case, if they use fewer of the outputs, since computing the prediction matrices is $\mathcal{O}(n)$ in the number of outputs used. The computational cost could be further reduced by limiting the state information used by each sub-controller to generate its prediction matrices ($\mathcal{O}(n^3)$ in the number of states used), however this approach is not considered here. These effects are quantified and discussed further in Section 5.2.

4.6 Centralized and Distributed Cost Functions

4.6.1 Definition

The cost functions used in an MPC controller combine (generally quadratic) weights on certain inputs and outputs of the system, which determine which variables should be minimized by the optimization. The centralized, cooperative and non-cooperative controllers differ in the inputs and outputs to which they assign nonzero weights.

The centralized controller optimizes for all inputs of both compressors and thus assigns weights to each component of the input vector, as well as to all outputs that are to be controlled.

A distributed controller contains multiple sub-controllers, each of which performs its own optimization using its own cost function. Two different types of distributed controllers are defined: cooperative and non-cooperative, which differ in how their cost functions are structured. In cooperative control, the sub-controllers share a single cost function, while in non-cooperative control they optimize individual cost functions relating to their own subsystems. Non-cooperative control therefore allows the possibility for each sub-controller to consider fewer outputs in the cost function, thereby reducing the computational cost of generating the QP to be solved.

A sample cost function from the centralized controller for the parallel system ($J_{\mathrm{p,k}}^{\mathrm{cen}}$) is given by:

$$J_{\mathrm{p,k}}^{\mathrm{cen}} = \sum_{i=1}^{p} \left(\boldsymbol{y}_{\mathrm{p},k+i}^{\mathrm{cen}}\right)^{\mathsf{T}} W_{\mathrm{p,y}}^{\mathrm{cen}} \left(\boldsymbol{y}_{\mathrm{p},k+i}^{\mathrm{cen}}\right) + \left(\boldsymbol{u}_{\mathrm{p},k+i}^{\mathrm{cen}}\right)^{\mathsf{T}} W_{\mathrm{p,u}}^{\mathrm{cen}} \left(\boldsymbol{u}_{\mathrm{p},k+i}^{\mathrm{cen}}\right) \tag{4.9}$$

where $\boldsymbol{y}_{\mathrm{p},k}^{\mathrm{cen}}$and $\boldsymbol{u}_{\mathrm{p},k}^{\mathrm{cen}}$ contain the components of $\boldsymbol{y}_{\mathrm{p}}$ and $\boldsymbol{u}_{\mathrm{p}}$, respectively, that have a

Table 4.1: Weighted input and output vectors used in cost functions for centralized, cooperative and non-cooperative controllers.

Controller type		Parallel		Serial	
		$\boldsymbol{y}_{\mathrm{p},k}^{(\cdot)}$	$\boldsymbol{u}_{\mathrm{p},k}^{(\cdot)}$	$\boldsymbol{y}_{\mathrm{s},k}^{(\cdot)}$	$\boldsymbol{u}_{\mathrm{s},k}^{(\cdot)}$
Centralized	(cen)	$\begin{bmatrix} SD_{1,k} & SD_{2,k} & p_{\mathrm{t,k}} \end{bmatrix}^{\mathsf{T}}$	$\boldsymbol{u}_{\mathrm{p}}$	$\boldsymbol{y}_{\mathrm{s}}$	$\boldsymbol{u}_{\mathrm{s}}$
Cooperative 1	(co,1)	$\begin{bmatrix} SD_{1,k} & SD_{2,k} & p_{\mathrm{t,k}} \end{bmatrix}^{\mathsf{T}}$	$\boldsymbol{u}_{\mathrm{n},1}$	$\boldsymbol{y}_{\mathrm{s}}$	$\boldsymbol{u}_{\mathrm{n},1}$
Cooperative 2	(co,2)	$\begin{bmatrix} SD_{1,k} & SD_{2,k} & p_{\mathrm{t,k}} \end{bmatrix}^{\mathsf{T}}$	$\boldsymbol{u}_{\mathrm{n},2}$	$\boldsymbol{y}_{\mathrm{s}}$	$\boldsymbol{u}_{\mathrm{n},2}$
Non-cooperative 1	(nc,1)	$\begin{bmatrix} SD_{1,k} & p_{\mathrm{t,k}} \end{bmatrix}^{\mathsf{T}}$	$\boldsymbol{u}_{\mathrm{n},1}$	$\boldsymbol{y}_{\mathrm{n},1}$	$\boldsymbol{u}_{\mathrm{n},1}$
Non-cooperative 2	(nc,2)	$\begin{bmatrix} SD_{2,k} & p_{\mathrm{t,k}} \end{bmatrix}^{\mathsf{T}}$	$\boldsymbol{u}_{\mathrm{n},2}$	$\boldsymbol{y}_{\mathrm{n},2}$	$\boldsymbol{u}_{\mathrm{n},2}$

nonzero weight in the centralized controller, and $W_{\mathrm{p,y}}^{\mathrm{cen}}$ and $W_{\mathrm{p,u}}^{\mathrm{cen}}$ are the weighting matrices that assign the relative cost of each of the inputs and outputs. These weighting matrices are tuning parameters for the controllers and can be adjusted to, for example, reduce the aggressiveness of the controller, or assign more weight to a specific output.

Cost functions using analogous weighted input and output vectors, and corresponding weighting matrices are similarly defined for the 5 other controllers. The weighted input and output vectors for each of the controllers are given in Table 4.1.

4.6.2 Choice of Weighted Variables

The distributed controllers, both cooperative and non-cooperative, have weights only on the inputs corresponding to the compressor they control (either $\boldsymbol{u}_{\mathrm{n},1}$ or $\boldsymbol{u}_{\mathrm{n},2}$). The centralized controllers optimize for all inputs, and as a result have weights on all inputs of both compressors.

The outputs that are weighted vary depending on the system and type of controller. For the parallel system, the controller should maintain the discharge pressure of the common tank at the setpoint while keeping both compressors away from surge. Accordingly, the centralized controller assigns weights to the discharge pressure of the common tank and the surge distances of both compressors. The cooperative controllers are, by definition, minimizing the same cost function, so they assign weights to the same outputs as the centralized controller. In industrial applications, a 4th output could be added to both the centralized and cooperative controllers to account for load sharing between the compressors, however for the simulation case considered here with two identical compressors such an output would have no effect.

The non-cooperative controllers minimize two separate cost functions related to their own outputs. For this reason, they each assign weight to only one compressor's surge distance and to the process variable (the common tank discharge pressure). As for the other controllers, load sharing could be considered by adding weight to the individual discharge pressures of each compressor.

The serial system has similar control requirements to the parallel system: the discharge pressure of the downstream compressor should be maintained at the setpoint and both compressors should be kept out of surge. In industry, a load sharing target defining, for example, the desired pressure ratios across each of the compressors would also be given in order to achieve optimal system efficiency. Since the two compressors in this model are considered to be identical, the desired pressure ratios were defined to be equal for both compressors. Unlike in the parallel system, the two compressors in the serial system are not operating under identical conditions, and this requirement had to be explicitly specified in the form of a setpoint for the discharge pressure of the upstream compressor.

To achieve these three objectives, the centralized and cooperative controllers weighted both the surge distances and the discharge pressures of both compressors, while the non-cooperative controllers considered the surge distance and discharge pressure of a single compressor.

4.6.3 Tuning

The primary goal of all of the controllers was anti-surge control, with process control (i.e. pressure setpoint tracking) a secondary consideration; the surge distances were accordingly assigned higher weights than the pressures in each of the cost functions. For the serial controller, the output pressure of the second (downstream) compressor was also weighted higher than that of the first to assign more importance to process control over load sharing.

The weights were assigned to react quickly to disturbances in the surge distance and limit the minimum surge distance reached after the disturbances were applied. The inputs were weighted in such a way as to prevent the controller from becoming too aggressive and oscillatory. The settling time for the discharge pressure in both the parallel and serial systems was not considered directly, however increasing weights on these pressures were found to actually improve the surge distance response.

This effect is due to the fact that MPC is performed on a linearized model, in which the torque input to a compressor has a positive effect on the surge distance. In reality, the torque has only a short-term positive effect on the surge distance, after which time the increased discharge pressure causes it to decrease – an effect that is not well-captured by the linearized models. Increasing the weight on the pressures limits this effect by keeping the discharge pressures lower, thus improving the surge distance.

4.7 State Estimation

All controllers are assumed to have full state information available to generate their prediction matrices. The state estimate is obtained using an observer with a static gain M as follows:

$$\begin{aligned}\hat{\boldsymbol{x}}^a_{k|k-1} &= A^a_{k-1}\left(\Delta\hat{\boldsymbol{x}}^a_{k-1} - \boldsymbol{u}_{\mathrm{r},k-2}\right) + B^a_{k-1}\Delta\boldsymbol{u}_{k-1}\\ \hat{\boldsymbol{x}}_k &= \hat{\boldsymbol{x}}^a_{k|k-1} + M\left(\boldsymbol{y}_k - \boldsymbol{y}_{k-1} - C^a_k\hat{\boldsymbol{x}}^a_{k|k-1}\right)\end{aligned} \tag{4.10}$$

where $\hat{\boldsymbol{x}}^a_{k|k-1}$ is the a priori estimate of $\hat{\boldsymbol{x}}^a_k$ and $\boldsymbol{u}_{\mathrm{r},k-2}$ is the recycle valve input about which the system was linearized, as defined in (4.3).

The observer gain M was computed offline using a Kalman filter based on the linearized model at the design point of the system. It is assumed to be constant and not updated at each time step with the new linearized model for simplicity. Additionally, in the physical installation, the states are measured directly using sensors and the observer gain is not crucial.

4.8 Implementation

The simulation was first implemented in `Simulink` using the `ode45` solver for numerical integration of the state equations. The MPC controllers were also implemented in `Simulink` using `Embedded MATLAB` scripts. The `Embedded MATLAB` syntax allows C code to be automatically generated and compiled for use in simulation or deployment on embedded systems.

Some steps were taken to decrease the number of computations required, particularly for generating the prediction matrices. To generate these matrices, the terms $C^a_k A^{a\,1}_k$, $C^a_k A^{a\,2}_k$, ..., $C^a_k A^{a\,p}_k$ must be calculated, taking the bulk of the computation time. To speed this process up, the result of the multiplication $C^a_k A^{a\,i}_k$ was saved as an intermediate variable and used to calculate $C^a_k A^{a\,i+1}_k$.

For faster computational performance – particularly on embedded systems – and for a more accurate characterization of the computational efficiency of each controller, the controllers were also implemented and tested in C++. The `qpOASES` library presented in [19] was used to solve quadratic programs, and linear algebra calculations were performed using the `Eigen`[22] C++ library.

One of the advantages of the C++ implementation compared to that in `Simulink` is that it is able to use knowledge of the structure of the augmented matrices (A_k^a, B_k^a and C_k^a) to multiply them more efficiently. The matrices have both dense sections, corresponding to the non-augmented states, sparse sections, corresponding to the augmented states, and sections which are zero by definition. New types were therefore defined for each of the 3 matrices that took this structure into account. For A_k^a, for example, the matrix is as follows (with dimensions for each section for the parallel and serial systems shown in parentheses):

$$A_k^a = \begin{array}{c} \begin{array}{cccc} \scriptstyle (10-11) & \scriptstyle (2) & \scriptstyle (78) & \scriptstyle (4) \end{array} \\ \begin{bmatrix} A_k & B_k^{\text{delay}} & 0 & 0 \\ 0 & 0 & A^{\text{delay}} & 0 \\ 0 & 0 & 0 & I_{4\times4} \end{bmatrix} \end{array} \tag{4.11}$$

The A_k and B_k^{delay} originate from the non-augmented system and are dense. The A^{delay} section, in contrast, is quite sparse: its number of nonzero entries is given by the number of total delayed states, in this case 80, minus the number of inputs that are delayed, in this case 2. All of the nonzero entries are furthermore equal to 1 by definition, since their purpose is to shift a delayed state up one row. To take advantage of this structure, both the A_k and B_k^{delay} sections are implemented as dense matrices of double values, while A^{delay} and $I_{4\times4}$ are combined into a single sparse matrix of booleans with one entry per column. This modification was found to reduce the computation time required for a single controller iteration by a factor of roughly 4 compared to an implementation with fully dense matrices.

5

Results

5.1 Controller Performance

In order to test the controller performance of the various control implementations, a single benchmark disturbance case was chosen. It is designed to mimic a typical disturbance downstream of the system such as the shutdown of another compressor. The disturbances used were as follows:

- **parallel**: common tank discharge valve (u_{t}) 70% -> 40% open;
- **serial**: downstream compressor discharge valve ($u_{\mathrm{d},2}$) 39% -> 29% open.

In the following plots, the disturbances are persistent and applied at 50 s, after the system has reached steady state.

The distributed controllers in this section were implemented using 3 controller iterations. As discussed in Section 4.5.2, this number was determined sufficient for convergence of the applied inputs.

5.1.1 Parallel System

The time response of each controller for the parallel system is shown in Figure 5.1. The responses obtained using each of the three controllers are virtually identical – in this case, using a distributed control approach has no performance penalty and a potential decrease in computational cost (see below). The controller weights used to generate these results are summarized in Table 5.1.

The disturbance applied has the effect of decreasing the mass flow out of, and increasing the pressure in, the common discharge tank, thus pushing both compressors toward surge. At this point, the recycle valve opens but with a 2 s delay. During this delay

Table 5.1: Controller weights used for the parallel system. Surge distance is normalized by a factor of 1000 to have similar weight ranges.

		Centralized	Coop.	Non-coop. 1	Non-coop. 2
Torque	$T_{d,1}$	20	19	22	0
	$T_{d,2}$	20	19	0	22
Recycle valve	$u_{r,1}$	200	190	220	0
	$u_{r,2}$	200	190	0	220
Surge distance	SD_1	1	1	1	0
	SD_2	1	1	0	1
Common tank pressure	p_t	0.5	0.42	0.6	0.6

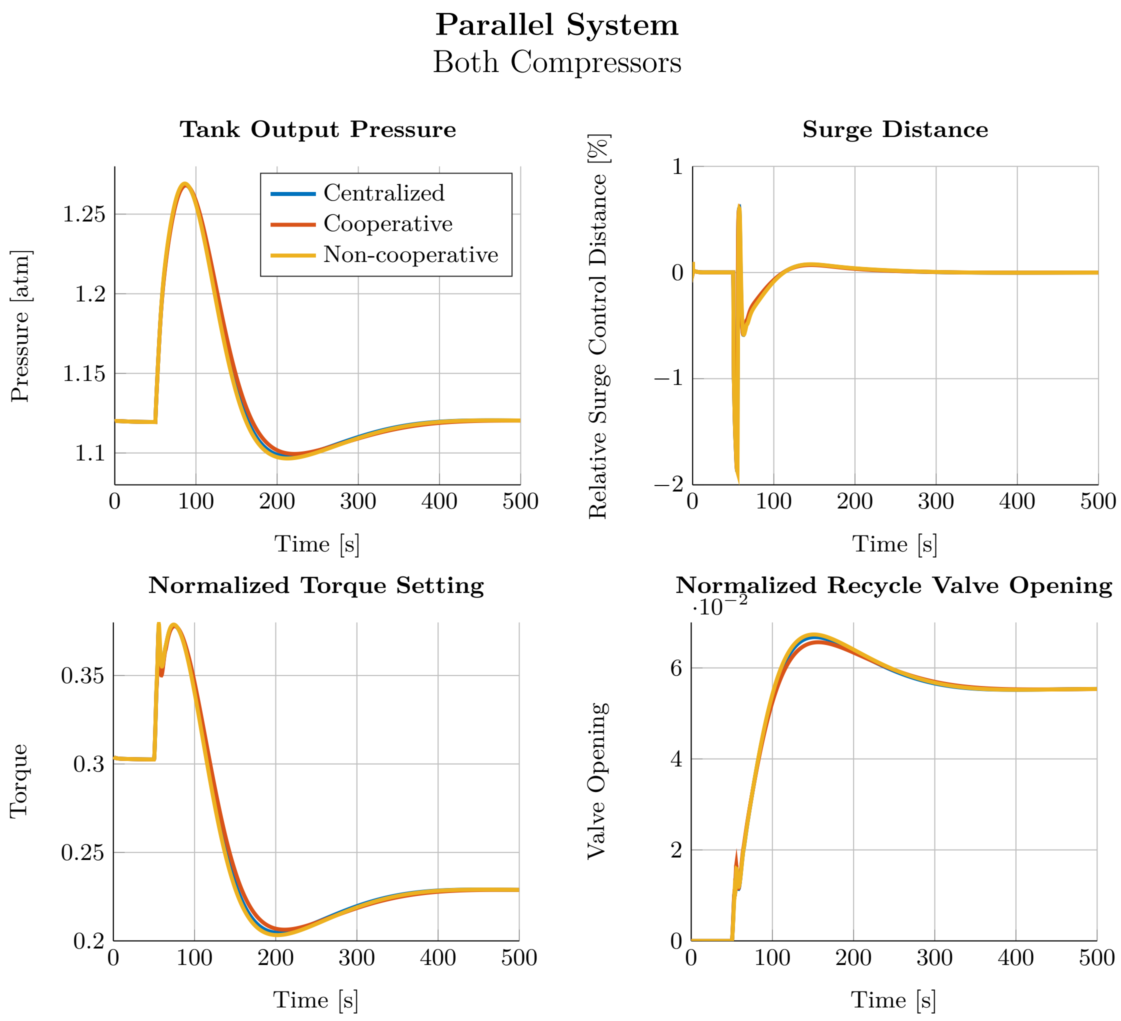


Figure 5.1: Comparison of time responses of centralized and distributed controllers. Distributed controllers use 3 solver iterations. The disturbance applied is a closing of the common tank discharge valve from 70% to 40% at time 50 s.

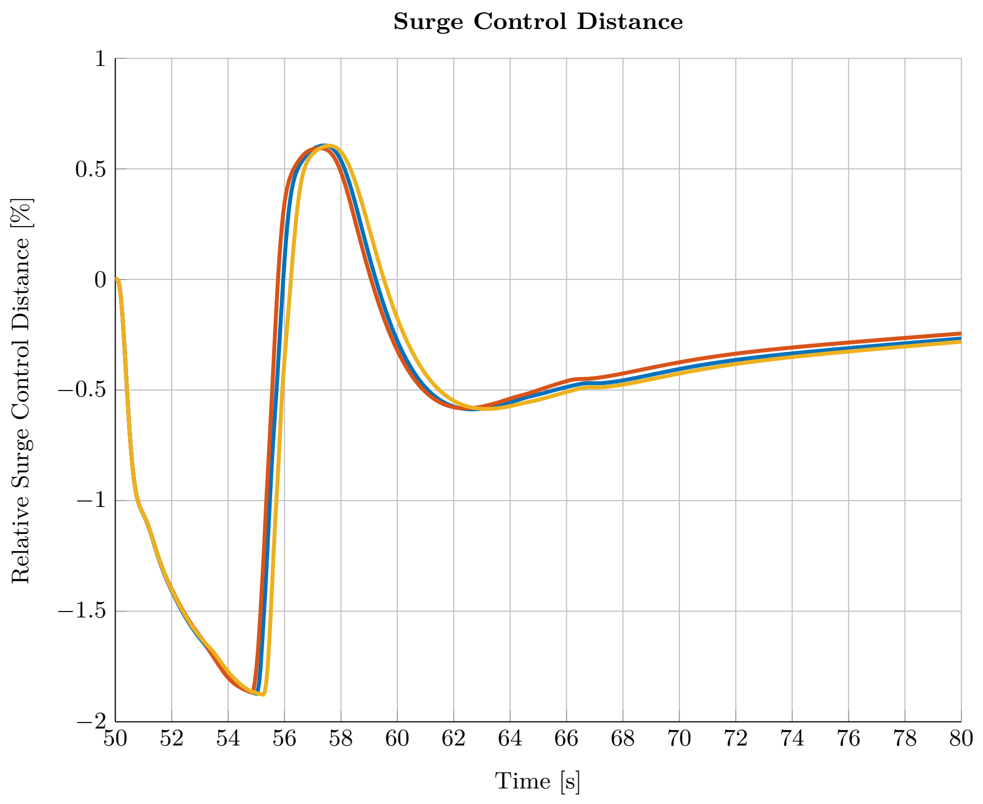


Figure 5.2: Zoomed view of surge distance time response given in Figure 5.1. The disturbance applied is a closing of the common tank discharge valve from 70% to 40% at time 50 s.

Table 5.2: Integral squared error (ISE) and integral absolute error (IAE) measures for parallel controllers.

	Centralized		Cooperative		Non-cooperative	
	ISE	IAE	ISE	IAE	ISE	IAE
T_d	0.0026	0.027	0.0026	0.028	0.0026	0.028
u_r	0.00012	0.0057	0.00011	0.0056	0.00012	0.0059
SD	0.035	0.057	0.033	0.054	0.036	0.059
p_t	0.0023	0.025	0.0024	0.026	0.0023	0.025

Table 5.3: Controller weights used for the serial system. Surge distance is normalized by a factor of 1000 to have similar weight ranges.

		Centralized	Coop.	Non-coop. 1	Non-coop. 2
Torque	$T_{\mathrm{d},1}$	20	41	30	0
	$T_{\mathrm{d},2}$	20	25	0	30
Recycle valve	$u_{\mathrm{r},1}$	250	500	500	0
	$u_{\mathrm{r},2}$	250	500	0	500
Compressor output pressure	$p_{\mathrm{d},1}$	0.2	0.75	1.9	0
	$p_{\mathrm{d},2}$	1	1.5	0	1.9
Surge distance	SD_1	1	4	3	0
	SD_2	8	8	0	3

period, the controller also increases the torque applied to the driver, which increases the mass flow through the compressor, quickly pulling it away from the surge line.

The increased torque from both compressors further increases the pressure in the common discharge tank, though the pressure reacts on a much slower time scale than the surge distance. Once the recycle valve begins to take effect, the torque applied to both compressors is decreased by the controller in order to bring the discharge pressure back to its setpoint.

The integral squared error (ISE) and integral absolute error (IAE) are shown in Table 5.2 for all controllers.

5.1.2 Serial System

The time response of the serial system is shown in Figure 5.3. A zoomed view of the initial surge distance response is shown in Figure 5.4. The controller weights used to generate these results are summarized in Table 5.3.

In contrast to the parallel case, differences in the controller response and performance can be observed for each of the control approaches. The cooperative controller response is qualitatively similar to that of the centralized controller. It is, however, somewhat less aggressive in its regulation of the output pressure, evidenced by its less extreme

Table 5.4: Integral squared error (ISE) and integral absolute error (IAE) measures for serial controllers.

	Centralized		Cooperative		Non-cooperative	
	ISE	IAE	ISE	IAE	ISE	IAE
$T_{\mathrm{d},1}$	0.0012	0.012	0.001	0.0079	0.0015	0.019
$u_{\mathrm{r},1}$	8.6e-05	0.0037	5.6e-05	0.0024	0.00011	0.0053
$p_{\mathrm{d},1}$	0.00096	0.011	0.0006	0.0067	0.00081	0.014
SD_1	0.052	0.063	0.031	0.041	0.078	0.13
$T_{\mathrm{d},2}$	0.0027	0.016	0.00095	0.0073	0.0017	0.021
$u_{\mathrm{r},2}$	1.1e-05	0.0011	2.7e-05	0.0016	3.2e-05	0.0026
$p_{\mathrm{d},2}$	0.00047	0.0051	0.00058	0.0049	0.00029	0.005
SD_2	0.056	0.032	0.064	0.035	0.1	0.064

change from high to low torque in the downstream compressor and faster return to the steady-state torque value, as shown in Figure 5.3g. As a result, the output pressure of the downstream compressor has a much lower overshoot than either the centralized or non-cooperative controller.

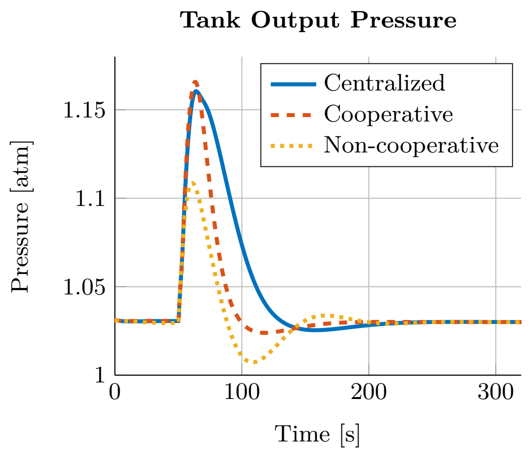
Similarly, in the upstream compressor, the cooperative controller's torque input – shown in Figure 5.3c – is smoother than that of the centralized controller with no uptick near 70 s, and a faster convergence of both inputs to their steady-state values. Accordingly, the upstream compressor's output pressure also converges faster for the cooperative controller than for the centralized one. The surge distances of the centralized and cooperative controllers, however, show an almost identical response for the downstream compressor and a similar response upstream, though that of the cooperative controller has a greater increase near 58 s (see Figure 5.4).

The non-cooperative controller response has a much different characteristic than that of the centralized controller. As shown in Figs. 5.3c and 5.3g, there is no significant initial increase in the torque input to either compressor when the disturbance is applied; as a result, both compressors are pushed further towards surge than in the centralized or cooperative case, and the surge distances are also slower to converge. The maximum disturbance to the output pressures of both compressors for the non-cooperative controller is accordingly reduced by approximately 40% when compared to the centralized case.

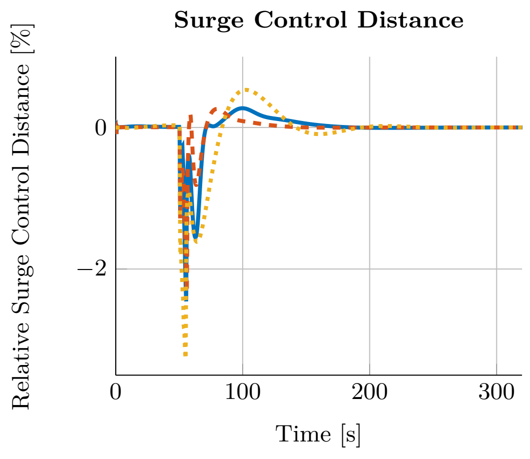
The integral squared error (ISE) and integral absolute error (IAE) are shown in Table 5.4 for all controllers.

Serial System

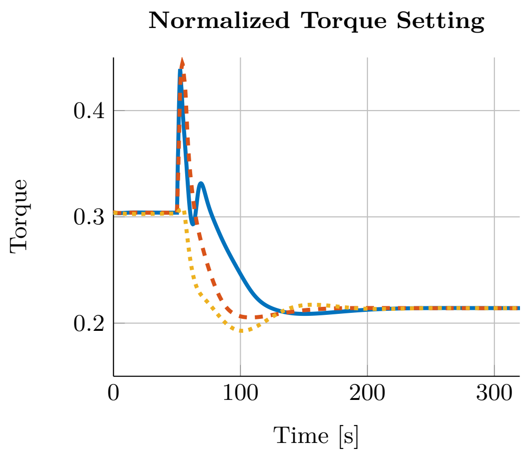
Upstream Compressor



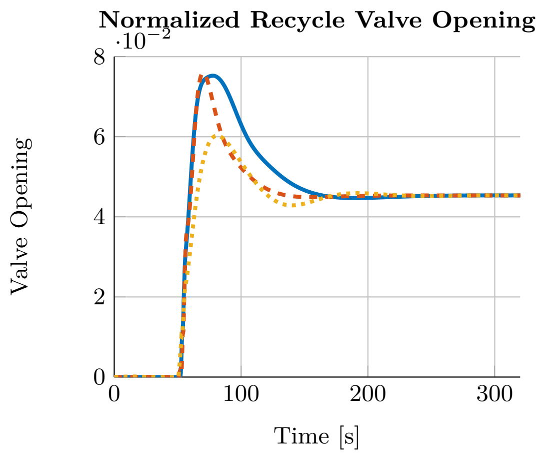
(a)



(b)

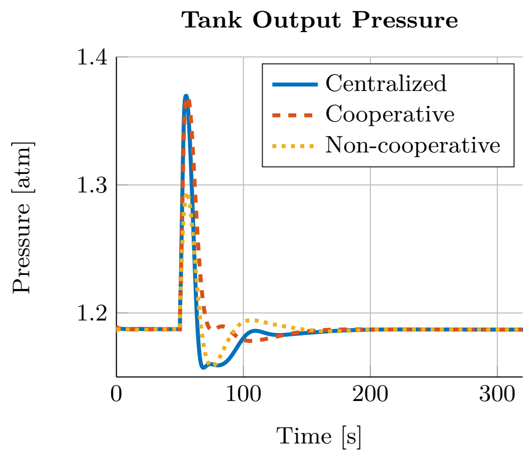


(c)

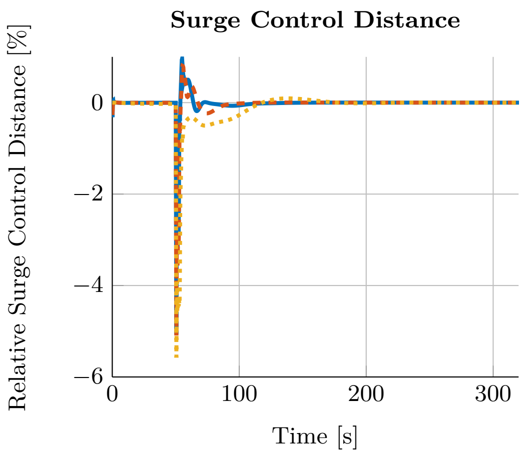


(d)

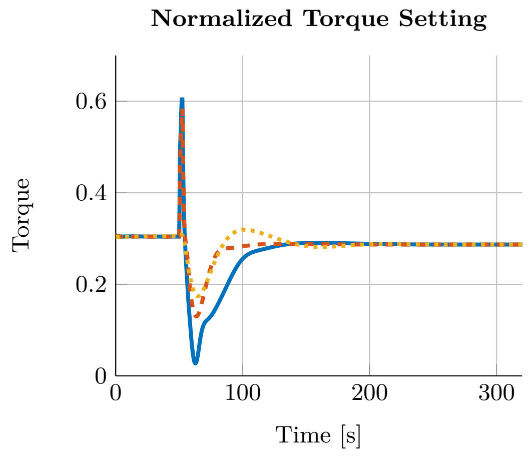
Serial System
Downstream Compressor



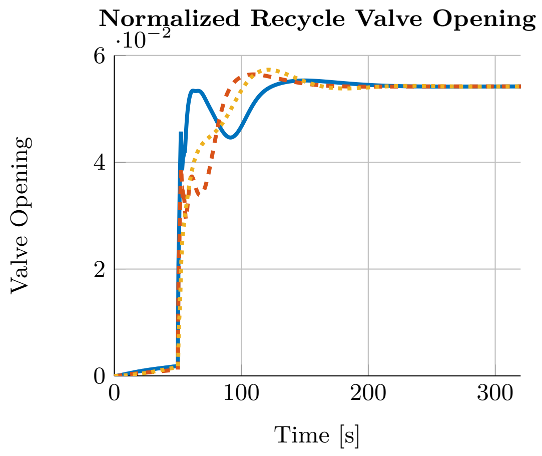
(e)



(f)



(g)



(h)

Figure 5.3: Comparison of time responses of centralized and distributed controllers. Distributed controllers use 3 solver iterations. Disturbance applied is a closing of the downstream compressor's discharge valve from 39% to 29% at time 50 s.

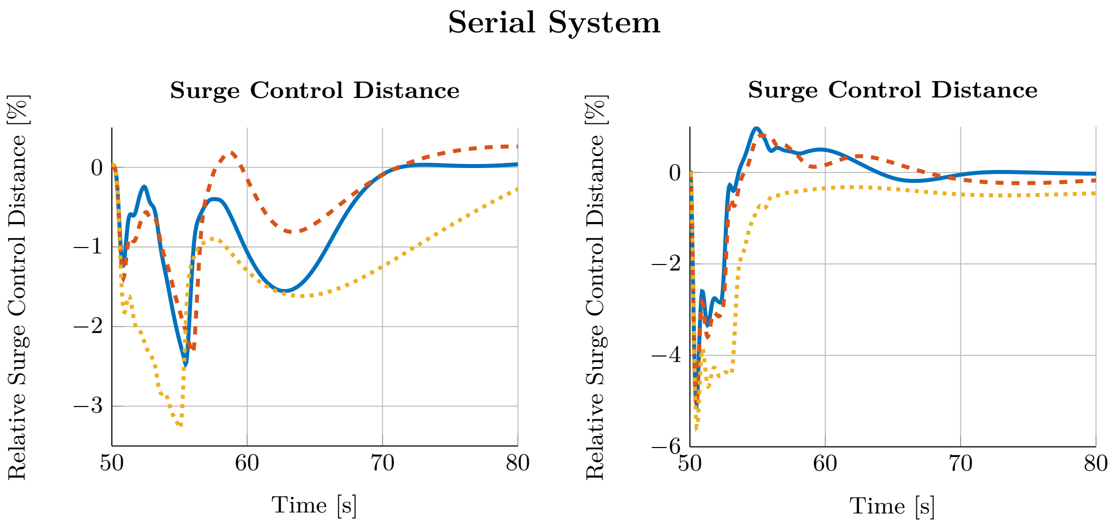


Figure 5.4: Zoomed view of surge distance time response given in Figure 5.3. Disturbance applied is a closing of the downstream compressor's discharge valve from 39% to 29% at time 50 s.

5.2 Computational Efficiency

The computational cost of the centralized and distributed control approaches was evaluated and is presented in Figure 5.5. [1] The computation times for the distributed controllers assume a parallelized implementation where each sub-controller is executed using a separate processor.

As expected, the non-cooperative controller achieves the lowest computation times for both the parallel and the serial system. In both cases, the cooperative controller has a computation time approximately on par with the centralized controller, though slightly lower for few solver iterations. The computational cost of the cooperative controller relative to the centralized controller is dependent on the operating point: for the parallel system, for example, at the non-disturbed, initial state, a single solver iteration of the cooperative controller requires as much computation time as the centralized controller, but the cooperative controller executes faster at the disturbed operating point. The number of solver iterations for which the cooperative approach requires more computation time than the centralized approach is thus shifted depending on the simulation case considered.

The computation time for the cooperative controller in the serial case has a zig-zag behavior, where increasing the number of solver iterations from an odd number to an even number actually decreases the computation time. From Figure 5.6, it is evident

[1] Tests performed using the C++ implementation described in Section 4.8 and run on a Intel® Core™ i5-540M 2.53 GHz processor.

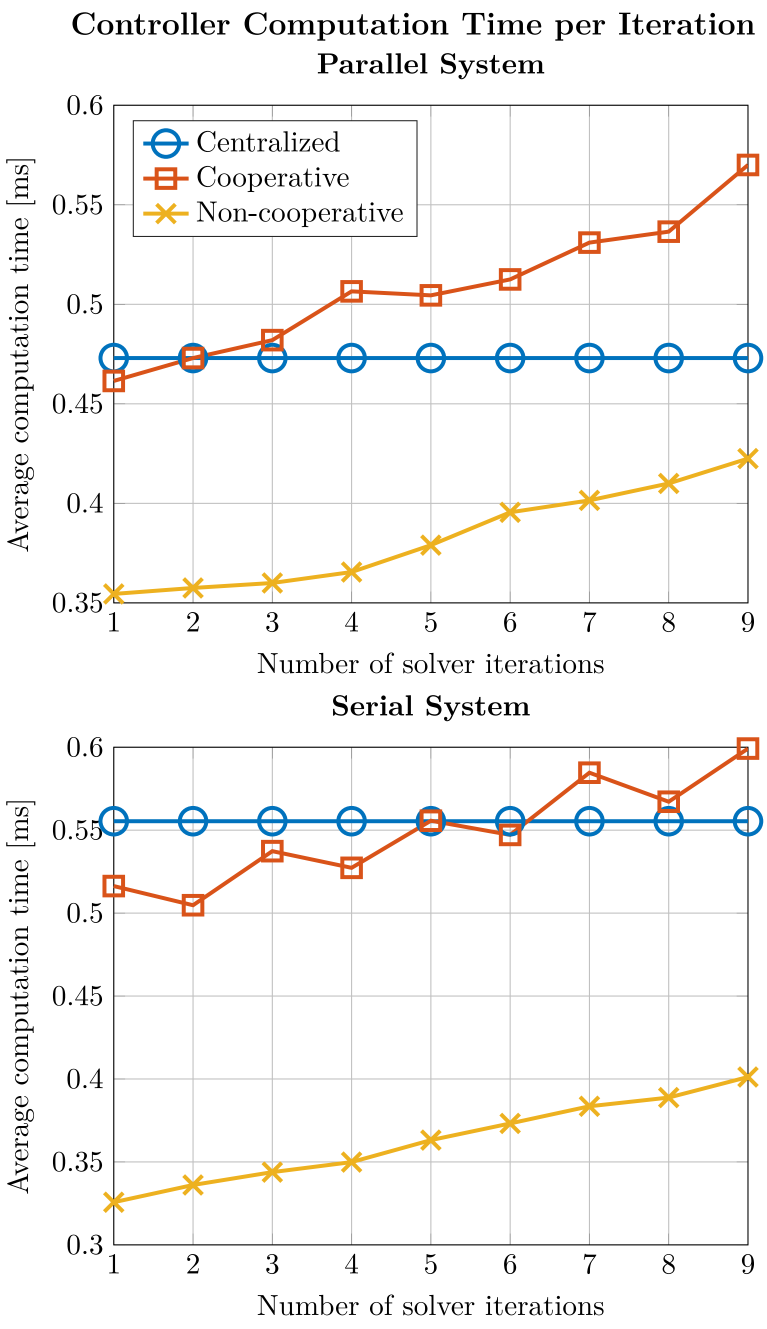


Figure 5.5: Average controller computation time per iteration for centralized and distributed controllers, as a function of the number of solver iterations used.It should be noted that performance gains for solver iterations greater than 2 are marginal. Disturbance (described in Section 5.1) is applied at 50 s and total simulation time is 500 s.

that the pattern originates in QP solver computation time, however the reason for this is unclear.

It should be noted that increasing the number of solver iterations above 3 had no measurable effect on performance (see Section 4.5.2).

The significant advantage in computation time demonstrated by the non-cooperative controller is a result of the reduced size of its prediction matrices (S_{U_k}, S_{x_k} and S_{f_k}), which are multiplied as described in Section 4.2 to generate the QP. The computational cost of the QP generation is approximately linear in the number of outputs used, and the non-cooperative approach considers fewer outputs than the centralized or cooperative approaches (see Table 5.1).

For the systems considered here, the QP-generation step has a much higher computational cost than the QP-solving step. The percentage of computation time spent on solving QPs (including the time required to update the linear QP term after each solver iteration for the distributed controllers) is shown in Figure 5.6. The remaining time is largely dedicated to QP generation. These results are implementation dependent and the computation time required for QP generation could be reduced by using a more efficient linear algebra library that that provided in `Eigen`, however it would still contribute significantly to the total computational cost.

Using 2 or 3 solver iterations, less than 10% of the total computation time is spent solving QPs for all control approaches. There is thus a limited scope for decreasing the required computation time without decreasing either the number of outputs or states used to generate the QP problem. Furthermore, the number of outputs used for the cooperative QP is fixed and cannot be decreased, since the sub-controllers have the same cost function, and for the non-cooperative controller only 2 outputs are used so it cannot be further decreased. Further gains in computational efficiency would therefore require a reduced number of states to be considered when generating the QP problem.

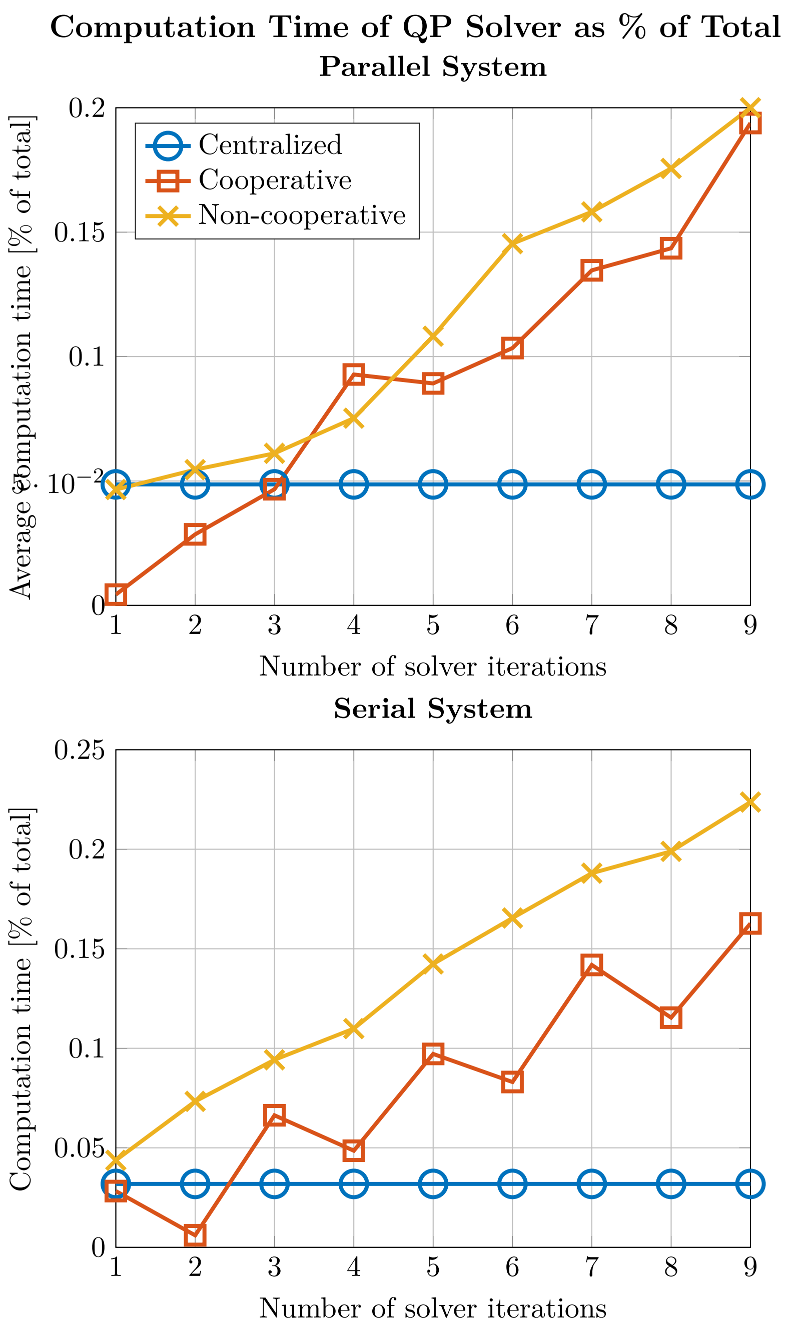


Figure 5.6: Computation time required for QP solver as % of total controller computation time. Includes time required to update linear term of QP at each solver iteration, as described in Section 4.5. Remaining time is used for generating the initial QP problem. Disturbance (described in Section 5.1) is applied at 50 s and total simulation time is 500 s.

6

Conclusion

In this work, the development of a distributed model predictive control scheme combining process and anti-surge control for systems of compressors was presented. Two compressor networks were studied: a parallel and a serial configuration, each with two compressors. The control scheme was based on linearized MPC using a linear model of the system updated at each time step. Distributed controllers using both cooperative and non-cooperative cost functions were developed.

The controllers were implemented in `Simulink` and in C++ using the QP solver `qpOASES`. Both implementations were developed to be suitable for deployment on embedded hardware. In the C++ implementation in particular, the known structure of the compressor model matrices was taken into account to reduce computation time.

The performance of the distributed controllers was then evaluated in simulation and compared to the benchmark established by a centralized MPC controller. Both the time responses of the distributed controllers as well as their computation time were evaluated. For the parallel system, both cooperative and non-cooperative controllers achieved virtually identical performance to the centralized controller. The cooperative controller had a 2% higher average computation time than in the centralized case, while the non-cooperative scheme reduced the average computation time by 25%.

The serial system had different controller performances for each of the centralized, cooperative and non-cooperative controllers. The cooperative controller achieved very similar performance in the time response as the centralized controller, with $< 1\%$ difference in both the maximum discharge pressure and minimum surge control distance in the downstream compressor. The non-cooperative controller also had a qualitatively similar response, however it did not perform as well in anti-surge control, allowing a 9% decrease in the minimum surge control distance reached in the downstream compressor compared to the centralized controller. Its reduced performance in surge control was combined with improved process control: its maximum discharge pressure was 40% lower than in the centralized case. Again, the non-cooperative controller outperformed the cooperative controller in terms of computation time, giving a 38% decrease compared to the centralized case, while the cooperative controller only achieved a 3% reduction.

The computational efficiency of the distributed controllers could be further improved by investigating the effect of reducing the model order at the sub-controller level on overall controller performance. In particular, reducing the number of states used to generate the prediction matrices could lead to relatively high performance gains as the prediction matrix generation is $\mathcal{O}(n^3)$ in the number of states. On the implementation side, the computational cost could be reduced by replacing the `Eigen` linear algebra library with a `BLAS`-based library, which is the benchmark for computational efficiency when performing basic linear algebra computations.

A

Generation of Prediction Matrices

The prediction matrices S_{U_k}, S_{x_k}, S_{f_k} are generated based on the linearized model of the compressor system, and the dynamics given by (4.3). They should satisfy:

$$\Delta Y_k = S_{U_k}\Delta U_k + S_{x_k}\Delta\hat{\boldsymbol{x}}^a_k + S_{f_k}\boldsymbol{f}_{\mathrm{d},k}, \tag{A.1}$$

where Y_k is the output vector, stacked along the prediction horizon, U_k contains the input vector, stacked along the move horizon, $\Delta\hat{\boldsymbol{x}}^a_k$ is the augmented state of the system and $\boldsymbol{f}_{\mathrm{d},k}$ is the discretized derivative of the system, evaluated at the linearization point. $\Delta U_{\min}$ and $\Delta U_{\max}$ contain the input constraints (both absolute bounds and rate constraints).

Defining:

$$\Delta Y_k = \begin{bmatrix} \Delta\boldsymbol{y}_{k+1} \\ \Delta\boldsymbol{y}_{k+2} \\ \vdots \\ \Delta\boldsymbol{y}_{k+p} \end{bmatrix} \qquad \Delta U_k = \begin{bmatrix} \Delta\boldsymbol{u}_{k} \\ \Delta\boldsymbol{u}_{k+1} \\ \vdots \\ \Delta\boldsymbol{u}_{k+m-1} \end{bmatrix}$$

and substituting (4.3) into (A.1), we obtain:

$$\begin{aligned} \Delta\boldsymbol{y}_{k+1} &= C^a_k\Delta\hat{\boldsymbol{x}}^a_{k+1} \\ &= C^a_k\left[A^a_k\left(\Delta\hat{\boldsymbol{x}}^a_k - \begin{bmatrix} 0 \\ \begin{bmatrix} u_{\mathrm{r},k-1} \\ 0 \end{bmatrix} \\ 0 \end{bmatrix}\right) + B^a_k\Delta\boldsymbol{u}_k + \boldsymbol{f}_{\mathrm{d},k}\right] \end{aligned}$$

$$\begin{aligned}
\Delta \boldsymbol{y}_{k+2} &= C_k^a \left[A_k^a \left(\Delta \hat{\boldsymbol{x}}_{k+1}^a - \begin{bmatrix} 0 \\ \begin{bmatrix} u_{\mathrm{r},k-1} \\ 0 \end{bmatrix} \\ 0 \end{bmatrix} \right) + B_k^a \Delta \boldsymbol{u}_{k+1} + \boldsymbol{f}_{\mathrm{d},k} \right] \\
&= C_k^a \left(A_k^a\right)^2 \left(\Delta \hat{\boldsymbol{x}}_k^a - \begin{bmatrix} 0 \\ \begin{bmatrix} u_{\mathrm{r},k-1} \\ u_{\mathrm{r},k-1} \\ 0 \end{bmatrix} \\ 0 \end{bmatrix} \right) + C_k^a A_k^a \left(B_k^a \Delta \boldsymbol{u}_k + \boldsymbol{f}_{\mathrm{d},k} \right) \\
&\quad + C_k^a \left(B_k^a \Delta \boldsymbol{u}_{k+1} + \boldsymbol{f}_{\mathrm{d},k} \right)
\end{aligned} \tag{A.2}$$

$$\begin{aligned}
\Delta \boldsymbol{y}_{k+i} = C_k^a \left(A_k^a\right)^i \left(\Delta \hat{\boldsymbol{x}}_k^a - \underbrace{\begin{bmatrix} 0 \\ \begin{bmatrix} u_{\mathrm{r},k-1} \\ \vdots \end{bmatrix} \\ 0 \end{bmatrix}}_{\boldsymbol{u}_{\mathrm{r},k-1}^{\mathrm{a}}} \right) + C_k^a \left(A_k^a\right)^{i-1} \left(B_k^a \Delta \boldsymbol{u}_k + \boldsymbol{f}_{\mathrm{d},k} \right) + \cdots \\
\cdots + C_k^a A_k^a \left(B_k^a \Delta \boldsymbol{u}_{k+i-2} + \boldsymbol{f}_{\mathrm{d},k} \right) + C_k^a \left(B_k^a \Delta \boldsymbol{u}_{k+i-1} + \boldsymbol{f}_{\mathrm{d},k} \right)
\end{aligned}$$

where $\Delta \boldsymbol{u}_k$ is used in place of $\Delta \boldsymbol{u}_k^{\mathrm{a}}$ because the recycle valve entry in the input vector must be offset by $u_{\mathrm{r},k-1}$, giving the definition of $\Delta \boldsymbol{u}_k$.

Writing (A.2) in matrix form gives:

$$\begin{aligned}
\Delta Y_k = &\begin{bmatrix} C_k^a A_k^a \\ C_k^a \left(A_k^a\right)^2 \\ \vdots \\ C_k^a \left(A_k^a\right)^p \end{bmatrix} \Delta \hat{\boldsymbol{x}}_k^a + \begin{bmatrix} C_k^a & & & \\ C_k^a + C_k^a A_k^a & & & \\ \vdots & & & \\ C_k^a + C_k^a A_k^a + & \cdots & + C_k^a \left(A_k^a\right)^{p-1} \end{bmatrix} \boldsymbol{f}_{\mathrm{d},k} \\
&+ \begin{bmatrix} C_k^a B_k^a & \cdots & & 0 \\ C_k^a B_k^a & C_k^a A_k^a B_k^a & & \vdots \\ \vdots & \vdots & \ddots & \\ C_k^a B_k^a & C_k^a A_k^a B_k^a & \cdots & C_k^a \left(A_k^a\right)^{p-1} B_k^a \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{u}_k \\ \Delta \boldsymbol{u}_{k+1} \\ \vdots \\ \Delta \boldsymbol{u}_{k+p-1} \end{bmatrix}
\end{aligned} \tag{A.3}$$

and considering that the move horizon (m) is smaller than the prediction horizon (p), we have:

$$\Delta \boldsymbol{u}_{k+i} = \Delta \boldsymbol{u}_{k+m-1}, \qquad \text{for } i \geq m \tag{A.4}$$

simplifying (A.3) to:

$$\Delta Y_k = \underbrace{\begin{bmatrix} C_k^a A_k^a \\ C_k^a \left(A_k^a\right)^2 \\ \vdots \\ C_k^a \left(A_k^a\right)^p \end{bmatrix}}_{S_{x_k}} \Delta \hat{\boldsymbol{x}}_k^a + \underbrace{\begin{bmatrix} C_k^a & & & \\ C_k^a + C_k^a A_k^a & & & \\ \vdots & & & \\ C_k^a + C_k^a A_k^a + & \cdots & + C_k^a \left(A_k^a\right)^{p-1} \end{bmatrix}}_{S_{f_k}} \boldsymbol{f}_{\mathrm{d},k}$$
$$+ \underbrace{\begin{bmatrix} C_k^a B_k^a & & & & & & & \\ C_k^a B_k^a & C_k^a A_k^a B_k^a & & & & & & \\ & & \ddots & & & & 0 & \\ & \vdots & & C_k^a \left(A_k^a\right)^{m-1} B_k^a & & & & \\ & & & C_k^a \left(A_k^a\right)^{m-1} B_k^a & + & C_k^a \left(A_k^a\right)^m B_k^a & & \\ & & & & & & \ddots & \\ C_k^a B_k^a & C_k^a A_k^a B_k^a & \cdots & C_k^a \left(A_k^a\right)^{m-1} B_k^a & + & C_k^a \left(A_k^a\right)^m B_k^a + & \cdots & + C_k^a \left(A_k^a\right)^{p-1} B_k^a \end{bmatrix}}_{S_{U_k}} \Delta U_k \tag{A.5}$$

Bibliography

[1] A. Cortinovis, H. Ferreau, D. Lewandowski, and M. Mercangöz, “Experimental evaluation of MPC-based anti-surge and process control for electric driven centrifugal gas compressors,” *Journal of Process Control*, vol. 34, pp. 13–25, 2015.

[2] B. Batson, “Invariant Coordinate Systems for Compressor Control,” in *International Gas Turbine and Aeroengine Congress & Exhibition*, 1996.

[3] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia, “Cooperative distributed model predictive control,” *Systems and Control Letters*, vol. 59, no. 8, pp. 460–469, 2010.

[4] E. M. Greitzer, “Surge and rotating stall in axial flow compressors—Part I: Theoretical compression system model,” *Journal of Engineering for Power*, vol. 98, no. 2, pp. 190–198, 1976.

[5] E. M. Greitzer and F. K. Moore, “A Theory of Post-Stall Transients in Axial Compression Systems : Part II — Application,” tech. rep., NASA, 1985.

[6] K. E. Hansen, P. Jørgensen, and P. S. Larsen, “Experimental and Theoretical Study of Surge in a Small Centrifugal Compressor,” *Journal of Fluids Engineering*, vol. 103, pp. 391–395, sep 1981.

[7] J. T. Gravdahl and O. Egeland, “Compressor Surge and Stall: An Introduction,” in *Compressor Surge and Rotating Stall*, pp. 1–62, Springer London, 1999.

[8] P. Aaslid, *Modelling of variable speed centrifugal compressors for anti-surge control.* Master’s thesis, Norwegian University of Science and Technology, 2009.

[9] J. Galindo, J. R. Serrano, H. Climent, and A. Tiseira, “Experiments and modelling of surge in small centrifugal compressor for automotive engines,” *Experimental Thermal and Fluid Science*, vol. 32, no. 3, pp. 818–826, 2008.

[10] S. Budinis and N. Thornhill, "Control of centrifugal compressors via model predictive control for enhanced oil recovery applications," in *IFAC Workshop on Automatic Control in Offshore Oild and Gas Production*, vol. 48, (Florianopolis, Brazil), pp. 9–14, Elsevier Ltd., 2015.

[11] T. Bentaleb, A. Cacitti, S. De Franciscis, and A. Garulli, "Multivariable control for regulating high pressure centrifugal compressor with variable speed and IGV," in *2014 IEEE Conference on Control Applications, CCA 2014*, (Antibes, France), pp. 486–491, 2014.

[12] T. Bentaleb, *Model-Based Control Techniques for Centrifugal Compressors.* Ph.d thesis, University of Siena, 2015.

[13] J. Smeulers, W. Bournan, and H. van Essen, "Model predictive control of compressor installations," in *International conference on compressors and their systems*, (London, UK), 1999.

[14] T. F. Øvervåg, *Centrifugal Compressor Load Sharing with the use of MPC.* PhD thesis, Norwegian University of Science and Technology, 2013.

[15] R. Scattolini, "Architectures for distributed and hierarchical Model Predictive Control - A review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.

[16] a.N. Venkat, J. Rawlings, and S. Wright, "Stability and optimality of distributed model predictive control," *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 6680–6685, 2005.

[17] P. A. Trodden and A. Richards, "Robust distributed model predictive control," *International Journal of Control*, vol. 80, no. 9, pp. 1517–1531, 2007.

[18] B. T. Stewart, S. J. Wright, and J. B. Rawlings, "Cooperative distributed model predictive control for nonlinear systems," *Journal of Process Control*, vol. 21, no. 5, pp. 698–704, 2011.

[19] 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.

[20] H. Peyrl and A. Cortinovis, "Computationally Efficient Solution of a Compressor Load Sharing Problem Using the Alternating Direction Method of Multipliers," in *2015 European Control Conference*, (Linz, Austria), pp. 2314–2319, 2015.

[21] R. Fausel and A. Verwee, "Challenges & Opportunities in Turbomachinery Control," Compressor Controls Corporation, 2010.

[22] G. Guennebaud, B. Jacob, and Others, "Eigen v3." http://eigen.tuxfamily.org, 2010.

[23] W. Blotenberg, H.-O. Jeske, and H. Voss, "Design, control and startup features of three parallel-working propane compressors each having three stage groups," 1984.

[24] B. Bøhagen and J. T. Gravdahl, "Active surge control of compression system using drive torque," *Automatica*, vol. 44, no. 4, pp. 1135–1140, 2008.

[25] K. O. Boinov, E. A. Lomonova, A. J. A. Vandenput, and A. Tyagunov, "Surge control of the electrically driven centrifugal compressor," *IEEE Transactions on Industry Applications*, vol. 42, no. 6, pp. 1523–1531, 2006.

[26] A. Cortinovis, H. J. Ferreau, D. Lewandowski, and M. Mercang, "Safe and Efficient Operation of Centrifugal Compressors using linearized MPC," in *IEEE Conference on Decision and Control*, (Los Angeles, USA), pp. 3982–3987, 2014.

[27] A. Cortinovis, M. Zovadelli, M. Mercangoz, D. Pareschi, A. De Marco, and S. Bittanti, "Online adaptation of performance maps for centrifugal gas compressors," *2014 European Control Conference, ECC 2014*, pp. 1036–1041, 2014.

[28] M. L. Darby and M. Nikolaou, "MPC: Current practice and challenges," *Control Engineering Practice*, vol. 20, no. 4, pp. 328–342, 2012.

[29] H. Ghorbani, "Constrained Model Predictive Control Implementation for a Heavy-Duty Gas Turbine Power Plant ine System m Descript tion," *WSEAS Transactions on Systems and Control*, vol. 3, no. 6, pp. 507–516, 2008.

[30] J. T. Gravdahl, *Modeling and control of surge and rotating stall in compressors.* Phd thesis, Norwegian University of Science and Technology, 1998.

[31] J. T. Gravdahl and O. Egeland, "Passivity based compressor surge control using a close-coupled valve," in *COSY Workshop on Control of Nonlinear and Uncertain Systems*, (Zürich, Switzerland), 1997.

[32] J. T. Gravdahl, O. Egeland, and S. O. Vatland, "Drive torque actuation in active surge control of centrifugal compressors," *Automatica*, vol. 38, no. 11, pp. 1881–1893, 2002.

[33] J. Javadi Moghaddam, M. H. Farahani, and N. Amanifard, "A neural network-based sliding-mode control for rotating stall and surge in axial compressors," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 1036–1043, 2011.

[34] C. Li, S. Xu, and Z. Hu, "Experimental Study of Surge and Rotating Stall Occurring in High-speed Multistage Axial Compressor," *Procedia Engineering*, vol. 99, pp. 1548–1560, 2015.

[35] S. Lin, C. Yang, P. Wu, and Z. Song, "Active surge control for variable speed axial compressors," *ISA Transactions*, vol. 53, no. 5, pp. 1389–1395, 2014.

[36] S. Mirsky, W. Jacobson, D. Tiscornia, J. McWhirter, and M. Zaghloul, "Development and Design of Antisurge and Performance Control Systems," in *42nd Turbomachinery Symposium*, (Houston, Texas), 2013.

[37] P. Z. Molenaar, *Model Predictive Compressor Surge Control.* Master's thesis, Technische Universiteit Eindhoven, 2007.

[38] K. R. Muske and T. A. Badgwell, "Disturbance modeling for offset-free linear model predictive control," *Journal of Process Control*, vol. 12, no. 5, pp. 617–632, 2002.

[39] F. Paparella, L. Domínguez, A. Cortinovis, M. Mercangoz, D. Pareschi, and S. Bittanti, "Load sharing optimization of parallel compressors," in *2013 European Control Conference*, (Zürich, Switzerland), pp. 4059–4064, 2013.

[40] G. Quartarone, N. Anglani, and S. Riverso, "Model Predictive Control: first application of a novel control strategy for adjustable speed drive compressors," *Proceedings of the 39th Annual Conference of the IEEE Industrial Electronics Society*, pp. 7893–7897, 2013.

[41] R. S. Shehata, H. A. Abdullah, and F. F. G. Areed, "Variable structure surge control for constant speed centrifugal compressors," *Control Engineering Practice*, vol. 17, no. 7, pp. 815–833, 2009.

[42] D. P. Xenos, M. Cicciotti, G. M. Kopanos, A. E. F. Bouaswaig, O. Kahrs, R. Martinez-Botas, and N. F. Thornhill, "Optimization of a network of compressors in parallel: Real Time Optimization (RTO) of compressors in chemical plants - An industrial case study," *Applied Energy*, vol. 144, pp. 51–63, 2015.

[43] J. Zhou, L. Fiorentini, M. Canova, and Y.-Y. Wang, "Coordinated Performance Optimization of a Variable Geometry Compressor With Model Predictive Control for a Turbocharged Diesel Engine," *IEEE Transactions on Control Systems Technology*, pp. 1–1, 2015.

[44] E. Camponogara, D. Jia, B. Krogh, and S. Talukdar, "Distributed model predictive control," *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 44–52, 2002.

[45] A. Alessio, D. Barcelli, and A. Bemporad, "Decentralized model predictive control of dynamically coupled linear systems," *Journal of Process Control*, vol. 21, no. 5, pp. 705–714, 2011.

[46] F. Borrelli, A. Bemporad, and M. Morari, "Predictive control for linear and hybrid systems," *Preparation, Available Online At Http://Www. Mpc. . . .*, 2015.