Henrik Manum

Simple implementation of optimal control for process systems

Doctoral thesis
for the degree of philosophiae doctor

Trondheim, November 2010

Norwegian University of Science and Technology
Faculty of Natural Sciences and Technology
Department of Chemical Engineering

NTNU

Norwegian University of Science and Technology

Doctoral thesis
for the degree of philosophiae doctor

Faculty of Natural Sciences and Technology
Department of Chemical Engineering

ISBN 978-82-471-2420-8 (printed version)
ISBN 978-82-471-2421-5 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2010:217

Printed by NTNU-trykk

Abstract

The main focus of this thesis is to find simple ways of *implementing* optimal operation of process plants. The work is in the spirit of "self-optimizing control", which can be summarized as [Skogestad, 2000b]:

> "The goal is to find a self-optimizing control structure where acceptable operation under all conditions is achieved with constant setpoints for the controlled variables. More generally, the idea is to use the model off-line to find properties of the optimal solution suited for (simple, model-free) on-line implementation."

In the first part of the thesis, the problem of static output feedback is addressed. This is one of the open problems in control [Syrmos et al., 1997], and we derive a novel approximation to this problem by using links to self-optimizing control. The approximation can be used to calculate multiple input – multiple output proportional-integral-derivative (MIMO-PID) controllers, which can be of great practical interest.

We further extend parts of the theory of self-optimizing control to cover changes in the active set. This is done by using results from explicit model predictive control (MPC) and the results are exact for a quadratic approximation around the optimum. By using an ammonia production plant as an example, we show that the results may also be applied to more general processes, and that the method is particularly interesting for cases where the set of active constraints is expected to change frequently.

Thereafter we develop a mathematical framework for analysis of the performance loss when "speedups" are applied to an MPC formulation. Such speedups can be model reduction, move blocking, shortening the horizon in the controller or changing the sample time of the internal model in the MPC. By using the method on a model of a distillation column, we find that the so-called "delta-move blocking" has a good performance to speed ratio.

We then use the same mathematical program to prove stability of simple control schemes by calculating the maximum distance to a robust controller; if the

distance is within the robustness margin of the robust controller, then the simple controller is proven to be stable. Several “simple controllers” can be analyzed in this scheme, for example partial enumeration of an explicit MPC and the linear quadratic regulator with saturation.

Finally, in the appendices of the thesis, we give mathematical links between the problem of self-optimizing control and explicit MPC, and we give some means of simplification of the implementation of explicit MPC. In addition we give some extra information regrading the static output feedback problem.

Acknowledgements

First of all I want to thank Professor Sigurd Skogestad, who accepted me as his student back in 2006, and thereafter spent a great amount of work on me to get me to understand the world of research. His determination to not let a single detail slip away is truly impressive, and probably the most important factor for the fulfillment of this thesis. Sigurd, thank you for your endless patience, encouragement, and faith in me and this thesis.

I am very thankful for all the journeys I was able to make during the last four years, which has enabled me to see both the east and the west of the world.

In particular, I am really thankful for the 6 months I got to stay at IfA (Insitut für Automatik) at ETH, Switzerland. Looking back, it is hard to understand how I could have finished the thesis without that stay. The group is both professionally and socially excellent, a big thanks to all of you for welcoming me as a guest. A special thanks goes to Colin N. Jones, who helped me to "rediscover" the fun of optimization, and how to relate optimization to control. In addition I would like to thank Daniel Axehill for all the fun we had both at the office and in the Alps, your presence really made the stay a lot better!

I further want to thank the Jim Rawlings Research Group for their warm welcome when I spent 10 days there before going to the American Control Conference in June 2008. In retrospect I wish I was more open for doing collaboration with Professor Rawlings. Now, two years later, you should know that your papers and books have been a big help for getting a chemical engineer to understand the principles of model predictive control.

I am also very thankful for the great welcome I got as a guest in Bratislava during February 2010. The software developed by Michal Kvasnica was what got me going when I first started the work of the thesis, now I can also say that I know the author as a great host, together with Professor Fikar and the rest of the group. Thanks!

The biggest thanks of all (by far!) goes to the love of my life, who I met three years ago. Right now you are working in what is to be our second home together, while I am putting down the final words of the thesis. You have always given me

all you possibly can, including living alone in our cold apartment for six months while I was in Switzerland (just after moving in together!). To you I dedicate this thesis.

Finally I give big thanks to my family who have always been there for me (many of which are also working in our new apartment at the moment), and to the family of Valentina, in which I feel like a natural part now. A very big thanks to all of you.

Trondheim, October 23, 2010

Henrik Manum

Contents

Chapter 1

Introduction

1.1 Motivation

The main focus of this thesis is to find simple implementations of solutions to optimal control of process systems. In order words, the focus if *not* on *how to solve an optimal control problem*, but on *how to implement the solution to an optimal control problem*. One can identify two different paradigms for implementation of the solution to an optimal control problem [Narasimhan and Skogestad, 2007]:

Paradigm 1. On-line optimizing control where measurements are primarily used to update the model. With arrival of new measurements, the optimization problem is resolved for the inputs.

Paradigm 2. Pre-computed solutions based on off-line optimization. Typically, the measurements are used to (indirectly) update the inputs using feedback control schemes.

Open-loop optimization is typically used in Paradigm 1, while closed-loop solution is the preferred alternative for Paradigm 2.

An open loop approach to Paradigm 1 may however be quite conservative, as is addressed by many authors in the context of robust model predictive control (MPC), see e.g. the survey paper by Mayne et al. [2000]. The approach is conservative because the open-loop optimization problems needs to take into account disturbances that in closed loop will be attenuated by the controller, but not in the open loop predictions of the optimization problem, see e.g. [Bemporad, 1998, Bemporad and Morari, 1999]. One would therefore like to use closed-loop optimization also in Paradigm 1, but in practice it is usually difficult to take feedback into account in on-line optimization, so in most cases one would be content with an open-loop formulation.

In this thesis we are mostly interested in Paradigm 2, however combinations of the two are also interesting.

Let us consider the following example to better understand the differences between the paradigms:

Example: Linear quadratic control. Consider the quadratic optimization problem

$$J^*(x_0) = \min_{u,x} \; x_N' Q_N x_N + \sum_{i=0}^{N-1} x_i' Q x_i + u_i' R u_i \tag{1.1}$$

$$\text{s.t. } x_{k+1} = Ax_k + Bu_k, \quad k = 0, 1, \cdots, N-1 \tag{1.2}$$

where $u = (u_0, u_1, \cdots, u_{N-1})$ is a sequence of inputs and $x = (x_1, x_2, \cdots, x_N)$ is a sequence of states. We assume that Q_N, Q, R are symmetric matrices and that $Q_N > 0$, $Q \geq 0$, $R > 0$. Further, $x_{k+1} = Ax_k + Bu_k$ is a linear model of some plant we want to control. There exists many solvers [Wang and Boyd, 2010] that can be implemented on-line such that when a new measurement of x_0 occurs, problem (1.1)-(1.2) is resolved for the inputs u. The idea of receding horizon control [Richalet et al., 1978, Cutler and Ramaker, 1980] is to implement only the first part of the sequence u, and then resolve the optimization problem when new measurements of the state x_k becomes available. This solution, where *open-loop optimization* is used in closed loop, belongs to Paradigm 1.

However, by using dynamic programming, one can find a *feedback policy* $u_k = K_0 x_k$ that is optimal for *any* state x_k, in order words, the feedback gain K_0 does not depend on the current state. For the quadratic program (1.1)-(1.2) one can define the backward Riccati iteration [Rawlings and Mayne, 2009]

$$\Pi_{k-1} = Q + A'\Pi_k A - A'\Pi_k B(B'\Pi_k B + R)^{-1} B'\Pi_k A, \quad k = N, N-1, \ldots, 1 \tag{1.3}$$

with terminal condition

$$\Pi_N = Q_N. \tag{1.4}$$

The optimal gain at time k is computed from the Riccati matrix at time $k+1$:

$$K_k = -(B'\Pi_{k+1} B + R)^{-1} B'\Pi_{k+1} A, \quad k = N-1, N-2, \ldots, 0. \tag{1.5}$$

All these calculations can be performed *off-line* and implementation, by using receding horizon control, is simply

$$u_k = K_0 x_k. \tag{1.6}$$

Implementation of this *state feedback* does not depend on any online optimization and belongs to Paradigm 2.

Once the state-feedback $u_k = K_0 x_k$ is found one may also use this as a re-parameterization of the open loop problem by introducing [Rossiter et al., 1997]:

$$u_k = K_0 x_k + w_k \tag{1.7}$$

and rewriting the plant model to

$$\begin{aligned} x_{k+1} &= \underbrace{(A + BK_0)}_{A_c} x_k + Bw_k \\ &= A_c x_k + Bw_k. \end{aligned} \tag{1.8}$$

An open loop optimization in the re-parameterized inputs w_k should now converge to a solution close to $w = 0$ for any disturbance, hence finding initial values and consequently solving the re-parameterized problem should be easier than the original formulation (1.1)-(1.2) [Mayne et al., 2005].

In the last decade, parts of the control community have investigated *explicit solutions* to model predictive control, see e.g. the recent survey by [Alessio and Bemporad, 2008]. In order to compute an explicit MPC one usually first formulates an *open-loop* problem, for example on the form of problem (1.1)-(1.2), but with constraints added on inputs and states. Then one solves this problem parametrically [Bemporad et al., 2006] to get a piece-wise affine feedback law on the form

$$u_k := K^i x_k + c^i \quad \text{if } x_k \in P_i, \tag{1.9}$$

where P_i represent a polytopic division of the state space and K^i, b_i are corresponding state feedback laws.

Implementation of explicit and on-line MPC will yield the same value of the cost function, as their solutions are equivalent. However, the explicit solution can itself be very complicated (with many polytopes P_i in (1.9)), and researches have therefore started to look for simplifications of the explicit control law, see for example the double description method by Jones and Morari [2008]. In this thesis Chapter 7 is devoted to analysis for such simplifications.

When one does a simplification a non-negative loss is introduced [Skogestad, 2000b]:

$$L(u,d) = J(u,d) - J_{\text{opt}}(d), \tag{1.10}$$

where $J(u,d)$ is the cost of a particular policy and $J_{\text{opt}}(d)$ is the optimal cost for a given disturbance d. The idea of self-optimizing control, which has been a major source of inspiration for this thesis, is to [Skogestad, 2004]:

> "Find a self-optimizing control structure where acceptable operation under all conditions is achieved with constant setpoints for the controlled variables. More generally, the idea is to to use the model

> off-line to find properties of the optimal solution suited for (simple, model-free) on-line implementation"

Here "acceptable operation" can imply that the loss $L(u,d)$ is below some acceptable value. The idea of self-optimizing control can be applied to control of complex systems, as described in the following example, which is borrowed from [Skogestad and Postlethwaite, 2005]:

Example: Long-distance running. Consider the optimal operation of a long-distance runner. The manipulated input is the muscle power. The objective is to minimize the time of the race. Using paradigm 1, on-line optimization, is clearly very difficult, because we would need to identify a model of the many complex mechanisms that occur in the human body and re-optimize this model online when disturbances occur. Probably just getting a model that is suitable for on-line optimization would take a very long time. We therefore consider the approach of Paradigm 2, to look for some feedback strategy that will give acceptable operation in closed loop. Moreover, let us search for some *constant setpoint policy* that we can use in closed loop. The question in thus: Can we find some variable c that when controlled to a constant setpoint c_s yields acceptable operation?

It is clear that running at maximum power is not a good strategy. This would give a high speed at the beginning, but a slower speed towards the end with an overall low average speed. A better policy would be to keep constant speed. The trainer will then choose an optimal setpoint for the speed, and this is implemented by the runner. Alternative strategies, which may work better in a hilly terrain, are to keep a constant heart rate or a constant lactate level.

Chapter 5 is devoted to self-optimizing control, where we extend some results to cover changes in the active set. However in a broader sense, most of the thesis is based on the general idea of finding *simple solutions* that gives *acceptable performance in closed loop.*

1.2 Outline of thesis

In **Chapters 2-4**, we give a convex *approximation* to the static output feedback problem, which is one of the open problems in control [Syrmos et al., 1997]. We do this by exploiting a link to self-optimizing control, where we minimize the *loss from an optimal controller*. Since we only find an approximation, the problem still remains open, but we show that our approximation may be used to synthesize multi input − multi output proportional-integral-derivative (MIMO-PID) controllers, which can be of great practical interest.

In **Chapter 5**, we extend the ideas of self-optimizing control to handle changes in the active set, by using results from explicit MPC.

In **Chapter 6**, we formulate a mathematical program that can analyse different speedups in MPC, such as move blocking. In **Chapter 7**, we use a similar program to analyse different low-complexity controllers.

In **Appendices A-B**, we give some ideas on simplification of explicit MPC and extensions to output feedback.

In **Appendix C**, we give some more details on the static output feedback problem.

1.3 Publications

During the work on my thesis I was the main author the publications given in Table 1.1. In addition I was a co-author on the papers listed in Table 1.2. I was also working on the papers listed in Table 1.3, which were related to my Master's thesis, but not to the work reported in this thesis.

Publication	Chapter
H. Manum and S. Skogstad: **Bilevel programming for Analysis of Reduced Models for use in Model Predictive Control**, *Journal of Cybernetics and Informatics*, 2010 (9), pp. 3-12 (Also presented at "Cybernetics and Informatics Conference", Vyšná Boca, Slovak Republic, February 2010)	
H. Manum, C.N. Jones, J. Löfberg, M. Morari, and S. Skogestad: **Bilevel programming for analysis of low-complexity control of linear systems with constraints**, *Proc. Conference on Decision and Control*, Shanghai, China, December 2009, pp. 946-951	Chapter 7
H. Manum, S. Skogestad and J. Jäschke: **Convex initialization of the $\mathcal{H}_2$-optimal static output feedback problem**, *Proc. American Control Conference*, St. Louis, USA, June 2009, pp. 1724-1729	Appendix C
H. Manum and S. Skogestad: **Design of multivariable LQ-optimal PID controllers based on convex optimization**, 15th Nordic Process Control Workshop, Porsgrunn, Norway, January 2009	
H. Manum, S. Narasimhan, and S. Skogestad: **Explicit MPC with output feedback using self-optimizing control**, *Proc. IFAC World Congress*, Seoul, Korea, 2008, pp. 6956-6961	Appendix B
H. Manum, S. Narasimhan, and S. Skogestad: **A new approach to explicit MPC using self-optimizing control**, *Proc. American Control Conference*, Seattle, USA, June 2008, pp. 435-440	Appendix A
H. Manum, S. Narasimhan, and S. Skogestad: **A new approach to explicit MPC**, AIChE Annual Meeting, Salt Lake City, USA, November 2007	
H. Manum, S. Narasimhan, and S. Skogestad: **Implementation of optimal operation of quadratic programs using off-line calculations: Application to heat exchanger networks and explicit MPC**, 14th Nordic Process Control Workshop, Helsinki, Finland, August 2007	
H. Manum, S. Narasimhan, and S. Skogestad: **A new approach to explicit MPC using self-optimizing control**, Internal report, NTNU, July 2007	

Table 1.1: Publications as first author.

Publication
R. Yelchuru, S. Skogestad, and H. Manum: **MIQP formulation for Controlled Variable Selection in Self Optimizing Control**, *Proc. DYCOPS*, Leuven, Belgium, July 2010, pp. 61-66
J. Jäschke, H. Smedsrud, S. Skogestad, and H. Manum: **Optimal operation of a Waste Incineration Plant for District Heating**, *Proc. American Control Conference*, St. Louis, USA, June 2009, pp. 665-670

Table 1.2: Co-authored publications.

Publication
H. Manum. C. Ghelardoni, and C. Scali: **Chapter 6: Automatic Diagnosis of Valve Stiction by Means of a Qualitative Shape Analysis Technique**, In book: *Modelling, Control, Simulation and Diagnosis of Complex Industrial and Energy Systems*, ISBN/ID: 978-1-934394-90-8
H. Manum and C. Scali: **Analisi di una tecnica di riconoscimento delle forme per la diagnostica automatica dell'attrito nelle valvole**, *Automazione e strumentazione* 2007 (6), pp. 100-105
H. Manum and C. Scali: **Closed Loop Performance Monitoring: Automatic Diagnosis of Valve Stiction by means of a Technique based on Shape Analysis Formalsm** (8 pages), Symposium ANIPLA Methodologies for Emerging Technologies in Automation, Rome, Italy, November 2006

Table 1.3: Publications not related to the thesis.

Chapter 2

Convex approximation of the static output feedback problem

with applications to design of multivariable PID controllers

In this chapter we derive a convex approximation to the $\mathcal{H}_2$-optimal static output feedback problem and show how the approximation can be used to synthesize multiple input − multiple output proportional-integral-derivative (MIMO-PID) controllers. The approximation is done in two steps. First, instead of minimizing directly the quadratic cost, we find the static output feedback controller that minimizes the "loss" compared to the optimal state feedback controller. Use of this modified cost seems to have a small effect on the resulting controller. Second, we approximate the resulting nonconvex problem by a convex quadratic program (QP). This approximation introduces some sub-optimality, but numerical examples show that the controller is close to the optimal in most cases, except for cases where the controllability using output feedback is far away from the optimal state feedback. Alternatively, we can use it as initial value for a nonlinear search.

2.1 Introduction

Consider the linear process

$$x_{k+1} = Ax_k + Bu_k \tag{2.1}$$

$$y_k = Cx_k + Du_k \tag{2.2}$$

$$x_0 \backsim N(0, W_{x_0}^{1/2}) \tag{2.3}$$

where equation (2.3) means that the initial state x_0 is a normally distributed random variable with zero mean and covariance matrix W_{x_0}. The objective is to find a *static output feedback*

$$u_k = -K^y y_k \tag{2.4}$$

that minimizes the *expected value* $E\{J^\infty\}$, where the cost

$$J^\infty = \lim_{T\to\infty} \left\{ \frac{1}{T} \sum_{i=0}^{T} x_i' Q x_i + u_i' R u_i \right\}. \tag{2.5}$$

Here $Q = Q' = C_z' Q^y C_z$, where $Q^y \geq 0$ is a quadratic weight on the controlled outputs $z_k = C_z x_k$, and $R = R' > 0$ is a weight on input usage. The infinite horizon objective function may be approximated by a finite horizon cost function (see e.g. Chmielewski and Manousiouthakis [1996] and Scokaert and Rawlings [1998]),

$$J = x_N' Q_N x_N + \sum_{i=0}^{N-1} x_i' Q x_i + u_i' R u_i. \tag{2.6}$$

Remark 2.1. *There are several ways of choosing the final state cost matrix Q_N such that the finite horizon problem corresponds as closely as possible to the infinite horizon problem. In this chapter we use the method of first finding the corresponding linear quadratic regulator $u_k = -K_{LQR} x_k$, which can be found by standard software, and further we let $A_K = A - BK_{LQR}$, $Q_f = Q + K_{LQR}' R K_{LQR}$, and let $Q_N > 0$ satisfy the Lyapunov equation*

$$A_K' Q_N A_K + Q_f = Q_N \tag{2.7}$$

With this method, the cost functions of the infinite horizon and finite horizon problems are the same for any value of prediction horizon N if full information state feedback is used, see [Chmielewski and Manousiouthakis, 1996, Scokaert and Rawlings, 1998].

Example: MIMO-PID. A controller of great practical interest that can be cast into a static output feedback problem is the multiple input − multiple output (MIMO) proportional-integral-derivative (PID) controller. To see this, let σ_k denote the integrated output and consider the augmented plant with augmented state dynamics

$$\begin{bmatrix} x_{k+1} \\ \sigma_{k+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ C & I \end{bmatrix} \begin{bmatrix} x_k \\ \sigma_k \end{bmatrix} + \begin{bmatrix} B \\ D \end{bmatrix} u_k, \tag{2.8}$$

and augmented outputs

$$\begin{bmatrix} y_k^P \\ y_k^I \\ y_k^D \end{bmatrix} = \begin{bmatrix} C & 0 \\ 0 & I \\ \frac{1}{T_s} C(A-I) & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \sigma_k \end{bmatrix} + \begin{bmatrix} D \\ 0 \\ B \end{bmatrix} u_k. \tag{2.9}$$

The PID-controller can then be written on the form (2.4) with

$$K^y = \begin{bmatrix} K_P & K_I & K_D \end{bmatrix}. \tag{2.10}$$

In (2.9) we have used the forward difference to approximate the derivative, $\frac{\partial y_k}{\partial t} \approx \frac{y_{k+1}-y_k}{T_s}$, which gives a direct feed through term in the model. We can also use other approximations of the derivative including filters, but the filter constant needs to be set a priori and is not considered a degree of freedom for the optimization. The MIMO-PID is covered in more details in the examples.

Remark 2.2. *Rather than only integrating the outputs as in equation* (2.8) *one could also add integrators on the inputs to get better performance for integrating disturbances at the inputs. Then one would need to estimate these disturbances by for example a Kalman filter, and include the output of the Kalman filter as an additional model output, to be used in the controller design. Issues regarding offset-free tracking and disturbance rejection for MPC are treated in [Muske and Badgwell, 2002, Pannocchia and Rawlings, 2003], where rank-requirements for the matrices in the estimator are given.*

By augmenting the plant as shown above, any fixed structure control problem may be posed as a static output feedback problem. In process control an interesting controller could be *decentralized* PI-control, but the fixed structure problem may also be interesting in other fields, see e.g. Gadewadikar et al. [2009] which studied static output feedback control for rotorcraft and Maithripala et al. [2005] which considered static output feedback of an electrostatic microelectromechinal system.

Difficulty of solving static output feedback. In the literature, it is proved that problems closely related to static output feedback belongs a class of problems that cannot be solved by polynomial time algorithms (NP-hard problems), and it is therefore conjectured that also the SOF problem is an NP-hard problem [Blondel and Tsitsikilis, 1997]. For a survey of different approaches to this problem the reader is referred to Syrmos et al. [1997].

There is still active research in this topic, see for example Bara and Boutayeb [2005], which addresses static output feedback stabilization with $\mathcal{H}_\infty$ performance, Fujimori [2004], which approaches static output feedback with a substitutive LMI formulation, and Maruta et al. [2009] which addresses fixed-structure synthesis using particle swarm optimization. For applications to uncertain systems see e.g. [Shu et al., 2010, Wu, 2008], and for an application to stabilization and control of networks see [Menon and Edwards, 2009]. The subject of static output feedback is also still frequent at conferences, see e.g. the Conference on Decision and Control [Pakshin and Peaucelle, 2009, Nagashio and Kida, 2009, Mukadidani and Xu,

2009, Qiu et al., 2009, Bouarar et al., 2009] and American Control Conference [Haidar et al., 2009, Du and Yang, 2009, Ding and Yang, 2009a,b].

It is interesting to note that a fixed structure feed*forward* control design problem can at least for some cases be posed as a convex problem. This is demonstrated by van der Meulen et al. [2008], who investigated fixed structure controller design by exploiting iterative trials. They demonstrated their method on a wafer stage and a desktop printer.

Contribution of this chapter. Rather than solving for the output feedback controller that directly minimizes the expected value of the cost function (2.5), in this chapter we derive a convex approximation to the problem of minimizing the expected loss $E\{L\}$, where the non-negative loss is given by

$$L(u_k = -K^y y_k, x_0) = J(u_k = -K^y y_k, x_0) - J_{\text{opt}}(x_0), \tag{2.11}$$

where $J(u, x_0)$ is the cost function (2.6) evaluated for a static output feedback controller and $J_{\text{opt}}(x_0)$ is the optimal value of the objective function for any given disturbance x_0, which can be realized with a LQ-optimal full-information controller (i.e. state feedback $u_k = -K_{\text{LQR}} x_k$). Note that minimizing the loss (2.11) may be more reasonable from an engineering point of view, as we avoid penalizing deviations that cannot be handled even with the best controller. In some sense we get "for free" a reference model which is often used as a tuning factor in controller design, see e.g. [Skogestad and Postlethwaite, 2005]. In any case, numerical evidence suggests that the differences are small.

Chapter overview. The rest of this chapter is organized as follows: First, we review background material on the finite horizon linear quadratic regulator, infinite horizon static output feedback, and finite horizon static output feedback. Then we give our main result, which is a convex approximation to the problem of minimizing the expected loss from optimal control. Thereafter two examples are given, and we finally discuss our findings.

2.2 Background material

2.2.1 Finite horizon linear quadratic regulator

Let us consider the finite horizon optimal control problem, also known as the linear quadratic regulator (LQR) problem (see for example [Kalman, 1960] and references therein). This problem can be posed as an *open-loop* problem in the inputs

$u = (u_0, u_1, \cdots, u_{N-1})$ by first writing the linear model $x_{k+1} = Ax_k + Bu_k$ as

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}}_{G^x_{x_0}} x_0 + \underbrace{\begin{bmatrix} B & & & \\ AB & B & & \\ \vdots & & \ddots & \\ A^{N-1}B & \cdots & \cdots & B \end{bmatrix}}_{G^x_u} u \tag{2.12}$$

By further defining $\bar{Q} = \text{diag}(Q, \cdots, Q, Q_N)$, and $\bar{R} = \text{diag}(R, \cdots, R)$, the LQR problem can be written as

$$J_{\text{opt}}(x_0) = \min_u \begin{bmatrix} u \\ x_0 \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ux_0} \\ J'_{ux_0} & J_{x_0x_0} \end{bmatrix} \begin{bmatrix} u \\ x_0 \end{bmatrix} \tag{2.13}$$

with

$$J_{uu} = G^{x\prime}_u \bar{Q} G^x_u + \bar{R}, \tag{2.14}$$

$$J_{ux_0} = G^{x\prime}_u \bar{Q} G^x_{x_0}, \tag{2.15}$$

$$J_{x_0x_0} = G^{x\ \prime}_{x_0} \bar{Q} G^x_{x_0}. \tag{2.16}$$

This is not formulated as a feedback problem, but if we assume a moving horizon problem and only implement the first move (of the input vector u), then by the principle of optimality [Bellman, 1954] an optimal feedback policy results. Actually, without knowledge of dynamic programming, one can deduce that the solution can be written as a feedback policy by solving the first-order optimality conditions, which give $u = -J^{-1}_{uu} J_{ux_0} x_0$, and since a new disturbance x_0, which can be observed by the states, can happen at every sample time, we simply keep implementing the first part of this vector (corresponding to the first input u_0) and we have solved the moving horizon control problem. Note that a Riccatti-iteration can be used to find K_{LQR}, which is the first part of the matrix $J^{-1}_{uu} J_{ud}$, see e.g. [Rawlings and Muske, 1993].

The optimal value of the cost function (denoted "value function"), which we get by inserting the optimal input $u = -J^{-1}_{uu} J_{ux_0} x_0$ into problem (2.13) is

$$J_{\text{opt}}(x_0) = -x'_0 J'_{ux_0} J^{-1}_{uu} J_{ux_0} x_0 + x'_0 J_{x_0x_0} x_0. \tag{2.17}$$

Remark 2.3. *The finite horizon linear quadratic regulator problem as presented above is* not *set up as a feedback problem and at the first glance it may therefore not seem to be very interesting from a control point of view, where one is mostly looking for feedback controllers (that has a good trade-off between performance and robustness). However, since the solution can be written on the form* $u_k = -K_{LQR} x_k$,

where $K_{LQR} = J_{uu}^{-1} J_{ux_0}$, and all information about the past disturbances can be assumed to be accumulated into the states x_0, we in fact get a feedback solution when we implement the solution in a moving horizon fashion. Note however that in the problem formulation there is no information about any moving horizon, but using the solution as a moving horizon strategy happens to give good results. In summary, the LQR problem itself (posed as an open-loop problem in u) is not very interesting from a practical point of view, but it's implementation *as a state feedback is what makes this problem interesting.*

We further remind the readers about the guaranteed robustness margins of linear quadratic Gaussian control (LQG), for which "there are none" [Doyle, 1978], so using the state feedback with a Kalman filter in the loop should be done with care. Other researches have pointed out that this limitation of no guaranteed margins is not too important in practise, see e.g. [Pannocchia et al., 2005]. The practitioner should nevertheless be aware of this limitation.

2.2.2 Infinite horizon static output feedback

The problem of finding the static output feedback that minimizes the expected value of the infinite-horizon objective function J^∞ in equation (2.5) may be posed as minimizing the $\mathcal{H}_2$-norm of the lower fractional transform $F_l(P,K^y) = P_{11} + P_{12}(I - P_{22}K^y)^{-1}P_{21} \stackrel{S}{=} \left[\begin{smallmatrix} A_{F_l} & B_{F_l} \\ C_{F_l} & D_{F_l} \end{smallmatrix}\right]$ where P is given by e.g., [Skogestad and Postlethwaite, 2005]:

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \stackrel{S}{=} \left[\begin{array}{c|c:c} A & W_{x_0}^{1/2} & B \\ \hline Q^{1/2} & 0 & 0 \\ 0 & 0 & R^{1/2} \\ \hdashline C & 0 & 0 \end{array}\right]. \tag{2.18}$$

The $\mathcal{H}_2$-norm $\|F_l(P,K^y)\|_2$ can be computed by first solving a Lyapunov equation to obtain the "state covariance" matrix S [Bryson and Ho, 1975]:

$$S = A_{F_l} S A_{F_l}' + B_{F_l} B_{F_l}', \tag{2.19}$$

The "output covariance" is then given by

$$R = C_{F_l} S C_{F_l}' + D_{F_l} D_{F_l}' \tag{2.20}$$

and finally

$$\|F_l(P,K^y)\|_2 = \text{trace}(R). \tag{2.21}$$

This problem is nevertheless believed to be NP-hard [Blondel and Tsitsikilis, 1997], and the "brute-force" approach is to optimize directly on the gain matrix K^y, which works well as long as we have a good initial guess. A minimal requirement from

our experience is that the initial guess should stabilize the plant, as the $\mathcal{H}_2$-norm is only defined for stable plants.

Remark 2.4. *(Interpretation of $\mathcal{H}_2$-norm) The problem of regulating a system that is driven by white noise back to origin may seem a bit restrictive from the practitioner's point of view, since the dominating disturbances in e.g. a chemical process plant look more like sinusoids than white noise. By [Skogestad and Postlethwaite, 2005, Tables A.1-A.2, page 540] we have that one should minimize the $\mathcal{H}_2$-norm if*

1. *The input signal is assumed to be (series of) impulses and the error signal is evaluated by the 2-norm (energy).*

2. *The input signal is assumed to be bounded by it's energy (2-norm) and the error signal is evaluated by the ∞-norm (peak magnitude).*

In order to cover sinusoids one must use the maximum singular value, but especially point 2, energy bounded input and peak magnitude as a measure of error signal, should be quite interesting from a practical point of view.

2.2.3 Finite horizon static output feedback

Let us analyze the problem of minimizing the expected value of the finite horizon objective function J given in equation (2.6). For a given x_0 the objective function can be written as

$$J(x_0) = z'z \tag{2.22}$$

with

$$z = G^{1/2}x_0 \tag{2.23}$$

$$G = (A_c^N)'Q_N A_c^N + \sum_{i=0}^{N-1} (A_c^i)'(Q + (KC)'RKC)A_c^i \tag{2.24}$$

where the dependence on K^y enters through the state matrix A_c,

$$A_c = A - BK^yC. \tag{2.25}$$

Note that A_c^i is A_c raised to the power i, and further note that G is symmetric under the assumption that Q and R are symmetric matrices. It now follows that

$$\begin{aligned} E\{J\} &= E\left\{z'z\right\} = E\left\{\text{trace}(zz')\right\} = E\left\{\text{trace}(G^{1/2}x_0x_0'G^{1/2})\right\} \\ &= E\left\{\text{trace}(G^{1/2}G^{1/2}x_0x_0')\right\} \\ &= \text{trace}(G^{1/2}G^{1/2}E\left\{x_0x_0'\right\}) \\ &= \text{trace}(G^{1/2}G^{1/2}W_{x_0}^{1/2}W_{x_0}^{1/2}) \\ &= \text{trace}((G^{1/2}W_{x_0}^{1/2})'(G^{1/2}W_{x_0}^{1/2})) \\ &= \|G^{1/2}W_{x_0}^{1/2}\|_F^2 \end{aligned}$$

where we have used that $E\{x_0x_0'\} = \text{var}(x_0) = W_{x_0}$ and the identify $\text{trace}(AB) = \text{trace}(BA)$.

In order to synthesize a static output feedback controller one may use a non-linear search to minimize the norm $\|G^{1/2}W_{x_0}^{1/2}\|_F^2$, but this problem is expected to be NP-hard [Blondel and Tsitsikilis, 1997].

2.3 Minimization of expected loss from optimal control

In this section we first analyse the problem of minimizing the expected loss

$$E\{J(u,d) - J_{\text{opt}}(d)\} \tag{2.26}$$

and then we derive an upper bound for this problem that may be found by convex optimization.

2.3.1 Analysis

Let the plant model (2.1)-(2.2) be written in the from $x = G_{x_0}^x x_0 + G_u^x u$ as in equation (2.12) and let

$$G_u^y = \bar{C}G_u^x + \bar{D}, \quad G_{x_0}^y = \bar{C}G_{x_0}^x, \tag{2.27}$$

where

$$\bar{C} = \text{diag}(C,\ldots,D), \quad \bar{D} = \text{diag}(D,\ldots,D). \tag{2.28}$$

Further, let y denote the stacked vector of outputs,

$$y = \begin{bmatrix} y_0 \\ \vdots \\ y_N \end{bmatrix}. \tag{2.29}$$

Consider now the augmented linear model

$$\tilde{y} = \begin{bmatrix} u \\ y \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ G_u^y \end{bmatrix}}_{G^{\tilde{y}}} u + \underbrace{\begin{bmatrix} 0 \\ G_{x_0}^y \end{bmatrix}}_{G_d^{\tilde{y}}} \underbrace{W_d d}_{x_0} \tag{2.30}$$

where d is a normally distributed random variable with zero mean and unit covariance, and $W_d = W_{x_0}^{1/2}$, such that $x_0 = W_d d$. Note that

$$\mathrm{var}(x_0) = \mathrm{var}(W_d d) = \mathrm{var}(W_{x_0}^{1/2} d) = W_{x_0} \tag{2.31}$$

as expected. The static output feedback constraint $u = -\mathrm{diag}(K^y, \ldots, K^y)y$ can be written as

$$\underbrace{\begin{bmatrix} I & \mathrm{diag}(K^y, \ldots, K^y) \end{bmatrix}}_{H} \tilde{y} = 0. \tag{2.32}$$

Here, the block diagonal structure $\mathrm{diag}(K_0^y, K_1^y, \ldots, K_{N-1}^y)$ is required to have causality whereas $K^y = K_0^y = K_1^y = \cdots = K_{N-1}^y$ gives a time invariant controller.

Halvorsen et al. [2003] show that the loss $J(u,d) - J_{\mathrm{opt}}(d)$ for a given d of adding a constraint $H\tilde{y} = 0$ to an otherwise unconstrained optimization problem is given by

$$L(u,d) = J(u,d) - J_{\mathrm{opt}}(d) = \frac{1}{2} q' q \tag{2.33}$$

with

$$q = -XHFW_d d, \tag{2.34}$$

$$X = J_{uu}^{1/2} (HG^{\tilde{y}})^{-1}, \tag{2.35}$$

$$F = -(G^{\tilde{y}} J_{uu}^{-1} J_{ux_0} - G_{x_0}^{\tilde{y}}). \tag{2.36}$$

By exactly the same deviation as for the finite horizon static output feedback problem in section 2.2.3 we find that the expected value of the loss is

$$E\{L\} = \frac{1}{2} \| \underbrace{J_{uu}^{1/2} (HG^{\tilde{y}})^{-1} HF}_{G} W_d \|_F^2. \tag{2.37}$$

This expression is also derived by Kariwala et al. [2008].

2.3.2 Solution

Problem (2.37) of minimizing the expected loss can be restated as:

$$\min_H \; E\{L\} = \|XHFW_d\|_F^2 \tag{2.38}$$

$$\text{s.t. } X = J_{uu}^{1/2}(HG^{\tilde{y}})^{-1} \tag{2.39}$$

$$H \text{ on the form } H = \begin{bmatrix} I & \text{diag}(K^y,\ldots,K^y) \end{bmatrix} \tag{2.40}$$

This problem is non-convex due to the inversion of $HG^{\tilde{y}}$, and moreover it is probably as difficult as the original static output feedback problem of minimizing the cost (2.6), as the only difference is that we subtract a reference controller in the objective function.

However, if we for the moment let H be a full matrix, meaning that u could be a function of all outputs y (in general, non-causal), there exists a reformulation which makes the problem convex:

Theorem 2.1. *(Convex reformulation for full H [Alstad et al., 2009]) The optimal full H that minimizes the expected loss in equation* (2.37) *can be found by solving the* convex *optimization problem*

$$\begin{aligned} &\min_H \; \|HFW_d\|_F \\ &\textit{subject to } X = (HG^y) \textit{ being any fixed full rank matrix (e.g., } X = J_{uu}^{1/2}) \end{aligned} \tag{2.41}$$

The reason why we can omit $X = J_{uu}^{1/2}(HG^y)^{-1}$ is that the expression for the loss is not affected by a non-singular matrix D in front of H. That is, if H is an optimal solution that minimizes $\|XHF\|_F$, then so is $H_1 = DH$ where D is any non-singular matrix of appropriate dimensions [Alstad et al., 2009].

For our case, where we know that we are looking for a feedback solution, it can be satisfied by requiring that $H_1 = \begin{bmatrix} I & K \end{bmatrix}$ where use of the identity matrix guarantees full rank and where the remaining K is a full matrix. This gives no loss because we may choose the non-singular D to get $H_1 = DH$.

Remark 2.5. *Actually, [Alstad et al., 2009] require $X = I$ which is one way of guaranteeing that X has full rank. This can be relaxed even more as X does not need to be the identity matrix (I), but can be any full rank matrix Yelchuru [2010]. This also means that the matrix J_{uu} is not needed, and from this we finally get the result in Theorem 2.1 that it is enough to require that HG^y has full rank. Explicit knowledge about J_{uu} is not actually required when finding the optimal H using* (2.41)*, but it would be required to find a numerical value for the loss.*

Theorem 2.1 does not apply for the case of a structured $K = \text{diag}(K^y,\ldots,K^y)$. However, a suboptimal K^y is obtained by solving the following problem:

Definiton 2.1. *(Upper bound on the loss by using static output feedback) The following convex problem can be used to find an upper bound for the optimal static output feedback:*

$$\begin{aligned} &\min_{H} \|HFW_d\|_F^2 \\ &\text{s.t. } H \text{ on the form } H = \begin{bmatrix} I & \text{diag}(K^y, \ldots, K^y) \end{bmatrix}, \end{aligned} \tag{2.42}$$

where F is given by (2.36).

As we demonstrate in the examples, we have numerical evidence that indicates that for some interesting cases, we get solutions that are close enough to the true solution such that we can use the resulting controller as an initial condition for a nonlinear search (or simply use the sub-optimal controller directly).

Problem (2.42) can be solved as a quadratic program (QP) by vectorization. A so-called "large-scale algorithm" is a good candidate to solve the resulting QP, as the problem is structured and only equality constraints are present [Mat].

Remark 2.6. *Note that problem* (2.42) *differs from the original static output feedback problem in two ways: First, we minimize the loss from the LQR controller, rather than the cost function* (2.6) *directly. Second, we neglect $X = J_{uu}^{1/2}(HG^y)^{-1}$ which gives the effect of the term $X = J_{cc}^{1/2}$. Neglecting X is not expected to have a large effect as we conclude that it is exact for the case when K is full.*

2.4 Examples

We here give two examples of application of the static output feedback design method. First we discuss proportional-integral (PI) LQ-optimal control of a set of second-order underdamped plants $g(s) = k/(\tau^2 s^2 + 2\tau\zeta s + 1)$, where we change the damping coefficient ζ in the range $\zeta \in [0,1]$. As a second example we discuss multivariable PID control of a distillation column.

2.4.1 PI control of underdamped plant

Consider the following second-order system:

$$g(s) = \frac{k}{\tau^2 s^2 + 2\tau\zeta s + 1}. \tag{2.43}$$

We want to design a proportional-integral (PI) LQ-optimal controller for this plant, and we want to investigate our design method for different values of the

damping coefficient ζ. For plants with small values of ζ a PI controller is not expected to work very well, and thus the problem of finding PI parameters is expected to be difficult.

Using the observer canonical form, see e.g. [Skogestad and Postlethwaite, 2005], we write the system on state-space form:

$$\dot{x} = \underbrace{\begin{bmatrix} 0 & 1 \\ 1/\tau^2 & -2\zeta/\tau \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} 0 \\ k/\tau^2 \end{bmatrix}}_{B} u, \quad y_m = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} z \tag{2.44}$$

To design a PI controller, we augment the system with an integrated output, to get

$$\begin{aligned} \dot{z} = \begin{bmatrix} \dot{x} \\ \dot{\sigma} \end{bmatrix} &= \underbrace{\begin{bmatrix} A & & 0 \\ & & 0 \\ 1 & 0 & 0 \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} x \\ \sigma \end{bmatrix} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{\tilde{B}} u \\ y_m = \begin{bmatrix} y_P \\ y_I \end{bmatrix} &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\tilde{C}} z \end{aligned} \tag{2.45}$$

We here consider $\tau = k = 1$, and sample the plant with $T_s = 0.1$ to get a discrete-time model which we use for design of different static-output feedback controllers.

We define the quadratic weights for the controller design as:

$$Q = \begin{bmatrix} 0 & & \\ & 0 & \\ & & 1 \end{bmatrix}, \quad R = 1 \tag{2.46}$$

In addition we assume that the disturbances are equally distributed on all the states and therefore use $W_{x_0} = W_d^2 = I$.

Figure 2.1 shows closed loop norms $\|F_l(P,K^y)\|_2$ for the following controllers:

1: Optimal finite horizon PI for original cost function For a horizon of $N = 100$ we used nonlinear optimization to minimize the norm $\|G^{1/2}W_{x_0}^{1/2}\|_F$ as given in section 2.2.3.

2: Convex approximation of minimizing the expected loss For the same horizon of $N = 100$ we solve the convex problem (2.42).

3: Optimal infinite horizon PI for original cost function Here we used a nonlinear solver to solve the problem posed in section 2.2.2, i.e. we search for a K^y such that the norm $\|F_l(P,K^y)\|_2$ in equation (2.21) is minimized.

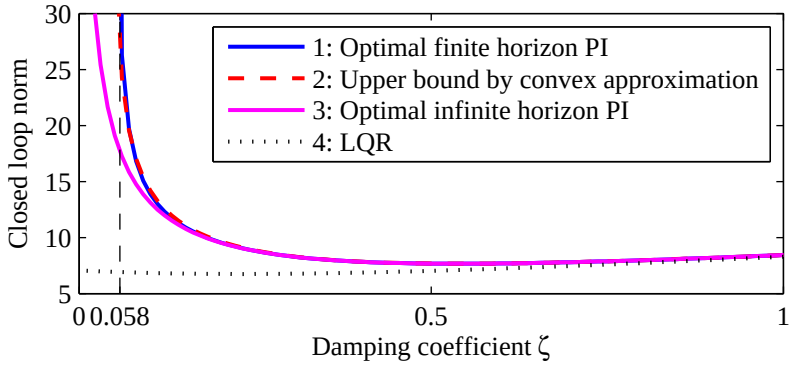

Figure 2.1: Closed-loop norms $\|F_l(P,K^y)\|_2$ for different design methods.

4: Optimal PI for loss formulation For an input horizon of $N = 100$ was problem (2.38)-(2.40) too time-consuming to solve. We discuss the properties of this solution for a smaller horizon after this example.

5: LQR controller An infinite horizon linear quadratic controller for the augmented plant.

Corresponding closed-loop responses for a damping coefficient $\zeta = 0.101$ are shown in Figure 2.2, and the specific controller gains for the same damping coefficient are given in Table 2.1. As already mentioned we used $N = 100$ in the design of the finite horizon controllers, which in this plot corresponds to 10 time units. For such a underdamped system, it seems like the prediction horizon is a bit too small. Nevertheless, we conclude that the controllers are comparable, and that they can be ranked as shown by the closed-loop norms in Figure 2.1, namely that optimal infinite horizon PI controller is the best static output feedback controller, but that the optimal finite horizon PI controller and the convex approximation are quite similiar with respect to closed-loop performance.

We observe that the LQR controller for the same process is superior in performance to the static output feedback controllers. This is expected as we here chose a “difficult” plant to control with the PI-structure of the controller. The LQR controller use a measurement of the whole state vector and does not suffer from the same controllability issues.

For both optimal control problems, which need to be solved by nonlinear optimization, we started with $\zeta = 1$, for which initial conditions can easily be found, and incrementally decreased the damping coefficient ζ to trace the optimum towards $\zeta = 0$.

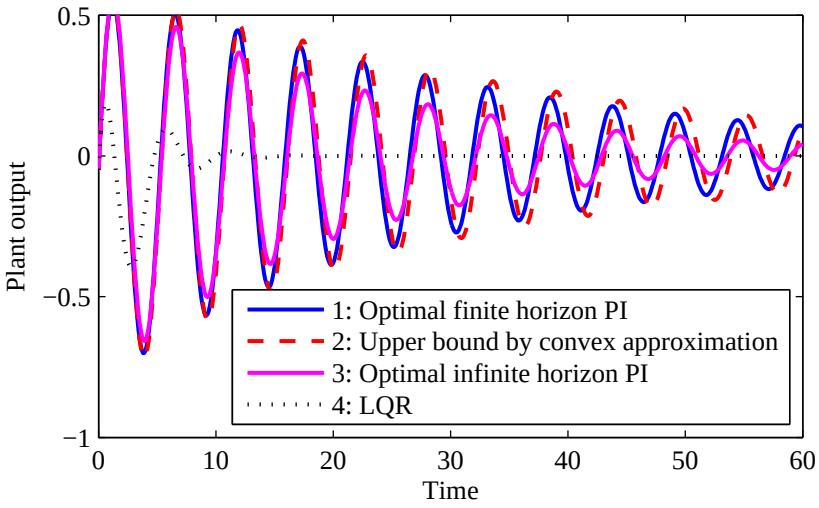

Figure 2.2: Closed loop responses for PI control with damping coefficient $\zeta = 0.101$. Disturbance at time $t = 0$ is $x_0 = (1,0,0)$.

Case	Controller gain
Optimal finite horizon PI	$\begin{bmatrix} 0.411 & 0.167 \end{bmatrix}$
Upper bound by convex approximation	$\begin{bmatrix} 0.372 & 0.173 \end{bmatrix}$
Optimal infinite horizon PI	$\begin{bmatrix} 0.389 & 0.124 \end{bmatrix}$
LQR	$\begin{bmatrix} 0.879 & 1.151 & 0.942 \end{bmatrix}$

Table 2.1: Output feedback (PI) controllers for damping coefficient $\zeta = 0.101$.

Expected value to be minimized	Controller gain	Closed loop norm (= infinite horizon cost)
Finite horizon cost J in (2.6)	$\begin{bmatrix}0.643 & 0.335\end{bmatrix}$	13.74
Approximated loss in (2.42)	$\begin{bmatrix}0.413 & 0.232\end{bmatrix}$	11.12
Exact loss in (2.37)	$\begin{bmatrix}0.345 & 0.161\end{bmatrix}$	10.95
Infinite horizon cost J^∞ in (2.5)	$\begin{bmatrix}0.447 & 0.190\end{bmatrix}$	10.81
LQR	$\begin{bmatrix}0.899 & 1.085 & 0.946\end{bmatrix}$	6.78

Table 2.2: Output feedback (PI) controllers for damping coefficient $\zeta = 0.15$ with a smaller horizon for the finite horizon costs, $N = 20$, compared to the rest of the example, where we used $N = 100$.

From Figure 2.1 we observe that in the range of processes for $\zeta \in [0.058, 1]$ is the convex approximation very close to the optimal finite horizon PI controller. For the very underdamped processes in the range $[0, 0.058]$ neither of the finite horizon PI controllers resulted in stable closed loop. This might have been improved by using a larger prediction horizon N in the design problem. However, even by taking an infinitely long prediction horizon there is a point where the "forced" PI structure of the controller is simply not suited for the process. This is illustrated by the infinite horizon PI controller, for which the closed-loop norm increases and eventually becomes infinity as $\zeta \to 0$. This is because derivative action (PID controller) is required to stabilize a process on the form $1/s^2$.

Comparison with exact loss minimization

In Table 2.2 we show the results of the different design methods when we use a shorter horizon, $N = 20$, for the finite horizon approximations. Minimizing the expected loss function in equation (2.37) is quite complicated, so we could not include this result in the rest of the example, where we used an horizon of $N = 100$ in the finite horizon approximations.

The results in Table 2.2 are quite interesting; we observe that in terms of the closed loop norm, our convex approximation of minimizing the loss is actually *better* than directly minimizing the finite horizon approximation. This may be related to the fact that in the convex approximation we are minimizing the loss to an LQR controller, which does not degenerate when the horizon is small. The static output feedback controller that directly minimizes the cost should be more sensitive to a short horizon, because the weight on the final state in the cost function

assumes an LQR controller, which is not correct. The weight matrix for the final state x_N was

$$Q_N = \begin{bmatrix} 14.73 & 9.58 & 14.16 \\ 9.58 & 11.17 & 10.01 \\ 14.16 & 10.01 & 20.08 \end{bmatrix}, \tag{2.47}$$

derived by the method of Remark 2.1.

We further notice that the exact loss is quite close in performance to minimizing directly the infinite cost. Then there is an additional (small) increase in the cost by using the approximated loss function, this is due to the convex relaxation of the problem.

2.4.2 Binary distillation

In this example we consider MIMO-PI and -PID control of "Column A" in [Skogestad, 1997]. The model is based on the following assumptions:

- binary separation,
- 41 stages, including reboiler and total condenser,
- each stage is at equilibrium, with constant relative volatility $\alpha = 1.5$,
- linearized liquid flow dynamics,
- negligible vapor holdup,
- constant pressure.

A sketch of the column is shown in Figure 2.3. The feed enters on stage 21.

We here consider the LV-configuration, where D and B are used to control the levels. With level controllers implemented (P-control with $K_c = 10$) the rest of the column is stable.

The model is first linearized around the nominal operating point given in Table 2.3. Balanced reduction was used to reduce the number of states from 82 to 16, with a largest neglected Hankel singular value of $8.067 \cdot 10^{-5}$. Then integrated outputs were added to the model, resulting in a model with 18 states. If we let the outputs of the model be P, I, and D, we get a model with the following structure:

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \dot{\sigma} \end{bmatrix} &= \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ \sigma \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u \\ \begin{bmatrix} y^P \\ y^I \\ y^D \end{bmatrix} &= \begin{bmatrix} C & 0 \\ 0 & I \\ CA & 0 \end{bmatrix} \begin{bmatrix} x \\ \sigma \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ CB \end{bmatrix} u \end{aligned}. \tag{2.48}$$

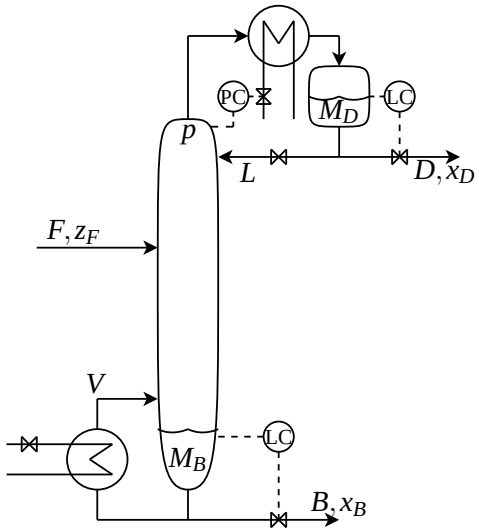

Figure 2.3: Sketch of distillation column.

Type	Description	Variable		Nominal Value
Input	Reflux flow	L	(u_1)	1.87
	Vapor flow	V	(u_2)	2.37
Disturbance	Feed flow	F	(d_1)	1.0
	Feed composition	z_F	(d_2)	0.5
	Liquid feed fraction	q_F		1.0
Output	Overhead composition (light component)	x_D	(y_1)	0.95
	Bottoms composition (light component)	x_B	(y_2)	0.05

Table 2.3: Variables for distillation example. Variable names in parenthesis indicate corresponding deviation variables, for example $u_1 = L - 1.87$. The liquid feed fraction q_F is assumed constant.

Description	Control equation	$\|F_l(P,K)\|_2$
Upper bound PI	$u_k = -\begin{bmatrix} 16.2850 & -4.5264 \\ 0.0938 & -8.2969 \end{bmatrix} y_k^P - \begin{bmatrix} 2.8640 & -0.6262 \\ 0.3900 & -2.5700 \end{bmatrix} y_k^I$	3.654
Optimal PI	$u_k = -\begin{bmatrix} 15.6967 & -4.9955 \\ 1.2471 & -9.9017 \end{bmatrix} y_k^P - \begin{bmatrix} 2.6874 & -0.6987 \\ 0.4058 & -2.7756 \end{bmatrix} y_k^I$	3.650
Upper bound PID	$u_k = -\begin{bmatrix} 16.7001 & -4.6139 \\ 0.1694 & -8.6518 \end{bmatrix} y_k^P - \begin{bmatrix} 2.9048 & -0.5948 \\ 0.3192 & -2.6017 \end{bmatrix} y_k^I - \begin{bmatrix} 1.0589 & -0.4309 \\ -1.4951 & -1.2063 \end{bmatrix} y_k^D$	3.650
Optimal PID	$u_k = -\begin{bmatrix} 16.5929 & -5.2730 \\ 2.3102 & -11.0965 \end{bmatrix} y_k^P - \begin{bmatrix} 2.6510 & -0.6209 \\ 0.1020 & -2.7143 \end{bmatrix} y_k^I - \begin{bmatrix} -2.4821 & -3.8819 \\ -7.0188 & -8.4573 \end{bmatrix} y_k^D$	3.630
LQR	Given by equation (2.50)	3.610

Table 2.4: Controller gains for the upper bound PI and PID controllers found by solving problem (2.42) and optimal controller for the distillation column example.

This model is sampled with $T_s = 1$ to get a discrete time model. We use the weights $Q = C' \begin{bmatrix} 0 & & \\ & I & \\ & & 0 \end{bmatrix} C$, and $R = 0.1 \cdot I$. For the finite horizon approximation we use a prediction horizon of $N = 80$, and we also for this example assume $W_d = I$.

We now look for controllers on the form

$$u_k = -\left(K^P y_k^P + K^I y_k^I + K_D y_k^D\right) \tag{2.49}$$

and we assume measurements of the compositions with a sample time of 1 minute is available.

Table 2.4 shows the result of using the approximation given by problem (2.42) and the assumed global optimum of the original problem for both MIMO-PI and -PID designs. In particular for the PI case is the solution of the convex approximation quite close to the optimal PI controller. The LQR controller $u_k = -K^{\mathrm{LQR}} x_k$, with gain

$$\begin{aligned} K^{\mathrm{LQR}} &= \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix}, \\ \text{where } k_1 &= \begin{bmatrix} -0.0022 & 0.0002 & -0.0004 & -0.0007 & 0.0016 & -0.0097 \\ 0.0008 & 0.0015 & -0.0016 & -0.0037 & 0.0079 & -0.0074 \end{bmatrix}, \\ k_2 &= \begin{bmatrix} -0.0036 & 0.0048 & 0.0116 & -0.0011 & -0.0213 & 0.0305 \\ -0.0066 & 0.0262 & 0.0610 & 0.0044 & 0.0093 & -0.0148 \end{bmatrix}, \\ k_3 &= \begin{bmatrix} 0.0149 & 0.0521 & 0.1349 & 0.1034 & 2.6897 & -0.5975 \\ 0.0233 & -0.0372 & -0.1607 & 0.0895 & -0.1350 & -2.5939 \end{bmatrix}, \end{aligned} \tag{2.50}$$

has a closed loop gain which is in the same range as the static output feedback controllers. We also notice with interest that adding derivative action does not improve the closed loop norm much, even for the optimal PID controller.

Figure 2.4 shows responses to step disturbances for the different controllers. The optimal PID controller seems to be a bit too aggressive, however this is ex-

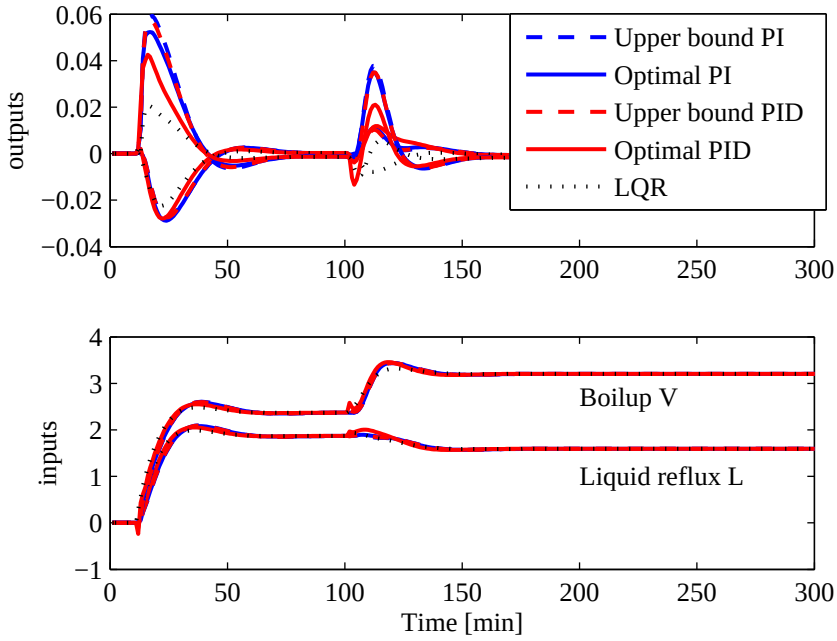

Figure 2.4: Closed loop responses for distillation example. Disturbances: step in feed rate at $t = 10$ and step in feed composition at $t = 100$.

pected as we did not include any penalty on the rate of change $(u_k - u_{k-1})$. Nevertheless, the controllers are quite comparable in performance, which is what we wanted to demonstrate with this example.

2.5 Proposed procedure

For plants where a MIMO-PID or another "fixed-structure" controller is desired, we propose the following design procedure in order to use the results in this chapter:

1. Design a good LQR controller by modifying the weights Q and R in the quadratic objective function (2.5).

2. Propose a *simple* fixed-structure controller, for example a diagonal PI controller, and use convex approximation (2.42) to find the controller gains.

3. Evaluate the difference in closed-loop costs between the LQR and the proposed simple controller.

(a) If the difference large, increase the complexity of the controller by for example allowing for interactions in the controller (i.e. add off-diagonal elements). In addition one can add filters in the PID formulations to change the dynamic properties of the controller.

(b) If the difference is acceptably small, finish.

In the procedure above we solve at each iteration a static decentralized control problem (where static in this chapter allows for MIMO-PID and other fixed-structure designs). For a given decentralized controller, we compare the cost to the LQR controller, and we add complexity if needed. Note that decentralized control is known to have non-unique solutions, see for example [Hovd and Skogestad, 1994, Example 6] and [Lundström and Skogestad, 1991]. It seems therefore likely that the convex approximation of the minimization of expected loss may perform badly for a controller with many zero elements. Investigation of this phenomenon is planned further work in this project. However, for a structure that is well-suited for the plant, the method should work satisfactory.

2.6 Discussion

2.6.1 Loss versus finite horizon objective

We observed in the example of PI control of an underdamped plant that the convex approximation of minimizing the loss actually performed *better* in terms of the closed-loop norm than minimizing a finite-horizon objective function. As already mentioned this is probably due to that the finite horizon cost function is not a good approximation for the infinite horizon cost for small horizons N when static output feedback is used. Minimizing the loss seems to be less sensitive to small horizons, and this is probably due to that we are approximating the cost function of the linear quadratic controller described in section 2.2.1 which is independent of N.

The problems with the finite horizon objective function for controller design may also be brought back to the issues with the moving horizon controller itself; usually when a moving horizon control problem is posed feedback is not taken into account, subsequently by only implementing the first move we have a difference between the controller used on the plant and the one found by the optimizer. These limitations have been acknowledged in the terms of robust model predictive control, see e.g. [Bemporad and Morari, 1999], but are often not addressed.

The limitations with the finite horizon approach further justifies our approach of minimizing the expected loss from the cost of using an optimal controller. The approximation may be as valid as simply using a finite horizon cost for output feedback design, and this is indeed demonstrated by the results of Table 2.2 for the example of controlling an underdamped plant.

2.6.2 Solution method

Recall the convex approximation (2.42):

$$\begin{aligned} &\min_H \|HFW_d\|_F \\ &\text{s.t. } H \text{ on the form } H = \begin{bmatrix} I & \operatorname{diag}(K^y,\ldots,K^y) \end{bmatrix}. \end{aligned}$$

We chose to solve this as a quadratic program by vectorization. Vectorization of the problem can be done by first defining

$$H \triangleq Z' = \begin{bmatrix} z_1' \\ z_2' \\ \vdots \\ z_n' \end{bmatrix} \tag{2.51}$$

and then

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \tag{2.52}$$

and finally

$$F_{\text{diag}} = \begin{bmatrix} FW_d & & & \\ & FW_d & & \\ & & \ddots & \\ & & & FW_d \end{bmatrix} \tag{2.53}$$

We now rewrite the objective function as:

$$\begin{aligned} \|HFW_d\|_F^2 = \|Z'FW_d\|_F^2 = \| \begin{bmatrix} z_1' \\ z_2' \\ \vdots \\ z_n' \end{bmatrix} FW_d \|_F^2 = \| \begin{bmatrix} z_1'FW_d \\ z_2'FW_d \\ \vdots \\ z_n'FW_d \end{bmatrix} \|_F^2 = \\ = \| \begin{bmatrix} z_1'FW_d & z_2'FW_d & \cdots & z_n'FW_d \end{bmatrix} \|_F^2 = \|z'F_{\text{diag}}\|_F^2 = \\ = \operatorname{trace}(z'F_{\text{diag}}F_{\text{diag}}'z) = z'F_{\text{diag}}F_{\text{diag}}'z \end{aligned} \tag{2.54}$$

Here we used the following identity: $\|A\|_F = \|A'\|_F = \operatorname{trace}(A'A) = \operatorname{trace}(AA')$.

In addition we add equality constraints on z such that H is constrained to be on the form $H = \begin{bmatrix} I & \operatorname{diag}(K^y,\ldots,K^y) \end{bmatrix}$, let $Az = b$ denote these constraints. In order to solve the convex approximation we therefore chose to solve the following QP:

$$\begin{aligned} &\min_z z'F_{\text{diag}}F_{\text{diag}}z \\ &\text{s.t. } Az = b \end{aligned} \tag{2.55}$$

		CPU time [minutes]	
N	size of optimization vector z	CPLEX	QUADPROG
80	102400 elements	1.4	1.4
100	160000 elements	5.0	5.3
120	230400 elements	43	out of memory

Table 2.5: CPU times for MIMO-PID design for the distillation example.

The optimization vector z has length $N^2 n_u(n_u+n_y)$, and the length of b is such that the remaining degrees of freedom (if we were to substitute the equality constraints into the objective function) are equal to the number of elements in the static output feedback gain K^y. Even though this is a convex QP we experienced that it took quite long time to solve the problem, as the following example illustrates.

Example: MIMO-PID design for distillation column. Let us consider the MIMO PID design in for the distillation example in section 2.4.2. In Table 2.5 we report CPU times for the commercial QP solver CPLEX version 12.1 called through the "IBM ILOG CPLEX Matlab connector" on a Dell PowerEdge 1950 with an Intel Xenon CPU E5410 at 2.33 GHz with 8GB RAM, running CentOS Linux release 5.2. In addition we tried to use Matlab's QUADPROG in "large-scale mode", which handles sparse Hessian and equality constraints. We observe that for small N the solvers have the same solution time, but QUADPROG causes Matlab to go out of memory for large N.

There is probably a significant potential to improve the solution of this QP by using a tailor-made solver. For instance, the "IBM ILOG CPLEX Matlab connector" converts sparse matrices to full matrices before it sends the problem to CPLEX. This is clearly not beneficial for our problem, as the Hessian matrix $F_{\text{diag}}F_{\text{diag}}'$ is a block diagonal matrix with FF' on the diagonal and zeros on the rest of the matrix. We leave improvements of the solution as further research for now.

2.6.3 Iterative method

The expected loss minimization,

$$\begin{aligned}&\min_H \|J_{uu}^{1/2}(HG^y)^{-1}HFW_d\|_F\\ &\text{s.t. } H \text{ on the form } \begin{bmatrix}H^y & H^u\end{bmatrix} = \begin{bmatrix}\text{diag}(K^y) & I\end{bmatrix},\end{aligned}$$

can be solved iteratively as

$$\begin{aligned} \min_{H_k} \ & \|Z_k F W_d\|_F \\ & \text{s.t. } H \text{ on the form } \begin{bmatrix} H^y & H^u \end{bmatrix} = \begin{bmatrix} \text{diag}(K^y) & I \end{bmatrix} \\ & Z_k = N_{k-1} H_k \end{aligned} \tag{2.56}$$

with

$$N_{k-1} = J_{uu}^{1/2} (H_{k-1} G^y)^{-1}. \tag{2.57}$$

However, we identified two problems with using this method. First, the method did not converge to the optimal solution of the nonlinear problem. This is due to that for a given N_{k-1}, the objective functions

$$J_1(H_k) = \|J_{uu}^{1/2} (H_k G^y)^{-1} H_k F W_d\|_F$$

and

$$J_2(H_k) = \|N_{k-1} H_k F W_d\|_F$$

are *not* the same. In the examples covered in this chapter this was not too important as $J_{uu}^{1/2} (H_k G^y)^{1/2}$ was close to unitary at the optimal point.

Second, we solved this problem as a quadratic program (QP) by vectorizing problem (2.56). For a large prediction horizon, just performing the vectorization takes a very long time ($>$ 1 hour). This can most likely be fixed by paying more attention to the coding, as we used "standard" Matlab code (sparse matrix functionality) to define the problem.

We therefore recommend to ignore the term N_{k-1} and rather solve the convex approximation, as this can be solved with considerably larger prediction horizon N for the same computational effort as the iterative scheme.

2.6.4 Static decentralized control

A static decentralized control problem may readily be posed as in problem (2.42), as one only has to add additional constraints that some parts of the static output feedback K^y should have zero elements. For example, one could search for a diagonal PI structure. As already mentioned in the design procedure, investigation of synthesis of diagonal controllers if planned further work in this project.

2.7 Conclusion

We have given a convex initialization for the static output feedback problem which can be used to initialize a non-linear search for finite and infinite horizon optimal

Variable	Value
Loss $L^* = \frac{1}{2}\overline{\sigma}^2(J_{uu}^{1/2}(HG^y)^{-1}HF)$	46.28
Corresponding input vector $d_{\max}$	$\begin{bmatrix}-0.05 & 0.83 & 0.56\end{bmatrix}'$
Finite horizon cost for $K^y_{\text{convex approximation}}$ starting from $x_0 = d_{\max}$	68.42
Finite horizon cost for K_{LQR} starting from $x_0 = d_{\max}$	22.14
Difference in costs	46.28
Bound on relative maximum loss by equation (2.58)	209%

Table 2.6: Loss calculation by the 2-norm for $\zeta = 0.101$.

static output feedback controllers. However, as examples indicate, the method may also be used on it's own, as the sub-optimal static output feedback controllers are quite close to the optimal ones for the examples studied.

We finally stress that this is a heuristic method with no guarantee of success for a given linear system, but that it performed well on the examples investigated in the chapter.

2.8 Appendix: Bounds on the loss

In this appendix we give some numerical verifications on our calculations and calculate a relative bound to the LQR controller by using the maximum singular value for the loss function. The maximum singular value will occur if we consider the worst case value of the loss function when the augmented disturbance $\tilde{d}' = \begin{bmatrix}d' & n'\end{bmatrix}$ is bounded in the 2-norm, see e.g. [Kariwala et al., 2008]. In this chapter we consider the expected loss (denoted as "average loss" by Kariwala et al. [2008]) when the augmented disturbance vector $\tilde{d}$ is assumed to be a random variable drawn from a normal distribution, and we showed that the resulting norm to be minimized is the Frobenius norm. Kariwala et al. [2008] showed that for the case of a full H (i.e. where the structural constraint "H on the form $\begin{bmatrix}H^y & H^u\end{bmatrix} = \begin{bmatrix}\text{diag}(K^y) & I\end{bmatrix}$") will an H that minimizes the Frobenius norm of the loss also minimize the maximum singular value, hence the optimal H is "super-optimal". This result cannot be applied to our case, since we have added the structural constraint that of static output feedback, but the norms should still be related.

Method	Loss	$J_{\text{approx}}(d_{\max})$	Bound
Upper bound PI	0.295	8.96	3.4%
Upper bound PID	0.252	8.90	2.9%

Table 2.7: Bounds of sub optimality for upper bound PI and PID controller calculated by expression (2.58).

2.8.1 Bound for the underdamped system

The loss can be used to calculate a relative bound between a static output feedback controller K^y and the optimal state feedback (finite horizon LQR) controller K_{LQR}. This is done in Table 2.6 for the case of damping coefficient $\zeta = 0.101$. We first use the singular value decomposition to find the maximum singular value of 9.62 and the corresponding worst-case disturbance $d_{\max} = \begin{bmatrix}-0.05 & 0.83 & 0.56\end{bmatrix}'$. We then calculate the finite horizon objective with $N = 100$ with $x_0 = d_{\max}$ for both the static output feedback controller and the LQR controller, and we observe that the difference in objective functions is indeed equal to the loss. Hence, a tight bound on the relative worst-case loss from optimality is:

$$\frac{J(K^y, d^{\max}) - J_{\text{opt}}(d^{\text{worst case}})}{J_{\text{opt}}(d^{\max})} = \frac{L^*}{J(K^y, d^{\max}) - L^*} \tag{2.58}$$

Remark 2.7. *The results in Table 2.6 gives bounds on the worst-case error for the given K^y that we found by using the convex approximation. Note however that we in this chapter use the Frobenius norm, and since H is not full it is not given that an optimal H in the Frobenius norm also minimizes the worst-case error, as is the case for full H, see [Kariwala et al., 2008].*

2.8.2 Bounds for distillation example

Also for the distillation case can we calculate the worst case relative difference in cost functions between optimal control and the static output feedback by equation (2.58). The results are reported in Table 2.7, and we observe that the relative difference between state feedback and static output feedback are a lot less that what was the case for the underdamped system. In this in accordance with the closed loop simulations in Figures 2.2 and 2.4.

Chapter 3

Static output feedback with noisy measurements

In this chapter we generalize the results in Chapter 2 to noisy measurements. This generalization has the following implications:

- The reference controller, which for the noise-free case was a finite horizon linear quadratic regulator, is for this case a non-causal controller which depends on information about the current and future disturbances.
- We need to consider disturbances on all the states over the time window of interest, because otherwise the noise will dominate the optimization.

The inclusion of noise gives us one extra tuning parameter, the noise weight W_n. This weight may be useful in applications, for example to reduce the derivative action in a multivariable PID controller, which can be achieved by simply adding more "fictitious" noise on the derivate output.

3.1 Introduction

Consider the linear process

$$x_{k+1} = Ax_k + Bu_k + d_k \tag{3.1}$$

$$y_k = Cx_k + Du_k + n_k \tag{3.2}$$

We assume that the disturbances d_k and noise terms n_k are normally distributed random variables with zero means and variances W, V, respectively.

Remark 3.1. *The difference in assumptions on model* (3.1)-(3.2) *from model* (2.1)-(2.3) *in Chapter* (2) *is that we have additional normally distributed disturbances*

d_k on all the states (and not only on the initial state x_0) and also allow for normally distributed noise n_k on all the measurements (while in Chapter 2 we neglected the noise).

The objective is to find a *static output feedback* $u_k = -K^y y_k$ that minimizes the *expected value* of the objective

$$J^\infty = \lim_{T\to\infty} \left\{ \frac{1}{T} \sum_{i=0}^{T} x_i' Q x_i + u_i' R u_i \right\}, \tag{3.3}$$

with $Q = C_z' Q^y C_z$ where $Q^y \geq 0$ is a quadratic weight on the controlled outputs $z_k = C_z x_k$ and $R > 0$ is a weight on input usage. The infinite horizon objective function may be approximated by a finite horizon cost function (see e.g. Chmielewski and Manousiouthakis [1996] and Scokaert and Rawlings [1998]) to

$$J = x_N' Q_N x_N + \sum_{i=0}^{N-1} x_i' Q x_i + u_i' R u_i. \tag{3.4}$$

Contribution of this chapter. In this chapter we derive a convex approximation to the problem of minimizing the *expected value* of the non-negative loss

$$L(u,d,n) = J(u,d,n) - J_{\text{opt}}(d) \tag{3.5}$$

where $J(u,d,n)$ is the value of the cost function (3.4) evaluated and for a given static output feedback policy for a specific realization of the disturbance

$$d = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{N-1} \end{bmatrix} \tag{3.6}$$

and noise

$$n = \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_{N-1} \end{bmatrix} \tag{3.7}$$

For more background information is the reader referred to Chapter 2.

Chapter overview. The rest of the chapter is organized as follows: First, we show how the background material in Chapter 2 needs to be adjusted in order to account for noise. That is, we first present the reference controller that is used in this chapter (corresponding to the linear quadratic regulator for the case of no noise in Chapter 2). Then some modifications to the problem of infinite and finite horizon static output feedback are discussed. Thereafter we present our main result, which is a convex approximation of the problem of finding a static output feedback such that the expected loss from an optimal controller is minimized. Finally, we demonstrate the method on a simple example.

3.2 Background material

3.2.1 Optimal non-causal control

Let us consider the extension from the finite horizon linear quadratic regulator in Chapter 2, section 2.2.1, when we let the optimal input $u = (u_0, u_1, \cdots, u_{N-1})$ be a function of all disturbances d (and not only the initial disturbance x_0 as in Chapter 2). (The resulting reference will then no longer be a causal controller.) First, let the linear model $x_{k+1} = Ax_k + Bu_k + d_k$ be written on the form

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} B & & & \\ AB & B & & \\ \vdots & & \ddots & \\ A^{N-1}B & \dots & \dots & B \end{bmatrix}}_{G_u^x} u + \underbrace{\begin{bmatrix} I & & & \\ A & I & & \\ \vdots & & \ddots & \\ A^{N-1} & \dots & \dots & I \end{bmatrix}}_{G_d^x} d. \tag{3.8}$$

By substitution of the model (3.8) into the finite horizon objective function (3.4) we obtain

$$J = \begin{bmatrix} u \\ d \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ud} \\ J_{ud}' & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix} \tag{3.9}$$

where

$$J_{uu} = G_u^{x\prime} \bar{Q} G_u^x + \bar{R} \tag{3.10}$$
$$J_{ud} = G_u^{x\prime} \bar{Q} G_d^x \tag{3.11}$$
$$J_{dd} = G_d^{x\prime} \bar{Q} G_d^x \tag{3.12}$$
$$\bar{Q} = \mathrm{diag}(Q, Q, \cdots, Q, Q_N) \tag{3.13}$$
$$\bar{R} = \mathrm{diag}(R, R, \cdots, R) \tag{3.14}$$

The optimizer of problem (3.9) may be found by simply completing the squares [Åström and Wittenmark, 1984] and is given by:

$$u_{\mathrm{opt}} = -J_{uu}^{-1} J_{ud} d \tag{3.15}$$

The optimal cost (for a given d) is then given by

$$J_{\text{opt}}(d) = -d'J'_{ud}J_{uu}^{-1}J_{ud}d + d'J_{dd}d. \tag{3.16}$$

The optimal input $u_{\text{opt}} = -J_{uu}^{-1}J_{ud}d$ cannot be realized as a feedback controller since it depends on the full disturbance vector d, including future disturbances. The optimal cost (3.16) is therefore a measure of how well we can control the plant with any controller, non-causal included.

Remark 3.2. *In Chapter 2, where only disturbances on x_0 were allowed, we found a nice interpretation of the resulting optimal input u, namely that it can be implemented as the solution to a finite horizon linear quadratic regulator problem. Unfortunately the corresponding expression for the case of more disturbances, equation* (3.15)*, does not have such an interpretation, as we cannot realize the control action by a feedback controller. Nevertheless, it is this optimal controller that we will use as reference for the loss minimization when noise is present, because this reference lets us derive a convex approximation similar to what we did in Chapter 2.*

3.2.2 Infinite horizon static output feedback with noise

The problem of minimizing the expected value of the infinite-horizon objective function J^∞ in equation (2.5) may be posed as minimizing the $\mathcal{H}_2$-norm of the lower fractional transform $F_l(P,K^y) = P_{11} + P_{12}(I - P_{22}K^y)^{-1}P_{21} \stackrel{S}{=} \left[\begin{smallmatrix} A_{F_l} & B_{F_l} \\ C_{F_l} & D_{F_l} \end{smallmatrix}\right]$ where P is given by e.g., [Skogestad and Postlethwaite, 2005]:

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \stackrel{S}{=} \left[\begin{array}{c|cc:c} A & W^{1/2} & 0 & B \\ \hline Q^{1/2} & 0 & 0 & 0 \\ 0 & 0 & 0 & R^{1/2} \\ \hdashline C & 0 & V^{1/2} & 0 \end{array}\right]. \tag{3.17}$$

The reason we can use the $\mathcal{H}_2$-norm is that this may be interpreted as the response to input signals that are white noise (by for example following a normal distribution as in this case) [Skogestad and Postlethwaite, 2005]. For more background material see Chapter 2.

Note that the only difference between the plant (3.17) and the corresponding plant in equation (2.18), Chapter 2, is that we have added an extra input channel to account for the measurement noise.

3.2.3 Finite horizon static output feedback with noise

Let us analyze the problem of minimizing the expected value of the finite horizon objective function J in equation (3.4) when noise is present. First, for a given static

output feedback $u_k = -K^y y_k$ we have that

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} I & & & \\ A_c & I & & \\ \vdots & & \ddots & \\ A_c^{N-1} & \dots & \dots & I \end{bmatrix}}_{G_d^x} d - \underbrace{\begin{bmatrix} BK^y & & & \\ A_cBK^y & BK^y & & \\ \vdots & & \ddots & \\ A_c^{N-1}BK^y & \dots & \dots & BK^y \end{bmatrix}}_{-G_n^x} n, \tag{3.18}$$

which on a compact form can be written as

$$x = \underbrace{\begin{bmatrix} G_d^x & G_n^x \end{bmatrix}}_{G_{\tilde{d}}^x} \underbrace{\begin{bmatrix} d \\ n \end{bmatrix}}_{\tilde{d}} = G_{\tilde{d}}^x \tilde{d}. \tag{3.19}$$

where

$$A_c = A - BK^yC. \tag{3.20}$$

In addition, the input u_k is given by

$$u_k = -K^y y_k = -K^y(Cx_k + n_k), \tag{3.21}$$

which means that the vector of stacked inputs u can be expressed as

$$\begin{aligned} u &= -(\overline{K^yC})x - \overline{K^y}n \\ &= -(\overline{K^yC})G_{\tilde{d}}^x\tilde{d} - \overline{K^y}n \\ &= -\begin{bmatrix} (\overline{K^yC})G_d^x & (\overline{K^yC})G_n^x + \overline{K^y} \end{bmatrix} \begin{bmatrix} d \\ n \end{bmatrix} \\ &= -G_{\tilde{d}}^u\tilde{d} \end{aligned}$$

where

$$(\overline{K^yC}) = \text{diag}(K^yC, \cdots, K^yC) \tag{3.22}$$

$$\overline{K^y} = \text{diag}(K^y, \cdots, K^y) \tag{3.23}$$

and

$$G_{\tilde{d}}^u = \begin{bmatrix} G_d^u & G_n^u \end{bmatrix} \tag{3.24}$$

$$G_d^u = -(\overline{K^yC})G_d^x \tag{3.25}$$

$$G_n^u = -(\overline{K^yC})G_n^x + \overline{K^y} \tag{3.26}$$

The finite horizon objective function $J = x'\bar{Q}x + u'\bar{R}u$, with $\bar{Q}$ and $\bar{R}$ given by equations (3.13)-(3.14), can be written as

$$\begin{aligned} J(x,u) &= x'\bar{Q}x + u'\bar{R}u \\ &= \tilde{d}'G_{\tilde{d}}^{x\prime}\bar{Q}G_{\tilde{d}}^{x}\tilde{d} + \tilde{d}'G_{\tilde{d}}^{u\prime}\bar{R}G_{\tilde{d}}^{u}\tilde{d}' \\ &= \tilde{d}'(G_{\tilde{d}}^{x\prime}\bar{Q}G_{\tilde{d}}^{x} + G_{\tilde{d}}^{u\prime}\bar{R}G_{\tilde{d}}^{u})\tilde{d} \end{aligned}$$

The problem of finding a static output feedback $u_k = -K^y y_k$ that minimizes the *expected value* of the objective J in equation (3.4) can now be analyzed by first defining

$$J(K^y, \tilde{d}) = z'z \tag{3.27}$$

with

$$z = G^{1/2}\tilde{d} \tag{3.28}$$

$$G = G_{\tilde{d}}^{x\prime}\bar{Q}G_{\tilde{d}}^{x} + G_{\tilde{d}}^{u\prime}\bar{R}G_{\tilde{d}}^{u} \tag{3.29}$$

Let

$$Z = \begin{bmatrix} \mathrm{diag}(W) & \\ & \mathrm{diag}(V) \end{bmatrix} \tag{3.30}$$

be the covariance matrix for the random variable $\tilde{d} = \left[\begin{smallmatrix} d \\ n \end{smallmatrix}\right]$. Now, by the same derivations that are used in Chapter 2, page 16, we can show that the expected value of the cost function for a given value of K^y is:

$$E\{J\} = \|G^{1/2}Z^{1/2}\|_F^2. \tag{3.31}$$

In order to synthesize a static output feedback controller one could use a nonlinear search to minimize the norm in equation (3.31), but this problem is expected to be NP-hard [Blondel and Tsitsikilis, 1997].

3.3 Minimization of expected loss

In this section, we first analyse the problem of minimizing the expected value of the loss

$$L(u,d,n) = J(u,d,n) - J_{\mathrm{opt}}(d) \tag{3.32}$$

where u is the result of a given static output feedback law. Then we derive a convex approximation for the problem of finding an optimal static output feedback.

3.3.1 Analysis

Similar to Chapter 2, section 2.3.1, we consider an augmented linear model

$$\tilde{y} = \begin{bmatrix} u \\ y \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ G_u^y \end{bmatrix}}_{G^{\tilde{y}}} u + \underbrace{\begin{bmatrix} 0 \\ G_d^y \end{bmatrix}}_{G_d^{\tilde{y}}} \bar{d} + \begin{bmatrix} 0 \\ W_n \end{bmatrix} \bar{n} \tag{3.33}$$

where

$$G_u^y = \mathrm{diag}(C) G_u^x, \quad G_d^y = \mathrm{diag}(C) G_d^x, \tag{3.34}$$

and G_u^x, G_d^x are defined by equation (3.8). Here $\bar{d}$ and $\bar{n}$ are normally distributed random variables with

$$W_d = \mathrm{diag}(W^{1/2}), \quad W_n = \mathrm{diag}(V^{1/2}), \tag{3.35}$$

such that the equalities

$$d = W_d \bar{d}, \quad n = W_n \bar{n} \tag{3.36}$$

hold. The static output feedback constraint $u = -\mathrm{diag}(K^y) y$ can be written as

$$\begin{bmatrix} I & \mathrm{diag}(K^y) \end{bmatrix} \tilde{y} = 0. \tag{3.37}$$

Halvorsen et al. [2003] derived an expression for the loss $L(u, \bar{d}, \bar{n}) = J(u, \bar{d}, \bar{n}) - J_{\mathrm{opt}}(\bar{d})$ where u is used to fulfill the static output feedback constraint (3.37):

$$L(u, \bar{d}, \bar{n}) = J(u, \bar{d}, \bar{n}) - J_{\mathrm{opt}}(\bar{d}) = \frac{1}{2} z' z, \tag{3.38}$$

where

$$z = -X H \tilde{F} \begin{bmatrix} \bar{d} \\ \bar{n} \end{bmatrix} \tag{3.39}$$

$$X = J_{uu}^{1/2} (H G^{\tilde{y}})^{-1} \tag{3.40}$$

$$\tilde{F} = \begin{bmatrix} F W_d & W_n \end{bmatrix} \tag{3.41}$$

$$F = -(G^{\tilde{y}} J_{uu}^{-1} J_{ud} - G_d^{\tilde{y}}) \tag{3.42}$$

where J_{uu}, J_{ud} are given by (3.10)-(3.11).

By following the derivation in Chapter 2, section 2.2.3, we find that the expected value of the loss is

$$E\{L\} = \frac{1}{2} \| J_{uu}^{1/2} (H G^y)^{-1} H \tilde{F} \|_F^2. \tag{3.43}$$

This expression is also derived by Kariwala et al. [2008].

3.3.2 Solution

In order to minimize the expected loss, one could solve the following non-convex problem

$$\min_H \; E\{L\} = \|XH\tilde{F}\|_F^2 \tag{3.44}$$

$$\text{s.t. } X = J_{uu}^{1/2}(HG^{\tilde{y}})^{-1} \tag{3.45}$$

$$\tilde{F} = \begin{bmatrix} FW_d & W_n \end{bmatrix} \tag{3.46}$$

$$H \text{ on the form } H = \begin{bmatrix} I & \text{diag}(K^y, \cdots, K^y) \end{bmatrix} \tag{3.47}$$

but this is expected to be as difficult as the original static output feedback problem. We therefore, by the same arguments as in Chapter 2, section 2.3.2, propose to solve the following *convex* program which gives an upper bound on the best possible static output feedback:

Definiton 3.1. *(Upper bound on noisy static output feedback) The following convex problem can be used to find an upper bound for the optimal static output feedback:*

$$\begin{aligned} &\min_H \|H\tilde{F}\|_F^2 \\ &\text{s.t. } H \text{ on the form } H = \begin{bmatrix} I & \text{diag}(K^y, \cdots, K^y) \end{bmatrix}, \end{aligned} \tag{3.48}$$

where $\tilde{F} = \begin{bmatrix} FW_d & W_n \end{bmatrix}$ *and* F *is given by* (3.41).

This problem can be vectorized and solved as a quadratic program.

Remark 3.3. *Note that this problem differs from the original static output feedback problem in two ways: First, we are as in Chapter 2 minimizing the loss from an optimal controller, and this controller happens to be non-causal with the formulation used in this chapter. Second, we neglect the term* $X = J_{uu}^{1/2}(HG^y)^{-1}$ *when minimizing* $\|XH\tilde{F}\|$*, which gives a convex problem, but the resulting* H *is then suboptimal.*

3.4 Example

Consider proportional (P) control of the discrete plant

$$x_{k+1} = 0.5x_k + u_k + d_k, \tag{3.49}$$

$$y_k = x_k + n_k, \tag{3.50}$$

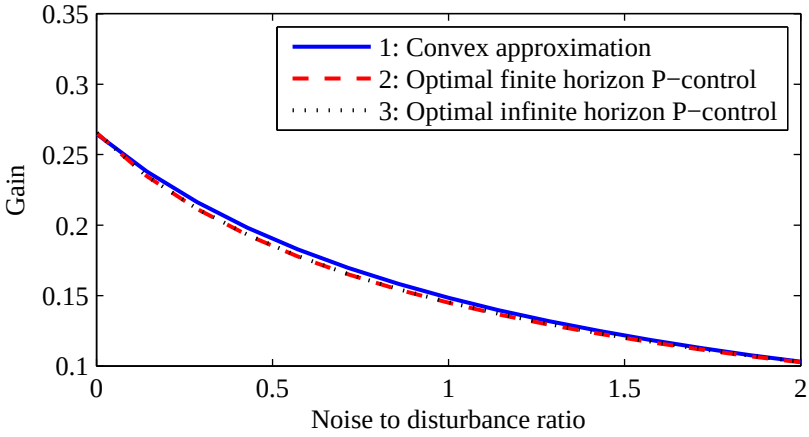

Figure 3.1: Gains for example of noisy process.

where d_k is a normally distributed variable with zero mean and unit variance. We want to study the effect of changing the variance V for the normally distributed noise n_k (which also has zero mean) on the resulting proportional gains.

We use a quadratic objective function to describe the trade-off between robustness and stability of the resulting controller, and we assume that the weights $(Q,R) = (1,1)$ give a good trade-off. We use an prediction length $N = 20$. By using the method outlined in Remark 2.1, Chapter 2, we calculate the final weight matrix as $Q_N = 1.13$.

Figure 3.1 shows the resulting gains for the following design methods:

1: Convex approximation. Design a P-controller by solving the convex problem (3.48).

2: Optimal finite horizon P-control. Found by minimizing the norm

$$\|G^{1/2}Z^{1/2}\|_F$$

in equation (3.31).

3: Optimal infinite horizon P-control. Found by minimizing the $\mathcal{H}_2$-norm of the plant found by taking the lower fractional transform of the plant P in equation (3.17) with the P-controller.

It is clear from the figure that the gain resulting from the convex approximation is similar to both optimal controllers for the range of noise to disturbance ratios studied.

3.5 Conclusion

In this chapter we have formally extended the results of Chapter 2 to include noisy measurements. As in Chapter 2, we minimize the expected loss to an optimal controller, which unfortunately is non-causal for the noisy case because of the need to include future disturbances to avoid that the noise dominates and makes it optimal to use $K^y = 0$ as the optimal controller. Nevertheless, the method seems interesting also for the case of noisy measurements, and we hope to conduct larger case studies in the future.

Chapter 4

Additional results regarding the static output feedback problem

In this chapter we review implementation of the first move of a series of open loop optimal inputs, as is done in moving horizon control. We show that for the case of full state information, there exist *linear* relationships between the states and the inputs such that when added to the original open loop problem, the solution does not change. We call these relationships "invariants". For state feedback, one invariant is $u_0 = -K_0 x_0$, i.e. that the first move can be written as a linear function of the initial state x_0, and that K_0 is *not* a function of x_0. We then show that the same does not hold for static output feedback, and consequently that the moving horizon idea does not work very well for static output feedback.

We further give a rule for how the optimal gains varies with increasing noise in the measurements. The rule is not exact due to the same reasons as for static output feedback, namely that the result cannot in general be implemented as a moving horizon controller. However, we show by an example that it gives reasonable approximations to the optimal controller.

4.1 Introduction

In this Chapter we discuss some additional results that we found when exploiting the link between self-optimizing control and static output feedback (that lead to the results in Chapters 2-3). First, we prove the existence of a state feedback implementation to the open loop problem normally posed in model predictive control: "Find a sequence of inputs u such that a linear system is regulated back to the origin while minimizing quadratic penalties on both inputs and outputs." Second, we discuss how this insight may be used to make an approximation to the static output feedback problem, but that implementing only the first part of the corresponding

input sequence does not work as good as for state feedback.

Finally we derive a short-cut method for how one should change the gain if one has designed a state feedback controller based on noise-free measurements and one wants to apply this controller to a plant where there are noisy state measurement available. The controller is not optimal, since we, as for noise-free state feedback, only implement the first part of a sequence of inputs, but it has the advantage of being very easy to compute, and may be used as initial condition for further nonlinear optimization.

4.2 Finite horizon linear quadratic regulator

Let us again look at the problem of regulating the linear system $x_{k+1} = Ax_k + Bu_k$ from an initial state x_0 to the origin when we have full information about the states available for feedback. As shown in section 2.2.1 in Chapter 2, this problem can be written on the form

$$J_{\text{opt}}(x_0) = \min_u \begin{bmatrix} u \\ x_0 \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ux_0} \\ J'_{ux_0} & J_{x_0x_0} \end{bmatrix} \begin{bmatrix} u \\ x_0 \end{bmatrix}, \tag{4.1}$$

where the expressions for the matrices in the cost function can be found in Chapter 2. By using the method of completing the squares [Åström and Wittenmark, 1984], the optimal input can be written on the form $u = -J_{uu}^{-1} J_{ux_0} x_0$. Interestingly, the same result can be found by using the following Theorem:

Theorem 4.1. *(Linear invariants for quadratic optimization problem [Alstad et al., 2009]) Consider an unconstrained quadratic optimization problem in the variables u (input vector of length n_u) and d (disturbance vector of length n_d)*

$$\min_u J(u,d) = \begin{bmatrix} u \\ d \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ud} \\ J'_{ud} & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix}. \tag{4.2}$$

In addition, there are "measurement" variables $y = G^y u + G^y_d d$. If there exists $n_y \geq n_u + n_d$ independent measurements (where "independent" means that the matrix $\tilde{G}^y = \begin{bmatrix} G^y & G^y_d \end{bmatrix}$ has full rank), then the optimal solution to (4.2) *has the property that there exists $n_c = n_u$ linear variable combinations*

$$c = Hy \tag{4.3}$$

that are invariant to the disturbances d, meaning that their optimal value ($c = 0$) is independent of d. The corresponding measurement combination matrix H can be found by selecting H such that

$$HF = 0, \tag{4.4}$$

That is, H is in the left nullspace of F, where $F = \frac{\partial y^{\text{opt}}}{\partial d}$ is the optimal sensitivity matrix which can be obtained from

$$F = -(G^y J_{uu}^{-1} J_{ud} - G_d^y). \tag{4.5}$$

In the notion of Theorem 4.1, let the measurement variables be

$$y = \begin{bmatrix} u \\ x_0 \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ 0 \end{bmatrix}}_{G^y} u + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{G_d^y} x_0, \tag{4.6}$$

where u is a vector of stacked inputs $(u_0, u_1, \cdots, u_{N-1})$. Further, we assume that all disturbances can be represented by the initial state x_0, hence we treat the disturbance d in Theorem 4.1 as the initial state x_0. We observe that the matrix $G^{\tilde{y}} = \begin{bmatrix} G^y & G_d^y \end{bmatrix}$ has full rank, so by Theorem 4.1 there exists a variable combination $c = Hy$ that is optimally invariant to the disturbances x_0 (which implies that we can add the constraint $c = Hy = 0$ to the original problem (4.1) and the solution will not change).

Let us find this measurement combination by the method outlined in Theorem 4.1: First, we form the optimal sensitivity matrix F:

$$F = -(G^y J_{uu}^{-1} J_{ux_0} - G_d^y) = -\left(\begin{bmatrix} J_{uu}^{-1} J_{ux_0} \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ I \end{bmatrix} \right) = \begin{bmatrix} -J_{uu}^{-1} J_{ux_0} \\ I \end{bmatrix}. \tag{4.7}$$

Second, let $H = \begin{bmatrix} H^u & H^{x_0} \end{bmatrix}$. In order to avoid a trivial solution H should have full column rank, however there is some degree of freedom in choosing such an H, because for a given H such that $HF = 0$ we also have that for $H_1 = DH$ with D square and full rank, that $H_1 F = 0$. This means that we can specify some parts of H, for example $H^u = I$ (This corresponds to normalizing the vectors that span the left nullspace of F.) With this choice of H^u, we have that

$$\begin{aligned} HF &= \begin{bmatrix} I & H^{x_0} \end{bmatrix} \begin{bmatrix} -J_{uu}^{-1} J_{ux_0} \\ I \end{bmatrix} = -J_{uu}^{-1} J_{ux_0} + H^{x_0} = 0 \\ &\Updownarrow \\ H^{x_0} &= J_{uu}^{-1} J_{ux_0} \end{aligned}$$

This means that the optimal variable combination is

$$Hy = 0 \Leftrightarrow \begin{bmatrix} I & J_{uu}^{-1} J_{ux_0} \end{bmatrix} \begin{bmatrix} u \\ x_0 \end{bmatrix} = 0 \Leftrightarrow u = -J_{uu}^{-1} J_{ux_0} x_0 \tag{4.8}$$

as expected. Let us analyze this solution. By setting $K = -J_{uu}^{-1} J_{ux_0}$ we have that

$$\begin{aligned} u_0 &= K_0 x_0, \\ u_1 &= K_1 x_0, \\ &\vdots \\ u_{N-1} &= K_{N-1} x_0, \end{aligned}$$

where $K' = \begin{bmatrix} K_0' & K_1' & \cdots & K_{N-1}' \end{bmatrix}$. We notice that neither of these gains are a function of the initial state x_0, only of the problem parameters J_{uu} and J_{ux_0}, and therefore can we use this solution in a moving horizon strategy by implementing only the first move $u_k = K_0 x_k$ as we get new state measurements (x_k). This is further discussed in Chapter 2, section 2.2.1.

We finally notice that K_0 can also be found from a Riccatti equation that can be derived from the structure of J_{uu} and J_{ud}, i.e. *without* forming the inverse of J_{uu} [Rawlings and Muske, 1993].

For long horizons N we further have that

$$u_1 = K_1 x_0 = \underbrace{K_1 (A + BK_0)^{-1}}_{=K_0} x_1 = K_0 x_1, \tag{4.9}$$

$$u_2 = K_2 x_0 = \underbrace{K_2 (A^2 + ABK_0 + BK_1)^{-1}}_{K_0} x_2 = K_0 x_2 \tag{4.10}$$

$$\begin{aligned} u_3 &= K_3 x_0 = \cdots \\ &\vdots \end{aligned}$$

that is, the gain remains constant also throughout the horizon, however this is not the case for short horizons. (I.e. first move implementation coincides with the open loop solution.)

Remark 4.1. *It may not be so surprising that Theorem 4.1 gives the same result as completing the squares, if we got some other result then we would be doing something wrong. Theorem 4.1 may however be used in a more general manner, for example we can let a measurement be $x_1 = Ax_0 + Bu_0$ and use the Theorem to prove that there must exists an optimal* linear *relationship between u_1 and x_1. The linear model is for then*

$$y = \begin{bmatrix} u \\ x_1 \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ \hat{B} \end{bmatrix}}_{G^y} u + \underbrace{\begin{bmatrix} 0 \\ A \end{bmatrix}}_{G_d^y} x_0, \tag{4.11}$$

where

$$\hat{B} = \begin{bmatrix} B & 0 & \cdots & 0 \end{bmatrix}. \tag{4.12}$$

The augmented plant $\tilde{G}^y = \begin{bmatrix} G^y & G^y_d \end{bmatrix}$ *has full rank in this case (given that A has full rank), so Theorem 4.1 holds, and may be used to find an H such that when the constraint* $c = Hy = 0$ *is added to problem* (4.1) *the solution will not change.*

For a problem with relatively long input horizon, this linear relationship should correspond to K_0 *above. Actually, we can use Theorem 4.1 to prove many such optimal linear relationships that when added as constraints to the original problem does not change the solution. The reason we considered* x_0 *as a measurement above it that this is the usual assumed available measurement in receding horizon control.*

4.3 First move output feedback

Let us slightly change the problem formulation from state feedback as in the previous section to output feedback, where an output $y_0 = Cx_0$ is available, but not the full state vector. Following the notation of Theorem 4.1 we consider the augmented measurement vector as

$$y = \begin{bmatrix} u \\ y_0 \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ 0 \end{bmatrix}}_{G^y} u + \underbrace{\begin{bmatrix} 0 \\ C \end{bmatrix}}_{G^y_d} x_0 \tag{4.13}$$

Since the dimension of the *disturbance* x_0 is larger than the dimension of the output y_0 is not the assumption that "there exists $n_y > n_u + n_d$ measurements" fulfilled, and Theorem 4.1 cannot be applied to this case. However, Theorem 2.1 in Chapter 2 can be applied. As above, let

$$H = \begin{bmatrix} H^u & H^{y_0} \end{bmatrix} = \begin{bmatrix} I & H^{y_0} \end{bmatrix}, \tag{4.14}$$

where H^{y_0} is a full matrix. With this normalization of H we have that

$$HG^y = \begin{bmatrix} I & H^{y_0} \end{bmatrix} \begin{bmatrix} I \\ 0 \end{bmatrix} = I,$$

which is always full rank, so finding the H that minimizes the expected loss from adding the constraint $Hy = 0$ to problem (4.1) is given by the H that minimizes the norm

$$\|HFW_d\|_F, \tag{4.15}$$

where as in Chapter 2 is $W_d = W_{x_0}^{1/2}$ the square of the variance of x_0. Note that we are now trying to combine the input vector u with the output y_0 on a form such that

when a constraint $Hy = 0$ is added to the original problem the loss from optimality is minimized. Thereafter, the goal is to implement only the first part of the input vector in a moving horizon strategy, as was the case for the state feedback above.

Remark 4.2. *This is not the same as what we did in Chapters 2-3, where we explicitly added the constraint $u_k = -K^y y_k$. That approach is different in two ways; first we are requiring that K^y is constant throughout the horizon and second, since all output y_k are included in the vector of "measurements" (in $\tilde{y}$) we had to add* structural constraints *in the form of zeros in the H matrix, such that one u_k is only a function of one y_k (and not for example y_{k-1} and y_{k+1}).*

Let us analyse the resulting problem. First, the sensitivity matrix F is

$$F = -(G^y J_{uu}^{-1} J_{ux_0} - G_d^y) = \begin{bmatrix} -J_{uu}^{-1} J_{ux_0} \\ C \end{bmatrix}, \tag{4.16}$$

and therefore

$$\|HFW_d\|_F = \|(-J_{uu}^{-1} J_{ux_0} + H^{y_0} C) W_d\|_F. \tag{4.17}$$

We can find an H that minimizes an upper bound to this expression by choosing

$$H^{y_0} = J_{uu}^{-1} J_{ux_0} C^\dagger, \tag{4.18}$$

so that

$$\begin{aligned} \|HFW_d\|_F &\leq \|(-J_{uu}^{-1} J_{ux_0} + J_{uu}^{-1} J_{ux_0} C^\dagger C)\|_F \\ &\leq \|J_{uu}^{-1} J_{ux_0}\|_F \cdot \|I - C^\dagger C\|_F \cdot \|W_d\|_F \end{aligned} \tag{4.19}$$

Clearly, if C has full row rank, then will the choice of H in (4.16) make the expected loss from optimality zero, which is expected as we regain the state feedback problem in this case.

The resulting sequence of inputs is thus

$$\begin{aligned} u_0 &= K_0 C^\dagger y_0 \\ u_1 &= K_1 C^\dagger y_0 \\ &\vdots \\ u_{N-1} &= K_{N-1} C^\dagger y_0, \end{aligned}$$

where K is the same gain as for the case of state feedback in the previous section.

Unfortunately, we find that the first input of this sequence is not well suited in a moving horizon strategy. For the case of state feedback the controller has full information about at the disturbances, and then the implementation $u_k = K_0 x_k$ is

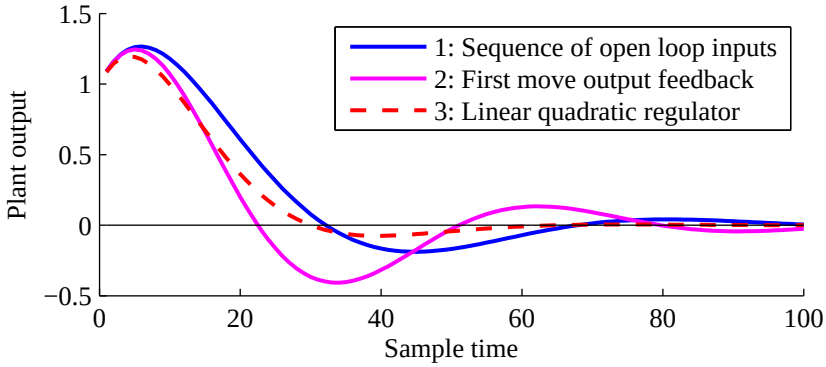

Figure 4.1: Comparison of open and closed loop implementations of the output feedback design method termed "first-move". The simulation starts from $x_0 = (1,1)$, with output $y_0 = 1$.

valid. However, the corresponding output feedback $u_k = K_0 C^\dagger y_k$ does not benefit from full information about the disturbances, and thus it should not be "reset" at every time-step. Moreover, the upper bound on the loss in equation (4.19) assumes that the whole sequence of u's is implemented, and it does not hold for the case of implementing only the first move in a moving horizon strategy. This is further illustrated in the following example section.

4.3.1 Examples

Open loop versus first move as closed loop. Consider the process $g(s) = 1/(s^2 + 2 \cdot 0.4s + 1)$, sampled with $T_s = 0.1$ to get

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 0.9951 & 0.0959 \\ -0.0959 & 0.9184 \end{bmatrix} x_k + \begin{bmatrix} 0.0049 \\ 0.0959 \end{bmatrix} u_k \\ y_k &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_k. \end{aligned} \tag{4.20}$$

We use a finite horizon quadratic objective function with $Q = I$, $R = 1$, and $N = 100$. For such a long input horizon will the final weight matrix not influence the solution, so we set $Q_N = I$ as well. We compare three different controllers:

1: Sequence of open-loop inputs. The inputs are calculated for the first value of the output, $u = -J_{uu}^{-1} J_{ux_0} C^\dagger y_0$, and then we implement these inputs *without taking any feedback into account*.

2: First move output feedback. This controller uses the first part of $J_{uu}^{-1} J_{ux_0} C^\dagger$ as a feedback controller, $u_k = K_0 C^\dagger y_k$, with $K_0 C^\dagger = -0.3608$.

3: Linear quadratic controller. State feedback controller as discussed in Section 4.2, $u_k = K_0 x_k$, with $K_0 = -\begin{bmatrix}0.3608 & 0.7309\end{bmatrix}$.

Results for an initial step disturbance in the states of $x_0 = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$ is shown in Figure 4.1. It is evident that the open loop inputs do not coincide with the first move output feedback (as would be the case for state feedback if the input horizon is long enough), and moreover we observe that the open-loop controller performs better than the first move controller.

This simulation is of course a function of the initial state, for disturbances that can be seen "directly" on the outputs, for example a disturbance on the first state for this example, we will get that the sequence of open loop inputs and the LQR implementation are coinciding. However, for disturbances that does not affect the initial output y_0, the open loop sequence of inputs will be kept to zero, and the first-move implementation will be better. Nevertheless, in the average case we found that the open loop sequence was performing better than using only the first move as a feedback implementation, and this is illustrated by this example.

Comparison with controllers from Chapter 2. In Chapter 2 we investigated control of the underdamped plant

$$g(s) = \frac{k}{\tau^2 s^2 + 2\tau\zeta s + 1}$$

for different values of the damping coefficient ζ. For the same design parameters as in Chapter 2 we designed a first move output feedback controller based on the first part of $-J_{uu}^{-1} J_{ux_0} C^{\dagger}$. The resulting closed loop gains are shown in Figure 4.2, were we observe that the closed loop gain using the first move output feedback controller is significantly higher than for the other design methods for the highly underdamped plants, and that for plants with a damping factor ζ less than 0.28 did the first move output feedback actually destabilize the plant.

4.3.2 Discussion

The first move output feedback as derived in this chapter is easy to compute, as we simply use the pseudo inverse of the measurement matrix C, so no optimization problem needs to be solved. This controller has the nice property that it for a full rank measurement matrix C coincides with the finite horizon linear quadratic regulator. However, when C is not full rank we observe that implementing only the first move of a sequence of "open loop optimal" inputs does not work very well. This may be explained in two ways; First, we are not solving a feedback problem in the original problem, see equation (4.1), so even though the sequence of outputs u may give a good transient performance, we are not guaranteed that implementing

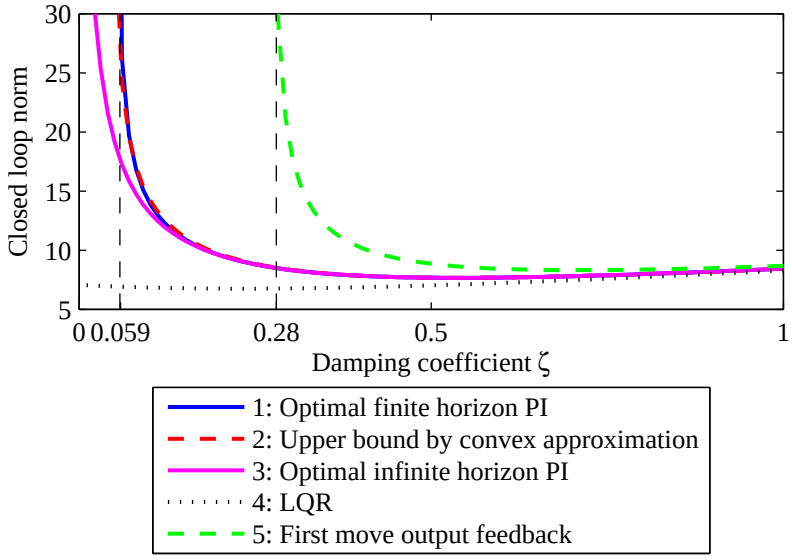

Figure 4.2: Closed loop norms $\|F_l(P,K^y)\|_2$ for different output feedback (PI) controllers

only the first part of this sequence gives a good closed-loop behaviour. Second, for state feedback, when we have full information about the disturbances, the problem can be restarted at every time sample, and it therefore makes sense to implement only the first move in a moving horizon strategy. However, for output feedback do we not have full information, and we would need some steps in order to estimate the states, but the problem is not set up to handle this. On the other hand, the static output feedback controllers that we synthesise in Chapters 2-3 are set up as feedback problems to begin with and do not suffer from the problem that we are trying to implement parts of an open loop solution as a feedback. (But we do the approximation of ignoring $X = J_{uu}^{1/2}(HG^y)^{-1}$ when minimizing $\|XHFW_d\|_F$.)

In appendices B and C is it discussed how we can use the ideas presented in this paper to find controllers which uses past and present information about the outputs. The performance of these controllers are better than using only present information, but they suffer from the same problem as discussed in this chapter, namely that they are implementations of a part of an open loop sequence. One could also use the framework presented in Chapters 2-3 to synthesise controllers on this form, with an expected better closed-loop performance, since we are solving a feedback problem. These results are not discussed further here because we now want the focus to be on the *fundamental properties* of the controllers derived the

current chapter and Chapters 2-3.

4.4 Gain reduction rule

We here consider the special case of the following plant

$$x_{k+1} = Ax_k + Bu_k, \tag{4.21}$$

$$y_k = x_k + n_k, \tag{4.22}$$

$$x_0 \backsim N(0, I), \tag{4.23}$$

where n_k is a normally distributed random variable with zero mean and variance $W = \alpha I$. The objective is to find a static output feedback $u_k = K^y y_k$ and to relate this controller to the corresponding linear quadratic controller for the same system. The controller should minimize some finite horizon objective function (see e.g. equation (2.6), Chapter 2):

$$J = x_N' Q_N x_N + \sum_{i=0}^{N-1} x_i' Q x_i + u_i' R u_i.$$

Consider now an augmented "measurement" y of

$$y = \begin{bmatrix} y_0 \\ u \end{bmatrix} = \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{G^y} u + \underbrace{\begin{bmatrix} C \\ 0 \end{bmatrix}}_{G_d^y} x_0$$

Let J_{uu} and J_{ux_0} be defined as in Chapter 2, Section 2.2.1. By the results from Chapters 2-3 we be pose the problem of finding a combination matrix $H = \begin{bmatrix} H^{y_0} & H_u \end{bmatrix}$ that minimizes the expected loss from an finite horizon linear quadratic controller when the constraint $c = Hy = 0$ is added to the problem as

$$\min_H \|XH\tilde{F}\|_F \tag{4.24}$$

$$\text{s.t. } X = J_{uu}^{1/2}(HG^y)^{-1} \tag{4.25}$$

$$\tilde{F} = \begin{bmatrix} FW_d & W_n \end{bmatrix}, \tag{4.26}$$

where $W_d = I$ and $W_n = \sqrt{\alpha} I$. In this problem, the disturbance only occurs on the initial state and we do not have any structural constraints on H from the causality requirement. We can then actually solve problem (4.24)-(4.26) analytically by using an expression from [Alstad et al., 2009] and show that the first move can optimally be written as:

$$u_k = -\frac{1}{1+\alpha} K_0 x_k,$$

where K_0 is the first part of $J_{uu}^{-1} J_{ux_0}$.

Proof. Problem (4.24)-(4.26) can be solved analytically by [Alstad et al., 2009]:

$$H' = (\tilde{F}\tilde{F}')^{-1}G^y\left(G^{y\prime}(\tilde{F}\tilde{F}')^{-1}G^y\right)^{-1}J_{uu}^{1/2} \tag{4.27}$$

Let the combination matrix H be partitioned as $H = \begin{bmatrix} H^{y_0} & H^u \end{bmatrix}$ for which we have

$$G^y = \begin{bmatrix} 0 \\ I \end{bmatrix}, \quad G^{y_d} = \begin{bmatrix} I \\ 0 \end{bmatrix}, \tag{4.28}$$

and J_{uu} and J_{ux_0} are derived in Chapter 2, Section 2.2.1. Now, let

$$W_n = \begin{bmatrix} \sqrt{\alpha}I_{n_x} & \\ & \sqrt{\beta}I_{Nn_u} \end{bmatrix} \tag{4.29}$$

where the upper left part corresponds to the noise on x_0 and the lower left part corresponds to noise on u. (We will show that β does not affect the solution.) Define $Y = -J_{uu}^{-1}J_{ux_0}$. We have that $\tilde{F}\tilde{F}' = FW_dW_d'F' + W_nW_n'$. By the above assumptions we get that

$$F\underbrace{W_dW_d'}_{I}F' = FF' = \begin{bmatrix} I & Y' \\ Y & YY' \end{bmatrix} \tag{4.30}$$

Due to the assumptions on W_n we get

$$\begin{aligned} \tilde{F}\tilde{F}' &= FW_dW_d'F' + W_nW_n' = \\ &= \begin{bmatrix} (1+\alpha)I & Y' \\ Y & YY' + \beta I \end{bmatrix} \end{aligned} \tag{4.31}$$

This matrix has to be inverted. This can be done using Lemma A.2 (Inverse of a partitioned matrix) in Skogestad and Postlethwaite [2005], with $A_{11} = (1+\alpha)I$, $A_{12} = Y'$, $A_{21} = Y$, $A_{22} = YY' + \beta I$. Further we have $X = A_{22} - A_{21}A_{11}^{-1}A_{12} = \cdots = \left(\frac{\alpha}{1+\alpha}YY' + \beta I\right)$. We observe that the inverse of X exists. Using the Lemma, we get that the inverse of $\tilde{F}\tilde{F}'$ is:

$$\left(\tilde{F}\tilde{F}'\right)^{-1} = \begin{bmatrix} \frac{1}{1+\alpha}I + \frac{1}{(1+\alpha)^2}Y'X^{-1}Y & -\frac{1}{1+\alpha}Y'X^{-1} \\ -\frac{1}{1+\alpha}X^{-1}Y & X^{-1} \end{bmatrix} \tag{4.32}$$

We now need to evaluate $G^{y\prime}(\tilde{F}\tilde{F}')^{-1}G^y$. For the current problem formulation we have that $G^{y\prime} = \begin{bmatrix} 0 & I \end{bmatrix}$, and after doing the multiplication we get that

$$G^{y\prime}(\tilde{F}\tilde{F}')^{-1}G^y = X^{-1} \;\Rightarrow\; \left(G^{y\prime}\left(\tilde{F}\tilde{F}'\right)^{-1}G^y\right)^{-1} = X \tag{4.33}$$

Further,

$$\left(\tilde{F}\tilde{F}'\right)^{-1} G^y = \begin{bmatrix} -\frac{1}{1+\alpha} Y'X^{-1} \\ X^{-1} \end{bmatrix} \tag{4.34}$$

and finally we get that

$$H' = \left(\tilde{F}\tilde{F}'\right)^{-1} G^y \left(G^{y\prime} \left(\tilde{F}\tilde{F}'\right)^{-1} G^y\right)^{-1} J_{uu}^{1/2} \tag{4.35}$$

$$= \begin{bmatrix} \frac{1}{1+\alpha} (J_{uu}^{-1} J_{ux_0})' J_{uu}^{1/2} \\ J_{uu}^{1/2} \end{bmatrix}, \tag{4.36}$$

or

$$H = \begin{bmatrix} \frac{1}{1+\alpha} J_{uu}^{1/2} J_{uu}^{-1} J_{ux_0} & J_{uu}^{1/2} \end{bmatrix} \tag{4.37}$$

We now scale H matrix by $J_{uu}^{-1/2}$ to decouple the inputs and to get an expression for the controller gains:

$$(J_{uu}^{1/2})^{-1} H = \begin{bmatrix} \frac{1}{1+\alpha} J_{uu}^{-1} J_{ux_0} & I \end{bmatrix}, \tag{4.38}$$

and we observe that optimally we should reduce the controller gains by $1/(1+\alpha)$ when there is noise on the states with variance αI. To see this, remember that $y = (x,u)$, and hence we get c's on the form

$$c = Hy = \frac{1}{1+\alpha} J_{uu}^{-1} J_{ux_0} x_0 + u, \tag{4.39}$$

□

Note that we are not solving the original problem in terms of the cost J, but rather to find the output feedback which is closest to the optimal state feedback solution. What is more serious, is that we cannot apply the "moving horizon" argument because each move is no longer optimal. Thus simply implementing the first move is not optimal and the gains will vary, that is, we not not have $K_0 = K_1$ or $K_0 = K_2$ as is the case for state feedback, see equations (4.9)-(4.10).

4.4.1 Example

Figure 4.3 shows that the gain reduction rule used on the plant

$$x_{k+1} = Ax_k + Bu_k + d_k$$
$$y_k = x_k + n_k$$

studied in Chapter 3, Section 3.4, with the same controller weights as used in Chapter 3 is quite close to the gains found by the convex approximation, however this method is a lot simpler as no optimization problem needs to be solved (except for a Riccatti equation to find the state feedback gain) in other to calculate the gain.

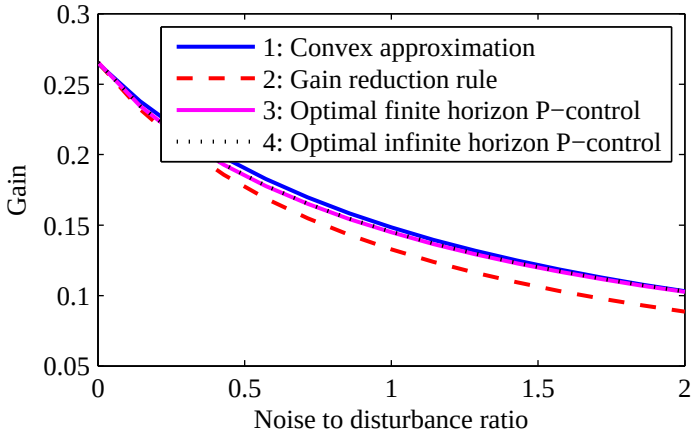

Figure 4.3: Gains for example of noise process

4.5 Conclusions

In this chapter we have discussed the existence of a state feedback on the form $u_k = K_0 x_k$ as an implementation to the typical problem formulation used in model predictive control, where one formulates an optimization problem to regulate a system from an initial state x_0 to the origin (or more general to some pre-defined trajectory). Further we showed that such a solution does not exist for output feedback, i.e. that using only the first part of an open loop optimal sequence is not very good for feedback control when the full state vector is not available for measurement.

Finally we derived a simple rule for reduction of the gain for the special case when a noisy state measurement is available for feedback, and one wants to use a static controller from this measurement in closed loop. The solution has the same problem as the “first move output feedback”, i.e. that we don’t solve a feedback problem, but it may still be interesting a fast way of calculating initial conditions for further nonlinear optimization.

Chapter 5

Self-optimizing control with active set changes

In this chapter we extend the "nullspace method" by Alstad and Skogestad [2007] to cover changes in active set. The extension in based on recent results from explicit model predictive control by Baotič et al. [2008]. The nullspace method is a method for selecting controlled variables, assuming that the set of active constraints does not change. With the extension presented here, we show that by applying the nullspace method for several different regions, where the regions are found by a parametric program, we can use the value of the controlled variables for each region to decide when to switch regions.

The proposed method is demonstrated on a simple model of an ammonia production plant, and the results are comparable to real-time optimization of the same plant.

5.1 Introduction

In this chapter we extend some recent results on implementation of quadratic programs [Alstad et al., 2009] to cover changes in the active set. The work is in the field of "self-optimizing control", where the focus is to select the right variables c to control, such that acceptable operation under all conditions is achieved with constant setpoints for the controlled variables [Skogestad, 2000a].

A more direct approach for ensuring optimal operation is real-time optimization (RTO) [Marlin and Hrymak, 1996]. Using RTO, the optimal values (setpoints) for the controlled variables c are computed online based on online measurements, and a model of the process [Alstad et al., 2009]. In control of chemical processes an hierarchal structure [Findeisen et al., 1980] is often used. RTO is then used to calculate setpoints c_s for the controlled variables c for the supervisory control

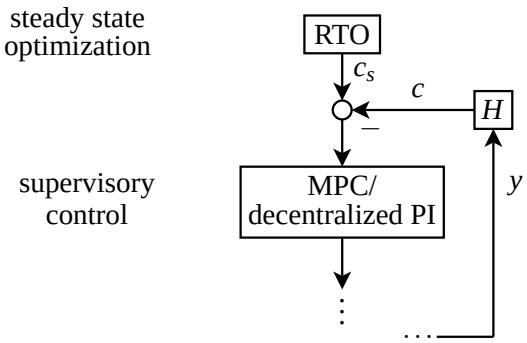

Figure 5.1: Interconnection between RTO and supervisory control layer.

layer. In the supervisory layer model predictive control (MPC) [Morari and Lee, 1999, Mayne et al., 2000] is often used.

A typical hierarchical structure is outlined in Figure 5.1. In the RTO framework, the degree of freedom H is not exploited as an optimization variable, while for "self-optimizing control" finding a good H is the main focus. The two approaches are therefore complementary. In general H can be any non-linear map from measurements y to controlled variables c, but until now the focus has been mostly on static maps, i.e. H is a *static combination matrix* or a *selection matrix*.

One can identify at least four ways of choosing the combination matrix H for the controlled variables $c = Hy$:

1. Use $c = u_0$, i.e. open loop control. This is not expected to work very well unless the static optimization layer is updated rapidly.

2. Use $c = y_0$, where y_0 are presently used controlled variables in the supervisory layer. Also this choice is not expected to give a good performance unless the static optimization problem is resolved frequently. In fact, it does not have any particular advantage over choosing $c = u_0$.

3. Use $c = Hy$, where y is all available measurements, including u_0 and y_0. If H is chosen carefully, this choice is expected to give better performance between samples of the RTO than the choices above. In particular, H should be chosen such that even though we have large disturbances, the optimal values c_s of the controlled variables c does not change much. Such a choice of H may be beneficial in at least two ways. First, since RTO is in general a non-convex problem, the starting values for the optimization are important, and thus if the optimal values do not change much, such a choice of c should aid the success of a RTO implementation. Second, c_s can be updated less

frequent and the system will work better should the RTO be out of service. In the ideal "self-optimizing" case one identifies controlled variables such that the RTO layer may be eliminated altogether. During the last decade several methods for finding "good" controlled variables have been developed, such as the maximum gain rule [Hori and Skogestad, 2008], exact local method [Halvorsen et al., 2003], and the nullspace method [Alstad and Skogestad, 2007].

4. Even more general; we may choose to change the controlled variables c as operating conditions change. This is equivalent to letting the map H be a function of the operating conditions.

In this chapter we consider the last approach and develop a method for changing the combination matrix H when changes in the active set occur. The results are exact for quadratic problems, but an example of an ammonia production plant will show that the method may be applicable also to more general processes by local linearization.

The approach we use is to exploit a link between self-optimizing control and linear-quadratic explicit MPC [Manum et al., 2008b]. The link is exact for quadratic approximations of the self-optimizing control problem, because then the static optimal operation problem of self-optimizing control and explicit MPC have the same equation structure.

Using parametric programming [Kvasnica et al., 2004] and recent results from explicit MPC [Baotiċ et al., 2008] on implementation of the optimal solution, we show that combination matrices H^i, found by using the nullspace method, can be used to track changes in the active set using only information about the outputs. In the multivariable case, a scalar function of the outputs is enough to track changes in the active set. We have already proposed similar results earlier, see [Manum et al., 2008b,c], where we used controlled variables (invariants) from the nullspace method to track changes in the optimal active set. However, we proposed to keep track of the whole vector $c = Hy \in \mathbb{R}^{n_c}$, where n_c is the number of controlled variables, and not the considerably simpler method of tracking $m'Hy \in \mathbb{R}$.

The rest of the chapter is organized as follows: First, we review theory from self-optimizing control and implementation of solutions to quadratic optimization problems using *descriptor functions*. Then we show how continuous piecewise-affine (PWA) descriptor functions from measurements can be constructed by using the *nullspace method*. We then discuss how to match constraints between measured constraints and constraints in the model. This *constraint matching* may in some cases have a significant effect on the economical operation of a given plant. Thereafter we collect our findings in an algorithm for design of a control structure that handles changes in the active set, and finally we show how this method can be

used on an example of an ammonia production plant.

5.2 Background

5.2.1 Quadratic approximation to RTO

We consider the problem

$$\begin{aligned} \min_{x,u_0} \; & J_0(x,u_0,d_0) \\ \text{s.t. } & f_0(x,u_0,d_0)=0 \\ & g_0(x,u_0,d_0)\geq 0 \end{aligned} \tag{5.1}$$

where $x\in\mathbb{R}^{n_x}$ are states, $u_0\in\mathbb{R}^{n_{u_0}}$ are steady state degrees of freedom and $d_0\in\mathbb{R}^{n_{d_0}}$ are disturbances. Using the model equations $f_0(x,u_0,d_0)=0$ to formally eliminate the internal state x, we can rewrite problem (5.1) on the form

$$\begin{aligned} \min_{u_0} \; & J(u_0,d_0) \\ \text{s.t. } & g(u_0,d_0)\geq 0 \end{aligned} \tag{5.2}$$

Unconstrained case. Assume that for the nominal disturbance $\bar{d_0}$ the optimal input u_0^* is such that none of the inequality constraints $g(u_0^*,\bar{d_0})\geq 0$ are exactly equal to zero (i.e. they are not active). Further, introduce the following substitutions:

$$u = u_0 - u_0^* \tag{5.3}$$
$$d = d_0 - \bar{d_0} \tag{5.4}$$
$$M_u = -\nabla_{u_0} g|_{u_0^*,\bar{d_0}} \tag{5.5}$$
$$M_d = \nabla_{d_0} g|_{u_0^*,\bar{d_0}} \tag{5.6}$$
$$M = g(u_0^*,\bar{d_0}) \tag{5.7}$$

By a quadratic expansion of the objective function around the nominal optimum ($\nabla J(u_0^*,\bar{d_0})=0$) can we give the following quadratic approximation to problem (5.1), which we will use throughout the chapter:

$$\begin{aligned} \min_{u} \; & \frac{1}{2}\begin{bmatrix} u \\ d \end{bmatrix}\begin{bmatrix} J_{uu} & J_{ud} \\ J_{ud}' & J_{dd} \end{bmatrix}\begin{bmatrix} u \\ d \end{bmatrix} \\ \text{s.t. } & M_u u \leq M + M_d d \end{aligned} \tag{5.8}$$

Here the notation J_{uu} means the second derivative of the matrix J with respect to the inputs u, and so on.

Extension to constrained case. The case of nominally constrained optimum can also be posed on the form of problem (5.8), which we will now demonstrate. The only difference from the unconstrained case is that we do a change of variables to "eliminate" the effect of a non-zero first derivate J_u at the optimum.

First we define a Lagrangian as:

$$\mathcal{L}(u_0, d_0, \lambda) = J(u_0, d_0) - \lambda' g(u_0, d_0) \tag{5.9}$$

where λ are the Lagrange multipliers. Then we make a quadratic approximation of the nonlinear program (5.2) around the optimal point (u_0^*, λ^*) as [Nocedal and Wright, 1999, Ying and Joseph, 1999]:

$$\begin{aligned} \min_{u_0} \; & \underbrace{\nabla_{u_0} J|_{u_0^*, \bar{d_0}}}_{J_u} (u_0 - u_0^*) + \frac{1}{2} \begin{bmatrix} u_0 - u_0^* \\ d_0 - \bar{d_0} \end{bmatrix}' \begin{bmatrix} \mathcal{L}_{u_0 u_0} & \mathcal{L}_{u_0 d_0} \\ \mathcal{L}'_{u_0 d_0} & \mathcal{L}_{d_0 d_0} \end{bmatrix} \begin{bmatrix} u_0 - u_0^* \\ d_0 - \bar{d_0} \end{bmatrix} \\ & \text{s.t. } g(u_0^*, \bar{d_0}) + \nabla_{\left[\begin{smallmatrix} u_0 \\ d_0 \end{smallmatrix}\right]} g|_{u_0^*, \bar{d_0}} \begin{bmatrix} u_0 - u_0^* \\ d_0 - \bar{d_0} \end{bmatrix} \geq 0 \end{aligned} \tag{5.10}$$

where we have cancelled the term $\nabla J_{d_0}(d_0 - d_0^*)$ which can not be affected by the degrees of freedom u_0. All first and second derivatives are evaluated at the nominal optimum, (u^*, d_0). Under the assumption that $\mathcal{L}_{u_0 u_0}$ is positive definite (second-order optimality conditions) we introduce the following change of variables for the degrees of freedom u_0:

$$u = u_0 - u_0^* + \mathcal{L}_{u_0 u_0}^{-1} J_u \tag{5.11}$$

Note that this definition of u is not in conflict with definition (5.3) used for the unconstrained case, because for an unconstrained minimum $J_u = 0$ and the two definitions coincide. Now, by *defining* J_{uu}, J_{ud} and J_{dd} as

$$\left[\begin{array}{c|c} J_{uu} & J_{ud} \\ \hline J'_{ud} & J_{dd} \end{array}\right] \triangleq \left[\begin{array}{c|c} \mathcal{L}_{uu} & \mathcal{L}_{ud} \\ \hline \mathcal{L}'_{ud} & \mathcal{L}_{dd} \end{array}\right] \tag{5.12}$$

can also the nominally constrained case be written exactly on the form of problem (5.8). Note again the analogy to the unconstrained case: for the unconstrained optimum is the Hessian of the quadratic approximation equal to the Hessian of the objective function at the nominal operating point, while for the constrained case is the Hessian of the quadratic approximation equal to the Hessian of the Lagrange function of the original problem at the nominal point. Further note that the unconstrained case is a special case of the constrained case and is included here only to ease the presentation of the material, and because in the example we consider in this chapter the nominal optimum happens to be unconstrained.

Remark 5.1. *The matrix J_{dd} is not needed and may be set to zero.*

5.2.2 Nullspace method

The nullspace method by Alstad and Skogestad [2007] deals with the optimal selection of linear measurement combinations as controlled variables, $c = Hy$. In a recent chapter by the same authors [Alstad et al., 2009], their results are interpreted more generally as deriving linear invariants for quadratic optimization problems. More specifically, a key result is the following Theorem:

Theorem 5.1. *(Linear invariants for quadratic optimization problems [Alstad and Skogestad, 2007]) Consider an unconstrained quadratic optimization problem in the variables $u \in \mathbb{R}^{n_u}$, parameterized in $d \in \mathbb{R}^{n_d}$:*

$$J^*(d) = \min_u \left\{ J(u,d) = \begin{bmatrix} u \\ d \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ud} \\ J'_{ud} & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix} \right\}. \tag{5.13}$$

In addition, there are "measurement" variables $y = G^y u + G^y_d d$. If there exists $n_y \geq n_u + n_d$ independent measurements (where "independent" means that the matrix $\tilde{G}^y = [G^y \; G^y_d]$ has full row rank), then the optimal solution to (5.13) *has the property that there exists $n_c = n_u$ linear variable combinations (constraints) $c = Hy$ that are invariant to the disturbances d, meaning that their optimal value $(c = 0)$ is independent of d. Here, H may be found from the nullspace method using $H = \text{null}(F')$, where*

$$F = -(G^y J_{uu}^{-1} J_{ud} - G^y_d) \tag{5.14}$$

5.2.3 Implementation of solution to parametric quadratic programs

In this section we follow Baotić et al. [2008] unless otherwise noted. This implies that all Theorems, Lemmas, Algorithms and Definitions are taken from the reference unless otherwise noted.

Definiton 5.1. *Two polyhedra $P_i, P_j \in \mathbb{R}^{n_x}$ are called* neighboring polyhedra *if their interiors are disjoint and $P_i \cap P_j$ is $(n_x - 1)$-dimensional (i.e. is a common facet).*

Let $\{P_i\}_{i=1}^{N_p}$ be a polyhedral partition. For each polyhedron P_i we denote with C_i the list of all its neighbors,

$$C_i := \left\{ j \;\middle|\; \begin{array}{l} P_j \text{ is a neighbor of } P_i, \\ j = 1,\ldots,N_p, \quad j \neq i \end{array} \right\} \tag{5.15}$$

Throughout the chapter we assume that every facet is shared by only two neighboring polyhedral partitions, i.e. that the facet-to-facet property [Spjøtvold et al., 2006] holds.

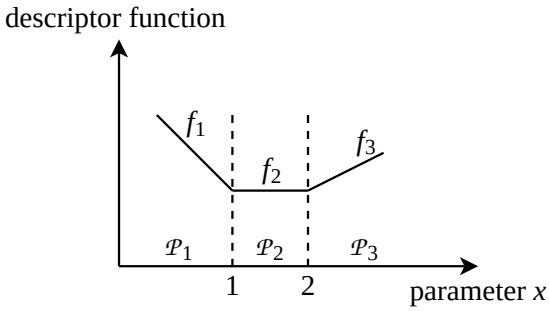

Figure 5.2: A scalar descriptor function over three polyhedra.

Definiton 5.2. *(PWA descriptor function) A scalar continuous real-valued PWA function* $f : X_f \mapsto \mathbb{R}$*,*

$$f(x) := f_i(x) = A_i'x + B_i \quad \textit{if} \quad x \in \mathcal{P}_i, \tag{5.16}$$

with $A_i \in \mathbb{R}^{n_x}$*,* $B_i \in \mathbb{R}$*, is called a* descriptor function *if*

$$A_i \neq A_j, \quad \forall j \in C_i, \quad i = 1, \ldots, N_p, \tag{5.17}$$

where $\cup_i \mathcal{P}_i = X_f \subset \mathbb{R}^{n_x}$*, and* C_i *is the list of neighbors of* $\mathcal{P}_i$*.*

See Figure 5.2 for an example of a scalar PWA descriptor function. This kind of function can be used to track changes in the optimal active set. We can do this because the sign of $f_i(x) - f_j(x)$ changes only when the point x crosses the separating hyperplane between $\mathcal{P}_i$ and $\mathcal{P}_j$. Thus for all $x \in \mathcal{P}_i$, the difference $f_i(x) - f_j(x)$ has the same sign.

In the figure, let $f_1 = -2x + 5$, $f_2 = 3$, and $f_3 = 0.5x + 2$. Assume that the parameter x is in $\mathcal{P}_2$ and we want to detect when x crosses into either $\mathcal{P}_1$ or $\mathcal{P}_3$ *without* measuring x itself (but we have a measurements available of f_1, f_2, f_3). We can do this using the descriptor function

$$f := f_i \quad \text{if} \quad x \in \mathcal{P}_i, \quad i = 1, 2, 3.$$

For $x \in \mathcal{P}_2$ we have that $\text{sign}(f_2 - f_1) = 1$ and $\text{sign}(f_2 - f_3) = 1$. Now, if either $\text{sign}(f_2 - f_1)$ or $\text{sign}(f_2 - f_3)$ changes sign, we deduce that x has moved to $\mathcal{P}_1$ or $\mathcal{P}_3$, respectively.

Definiton 5.3. *(Ordering function) Let* $f(x)$ *be a PWA descriptor function on the polyhedral partition* $\{\mathcal{P}_i\}_{i=1}^{N_p}$*. An ordering function* $O_i(x)$ *is defined as*

$$O_i(x) := [O_{i,j}(x)]_{j \in C_i} \tag{5.18}$$

where

$$O_{i,j} = \begin{cases} +1 & \text{if} \quad f_i(x) \geq f_j(x) \\ -1 & \text{if} \quad f_i(x) \leq f_j(x) \end{cases} \tag{5.19}$$

with $i \in \{1,\ldots,N_p\}$, $j \in C_i$.

Theorem 5.2. *Let* $f(x)$ *be a PWA descriptor function on the polyhedral partition* $\{P_i\}_{i=1}^{N_p}$. *Let* $\xi_i \in \mathbb{R}^{n_x}$ *be any point in the interior of* $\mathcal{P}_i$, *and define*

$$\begin{aligned} S_{i,j} &:= O_{i,j}(\xi_i) \\ S_i &:= O_i(\xi_i), \end{aligned} \tag{5.20}$$

with $i = 1,\ldots,N_p$, $j \in C_i$. *Then the following holds:*

$$\begin{aligned} x \in \text{int}(\mathcal{P}_i) \;&\Leftrightarrow\; O_{i,j}(x) = S_{i,j} \quad \forall j \in C_i \\ &\Leftrightarrow\; O_i(x) = S_i \end{aligned} \tag{5.21}$$

Theorem 5.2 states that the ordering function $O_i(x)$ and the vector S_i uniquely characterize $\mathcal{P}_i$. Therefore, to check on-line if the polyhedral region $\mathcal{P}_i$ contains the state x, it is sufficient to compute the binary vector $O_i(x)$ and compare it to S_i.

Vectors S_i are calculated off-line for $i = 1,\ldots,N_p$, by comparing the values of $f_i(x)$ and $f_j(x)$, $\forall j \in C_i$, in a point that belongs to $\text{int}(\mathcal{P}_i)$, for instance, the Chebysev center of $\mathcal{P}_i$.

Algorithm 5.1 GLOBAL. (Used for initialization and recovery)

1: $I = \{1,\ldots,N_p\}$
2: $i \leftarrow I$
3: $I = I\backslash\{i\}$, $C = C_i$
4: **while** $C \neq \emptyset$ **do**
5: $\quad j \leftarrow C$, $C = C\backslash\{j\}$
6: $\quad$ Compute $O_{i,j}(x)$
7: $\quad$ **if** $O_{i,j}(x) \neq S_{i,j}$ **then**
8: $\quad\quad$ **if** $j \notin I$ **then**
9: $\quad\quad\quad$ GOTO step 2
10: $\quad\quad$ **else**
11: $\quad\quad\quad$ $i = j$ and GOTO step 3
12: $\quad\quad$ **end if**
13: $\quad$ **end if**
14: **end while**

Algorithm 5.2 LOCAL.

Require: Current region i

1: $C = C_i$ and NotLost $= 1$
2: **while** NotLost **do**
3: Compute vector $O_i(x)$
4: **if** $O_i(x) \neq S_i$ **then**
5: **if** the difference is at element corresponding to j only **then**
6: Set $i = j$ and GOTO step 1.
7: **else**
8: Set NotLost $= 0$.
9: **end if**
10: **end if**
11: **end while**

Algorithm 5.3 Main program.

1: Run Algorithm 5.1 GLOBAL to find current region i.
2: **while** System is operational **do**
3: Run Algorithm 5.2 LOCAL.
4: **if** $NotLost = 0$ **then**
5: Run Algorithm 5.1 GLOBAL to find current region i.
6: **end if**
7: **end while**

Locating the current state. Algorithm 5.1 was proposed by Baotić et al. [2008] to find the current state $x(t)$ for explicit MPC. Here we extend this method by adding Algorithm 5.2 as a "local" algorithm that for the current polyhedral region only monitors the corresponding ordering function (and thus only "looks" at the neighboring regions). If one element of this vector changes sign, the algorithm updates the current region to the region *corresponding* to the element of the vector that changed sign. However, if more elements changed sign we deduce that the process did not change to a neighboring region and we must run Algorithm 5.1 again. This logic is covered in the main program in Algorithm 5.3. A similar logic is published in [Narasimhan and Skogestad, 2010].

Finding a scalar PWA descriptor function. A vector-valued PWA descriptor function is defined as:

Definiton 5.4. *(Vector-valued PWA descriptor function) A continuous vector-valued piece-wise affine (PWA) function*

$$m(x) := \bar{A}_i x + \bar{B}_i \quad if \quad x \in \mathcal{P}_i \tag{5.22}$$

is called a vector-valued PWA descriptor function if

$$\bar{A}_i \neq \bar{A}_j \quad \forall j \in C_i, \quad \forall i = 1, \cdots, N_p, \tag{5.23}$$

where $\bar{A}_i \in \mathbb{R}^{s \times n_x}$, $\bar{B}_i \in \mathbb{R}^s$, $s \in \mathbb{N}$, $s \geq 2$, *and* C_i *is the list of neighbors of* $\mathcal{P}_i$.

Next, the following Theorem gives a method for constructing a scalar PWA descriptor function from a vector-valued one.

Theorem 5.3. *([Baotić et al., 2008]) Let* $m : \mathbb{R}^{n_x} \mapsto \mathbb{R}^s$ *be a vector valued PWA descriptor function defined over a polyhedral partition* $\{\mathcal{P}_i\}_{i=1}^{N_p}$. *Then there exists a* $w \in \mathbb{R}^s$ *such that* $f(x) := w'm(x)$ *is a PWA descriptor function over the same polyhedral partition.*

Algorithm for finding w**.** For a given vector-valued PWA descriptor function we form a set of vectors $a_k \in \mathbb{R}^s$, $\|a_k\| = 1$, $k = 1, \cdots, N_a$, by taking one (and only one) nonzero column from each matrix $(\bar{A}_i - \bar{A}_j)$, $\forall j \in C_i$, $i = 1, \cdots, N_p$. Here $N_a := \sum_i |C_i|/2 \leq N_H$, and $|C_i|$ denotes the cardinality of set C_i. The vector $w \in \mathbb{R}^s$ satisfying the set of equations $w'a_k \neq 0$, $k = 1, \cdots, N_a$, can be constructed using Algorithm 5.4.

Algorithm 5.4 Construct the vector w.

```
w ← [1, ··· , 1]′
while k ≤ N_a do
    d ← w′a_k
    if 0 ≤ d ≤ R then
        w ← w + ½(R − d)a_k, R ← ½(R + d)
    else if −R ≤ d ≤ 0 then
        w ← w − ½(R + d)a_k, R ← ½(R − d)
    end if
end while
```

Properties of the solution of a parametric QP. Consider again the quadratic problem (5.8):

$$\min_u \frac{1}{2}\begin{bmatrix} u \\ d \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ud} \\ J'_{ud} & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix}$$
$$\text{s.t. } M_u u \leq M + M_d d$$

From [Baotiċ et al., 2008] we have the following properties of the solution to this problem:

Theorem 5.4. *Consider the parametric QP in* (5.8) *and let $J_{uu} > 0$. Then the set $\mathcal{D}$ of feasible parameters d is convex, the optimal input $u^* : \mathcal{D} \mapsto \mathbb{R}^{n_u}$ is continuous and piecewise affine.*

In addition the following Lemma is provided:

Lemma 5.1. *Let the optimal solution ("optimizer") be written on the form*

$$u = K_d^i d + k_d^i \quad \text{if } d \in \mathcal{P}_i \tag{5.24}$$

Then, for two neighboring polyhedra $\mathcal{P}_i, \mathcal{P}_j$ the gains $K_d^i \neq K_d^j$.

5.3 Measurement based descriptor function

From now on, the results are new unless otherwise noted.

Lemma 5.1 states that the optimizer to problem (5.8) can be written on the form

$$u = K_d^i d + k_d^i \quad \text{if } d \in \mathcal{P}_i. \tag{5.25}$$

We now want to eliminate the need of information about the disturbances d, but rather rely on plant output y_m.

Assuming a parametric solution exists, we form the following problem for a given set of active inequality constraints:

$$\begin{aligned}&\min_{\left[\begin{smallmatrix}u_1\\u_2\end{smallmatrix}\right]} J(u_1,u_2,d) = \\ &\quad \frac{1}{2}\begin{bmatrix}u_1\\u_2\\d\end{bmatrix}' \begin{bmatrix}J_{u_1u_1} & J_{u_1u_2} & J_{u_1d}\\ \star & J_{u_2u_2} & J_{u_2d}\\ \star & \star & J_{dd}\end{bmatrix}\begin{bmatrix}u_1\\u_2\\d\end{bmatrix} \\ &\quad \text{subject to } M_{u_1}u_1 + M_{u_2}u_2 = M + M_d d,\end{aligned} \tag{5.26}$$

where we choose u_1 (as a subset of the input vector u) such that M_{u_1} is invertible. This implies that we can write

$$u_1 = \underbrace{-{M_{u_1}}^{-1}M_{u_2}}_{K^{u_2}}u_2 + \underbrace{{M_{u_1}}^{-1}M_d}_{K^d}d + \underbrace{{M_{u_1}}^{-1}M}_{K}. \tag{5.27}$$

We can now do the following manipulations in order to get the problem on a form suitable for Theorem 5.1: First, we define $z = u_2 + J_{zz}^{-1}J_z$ and

$$J_{zz} = {K^{u_2}}'J_{u_1u_1}K^{u_2} + J_{u_2u_2} + 2{K^{u_2}}'J_{u_1u_2} \tag{5.28}$$
$$J_{zd} = {K^{u_2}}'J_{u_1u_1}K^d + {J_{u_1u_2}}'K^d + {K^{u_2}}'J_{u_1d} + J_{u_2d} \tag{5.29}$$
$$J_z = K'J_{u_1u_1}K^{u_2} + K'J_{u_1u_2} \tag{5.30}$$

With these definitions it can be shown that the objective function with the active equality constraints substituted into the objective can be written as

$$J(z,d) = \frac{1}{2}z'J_{zz}z + z'J_{zd}d. \tag{5.31}$$

In addition, we write the linear model as

$$\begin{bmatrix}u_1\\u_2\\y_m\end{bmatrix} = \begin{bmatrix}I & 0\\0 & I\\G_{u_1}^{y_m} & G_{u_2}^{y_m}\end{bmatrix}\begin{bmatrix}u_1\\u_2\end{bmatrix} + \begin{bmatrix}0\\0\\G_d^{y_m}\end{bmatrix}d \tag{5.32}$$

Using (5.27) we find that

$$\begin{bmatrix} u_1 \\ u_2 \\ y_m \end{bmatrix} = \begin{bmatrix} K^{u_2} \\ I \\ \underbrace{G_{u_1}^{y_m} K^{u_2} + G_{u_2}^{y_m}}_{=\tilde{G}_{u_2}^{y_m}} \end{bmatrix} u_2 + \begin{bmatrix} K^d \\ 0 \\ \underbrace{G_{u_1}^{y_m} K^d + G_d^{y_m}}_{=\tilde{G}_d^{y_m}} \end{bmatrix} d + \begin{bmatrix} K \\ 0 \\ G_{u_1}^{y_m} K \end{bmatrix} \tag{5.33}$$

$$= \underbrace{\begin{bmatrix} K^{u_2} \\ I \\ \tilde{G}_{u_2}^{y_m} \end{bmatrix}}_{G^y} (z - J_{zz}^{-1} J_z') + \underbrace{\begin{bmatrix} K^d \\ 0 \\ \tilde{G}_d^{y_m} \end{bmatrix}}_{G_d^y} d + \underbrace{\begin{bmatrix} K \\ 0 \\ G_{u_1}^{y_m} K \end{bmatrix}}_{\tilde{K}} \tag{5.34}$$

$$= G^y z + G_d^y d + \underbrace{\tilde{K} - G^y J_{zz}^{-1} J_z'}_{\gamma}. \tag{5.35}$$

Now, let $\bar{y} = y - \gamma$ and let $\bar{F} = -(G^y J_{zz}^{-1} J_{zd} - G_d^y)$ and further let $\tilde{H}$ be a full rank matrix that fulfills $\tilde{H}F = 0$. Due to Theorem 5.1 we have that optimally $\tilde{H}\bar{y} = \tilde{H}(y - \gamma) = 0$, hence the invariants are $c = \tilde{H}y$ with $c_s = \tilde{H}\gamma$. Due to the "extra" degrees of freedom in $\tilde{H}$ can we write the combination matrix on the form $\tilde{H} = \begin{bmatrix} I & H^{y_m} \end{bmatrix}$. The extra degrees of freedom in H arise from the fact that if $HF = 0$, then also $DHF = 0$. By letting D be a non-singular square matrix we can use this to scale the entries in H, or as above to introduce an identity matrix. This is further discussed in [Alstad et al., 2009]. Finally, we can show that this invariant can be written on the form:

$$u = -H^{y_m} y_m + \begin{bmatrix} (K - K^{u_2}) \\ -I \end{bmatrix} J_{zz}^{-1} J_z' + H^{y_m} [G_{u_1}^{y_m} (K - K^{u_2}) + G_{u_2}^{y_m}] J_{zz}^{-1} J_z'. \tag{5.36}$$

We observe that for a given set of active constraints, there is an affine optimal relationship between the input u and the measurement y_m. For several regions we can therefore pose the following optimal relationship:

$$u = K_{y_m}^i y_m + c_s^i, \quad \text{if } d \in \mathcal{P}_i, \tag{5.37}$$

where $K_{y_m}^i$ and c_s^i can be found by using the procedure above. The following Lemma shows that this functional relationship from measurement y_m to input u can be used as a vector-valued PWA descriptor function.

Lemma 5.2. *The invariants defined by* $\text{inv}^i := H^i y - c_s^i$ *can be used as a vector-valued descriptor function*

Proof. Theorem 5.2 states that the optimizer for for problem (5.2.3) can be written on the form of equation (5.25). According to Lemma 5.1, for two neighboring

polyhedra $\mathcal{P}_i, \mathcal{P}_j$, $K_d^i \neq K_d^j$, and hence the disturbance feedback law (5.25) is a vector-valued descriptor function.

We now consider the invariants $\text{inv}^i := H^i y - c_s^i$, and we assume that we have a perfect measurement of the input vector included in y on the form:

$$y = \begin{bmatrix} u \\ y_m \end{bmatrix} = \begin{bmatrix} I \\ G_u^{y_m} \end{bmatrix} u + \begin{bmatrix} 0 \\ G_d^{y_m} \end{bmatrix} d \tag{5.38}$$

With this partition of y we accordingly write $H^i = \begin{bmatrix} H^{u,i} & H^{y_m,i} \end{bmatrix}$. By assumption J_{uu} is positive definite (second-order optimality conditions), and hence the optimal input u must by Theorem 5.2 be unique and continuous. This has the following implications: First, we can form an equivalent invariant by

$$\text{inv}_{\text{fb}}^i = u - \underbrace{(H^{u,i})^{-1} H^{y_m,i}}_{K_{y_m}^i} y_m + \underbrace{(H^{u,i})^{-1} c_s^i}_{k_{y_m}^i}. \tag{5.39}$$

For optimality, by Theorem 5.1, this invariant should be controlled to zero, hence we have the measurement feedback form

$$u = K_{y_m}^i y_m + k_{y_m}^i \quad \text{if} \quad d \in \mathcal{P}_i, \quad \forall i = 1, \ldots, N_p \tag{5.40}$$

Inserting the equality constraint (5.38), we have that

$$\begin{aligned} u &= K_{y_m}^i (G^{y_m} u + G_d^{y_m} d) + k_{y_m}^i, \\ &= K_{y_m}^i G^{y_m} u + K_{y_m} G_d^{y_m} d + k_{y_m}^i, \\ \Rightarrow \quad (I - K_{y_m}^i G^{y_m}) u &= K_{y_m}^i G_d^{y_m} d + k_{y_m}^i, \\ \Rightarrow \quad u &= (I - K_{y_m}^i G^{y_m})^{-1} K_{y_m}^i G_d^{y_m} d \\ &\quad + (I - K_{y_m} G^{y_m})^{-1} k_{y_m}^i. \end{aligned} \tag{5.41}$$

Second, due to the uniqueness of the optimal input u, we have that the inverse of $(I - K_{y_m}^i G^{y_m})$ must exist and further that

$$(I - K_{y_m}^i G^{y_m})^{-1} K_{y_m}^i G_d^{y_m} = K_d^i \quad \forall i = 1, \ldots, N_p \tag{5.42}$$

Since both $(I - K_{y_m}^i G^{y_m})^{-1}$ and $G_d^{y_m}$ have full rank, we must have that if $K_d^i \neq K_d^j$, then $K_{y_m}^i \neq K_{y_m}^j$. Finally, since u is continuous we conclude that the function $K_{y_m}^i y_m + k_{y_m}^i$ can be used as a vector-valued PWA descriptor function. □

Remark 5.2. *To use the nullspace method we do not need to include a perfect measurement of u in y; it is sufficient that we have enough independent measurements $n_y \geq n_u + n_d$. Here we include u because it is then easier to prove that the*

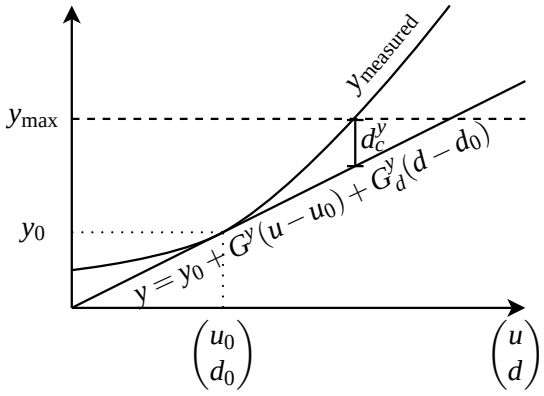

Figure 5.3: Additional disturbance d_c^y to match the model with the measured constraint.

resulting set of invariants can be used as a vector-valued PWA descriptor function. This means that we can use the method described in this chapter to construct a descriptor function as a function of measurements y_m, and include other measurements in the controlled variable selection problem. The only requirement is that the controlled variables gives zero loss from optimality when controlled to constant setpoints c_s^i.

5.4 Constraint matching

The linear approximation of the constraints as used in problem (5.8) may, as any model based scheme, lead to infeasibility when used on a real plant. However, this can to some extent be accounted for if the constraints are measured. We can then simply estimate a disturbance d_c as illustrated in Figure 5.3 and treat this as a measured disturbance in the problem formulation (problem (5.8)). For an output constraint we then have

$$\begin{aligned} y_{\min} \leq y_{\text{measured}} &\leq y_{\max} \\ \Updownarrow & \\ y_{\min} \leq y + d_c &\leq y_{\max} \\ \Updownarrow & \\ y_{\min} - d_c \leq y &\leq y_{\max} - d_c \end{aligned}$$

One should realize that this method can (and should) also be used on important manipulated variables u, where important here means inputs that have a strong economic effect, for example inputs that are affecting the throughput of a plant.

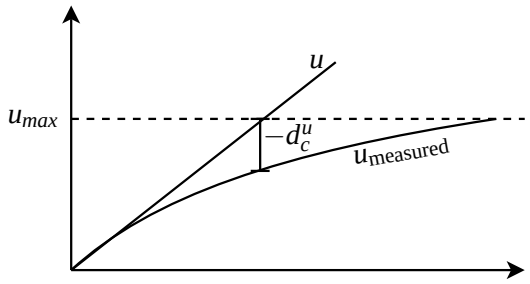

Figure 5.4: Matching of an input constraint by additional disturbance d_c^u.

Say, that for some values of the disturbances it is optimal to implement a certain input at its maximum value, but that there is some mismatch between the model and the reality, as illustrated in Figure 5.4. If a measurement of the actual value of u is available, then can this be corrected for by adding an extra disturbance d_c^u as indicated in the figure, and by using the procedure outlined above for the inputs to effectively change the value of $u_{\max}$ in the internal model of the controller.

5.5 Procedure for structure selection

We summarize our findings in the following procedure that may be used to find controlled variables for an economic problem that can be approximated as a quadratic program:

1. Define the optimal control problem, consisting of objective function $J_0(x,u,d)$, process model $f_0(x,u,d) = 0$, and operational constraints $g_0(x,u,d)$.
2. Approximate this problem around the nominal optimum as a QP by the method outlined in section 5.2.1:
   (a) Eliminate the model from the nonlinear problem.
   (b) Solve the resulting optimization problem for nominal disturbance d_0 to get optimal inputs u^* and optimal Lagrange multipliers λ^*.
   (c) Approximate this problem as a QP around (u^*, λ^*).
3. Add extra disturbances d_c as illustrated in section 5.4 for important constraints.
4. Solve the resulting problem as a parametric QP where the disturbances d are parameters. The solution will consist of a set of polyhedral regions $\mathcal{P}_i$ in the disturbance space and a list of active constraints for each region.

5. Identify available measurements and linearize to get

$$y_m = G_m^y u + G_d^{y_m} d.$$

6. In each region (in the disturbance space), use the nullspace method of Theorem 5.1 to find invariants $\text{inv}^i := H^i y - c_s^i$.

7. Use Lemma 5.2 to make vector-valued PWA descriptor function.

8. Use Algorithm 5.4 to construct a scalar PWA descriptor function to be used for region detection.

9. Use Algorithm 5.3 for region detection.

5.6 Example: Ammonia production plant

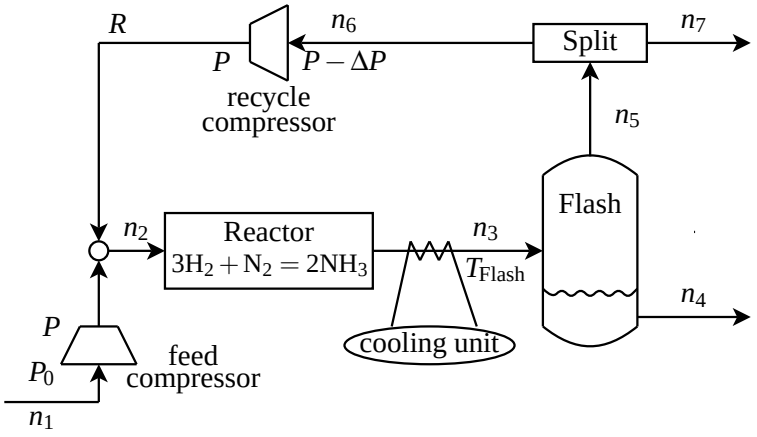

Figure 5.5: Sketch of an ammonia synthesis loop.

Consider the ammonia synthesis loop in figure 5.5. The objective is to maximize the produced ammonia in stream 4, while at the same time to minimize the use of compressor work and cooling with a given reactor temperature. The cost function is

$$\text{profit} = P_{W_{\text{feed}}} W_{\text{feed}} + P_{W_{\text{recycle}}} W_{\text{recycle}} + P_{W_{\text{cooling}}} W_{\text{cooling}} + P_{\text{NH}_3} n_4^{\text{NH}_3}, \qquad (5.43)$$

with prices given in Table 5.1.

In addition we must satisfy some operational constraints, namely a lower limit on the possible cooling (T_{flash}) and a high limit on the recycle (R) in the loop.

Price variable	Value [€/unit]
$P_{W_{\text{feed}}}$	-0.4
$P_{W_{\text{recycle}}}$	-10
$P_{W_{\text{cooling}}}$	-0.5
P_{NH_3}	10^4

Table 5.1: Prices for ammonia example.

There are three steady state degrees of freedom, which can be chosen to be the pressure, recycle ratio and flash temperature, and the temperature in the reactor is assumed constant.

The feed consists of a mix of H_2 and N_2, and the main disturbances are feed rate and feed composition.

5.6.1 Model

The model we use in this example is a nonlinear model of the ammonia plant that should explain the most dominant effects that influence the economic operation of the plant at steady state. The main properties of the model are:

- Equilibrium reactor.
- Henry's law (H_2, N_2) and Raoult's law (NH_3) describe the flash-tank.
- Ideal compressor works.
- Cooling work efficiency given by a Carnot factor.

The model consists only consist of a mass balance. The variables are the mole vector for each stream, n_i, $i = 1, \ldots, n_7$ and extent of reaction ξ. In addition we use as secondary variable x_i^j to indicate the mole fraction of component j in stream i. The components are ordered by H_2, N_2, NH_3.

Table 5.2 shows a list of constants used in the modelling. All constants are found in the book by Skogestad [2003b].

The mathematical model is given below:

Reactor feed. Mass balance over the feed point:

$$n_2 = n_1 + n_6 \tag{5.44}$$

Variable	Value	Unit
K_{eq}	$6.36 \cdot 10^{-5}$	
$H_0^{H_2}$	210688	
$H_T^{H_2}$	-656	
$H_0^{N_2}$	110816	
$H_T^{N_2}$	-342	
A	4.4854	
B	926.132	
C	-32.98	
d_1^0	5.1	mole/time
d_2^0	0.8	mole fraction

Table 5.2: Constants for the ammonia plant example.

Equilibrium reactor. Let $P_i = Px_3$ be the partial pressures in stream 3. The equilibrium relation is then

$$\frac{P_{\text{NH}_3}^2}{P_{\text{H}_2}^3 P_{\text{N}_2}} = K_{\text{eq}}. \tag{5.45}$$

Further, by using the extent of reaction ξ, we have that

$$n_3 = n_2 + S\xi, \tag{5.46}$$

where the stoichiometric matrix $S = \begin{bmatrix} -3 & -1 & 2 \end{bmatrix}'$.

Flash tank. We here assume Henry's law for H_2 and N_2 and Rault's law for NH_3. The K-values are given by

$$k_{\text{H}_2} = \frac{H_0^{\text{H}_2} + H_T^{\text{H}_2} T_{\text{flash}}}{P} \tag{5.47}$$

$$k_{\text{N}_2} = \frac{H_0^{\text{N}_2} + H_T^{\text{N}_2} T_{\text{flash}}}{P} \tag{5.48}$$

$$k_{\text{NH}_3} = \frac{10^{A - \frac{B}{T_{\text{flash}} + C}}}{P} \tag{5.49}$$

Let $K = \text{diag}(k_{\text{H}_2}, k_{\text{N}_2}, k_{\text{NH}_3})$ and we have that

$$x_5 = Kx_4. \tag{5.50}$$

In addition we use the Rachford-Rice equation to find the ratio $r = (\sum n_5)/(\sum n_3)$:

$$\sum_{i=\{\text{H}_2,\text{N}_2,\text{NH}_3\}} \frac{x_3^i (k_i - 1)}{1 + r(k_i - 1)} = 0. \tag{5.51}$$

Now,

$$e'n_5 = re'n_3 \tag{5.52}$$
$$e'n_4 = (1-r)e'n_3, \tag{5.53}$$

where $e' = \begin{bmatrix}1 & 1 & 1\end{bmatrix}$.

Split. The mass balance around the split is

$$n_5 = n_6 + n_7 \tag{5.54}$$

In addition we have that

$$n_6 = fn_5. \tag{5.55}$$

The reader may observe that the model depends on the temperature T_{flash} in the flash-tank and the pressure P in the reactor and flash-tank. We here treat the variables P, T_{flash} and the recycle ratio f as steady state degrees of freedom.

For the compressors we have the following models:

Feed compressor. The feed compressor increases the pressure in the feed from the nominal pressure P_0 to the reactor pressure P by

$$W_{\text{feed}} = \sum(n_2)RT\ln\frac{P}{P_0} \tag{5.56}$$

Recycle compressor. This compressor should counteract pressure drop in the system by

$$W_{\text{recycle}} = \sum(n_6)RT_{\text{flash}}\ln\frac{P}{P-\Delta P} \tag{5.57}$$

Cooling water. We assume that there is cooling water free of charge that can cool the product stream down to 15° C $= 288$ K. For further cooling, we have to use a cooling unit with the following work associated:

$$W_{\text{sub cool}} = \sum n_3 C_P(T_0 - T)(\frac{T_0}{T_c} - 1), \tag{5.58}$$
$$\text{where } T_c = \frac{T_0 - T}{\ln(T_0/T)}. \tag{5.59}$$

This means that the overall energy usage for cooling is

$$W_{\text{cooling}} = \begin{cases} 0 & \text{if } T_{\text{flash}} > 288 \text{ K} \\ W_{\text{sub cool}} & \text{otherwise} \end{cases} \tag{5.60}$$

5.6.2 Disturbances

The only disturbances acting on the system are the feed rate (d_1) and composition (d_2). The feed stream n_1 can therefore be expressed as

$$n_1 = (d_1^0 + d_1) \begin{bmatrix} d_2^0 + d_2 \\ (1 - d_2^0) - d_2 \\ 0 \end{bmatrix}, \tag{5.61}$$

with $d_1^0 = 5.1$ mole/time as the nominal feed flow and $d_2^0 = 0.8$ as the nominal mole fraction of hydrogen in the feed. The disturbances are assumed to be in the set

$$\mathcal{D} = \left\{ d \in \mathbb{R}^2 \;\middle|\; |d_1| < 1,\ |d_2| < 0.02 \right\}. \tag{5.62}$$

This corresponds to a maximum relative change in the feed rate of about 20% and a change in the composition of about 4%.

5.6.3 Operational constraints

There are two operational constraints that we need to address. First, the cooling unit can only cool the reactor product to -7° C $= 266$ K, therefore

$$T_{\text{flash}} \geq 266\,\text{K}. \tag{5.63}$$

We assume that this constraint can be implemented exactly, i.e. that an unbiased measurement of this temperature exists.

In addition is there an upper bound on the maximum flow of recycle. For this constraint, we include an extra disturbance ($d_3 = d_c$) as explained in Section 5.4 to make sure that we satisfy the upper limit on recycle at all times. (Note that this maximum recycle constraint is motivated by the fact that the recycle compressor has a high limit on the amount of fluid it can process.) We include this correction by the following procedure: First, we find a linear model from (u, d) to n_6 on the form

$$n_6 \approx n_{6,0} + G^{n_6} u + G_d^{n_6} d, \tag{5.64}$$

using for example finite differences. The we add the "constraint matching" disturbance $d_3 = d_c$ to get the following inequality that bounds the maximum recycle in the plant:

$$R_{\text{measured}} \leq R_{\max} \quad \Leftrightarrow \quad e' G^{n_6} u + e' G_d^{n_6} d + 1' n_{6,0} + d_3 \leq \text{r}_{\max}$$

so finally we have

$$e' G^{n_6} u \leq (\text{r}_{\max} - e' n_{6,0}) + \begin{bmatrix} -e' G_d^{n_6} & -1 \end{bmatrix} \begin{bmatrix} d \\ d_3 \end{bmatrix}, \tag{5.65}$$

Steady state degree of freedom	Nominal optimal value
Split-factor (recycle ratio)	0.6875
Pressure P in reactor	342.17 bar
Temperature T_{flash} in flash-tank	266.53 K

Table 5.3: Nominal optimal inputs.

n_2	n_3	n_4	n_5	n_6	n_6
$\begin{bmatrix}6.8473\\1.2163\\0.0314\end{bmatrix}$	$\begin{bmatrix}4.0455\\0.2837\\1.8995\end{bmatrix}$	$\begin{bmatrix}0.0176\\0.0011\\1.8589\end{bmatrix}$	$\begin{bmatrix}4.0279\\0.2826\\0.0427\end{bmatrix}$	$\begin{bmatrix}2.7683\\0.1953\\0.0304\end{bmatrix}$	$\begin{bmatrix}1.2586\\0.0883\\0.0133\end{bmatrix}$

Table 5.4: Nominal optimal stream data.

where d are the "economic" disturbances (d_1, d_2), and d_3 is the "constraint matching" disturbance.

5.6.4 Control structure selection

Nominal operating point. We used Tomlab$^{\text{TM}}$ under Matlab$^{\text{TM}}$ to find the nominal operating point, as reported in Tables 5.4 and 5.3. The inputs u used in the sequel are deviation variables from this nominal operating point.

Approximation to a QP. At the nominal optimum no constraints are active, so we can use the Hessian of the nonlinear problem (rather than using the Lagrangian) to find J_{uu} and J_{ud}. The resulting matrices are:

$$\left[\begin{array}{c|c} J_{uu} & J_{ud} \\ \hline J_{ud}' & J_{dd} \end{array}\right] = \left[\begin{array}{ccc|cc} 55720.17 & -8.01 & 2.62 & 59.53 & 340268.50 \\ -7.95 & 0.08 & -0.08 & 0.02 & 75.11 \\ 2.61 & -0.08 & 1.31 & -0.00 & 7.70 \\ \hline 59.52 & 0.02 & -0.00 & 0.04 & 15695.24 \\ 340081.83 & 74.33 & 7.73 & 15575.95 & 1702546.66 \end{array}\right] \tag{5.66}$$

This Hessian is found by finite differences. We observe that the matrix is not fully symmetric because of numerical inaccuracy, but it is close enough to symmetric for our purposes. We used the upper right part as J_{ud} in the calculations.

The linearized constraints are:

$$\begin{matrix}[\max R:\,]\\ [\min T_{\text{flash}}:\,]\end{matrix}\underbrace{\begin{bmatrix}9.6295 & -0.0033 & 0.0015\\ 0 & 0 & -1.0000\end{bmatrix}}_{M_u}\begin{bmatrix}u_1\\u_2\\u_3\end{bmatrix}\leq \underbrace{\begin{bmatrix}0.51\\0.53\end{bmatrix}}_{M}+\underbrace{\begin{bmatrix}-0.59 & -23.76 & -1.00\\ 0 & 0 & 0\end{bmatrix}}_{M_d}\begin{bmatrix}d_1\\d_2\\d_3\end{bmatrix} \tag{5.67}$$

In addition there are non-negative constraints on all the compositions and a lower limit on the pressure in the system, but these constraints are not active for the disturbance space we chose to study, so we do not add them explicitly to the problem formulation.

Measurements for region detection. We have three disturbances, but one of them is assumed to be measured (the constraint matching for maximum recycle), hence we need to identify two measurements that we can use for region detection (see Theorem 5.1). Since the goal here is to demonstrate how to use this methodology, we simply chose the two first entries of the stream-vector n_2 as measurements, that is the flow of H_2 and N_2 in the reactor feed. This gives the following "measurements" y (in deviation variables) that we use for region detection:

$$y=\begin{bmatrix}u_1\\u_2\\u_3\\n_2^{\mathrm{H_2}}\\n_2^{\mathrm{N_2}}\\d_3\end{bmatrix}=\underbrace{\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\\9.6773 & -0.0024 & 0.0002\\-0.1411 & -0.0008 & 0.0002\\0 & 0 & 0\end{bmatrix}}_{G^y}\begin{bmatrix}u_1\\u_2\\u_3\end{bmatrix}+\underbrace{\begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\\1.3468 & 33.8399 & 0\\0.2371 & -10.3119 & 0\\0 & 0 & 1\end{bmatrix}}_{G_d^y}\begin{bmatrix}d_1\\d_2\\d_3\end{bmatrix} \tag{5.68}$$

where the matrices G^y and G_d^y come from linearization around the nominal optimal point.

Parametric solution. Using the "Multi-Parametric Toolbox" (MPT) [Kvasnica et al., 2004] we identify three regions for the solution of the QP-approximation, which are described in Table 5.5. We used three parameters in the optimization; (d_1, d_2) with search space defined in equation (5.62), and in addition $-0.5 < d_3 < 0.5$.

Region	Description
1	Unconstrained "nominal" region.
2	$R = R_{\max}$ (maximum throughput of the recycle compressor.)
3	$T_{\text{flash}} = T_{\text{flash,min}}$ (cooling unit can not decrease temperature further.)

Table 5.5: Regions of the parametric solution to the QP-approximation.

Vector-valued PWA descriptor function. Using the nullspace method in each region (as described in Lemma 5.2), we get the following invariants (which we can use also as controlled variables):

$$u = \begin{cases} \begin{bmatrix} -0.28 & 1.60 & 0 \\ -76.19 & 427.31 & 0 \\ -4.51 & 25.31 & 0 \end{bmatrix} \begin{bmatrix} y_m \\ d_3 \end{bmatrix} & \text{if } d \in \mathcal{P}_1 \\ \begin{bmatrix} -0.15 & 0.23 & -0.24 \\ -54.97 & 210.37 & -39.94 \\ -3.10 & 10.98 & -2.61 \end{bmatrix} \begin{bmatrix} y_m \\ d_3 \end{bmatrix} + \begin{bmatrix} 0.12 \\ 20.21 \\ 1.32 \end{bmatrix} & \text{if } d \in \mathcal{P}_2 \\ \begin{bmatrix} -0.28 & 1.58 & 0 \\ -70.74 & 396.73 & 0 \\ -0.00 & 0.00 & 0 \end{bmatrix} \begin{bmatrix} y_m \\ d_3 \end{bmatrix} + \begin{bmatrix} -0.0004 \\ -0.6408 \\ -0.5305 \end{bmatrix} & \text{if } d \in \mathcal{P}_3 \end{cases} \tag{5.69}$$

In order to check the calculations the reader is referred to section 5.3. Next, using algorithm 5.4, we identify the following function which can be used for tracking changes in the active set:

$$f(y_m) := \begin{cases} f_1 = \begin{bmatrix} -81.0 & 454.2 & 0 \end{bmatrix} \begin{bmatrix} y_m \\ d \end{bmatrix} & \text{if } d \in \mathcal{P}_1 \\ f_2 = \begin{bmatrix} -58.2 & 221.6 & -42.8 \end{bmatrix} \begin{bmatrix} y_m \\ d \end{bmatrix} + 21.7 & \text{if } d \in \mathcal{P}_2 \\ f_2 = \begin{bmatrix} -71.0 & 398.3 & 0 \end{bmatrix} \begin{bmatrix} y_m \\ d \end{bmatrix} - 1.18 & \text{if } d \in \mathcal{P}_2 \end{cases} \tag{5.70}$$

Table 5.6 shows neighbors and correct signs for the functions f_i in equation (5.70).

Region	Neighbor(s)	sign($f_i - f_j$)
1	$(2,3)$	$(-1,1)$
2	1	-1
3	1	1

Table 5.6: Neighbors and correct signs for the scalar PWA descriptor $f(y_m)$ as defined in equation (5.70).

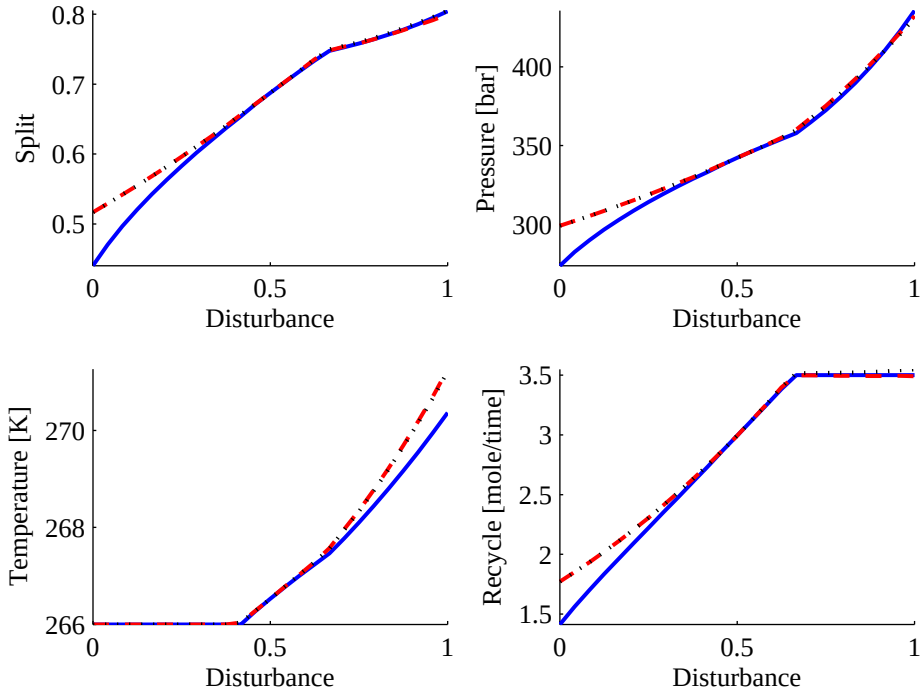

Figure 5.6: Calculated steady state degrees for freedom for the ammonia example for various disturbances. In addition we have plotted the resulting recycle R. The blue line represents the RTO, the red dashed line is the approach of this chapter, and the black dotted line is the approach of this paper *without* constraint matching. The disturbance axis represent traversing the disturbance space from $d_{\text{start}} = (-1, 0.02)$ to $d_{\text{end}} = (1, -0.02)$ in a straight line.

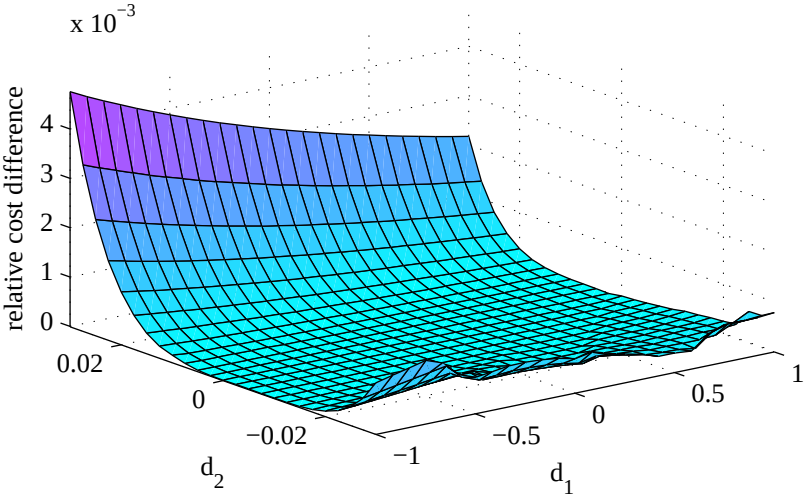

Figure 5.7: Relative difference in cost functions with constraint matching implemented.

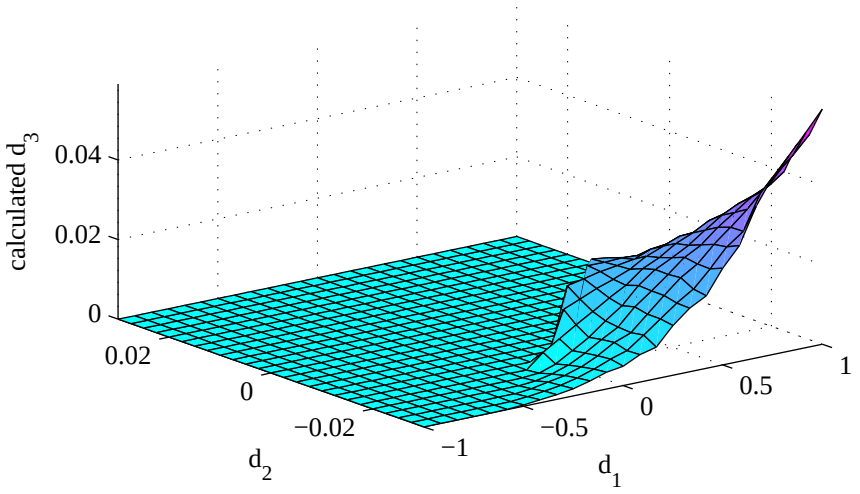

Figure 5.8: Constraint matching term d_3.

5.6.5 Simulation results

Figure 5.6 shows the result of simulating the proposed control structure for a range of disturbances corresponding to the search-space of the parametric program, that is $-1 \leq d_1 \leq 1$ and $-0.02 \leq d_2 \leq 0.02$. The figure shows a traversal of the disturbance space from $d_{\text{start}} = (-1, 0.02)$ to $d_{\text{end}} = (1, -0.02)$ by following a straight line. We chose this representation because this direction was the direction where the methods differed the most. For comparison we have solved the original nonlinear program for the same disturbances.

From Figure 5.6 we observe that the two methods are quite close, especially around the nominal disturbance. Figure 5.7 shows the difference in cost functions scaled with the absolute value of the optimal cost. We observe that the difference in this metric is quite small, and less than 1% for the cases studied. Probably this will be an "acceptable loss" and we therefore have an implementation that is close to optimal but simple, which is exactly in the spirit of "self-optimizing control."

Figure 5.8 shows the estimated nonlinear correction d_3. At saturation of the recycle the actual value is 3.5 mole/time, so the error in predicted recycle by the linear model is about 1.4% at maximum. We also simulated the system without this disturbance included as a measurement, and then the constraint on maximum recycle was violated with about 1%. This can be observed from Figure 5.6.

5.7 Discussion

In this chapter we use the descriptor function defined by Baotić et al. [2008] to implement the solution of a parametric quadratic program. Our main contribution is to relate descriptor functions to implementation of static optimization problems. In particular for quadratic problems, we show that we can identify descriptor functions based on "linear measurements" $y = G^y u + G^y_d d$ by using the nullspace method in each region of the parametric quadratic program. As a result we can make a list of constant setpoint policies, one for each region of the problem at hand, and a simple method for how to change between these policies, based on the outputs only.

The results are exact for quadratic problems, but as we have shown with an example, it seems like the method may also be applied to more general problems by quadratic approximations. This opens up some interesting research topics, as it seems like we can use the current method for region detection, and use other methods, such as the exact local method [Alstad et al., 2009] to account to noise, too few measurements, and other issues, as long as the loss from optimality is sufficiently small. What "sufficiently" means, and how to construct a control policy (that handles changes in the active set) that is robust to modelling errors and other

sources of errors, are open research issues.

In this chapter we have only considered steady state, and in fact the region detection scheme assumes that the system is at steady state at all times. This assumption will of course not be valid for real processes, and therefore dynamic studies of control policies with dynamic controllers and dynamic model of the plant should be conducted before application.

Static part of MPC. MPC is usually implemented with a static optimization problem that adjusts the setpoints of the controlled variables such that feasibility of the dynamic problem is guaranteed. The problem is often referred to as "target calculation", and may have the following structure [Rawlings, 2000]:

$$\min_{x_s,u_s,\eta} \frac{1}{2}(\eta' W_s \eta + (u_s-\bar{u})R_s(u_s-\bar{u})) + q_s'\eta$$

s.t.

$$\begin{bmatrix} I-A & -B & 0 \\ C & 0 & I \\ C & 0 & -I \end{bmatrix} \begin{bmatrix} x_s \\ u_s \\ \eta \end{bmatrix} \begin{Bmatrix} = \\ \geq \\ \leq \end{Bmatrix} \begin{bmatrix} Bd \\ \bar{y}-p \\ \bar{y}-p \end{bmatrix}$$

$$\eta \geq 0$$

$$u_{\min} \leq Du_s \leq u_{\max}$$

$$y_{\min} \leq Cx_s + p \leq y_{\max}$$

Here $\bar{y}$ and $\bar{u}$ are desired (assumed economically optimal) values for the measurements and inputs, while η is a slack variable.

However, our method may also guarantee feasibility if we can estimate (by using "constraint matching") the deviation from predicted and actual value of the output constraints. This is because the controlled variables $c = Hy$ are by construction feasible at their setpoints c_s (also for the *actual plant* when "constraint matching" is used). Hence, the method presented in this chapter may be used as an alternative to the steady state part of the MPC, with the benefit of improved economic performance of the plant.

A similar idea is presented in [Ying and Joseph, 1999], but the authors do not consider "feedback implementations" on the form of controlling $c = Hy$ to a setpoint c_s, rather they consider an open loop implementation of the static problem. Similar to what we do in this chapter, Ying and Joseph also suggest to use the Hessian of the Lagrangian of a quadratic approximation of the RTO as a quadratic weight in the feasibility problem of the MPC.

Using self-optimizing control with RTO. In the example we assumed that the feed composition could change with about 4%. Optimal economic operation of

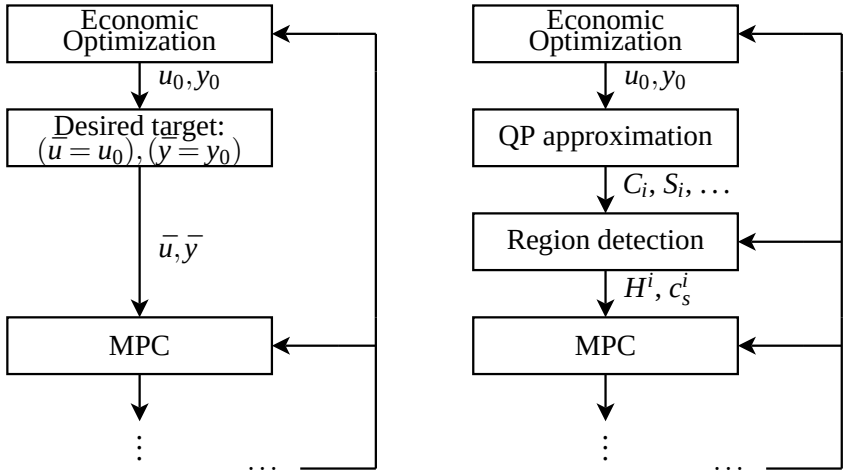

Figure 5.9: **Left figure:** Typical situation when self-optimizing control is not implemented, the controller should track the optimal values (u_0, y_0) from the real-time optimization (RTO). **Right figure:** A possible implementation of self-optimizing control with an RTO layer above. The controller is tracking controlled variables that should give an acceptably small loss from optimality between RTO updates when disturbances occur.

the plant was found to be a strong function of this disturbance, which is also clear from e.g. Figure 5.6 where one observes that one has to change the inputs considerably when the feed composition changes. For larger disturbances in the feed composition, say 10%, the self-optimizing control scheme will generate inputs that are quite far away from the optimal inputs, and there will be a significant loss. In these situations it would be fruitful to *update* the self-optimizing control policy by using an RTO layer (economic optimization) above the self-optimizing layer. A flow-sheet of a possible implementation is shown in Figure 5.9. In the figure we show both a typical scheme where an economic optimization layer sends a desired target value to the control layer, and a situation where one implements the scheme presented in this chapter. The scheme on the right hand side of the figure would typically be interesting if the economic optimization is updated every now and then (assuming that a fast update is too difficult). Such a scheme should be able to handle larger disturbances in the feed composition and still have an acceptably small loss from optimality.

5.8 Conclusion

Based on a recent contribution by Baotič et al. [2008] we have presented a generalization of the nullspace method [Alstad and Skogestad, 2007] to include changes in the optimal active set. The method has been demonstrated on a model of an ammonia production facility. We identified three different regions of operation, and the method was comparable in performance to real-time optimization of the same plant.

Chapter 6

Analysis of methods used to speed up model predictive control

Based on the paper "Bilevel Programming for Analysis of Reduced Models for use in Model Predictive Control" published in *Journal of Cybernetics and Informatics*, pages 3-12, volume 9, 2010.

In this chapter we develop a mathematical program that identifies the disturbance that maximizes the difference between two model predictive controllers, one candidate controller and one reference controller. The reference controller is assumed to be tuned to give a good trade-off between performance and robustness, but it is too computationally demanding to be implemented. The candidate controller is an approximation to the reference controller, where some "speedup" has been used. In this paper we consider move blocking, model reduction and changing the input horizon as possible speedups. For several different candidate controllers, one may use the proposed mathematical program to choose which controller gives the best performance to computational demand ratio.

We apply the proposed method to model predictive control of a distillation column, and we find that blocking the difference of moves in the MPC is more efficient that directly blocking the moves. We also find that the reducing the horizon is not very good, as this gives a high performance loss.

6.1 Introduction

In the literature, one can find numerous variations of discrete-time finite horizon optimal control, see e.g. the survey by Mayne et al. [2000]. In most implementations, the optimal sequence of inputs is computed at each time-step, and subsequently only the first element of the sequence is applied [Cagienard et al., 2007].

Method	CPU-time
Dense quadratic program	$\mathcal{O}((Nn_u)^3)$
Interior point method that exploits structure	$\mathcal{O}(N(n_x+n_u)^3)$

Table 6.1: Expected order of magnitude $\mathcal{O}(\cdot)$ for number of operations per step using either a dense formulation or a structured (sparse) formulation [Rao et al., 1998, Wang and Boyd, 2010].

At the next time step, the horizon is moved one step and with new measurements the calculations are repeated. This policy is referred to as model predictive control (MPC). In a recent paper Wang and Boyd [2010] claim that "a widely recognized shortcoming of MPC is that it can usually only be used in applications with slow dynamics". For an MPC problem with linear model, polytopic constraints, and quadratic objective, with state dimension n_x, input dimension n_u, and prediction horizon N, the order of magnitude of expected number of operations per step is given in Table 6.1. For both methods, the computational time is increasing in the third power of n_u, but whereas the computational time of a dense program is increasing in the third power also in N, the computational time for a solver that exploits structure is linear in N. Finally observe that the states are removed in the dense formulation, but they are present in the structured formulation, and the corresponding computational time is increasing in third order with n_x.

It is in addition recognized that the expected number of operations is also a function of the number of constraints of the problem [Qin and Badgwell, 2003].

In order to reduce the computational effort in conventional MPC one can

- Perform a *model reduction* of the internal model in the MPC to reduce the number of states n_x.
- Introduce *move blocking* strategies to reduce the number of degrees of freedom (given e.g. by the product Nn_u for a dense formulation). This strategy is used in numerous industrial implementations, see e.g. the survey by [Qin and Badgwell, 2003]. For an overview of different move blocking possibilities see [Cagienard et al., 2007]. The effect of move-blocking on the computational demand is clear in the case of a dense formulation (cubic decrease), but less clear when a structure-exploiting method is used. (Though there should be possible to exploit the extra structure given by the move-blocking and hence have some decrease in expected number of operations per step.)
- Reduce the *input horizon* N. Though this is an important parameter, there are few clear guidelines for how to choose it. It is recognized that the constraints

will only remain active for a finite number of time steps, and for a long enough horizon one can eventually force the system to an invariant set $O_\infty = \left\{ w \mid (A+BK)^j w \in W_K, \forall j \geq 0 \right\}$ such that the unconstrained feedback law $v = Kw$ is feasible for all future times, see e.g. [Rawlings, 2000]. Denote such an horizon N^*. Chmielewski and Manousiouthakis [1996] found an upper bound on N^*, but it is considered to be very conservative in many cases, see e.g. [Grieder et al., 2004]. Due to this we believe that for many applications the input horizon is chosen on an ad-hoc basis and may be used as a tuning to reduce the computational demand for the MPC.

- Reduce the *sample time* T_s of the model. This may be seen as an alternative to reducing the input horizon, where one instead uses a coarser model. One may also use models with different sampling times inside the controller, typically one would use a small sampling time for the first times, and then increase the sampling time later. See [Halldorsson et al., 2005] for more details.

In this work we develop a framework for *performance loss analysis* for such speed-up schemes by comparing a "candidate controller" with a "reference controller". The "candidate controller" is an MPC for which one has used some means to reduce the computational load, whereas the "reference controller" is the controller one would like to use. The mathematical method is based on exploiting the optimality conditions of MPC to rewrite a bilevel program to a mixed-integer linear program.

Closed-loop simulation of the candidate controller seems to be the normal way of checking closed-loop performance for industrial implementations [Qin and Badgwell, 2003]. However, checking all possible constraint combinations is not in general possible, and there is therefore a need for automatic methods that identifies the worst case disturbances.

With the method developed in this chapter the goal is that the practitioner can easily scan through different methods to speed up the controller in order to get a good trade-off between computational effort and performance loss.

An alternative approach is to use an explicit solution of MPC, see e.g. the recent survey by Alessio and Bemporad [2008]. Here one avoids the problem of online optimization altogether, but instead one needs to identify the correct state region. In practise, this approach works well only for systems with small state and input dimensions (say, no more than five [Wang and Boyd, 2010]). When the dimensions grow, the number of regions can grow exponentially [Wen et al., 2009], which implies that both storing all the regions and locating the correct region for a given state becomes a difficult problem.

Hovland et al. [Hovland et al., 2006, Hovland and Gravdahl, 2008] propose

a scheme to use reduced models in explicit MPC. They perform a two-step procedure to analyse the reduced order controller: First, they analyse the model reduction using an open loop evaluation of the model mismatch (evaluated by the $\mathcal{H}_2$-norm). Then, they make a table of model order and resulting number of regions, and choose the model order that gives a satisfactory low number of regions combined with a low model mismatch. This approach does not take closed-loop performance explicitly into account, but one may assume that there is a relation between performance and number of regions in the controller.

Note that the results in this chapter are valid both for on-line and explicit MPC.

The rest of the chapter is organized as follows: First we give background material on MPC and two common methods to speed up MPC, namely move blocking and model reduction by balanced truncation. Then we give a mathematical formulation of the problem of identifying the maximum difference between two controllers in closed loop on the same system. Thereafter we discuss some computational aspects and present some simplifications. Finally, we demonstrate the proposed method on a linearized model of a distillation column.

6.2 Background

6.2.1 System to be controlled

Consider the linear system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k, \end{aligned} \tag{6.1}$$

where $x_k \in \mathbb{R}^{n_x}$ are the states, $u_k \in \mathbb{R}^{n_u}$ are the inputs, and $y_k \in \mathbb{R}^{n_y}$ are the outputs.

6.2.2 Model predictive control (MPC)

We pose the following *open-loop* control problem in order to control the linear system (6.1):

$$\min_{x_1,\dots,x_N,u_0,\dots,u_{N-1}} \frac{1}{2}x_N' Q_N x_N + \frac{1}{2}\sum_{i=0}^{N-1}(u_i' R u_i + x_i' Q x_i) \tag{6.2}$$

$$\text{s.t.} \begin{cases} x_{k+1} = Ax_k + Bu_k, & \forall k = 0,\dots,N-1 \\ y_k = Cx_k, & \forall k = 0,\dots,N \\ F^u u \le f_u, & \forall k = 0,\dots,N-1 \\ F^y y \le f_y, & \forall k = 1,\dots,N-1 \\ F_{x_N} x_N \le f_{x_N}, & \\ x_0 = \text{given}. & \end{cases} \tag{6.3}$$

Remark 6.1. *We have here included an inequality constraint on the final state x_N, because if this is chosen carefully together with the final weight $x_N' Q_N x_N$, feasibility and stability can be guaranteed in closed loop. There are many references on this subject, see e.g. [Mayne et al., 2000, Rawlings and Mayne, 2009].*

Remark 6.2. *We assume that the MPC problem at hand is a well-posed problem with $Q \ge 0$ and $R > 0$ and that there exists a non-empty set X_0 of initial conditions x_0 such that problem* (6.3) *has a feasible solution.*

Dense formulation. The MPC problem (6.2)-(6.3) can be written on a dense form, where the states are eliminated by substituting the model equations into the objective function and constraints: First, we define the following gains with notation $G^{\text{to}}_{\text{from}}$:

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}}_{G^x_{x_0}} x_0 + \underbrace{\begin{bmatrix} B & & & \\ AB & B & & \\ \vdots & & \ddots & \\ A^{N-1}B & \dots & \dots & B \end{bmatrix}}_{G^x_u} \underbrace{\begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}}_{u} \tag{6.4}$$

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-1} \end{bmatrix}}_{G^y_{x_0}} x_0 + \underbrace{\begin{bmatrix} CB & & & & 0 \\ CAB & CB & & & 0 \\ \vdots & & \ddots & & 0 \\ CA^{N-2}B & \dots & \dots & CB & 0 \end{bmatrix}}_{G^y_u} u \tag{6.5}$$

In addition, we define the following model, corresponding to the last row of (6.4):

$$x_N = \underbrace{A^N}_{G_{x_0}^{x_N}} x_0 + \underbrace{\begin{bmatrix} A^{N-1}B & \dots & AB & B \end{bmatrix}}_{G_u^{x_N}} u. \tag{6.6}$$

We further define the following matrices:

$$\hat{F}_u = \text{diag}(F_u, \cdots, F_u) \tag{6.7}$$
$$\hat{F}_y = \text{diag}(F_y, \cdots, F_y) \tag{6.8}$$
$$\hat{Q} = \text{diag}(Q, \cdots, Q, Q_N) \tag{6.9}$$
$$\hat{R} = \text{diag}(R, \cdots, R) \tag{6.10}$$

and vectors:

$$\hat{f}_u = \begin{bmatrix} f_u \\ \vdots \\ f_u \end{bmatrix} \qquad \hat{f}_y = \begin{bmatrix} f_y \\ \vdots \\ f_y \end{bmatrix} \tag{6.11}$$

With the above definitions can we rewrite the MPC problem (6.2)-(6.3) as

$$\begin{aligned} &\min_u \begin{bmatrix} u \\ x_0 \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ux_0} \\ J'_{ux_0} & J_{x_0x_0} \end{bmatrix} \begin{bmatrix} u \\ x_0 \end{bmatrix} \\ \text{s.t.} \ &\underbrace{\begin{bmatrix} \hat{F}^u \\ \hat{F}^y G_u^y \\ F_{x_N} G_u^{x_N} \end{bmatrix}}_{F} u \leq \underbrace{\begin{bmatrix} \hat{f}_u \\ \hat{f}_y \\ \hat{f}_{x_N} \end{bmatrix}}_{f} + \underbrace{\begin{bmatrix} 0 \\ -\hat{F}^y G_{x_0}^y \\ -\hat{F}_{x_N} G_{x_0}^{x_N} \end{bmatrix}}_{E} x_0 \end{aligned} \tag{6.12}$$

with

$$J_{uu} = G_u^{x\prime} \bar{Q} G_u^x + \bar{R}, \tag{6.13}$$
$$J_{ux_0} = G_u^{x\prime} \bar{Q} G_{x_0}^x, \tag{6.14}$$
$$J_{x_0x_0} = G_{x_0}^{x\ \prime} \bar{Q} G_{x_0}^x. \tag{6.15}$$

First order optimality conditions. The first order optimality conditions (referred to as Karush-Kuhn-Tucker (KKT) conditions, see [Nocedal and Wright, 1999] for an introduction), for the MPC problem (6.12) are:

$$J_{uu}u + J'_{ux_0}x_0 + F'\lambda = 0 \tag{6.16}$$
$$Fu \leq f + Ex_0 \tag{6.17}$$
$$\lambda \geq 0 \tag{6.18}$$
$$\lambda'(Fu - f - Ex_0) = 0 \tag{6.19}$$

where λ are the so-called Lagrange multipliers for the function

$$\mathcal{L}(u,\lambda) = \frac{1}{2}u'J_{uu}u + u'J_{ud}d - \lambda'(-Fu + f + Ex_0) \tag{6.20}$$

Mixed integer formulation of complimentary constraints. From equation (6.19), often referred to as a complimentary constraint between the inequality $Fu \leq f + Ex_0$ and the Lagrange multiplier λ, we see that the KKT-conditions are bilinear in u and λ. This equation can be replaced with the following two equations:

$$\lambda \leq Ms \tag{6.21}$$

$$Fu \geq f + Ex_0 - M(1-s) \tag{6.22}$$

Here $s \in \{0,1\}^{n_f}$ is a vector of binary variables corresponding to the number of inequality constraints in problem (6.12), and M is a constant large enough such that the solution does not change when equation (6.19) is replaced with the inequalities (6.21)-(6.22). (This is often referred to as a "big-M" formulation.) Note that even though some references claim that that the value of M should be set "arbitrarily large," it should only be as large as necessary to not affect the solution of the equations, but not larger, as a too large M will render the problem intractable [Camm et al., 1990]. This is because both solution time and the reciprocal of the precision of the solution will increase with an increasing value of M.

6.2.3 Move blocking

Move blocking schemes comes in many different fashions, see [Cagienard et al., 2007] for an overview. We here cover "input blocking" and "delta input blocking". In addition, one can choose to parameterize the input sequence in terms of deviation from a reference controller, typically the linear quadratic regulator (LQR). We then have $u_k = K_{\text{LQR}}x_k + c_k$, where the deviations c_k are the new degrees of freedom, and we add blocking constraints to these new variables. The methods presented in this chapter can be used also for the latter case, but we do not discuss this method further here.

Figure 6.1 shows a sketch of input blocking and delta input blocking. A special case of input blocking would be to keep all moves the same, which would correspond to a single-move MPC. The blocking schemes can be expressed on the from $Wu = 0$, and we now give examples for how to find W (the "blocking matrix") for input blocking and delta input blocking.

Input blocking. As in [Cagienard et al., 2007], consider a SISO system with prediction horizon $N = 4$ and corresponding degrees of freedom u_0, u_1, u_2, u_3. We

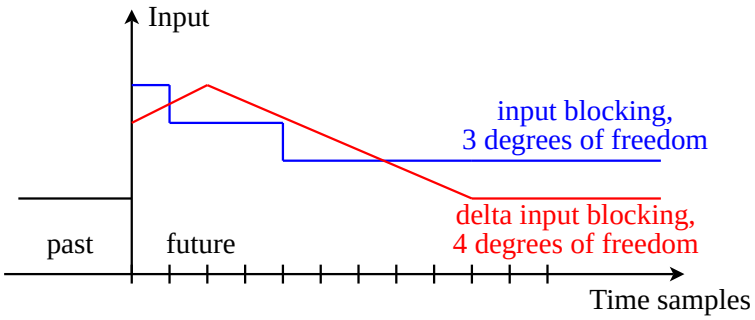

Figure 6.1: Different types of move blocking.

want to add a blocking such that u_0 is free, while $u_1 = u_2 = u_3$, thus reducing the degrees of freedom in a dense MPC formulation from 4 to 2. This is achieved by adding the equation

$$\begin{bmatrix} 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{6.23}$$

to the control problem.

Delta input blocking. Consider the same example, but now we want to constrain the *difference between the inputs* to remain constant. For the SISO example with the four degrees of freedom u_0, u_1, u_2, u_3 we can reduce the number of degrees of freedom Nn_u to 2 by adding the equations $u_1 - u_0 = u_2 - u_1 = u_3 - u_2$, which on matrix from can be written as

$$\begin{bmatrix} -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{6.24}$$

Optimality conditions for MPC with move blocking. Consider the optimality conditions (6.16)-(6.19) for the MPC problem (6.12). When move blocking is introduced we need to augment the Lagrangian function to

$$\mathcal{L}(u, \lambda, \gamma) = \frac{1}{2} u' J_{uu} u + x_0' J_{ux_0} u - \lambda'(-Fu + f + Ex_0) - \gamma' W u, \tag{6.25}$$

where γ are the Lagrange multipliers corresponding to the move blocking equality constraints $Wu = 0$. The KKT-conditions for the case of move blocking are then:

$$J_{uu}u + J_{ux_0}x_0 + F'\lambda - W'\gamma = 0 \tag{6.26}$$

$$Wu = 0 \tag{6.27}$$

Equations (6.17)-(6.19)

6.2.4 Balanced truncation

We here review model reduction by balanced truncation [Moore, 1981] as an example of a model reduction scheme that can be analyzed with the proposed method.

The model reduction by balanced truncation consists of two steps: First, we find a balanced representation of system (6.1), then we remove the states corresponding to the smallest Hankel singular values of the balanced representation.

Balanced representation. The controllability and observability Gramians of a linear system are defined as

$$AW_c + W_cA' + BB' = 0 \tag{6.28}$$

$$A'W_o + W_oA + C'C = 0 \tag{6.29}$$

A balanced representation of system (6.1) is obtained through a transformation matrix T, such that $\bar{W}_c$ and $\bar{W}_o$ (of the transformed system) are equal. Let $\tilde{z}_k$ denote the states of the balanced system, i.e. $\tilde{z}_k = Tx_k$. It can be shown that

$$\begin{aligned} \bar{W}_c = \bar{W}_o &= \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{n_x}) \\ \bar{W}_c &= TW_cT^{-1} \\ \bar{W}_o &= (T^{-1})'W_oT^{-1} \end{aligned} \tag{6.30}$$

where σ_i, $k = 1, 2, \ldots, n_x$ are the Hankel singular values of the balanced representation, ordered according to

$$\sigma_1 > \sigma_2 > \cdots > \sigma_{n_x} \geq 0.$$

Truncation. Let $\tilde{z}_k' = \begin{bmatrix} z_k' & z_k^{\text{res}\prime} \end{bmatrix}$. In balanced truncation we simply delete z_k^{res} from the vector of balanced states $\tilde{x}_k$. Denote T_l and T_r as

$$T = \begin{bmatrix} \overbrace{\begin{bmatrix} T_{11} & \cdots & T_{1n} \\ \vdots & & \vdots \\ T_{\tilde{n}1} & \cdots & T_{\tilde{n}n} \end{bmatrix}}^{T_l} \\ \vdots \\ T_{n1} \quad \cdots \quad T_{nn} \end{bmatrix}, \quad T^{-1} = \begin{bmatrix} \underbrace{\begin{bmatrix} T_{11}^{-1} & \cdots & T_{1\tilde{n}}^{-1} \\ \vdots & & \vdots \\ T_{n1}^{-1} & \cdots & T_{n\tilde{n}}^{-1} \end{bmatrix}}_{T_r} \cdots & T_{1n}^{-1} \\ & \vdots \\ & T_{nn}^{-1} \end{bmatrix} \tag{6.31}$$

We can now express the balanced and truncated result as

$$\begin{aligned} z_{k+1} &= T_l A T_r z_k + T_l B u_k \\ \bar{y}_k &= C^c T^r z_k + D u_k, \end{aligned} \tag{6.32}$$

and we note that the map from the full state vector x_k to the balanced and truncated system (6.32) is given by $z_k = T_l x_k$.

6.3 Problem formulation

Consider the controlled system in figure 6.2, where we have included a disturbance model and a state observer. The objective is to identify a worst-case disturbance sequence w^* such that the weighted difference

$$\text{diff} = \sum_{i=1}^{N_{\text{sim}}} \|Q(x_i^c - x_i^r)\|_\infty + \|R(u_i^c - u_i^r)\|_\infty \tag{6.33}$$

is maximized. Here superscript "c" means "candidate controller," and "r" means "reference controller". The parameter N_{sim} is the length of a time period for which we are comparing the controllers.

The idea is that the reference controller has been tuned to give a good trade-off between performance and robustness for the linear system to be controlled, but that this controller is too computationally demanding to be implemented. The candidate controller is a simplified alternative, where different simplifications have been used, i.e. model reduction or move blocking. The objective is to find the performance degradation in terms of the weighted difference 'diff' in equation (6.33).

6.3.1 Notation

The system to be investigated in Figure 6.2 consists of three linear systems (disturbance model, linear process and state observer) and one optimization problem (MPC).

Disturbance model. Let the disturbance dynamics be described by

$$d_{k+1} = A_d d_k + B_d w_k \tag{6.34}$$

$$d_k^O = C_d^O d_k \tag{6.35}$$

$$d_k^I = C_d^I d_k. \tag{6.36}$$

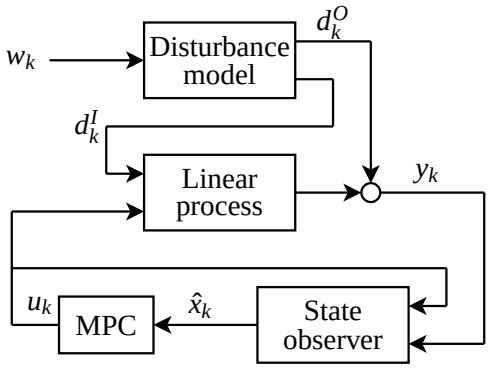

Figure 6.2: Linear system with MPC controller, state observer and disturbance model.

For a time-sequence $1, \cdots, N_{\text{sim}}$ we have

$$\underbrace{\begin{bmatrix} d_1^O \\ d_2^O \\ \vdots \\ d_{N_{\text{sim}}-1}^O \end{bmatrix}}_{d^O} = \underbrace{\begin{bmatrix} C_d^O A_d \\ C_d^O A_d^2 \\ \vdots \\ C_d^O A_d^{N_{\text{sim}}-1} \end{bmatrix}}_{G_{x_0}^{d^O}} d_0 + \underbrace{\begin{bmatrix} C_d^O B_d & & & \\ C_d^O A_d B_d & C_d^O B_d & & \\ \vdots & & \ddots & \\ C_d^O A_d^{N_{\text{sim}}-2} & \cdots & \cdots & C_d^O B_d \end{bmatrix}}_{G_w^{d^O}} \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N_{\text{sim}}-1} \end{bmatrix}}_{w} \tag{6.37}$$

$$\underbrace{\begin{bmatrix} d_1^I \\ d_2^I \\ \vdots \\ d_{N_{\text{sim}}-1}^I \end{bmatrix}}_{d^I} = \underbrace{\begin{bmatrix} C_d^I A_d \\ C_d^I A_d^2 \\ \vdots \\ C_d^I A_d^{N_{\text{sim}}-1} \end{bmatrix}}_{G_{x_0}^{d^I}} d_0 + \underbrace{\begin{bmatrix} C_d^I B_d & & & \\ C_d^I A_d B_d & C_d^I B_d & & \\ \vdots & & \ddots & \\ C_d^I A_d^{N_{\text{sim}}-2} & \cdots & \cdots & C_d^I B_d \end{bmatrix}}_{G_w^{d^I}} \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N_{\text{sim}}-1} \end{bmatrix}}_{w} \tag{6.38}$$

Linear process. The linear process

$$\begin{aligned} z_{k+1} &= A_l z_k + B_l^u u_k + B_l^{d_I} d_k^I \\ y_k &= C_l z_k + D_l^{d_O} d_k^O \end{aligned} \tag{6.39}$$

may have slightly different dynamics than the internal model in the MPC. Over N_{sim} sample times we write the model as

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N_{\text{sim}}} \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} C_l B_l^u & & & \\ C_l A_l B_l^u & C_l B_l^u & & \\ \vdots & & \ddots & \\ C_l A_l^{N_{\text{sim}}-1} B_l^u & \cdots & \cdots & C_l B_l^u \end{bmatrix}}_{G_u^y} \underbrace{\begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_{\text{sim}}-1} \end{bmatrix}}_{u} + \underbrace{\begin{bmatrix} C_l B_l^{d^I} & & & \\ C_l A_l B_l^{d^I} & C_l B_l^{d^I} & & \\ \vdots & & \ddots & \\ C_l A_l^{N_{\text{sim}}-1} B_l^{d^I} & \cdots & \cdots & C_l B_l^{d^I} \end{bmatrix}}_{G_{d^I}^y} d^I + \underbrace{\begin{bmatrix} D^{d^O} & & \\ & \ddots & \\ & & D^{d^O} \end{bmatrix}}_{G_{d^O}^y} d^O \tag{6.40}$$

State observer. We consider a simple Luenberger observer [Luenberger, 1966] on the form

$$\begin{aligned} \hat{x}_{k+1} &= A\hat{x}_k + L(y_k - \hat{y}_k) + Bu_k \\ \hat{y}_k &= Cx_k, \end{aligned} \tag{6.41}$$

or equivalently

$$\hat{x}_{k+1} = \underbrace{(A - LC)}_{A_O} \hat{x}_k + Ly_k + Bu_k \tag{6.42}$$

and we assume that A_O is stable. Over N_{sim} time samples we have

$$\underbrace{\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \vdots \\ \hat{x}_{N_{\text{sim}}} \end{bmatrix}}_{\hat{x}} = \underbrace{\begin{bmatrix} L & & & \\ A_O L & L & & \\ \vdots & & \ddots & \\ A_O^{N_{\text{sim}}-1} L & \cdots & \cdots & L \end{bmatrix}}_{G_y^{\hat{x}}} y + \underbrace{\begin{bmatrix} B & & & \\ A_O B & B & & \\ \vdots & & \ddots & \\ A_O^{N_{\text{sim}}-1} B & \cdots & \cdots & B \end{bmatrix}}_{G_u^{\hat{x}}} u. \tag{6.43}$$

MPC. Let K_{MPC} denote the map $\{K_{\text{MPC}} : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_u}\}$, i.e. the first part of the sequence u^* that solves an MPC problem. We then have that

$$u = \underbrace{\begin{bmatrix} K_{\text{MPC}} & & \\ & \ddots & \\ & & K_{\text{MPC}} \end{bmatrix}}_{\mathbb{K}_{\text{MPC}}} \hat{x}, \tag{6.44}$$

where $\mathbb{K}_{\text{MPC}}$ is a variable gain that depends on $\hat{x}$, where $\hat{x} = \begin{bmatrix} \hat{x}_1' & \hat{x}_2' & \dots & \hat{x}_{N_{\text{sim}}}' \end{bmatrix}'$.

6.3.2 Problem statement

The problem we attempt to solve can be summarized as:

$$\begin{aligned} &\max_{w \in \mathcal{W}} \sum_{i=1}^{N_{\text{sim}}} \|Q(y_i^c - y_i^r)\|_\infty + \|R(u_i^c - u_i^r)\|_\infty \\ &\text{s.t. } d^O = G_w^{d^O} w, \quad d^I = G_w^{d^I} w, \\ &\qquad y^c = G_u^y u^c + G_{d^I}^y d^I + G_{d^O}^y d^O, \quad y^r = G_u^y u^r + G_{d^I}^y d^I + G_{d^O}^y d^O, \\ &\qquad \hat{x}^c = G_y^{\hat{x}} y^c + G_u^{\hat{x}} u^c, \qquad\qquad \hat{x}^r = G_y^{\hat{x}} y^r + G_u^{\hat{x}} u^r, \\ &\qquad u^c = \mathbb{K}_{\text{MPC}}^c \hat{x}^c, \qquad\qquad u^r = \mathbb{K}_{\text{MPC}}^r \hat{x}^r. \end{aligned} \tag{6.45}$$

Here $\mathbb{K}_{\text{MPC}}^c$ indicates that a candidate MPC formulation is used, whereas $\mathbb{K}_{\text{MPC}}^r$ indicates the reference controller. The search space for the disturbances is given by the polytope $\mathcal{W}$.

Remark 6.3. *Rather than considering a linear state observer as done in problem* (6.45)*, one may include a finite horizon estimator. Note however that the complexity of the problem increases significantly when a finite horizon estimator is used.*

6.3.3 Solution by mixed integer linear programming

Problem (6.45) is referred to as a *bilevel program* [Bard, 1998, Colson et al., 2005] because we have two "levels" of optimization, the "upper level" of maximizing the difference between the controllers, and the "lower level" representing the solutions to the MPC problems (which are themselves optimization problems). If the lower-level problems happens to be convex and regular, they can be replaced by their first order optimality conditions, yielding a single-level optimization problem (see for example [Jones and Morari, 2009]). Under the assumption that we have a positive semi-definite weight $Q \geq 0$ on the states and a positive definite weight $R > 0$ on the inputs, the MPC formulations will fulfill this assumption. Thus, we do not look for an explicit solution (controller) to the lower-level problems, but we solve for the corresponding inputs by using the first-order optimality conditions.

MPC as mixed integer constraints. We define the operator

$$\text{KKT} : \mathbb{R}^{n_x} \mapsto \{\text{first-order optimality conditions on "big-}M\text{ form"}\}, \tag{6.46}$$

i.e. an operation that writes the KKT-conditions given in section 6.2 for a given MPC formulation. Further, let the operation

$$\mathbb{KKT} : \mathbb{R}^{N_{\text{sim}} n_x} \mapsto N_{\text{sim}} \text{ independent instances of the KKT operator above} \tag{6.47}$$

have as input the vector of stacked states $\hat{x}$. This operation then writes the KKT-conditions for N_{sim} steps forward in time, i.e. N_{sim} MPC calculations. We can now do the following substitution:

$$(u^c = \mathbb{K}^c_{\text{MPC}} \hat{x}^c,\, u^r = \mathbb{K}^r_{\text{MPC}} \hat{x}^r) \Leftrightarrow (\mathbb{KKT}^c \hat{x}^c,\, \mathbb{KKT}^r x^r) \tag{6.48}$$

We here substitute the lower level problems with linear constraints, and add new variables, some of which are binary.

Objective function. The objective function is still non-convex, but it can be made convex by a standard trick, see e.g. [Jones and Morari, 2009, Löfberg, 2004]: The objective function is on the form "max $\|t\|_\infty$". If we introduce binary variables n_i and p_i for each element of t and add the condition that the binary variable p_i is 1 if $\|t\|_\infty = t_i$ and n_i is 1 if $\|t\|_\infty = -t_i$, will the remaining problem be a mixed-integer linear program (MILP). In this work we used YALMIP [Löfberg, 2004] to pose the problem, and this software has a built-in facility for this "trick".

Summing up, if we replace the MPC problems with their respective KKT conditions and add binary variables to render the objective function linear do we get an MILP problem in the end, which we can solve using standard software. In this work, the commercially available software CPLEX© was used.

6.4 Computational aspects

After replacing the MPC problems with their respective first order optimality conditions (modelled using binary variables for the complimentary constraints) and noting that the maximization of an infinity norm can be rendered into a mixed-integer linear program we are left with an MILP problem:

$$\begin{aligned}
&\max_{w \in \mathcal{W}} \quad \sum_{i=1}^{N_{\text{sim}}} \|Q(y_i^c - y_i^r)\|_\infty + \|R(u_i^c - u_i^r)\|_\infty \\
&\text{s.t. } d^O = G_w^{d^O} w, \quad d^I = G_w^{d^I} w, \\
&\quad y^c = G_u^y u^c + G_{d^I}^y d^I + G_{d^O}^y d^O, \quad y^r = G_u^y u^r + G_{d^I}^y d^I + G_{d^O}^y d^O, \\
&\quad \hat{x}^c = G_y^{\hat{x}} y^c + G_u^{\hat{x}} u^c, \qquad\qquad \hat{x}^r = G_y^{\hat{x}} y^r + G_u^{\hat{x}} u^r, \\
&\quad\ \ \mathbb{KKT}^c \hat{x}^c, \qquad\qquad\qquad\ \ \mathbb{KKT}^r \hat{x}^r.
\end{aligned}$$

Origin	Number of binary variables
Objective function	$N_{\text{sim}}(2n_y + 2n_u)$
KKT conditions for MPC problems	$N_{\text{sim}}(N^c n_c + N^r n_c)$

Table 6.2: Number of binary variables by solving problem (6.33) as a MILP problem. The MPC problems have an input horizon of N^c for candidate controller and N^r for reference controller, both with n_c constraints.

Table 6.2 shows approximately how many binary variables that are used in the problem formulation. To get some insight into these numbers, consider the following example:

Example: Distillation column. We consider a typical distillation column, see e.g. [Skogestad, 1997], with 2 inputs and 2 measurements. Consider the simplest case of only input constraints on the form $u^l \leq u \leq u^l$, so the number of constraints $n_c = 2n_u = 4$. Assume that one decides that a prediction horizon of $N = 20$ is sufficient, and that both reference controller and candidate controllers have this prediction horizon. Further let us assume that we want to simulate the control for $N_{\text{sim}} = 40$ time steps. We then get the following number of binary variables in our formulation:

Origin	Number of binary variables
Objective function	$40(2 \cdot 2 + 2 \cdot 2) = 320$
KKT conditions for MPC problems	$40(20 \cdot 4 + 20 \cdot 4) = 51200$

A number of binary variables of 51520 corresponds to that the solver used for finding solving the MILP in the worst case has to explore 2^{51520} nodes, which may take extremely long time. It is well-known that MILP problems are NP-hard (NP-hard problems belong to a class of problems that cannot be solved in polynomial time [Blondel and Tsitsikilis, 1997]), but nevertheless there exists commercial software that can solve these problems within reasonable time. What renders the problem we have posed even more difficult, is probably the structure of the complimentary constraints originating from the KKT conditions of the lower level of the bilevel program, because even for the simplest instance of a bilevel program, the linear bilevel programming problem is shown to be NP-hard [Colson et al., 2005].

6.4.1 Simplifications

We now present some simplifications that effectively lets us only find a lower bound on the maximum difference (rather than the exact value), but that renders the remaining problem tractable.

Initial state of disturbance model as only disturbance. In the problem formulation shown graphically in Figure 6.2, the "generators" w_k drive the disturbance model. In the equations we have also included the initial state d_0 as a degree of freedom. In optimal control, see for example [Kwakernaak and Sivan, 1972, Theorem 3.9, page 260], is it well-known that for a linear system (without constraints), a linear feedback that is optimal for rejecting an initial disturbance d_0 is also optimal for sustained white noise disturbances on the form $x_{k+1} = (A - BK)x_k + w_k$.

This result is only valid for linear controllers on linear systems *without* constraints, but here we suggest to ignore the generators w_k and only search over initial states of the disturbance model, d_0, which should by the similarity to the result above from optimal control, lead to the same classification of the candidate controllers.

Using fewer points in the objective function. With the simplification of considering only the initial state as our disturbances, we suggest to simplify the problem by, rather than evaluating the whole trajectory in the objective function, i.e. $\sum_{i=1}^{N_{\text{sim}}} \|Q(y_i^c - y_i^r)\|_\infty + \|R(u_i^c - u_i^r)\|_\infty$, only considering a few points, for example the midpoint and the end point $t = T_s \cdot N_{\text{sim}}$. We here assume that an initial disturbance d_0^*, that maximizes the difference between the controllers at the end point, *also* maximizes the difference over the whole trajectory. However, for long "simulation times" N_{sim}, this problem maybe badly posed (as the difference may be very small at the end point for all d_0 in the set of initial disturbances that we want to investigate), so we then suggest to add some more points, for example the mid point, as indicated in Figure 6.3.

From Table 6.2 it seems like "simplifying" the objective function as outlined above, does not have a significant influence on the solution time, as most of the binary variables come from the KKT conditions of the MPCs. However, by numerical experiments we found that this simplification made the problem solve a lot faster.

Early termination of the MILP solver. While testing the method proposed in this chapter we found that the MILP solver (CPLEX) actually found an acceptable solution to the problem of maximizing the difference between two controller quite fast, but that it took a long time to converge to the actual solution. A typical

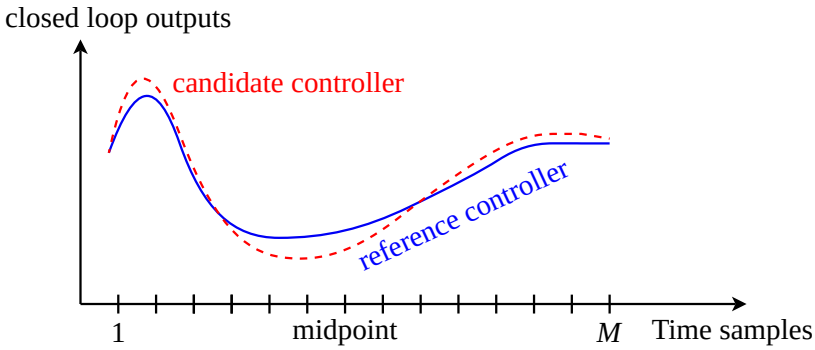

Figure 6.3: Illustration of two closed-loop trajectories, one for the reference controller (blue line) and another for the candidate controller (dashed red line).

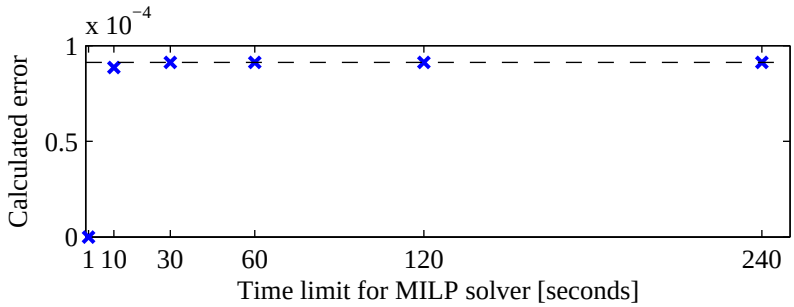

Figure 6.4: Sensitivity to time limit.

example is shown in Figure 6.4, where by changing the time limit on the solver we get essentially the same solution after 10 seconds as after two minutes. We also investigated the sensitivity to the tolerances in CPLEX (absolute and relative gaps, and absolute and relative objective function values), but we found that the most effective solution was to simply add a time-limit to the solver.

6.5 Example: Distillation

We here consider model predictive control of "Column-A" in [Skogestad, 1997], with the "LV-configuration". Muske and Badgwell [2002] also used the same column model as an example for offset-free MPC, and we here use the same operating point as they did, which is reported in Table 6.3.

In order to use the distillation model in this example, we first linearize the

Type	Description	Variable		Nominal Value
Input	Reflux flow	L	(u_1)	1.87
	Vapor flow	V	(u_2)	2.37
Disturbance	Feed flow	F	(d_1)	1.0
	Feed composition	z_F	(d_2)	0.5
	Liquid feed fraction	q_F		1.0
Output	Overhead composition (light component)	x_D	(y_1)	0.95
	Bottoms composition (light component)	x_B	(y_2)	0.05

Table 6.3: Variables for distillation example. Variable names in parenthesis indicate corresponding deviation variables, for example $u_1 = L - 1.87$. The liquid feed fraction q_F is assumed constant.

model around the nominal operating point to get an 82 state model. In addition, we sample the model to get a discrete system, with sample time $T_s = 2$ minutes. We further simplify the model by balanced truncation to find a corresponding 16 state model. We propose the following objective function for control:

$$J = x_{16}' Q_N x_{16} + \sum_{i=0}^{15} y_i' Q y_i + u_i' R u_i \tag{6.49}$$

with $Q = R = I$.

The disturbance space is:

$$-1 \leq d_i \leq 1, \quad i = 1,2. \tag{6.50}$$

By simulation one finds that, with the original formulation, input saturation is not likely to happen. In order to investigate this effect, we therefore tighten the input constraints and use in the following:

$$-0.02 \leq u_i \leq 0.02, \quad i = 1,2. \tag{6.51}$$

The final state weight matrix Q_N is found by first finding the LQR feedback gain $u_k = -K_{\mathrm{LQR}} x_k$ corresponding to an infinite objective function with weights $(C'QC, R)$ on states and inputs, respectively. Then, we define

$$A_K = (A - BK_{\mathrm{LQR}}) \tag{6.52}$$

$$Q_f = C'QC + K_{\mathrm{LQR}}' R K_{\mathrm{LQR}} \tag{6.53}$$

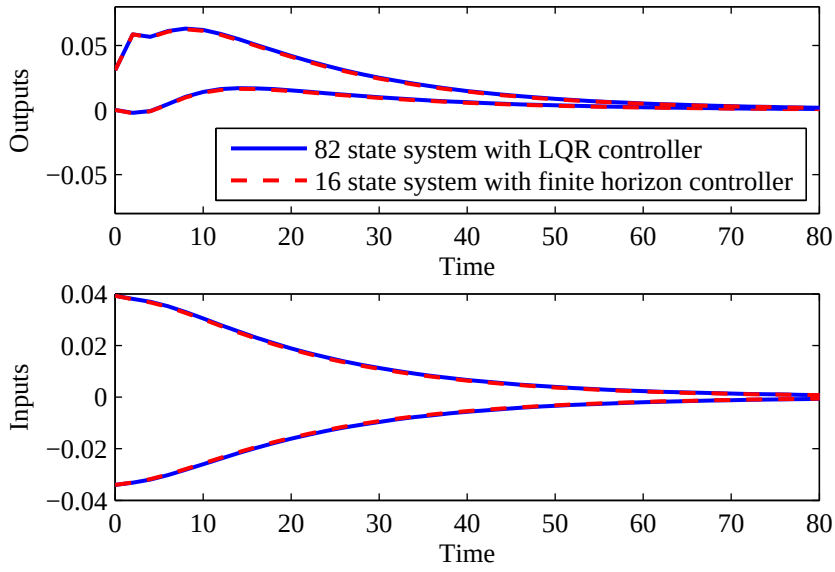

Figure 6.5: Responses of unconstrained systems from an impulse disturbance $d_0 = (-1, 1)$.

and let $Q_N > 0$ satisfy the Lyapunov equation

$$A_K' Q_N A_k + Q_f = Q_N, \tag{6.54}$$

see [Chmielewski and Manousiouthakis, 1996] for details. The matrix $Q_N =$

$$\begin{bmatrix}
0.8797 & -0.0072 & 0.0161 & -0.0241 & -0.0191 & 0.0125 & 0.0356 & -0.0078 & -0.0023 & 0.0182 & -0.0154 & 0.0023 & 0.0130 & -0.0074 & 0.0070 & -0.0062 \\
-0.0072 & 0.0975 & -0.0042 & -0.0082 & -0.0140 & -0.0031 & 0.0156 & -0.0039 & 0.0088 & -0.0006 & -0.0096 & -0.0118 & -0.0002 & -0.0037 & 0.0050 & 0.0014 \\
0.0161 & -0.0042 & 0.1009 & 0.0155 & 0.0171 & -0.0078 & -0.0258 & 0.0084 & 0.0029 & -0.0162 & 0.0106 & -0.0039 & -0.0111 & 0.0063 & -0.0045 & 0.0056 \\
-0.0241 & -0.0082 & 0.0155 & 0.0442 & 0.0263 & -0.0025 & -0.0222 & 0.0126 & -0.0138 & -0.0105 & 0.0120 & 0.0059 & -0.0092 & 0.0096 & -0.0054 & 0.0022 \\
-0.0191 & -0.0140 & 0.0171 & 0.0263 & 0.0221 & 0.0018 & -0.0131 & 0.0111 & -0.0111 & -0.0054 & 0.0082 & 0.0058 & -0.0057 & 0.0066 & -0.0039 & 0.0009 \\
0.0125 & -0.0031 & -0.0078 & -0.0025 & 0.0018 & 0.0138 & 0.0011 & -0.0000 & -0.0065 & 0.0066 & 0.0013 & 0.0080 & 0.0042 & -0.0001 & -0.0010 & -0.0029 \\
0.0356 & 0.0156 & -0.0258 & -0.0222 & -0.0131 & 0.0011 & 0.0200 & -0.0048 & 0.0057 & 0.0075 & -0.0096 & -0.0041 & 0.0055 & -0.0055 & 0.0043 & -0.0018 \\
-0.0078 & -0.0039 & 0.0084 & 0.0126 & 0.0111 & -0.0000 & -0.0048 & 0.0066 & -0.0041 & -0.0033 & 0.0031 & 0.0014 & -0.0032 & 0.0032 & -0.0014 & 0.0007 \\
-0.0023 & 0.0088 & 0.0029 & -0.0138 & -0.0111 & -0.0065 & 0.0057 & -0.0041 & 0.0110 & -0.0024 & -0.0053 & -0.0082 & 0.0001 & -0.0031 & 0.0027 & 0.0014 \\
0.0182 & -0.0006 & -0.0162 & -0.0105 & -0.0054 & 0.0066 & 0.0075 & -0.0033 & -0.0024 & 0.0092 & -0.0015 & 0.0044 & 0.0056 & -0.0025 & 0.0007 & -0.0028 \\
-0.0154 & -0.0096 & 0.0106 & 0.0120 & 0.0082 & 0.0013 & -0.0096 & 0.0031 & -0.0053 & -0.0015 & 0.0058 & 0.0042 & -0.0018 & 0.0030 & -0.0025 & 0.0001 \\
0.0023 & -0.0118 & -0.0039 & 0.0059 & 0.0058 & 0.0080 & -0.0041 & 0.0014 & -0.0082 & 0.0044 & 0.0042 & 0.0082 & 0.0020 & 0.0015 & -0.0021 & -0.0019 \\
0.0130 & -0.0002 & -0.0111 & -0.0092 & -0.0057 & 0.0042 & 0.0055 & -0.0032 & 0.0001 & 0.0056 & -0.0018 & 0.0020 & 0.0039 & -0.0022 & 0.0009 & -0.0017 \\
-0.0074 & -0.0037 & 0.0063 & 0.0096 & 0.0066 & -0.0001 & -0.0055 & 0.0032 & -0.0031 & -0.0025 & 0.0030 & 0.0015 & -0.0022 & 0.0023 & -0.0014 & 0.0005 \\
0.0070 & 0.0050 & -0.0045 & -0.0054 & -0.0039 & -0.0010 & 0.0043 & -0.0014 & 0.0027 & 0.0007 & -0.0025 & -0.0021 & 0.0009 & -0.0014 & 0.0012 & -0.0000 \\
-0.0062 & 0.0014 & 0.0056 & 0.0022 & 0.0009 & -0.0029 & -0.0018 & 0.0007 & 0.0014 & -0.0028 & 0.0001 & -0.0019 & -0.0017 & 0.0005 & -0.0000 & 0.0010
\end{bmatrix}.$$

Figure 6.5 shows a closed loop simulation of the state feedback controller (LQR) on the 82 state system and the LQR that minimizes objective (6.49) applied to the 16 state system. We observe that the input-output relationships of the two systems in closed loop are almost indistinguishable.

The reference controller is thus the following MPC:

$$\begin{aligned} J^*(x_0) = \min_{u_k, x_k} \; & x_{16}' P x_1 6 + \sum_{i=0}^{15} y_i' Q y_i + u_i' R u_i \\ \text{s.t.} \quad & x_{k+1} = A x_k + B u_k, \qquad k = 0, \ldots, 15 \\ & -0.02 \leq u_k \leq 0.02, \qquad k = 0, \ldots, 15 \end{aligned} \tag{6.55}$$

We consider both dense and structured formulations of the MPC problem. From Table 6.1 we observe that in order to reduce the CPU-time for a structured formulation, we can change the prediction horizon (N) or reduce the model order (n_x). For the dense formulation we can either change the prediction horizon N or the product Nn_u, which we can address using move blocking.

Definition of analysis problem. We consider impulse disturbances at time sample $t = 1$, and we define the objective to maximize the difference in outputs after 5 time steps (corresponding to 10 minutes), i.e. the difference function is

$$\text{diff} = \|y_5^r - y_5^c\|_\infty.$$

We use the same linear model as in the reference controller, and we assume that the state vector is available as a measured variable. For the case of a reduced internal model in the MPC, we assume that the state observer is simply given by $z_k = T_l x_k$, see section 6.2.4.

Results. The results for the dense formulation (where we assume a CPU-time in the order of $(Nn_u)^3$) are shown in Tables 6.4-6.5 and Figure 6.6. The first conclusion is that it is significantly better to apply move blocking than to reduce the horizon of the controller. This may be because the horizon was quite short in the reference controller, so the control action is sensitive to changes in the horizon.

The next conclusion is that delta-blocking generally gives better performance. Notice that for delta input blocking we have one case where we get better performance with less degrees of freedom (3 versus 5 degrees of freedom), and both blocking schemes seems to be reasonable. This is a bit counter intuitive, but here we are maximizing the difference between a candidate controller and a reference controller at a certain point in time, which is not exactly the same as checking the performance of the controllers directly (though the problems are expected to be related).

We further notice from Figure 6.6 that it seems to be better to keep the original horizon and block all the moves (corresponding to the original DMC controller [Cutler and Ramaker, 1980]), than to reduce the horizon to one.

Blocking type	DOF $= \frac{Nn_u - n_W}{n_u}$	Maximum error
First 13 free, last 2 blocked	14	$3.0 \cdot 10^{-8}$
First 12 free, last 3 blocked	13	$1.6 \cdot 10^{-7}$
First 11 free, last 4 blocked	12	$6.9 \cdot 10^{-7}$
First 10 free, last 5 blocked	11	$1.4 \cdot 10^{-6}$
First 9 free, last 6 blocked	10	$3.1 \cdot 10^{-6}$
First 8 free, last 7 blocked	9	$9.4 \cdot 10^{-6}$
First 7 free, last 8 blocked	8	$1.7 \cdot 10^{-5}$
$(1,2,2,4,6)$	5	$7.7 \cdot 10^{-6}$
$(1,3,4,7)$	4	$1.4 \cdot 10^{-5}$
$(1,7,7)$	3	$5.0 \cdot 10^{-5}$
First free, last 14 blocked	2	$1.5 \cdot 10^{-4}$
All blocked	1	0.0033
Reference controller	16	0

Table 6.4: Details of input blocking. The notation $(1,2,2,4,6)$ means that the first input is free, then the next two are blocked (constrained to be the same) and so on. The degrees of freedom (DOF) $(Nn_u - n_W)/n_u$ are the original degrees of freedom Nn_u minus the number of blocking relations n_W and divided by n_u to get a number that corresponds to an equivalent horizon length. For all the controllers, $N = 16$ and $n_u = 2$.

Blocking type	DOF $= \frac{Nn_u - n_W}{n_u}$	Maximum error
First 12 free, gradient of last 3 constant	14	$1.4 \cdot 10^{-9}$
First 11 free, gradient of last 4 constant	13	$8.2 \cdot 10^{-9}$
First 10 free, gradient of last 5 constant	12	$2.9 \cdot 10^{-8}$
First 9 free, gradient of last 6 constant	11	$7.6 \cdot 10^{-8}$
First 2 free, two gradients of length 7 constant	5	$8.7 \cdot 10^{-6}$
First free, last 14 same gradient	3	$8.1 \cdot 10^{-6}$
All have the same gradient	2	0.0017
Reference controller	16	0

Table 6.5: Details of delta input blocking. The degrees of freedom (DOF) $(Nn_u - n_W)/n_u$ are the original degrees of freedom Nn_u minus the number of blocking relations n_W and divided by n_u to get a number that corresponds to an equivalent horizon length. For all the controllers, $N = 16$ and $n_u = 2$.

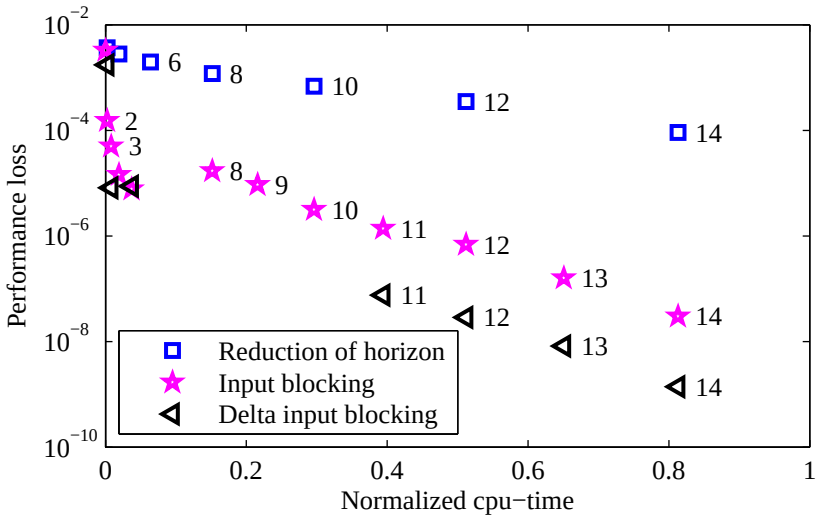

Figure 6.6: Dense formulation with Cpu-time in the order of $(Nn_u)^3$. The normalized Cpu-time in the plot is $(Nn_u)^3/(N^r n_u^r)^3$ where superscript ^{r} refers to the reference controller. The numbers next to the symbols correspond to the normalized degrees of freedom DOF $= (Nn_u - n_W)/n_u$ where n_W is the number of blocking relations applied to the problem.

Order of reduced model (number of states)	Maximum error
2	$2.11 \cdot 10^{-4}$
4	$3.61 \cdot 10^{-4}$
6	$1.45 \cdot 10^{-5}$
8	$1.26 \cdot 10^{-5}$
10	$1.24 \cdot 10^{-6}$
12	$1.43 \cdot 10^{-6}$
16 (Reference controller)	0

Table 6.6: Details of model reduction.

Horizon in candidate controller	Maximum error
2	$3.65 \cdot 10^{-3}$
4	$2.79 \cdot 10^{-3}$
6	$1.96 \cdot 10^{-3}$
8	$1.20 \cdot 10^{-3}$
10	$6.86 \cdot 10^{-4}$
12	$3.53 \cdot 10^{-4}$
14	$9.12 \cdot 10^{-5}$
16 (Reference controller)	0

Table 6.7: Details of changing the horizon length.

The details of the blocking schemes were selected by intuition and there may certainly be other schemes that are better given a certain number of degrees of freedom.

When using a solver that exploits the structure of the problem (with assumed number of operations per step in the order of $N(n_x + n_u)^3$), we observe from Figure 6.7 that it is far better to apply model reduction than to reduce the horizon. Additional details are given in Tables 6.6-6.7.

Finally, we show in Figure 6.8 a closed loop simulation for a candidate controller with reduced horizon of $N_c = 2$ from the worst-case disturbance

$$d_0' = \begin{bmatrix} 1 & -0.96 \end{bmatrix}.$$

Though perhaps difficult to observe from Figure 6.8, this gives a worst-case output difference between the candidate and reference controllers of $3.65 \cdot 10^{-3}$, which is the same as the calculated worst case error by the MILP solver (see Table 6.7). Figure 6.9 shows the result of a gridding over the disturbance space with corresponding errors $\|y_5^c - y_5^r\|_\infty$, and we observe that the worst case error is indeed at the location where the MILP solver found it, which validates our numerical procedure

Computation time. The resulting MILP needed to generate the results in this example turned out to take very long time to solve, so in order to get a fast screening of different methods we added a time-limit of 10 minutes to each optimization problem. A high computational demand to solve these problems is expected since they in general are NP-hard.

We did the calculations on a Dell PowerEdge 1950 with Intel(R) Xeon(R) CPU E5410 @ 2.33GHz and 8GB RAM. The MILP solver was set up in parallel mode,

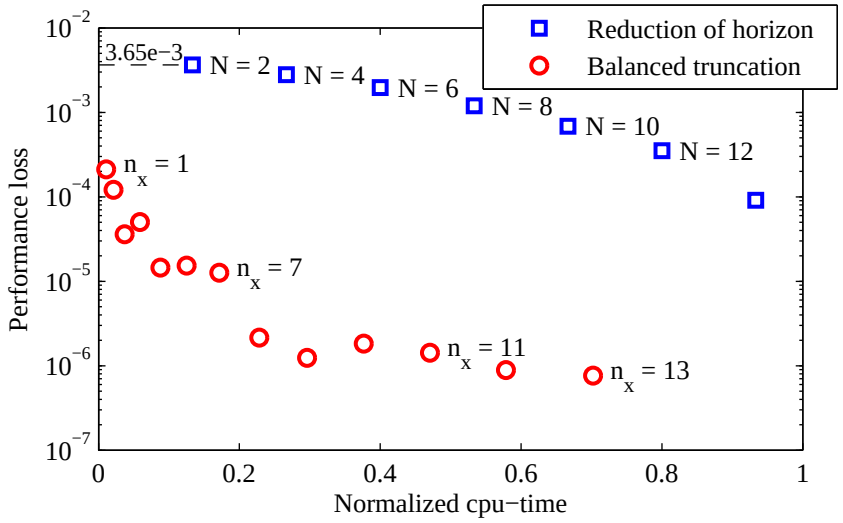

Figure 6.7: Structured formulation with the cpu-time is in the order of $N(n_x + n_u)^3$. The normalized Cpu-time in the plot is $N(n_x + n_u)^3/(N^r(n_x^r + n_u^r)^3)$ where superscript ^{r} refers to the reference controller.

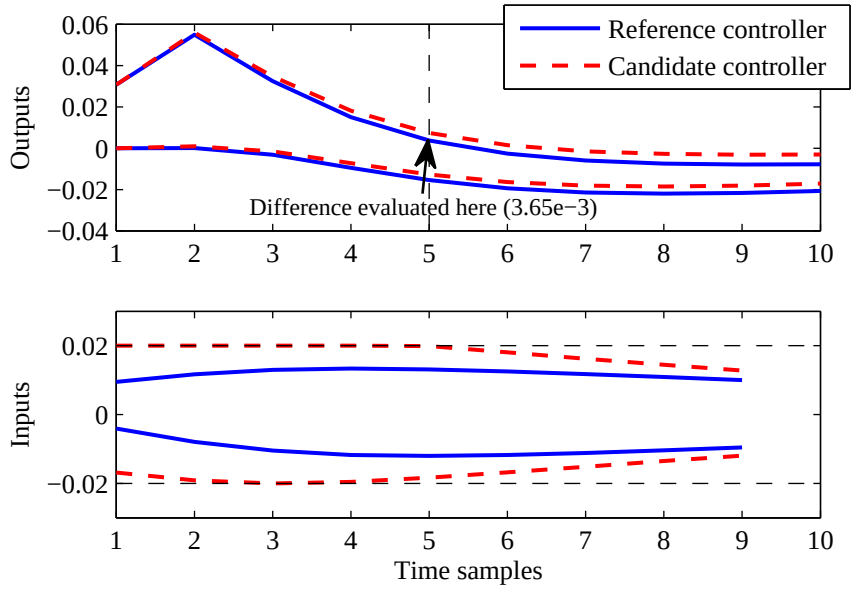

Figure 6.8: Closed loop simulation

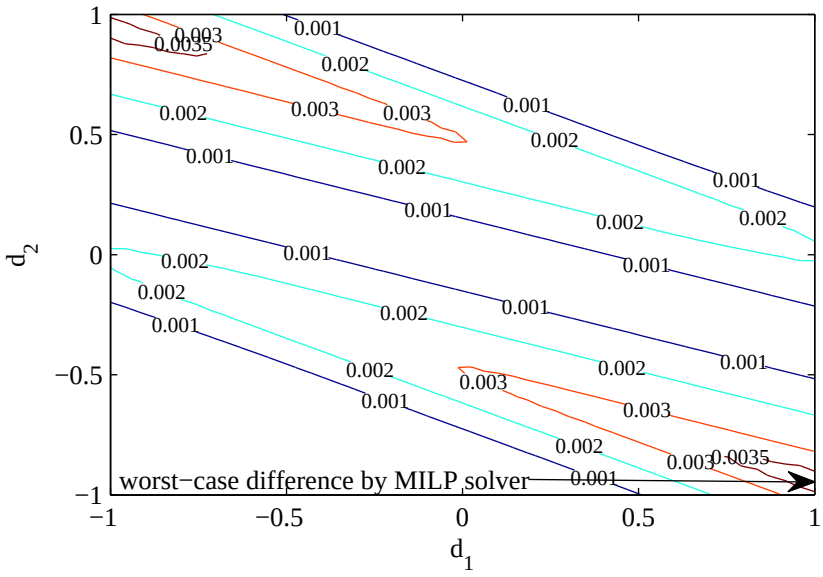

Figure 6.9: Contour plot of worst case difference for candidate controller with reduced horizon $N_c = 2$.

using up to 8 of 8 available CPU's.

6.6 Discussion

6.6.1 Numerical experiences and future work

As stated clearly in the online documentation for YALMIP [Löfberg, 2004], there are several pit falls when setting up a mixed integer problem. First of all, it is very important to bound all variables, and the bounds should be fairly tight to avoid excessive computation times. We added bounds in an iterative manner where we first solved a problem with quite conservative bounds, checked the solution, tightened the bounds accordingly, and finally resolved the problem to check that the solution was unaffected by the added bounds (i.e. that none of the bounds were active).

In addition, we experienced that the solution was very sensitive to the "big-M" value, and we found as indicated above acceptable values of M by iteration. The sensitivity to the value of the big-M is well known [Camm et al., 1990] and should always be addressed when setting up an MILP problem.

Other solution methods. Rather than adding constraints on big-M form and using an MILP solver as we did in this chapter, there are reports of methods that branch directly on the KKT conditions, see for example [Bard and Moore, 1990, Bard, 1998]. These methods could be investigated as one may avoid adding (the numerically dangerous) big-M constraints altogether.

Symmetry. The example in this chapter is actually symmetric in the inputs u_k and disturbances d_0, as we have a linear model with symmetric constraints on u_k. The symmetry can also be seen from the objective function plotted in Figure 6.9. It seems likely that if this symmetry would be exploited, the solution could be made faster.

Further, when we have a high and low limit on some variable, we know in advance that both constraints cannot be active at the same time, and this information should be used when solving the problem.

6.6.2 Extension to quadratic cost function

Practitioners might prefer to use a weighted 2-norm to judge the difference between controllers, rather than the infinity norm as used in this chapter. In this case one can simply use the same objective function in the performance evaluation as one is using in the MPC itself. A method for how to write this problem as an MILP is outlined in [Jones and Morari, 2009].

6.7 Conclusions

In this chapter we have presented a framework for analysis of different methods to speed up MPC. The framework can be used to rank different candidates in terms of expected performance degradation. The problem we pose is known to be NP-hard and thus can be very time-consuming. We showed by an example that by introducing an early stopping constraint on the MILP solver, we got interesting results within a reasonable time frame. For a case study, we found for a dense formulation that move blocking was more efficient than reducing the horizon. We also found that "delta input blocking" was better than "input blocking." For a structured formulation on of the same case study, model reduction was significantly better than reducing the horizon.

Several improvements on the numerical side has been addressed as possible future work.

Chapter 7

Bilevel programming for analysis of low-complexity control of linear systems with constraints

Published in "Proceedings of Conference on Decision and Control 2009, Shanghai, China."

In this paper we use bilevel programming to find the maximum difference between a reference controller and a low-complexity controller in terms of the infinity-norm difference of their control laws. A nominal MPC for linear systems with constraints, and a robust MPC for linear systems with bounded additive noise are considered as reference controllers. For possible low-complexity controllers we discuss partial enumeration (PE), Voronoi/closest point, triangulation, linear controller with saturation, and others. A small difference in the norm between a low-complexity controller and a robust MPC may be used to guarantee closed-loop stability of the low-complexity controller and indicate that the behaviour or performance of the low-complexity controller will be similar to that of the reference one. We further discuss how bilevel programming may be used for closed-loop analysis of model reduction.

7.1 Introduction

Bemporad et al. [2002] introduced an explicit solution of the model predictive control (MPC) problem for control of *linear systems with constraints* using a quadratic performance index. Later these results have been extended to cover a broader class of systems and performance objectives, see [Alessio and Bemporad, 2008] for a survey.

The main drawback of explicit MPC is that the control law, due to the combinatorial nature of the problem, can grow exponentially with the size of the optimal control problem [Wen et al., 2009].

Alessio and Bemporad [2008] proposed to reduce complexity of explicit MPC by either storing only the L regions with the highest Chebysev radius (if a full explicit solution is available), or to run extensive simulations of closed-loop MPC and collect the L most recurrent combinations of active constraints for implementation, similar to [Pannocchia et al., 2007]. (Storing only a subset of the possible regions of a MPC and using them for implementation is called partial enumeration (PE).)

Pannocchia et al. [2007] recently reported that by using a PE policy on an industrial example with more than 250 states, 32 inputs and a 25-sample control horizon, small look-up tables with only 25-200 entries gave a control that was less than 0.01% suboptimal compared to the full model predictive controller (MPC) for the same example. The MPC could theoretically enter $3^{800} = 4.977 \times 10^{381}$ regions.

In this paper we use bilevel programming to investigate the PE-schemes described above, but also more general low-complexity policies. The main idea is to calculate the maximum difference between a either a nominal or a robust MPC and the low-complexity policy, and then, based on this difference, draw conclusions about the proposed low-complexity controller.

In addition to guarantees of feasibility and stability the method can be used to give bounds on the sub-optimality of the low-complexity scheme, by using the value of the objective function of the reference controller as a difference-metric of the reference and low-complexity controller.

7.2 Notation and preliminaries

A *polyhedron* is the intersection of a finite number of halfspaces and a *polytope* is a bounded polyhedron. Given two sets $S_1, S_2 \subseteq \mathbb{R}^n$ the Minkowski sum is defined as $S_1 \oplus S_2 \triangleq \{s_1 + s_2 | s_1 \in S_1,\ s_2 \in S_2\}$, and the Pontryagin difference as $S_1 \ominus S_2 \triangleq \{s_1 | s_1 + s_2 \in S_1, s_2 \in S_2\}$. Boldface $\mathbf{x}$ and $\mathbf{u}$ means the sequences $\mathbf{x} = (x_0, x_1, \ldots, x_N)$ and $\mathbf{u} = (u_0, u_1, \ldots, u_{N-1})$, while boldface $\mathbf{1}$ is a vector of 1's of appropriate length.

We consider control of the following discrete-time linear system

$$x^+ = Ax + Bu, \tag{7.1}$$

where $x \in \mathbb{R}^{n_x}$ are the states and $u \in \mathbb{R}^{n_u}$ are the inputs, and x^+ above is a short-hand notation for $x_{k+1} = Ax_k + Bu_k$. In addition we have constraints such that $x \in \mathbb{X}$ and

$u \in \mathbb{U}$, where $\mathbb{X} = \{x \mid Fx \leq f\} \subset \mathbb{R}^{n_x}$ and $\mathbb{U} = \{u \mid Gu \leq g\} \subset \mathbb{R}^{n_u}$ are polytopic sets.

The solution of an explicit MPC with quadratic objective, linear process and polytopic constraints, can be written as a piecewise affine function of the state. A piecewise affine function $u(x) : X \mapsto \mathbb{R}^{n_u}$, where $X \subset \mathbb{R}^{n_x}$ is a polyhedral set, is piecewise affine if it is possible to partition X into convex polyhedral regions, CR_i, and $z(x) = K^i x + c^i, \quad \forall x \in CR_i$ [Bemporad et al., 2002]. In this paper "region" denotes CR_i, written "region i", and (K^i, c^i) is the corresponding optimal control law, i.e. the part of $u(x)$ that belongs to CR_i. In order to conform with notation used in [Alessio and Bemporad, 2008], we use $L_i = \left\{x \in \mathbb{R}^{n_x} \mid A^i x \leq b^i\right\}$ in the place of CR_i.

7.3 Bilevel optimization

The main focus of this paper is the application of bilevel optimization for analysis of low-complexity controllers. Here we give an introduction to bilevel optimization and solution methods, following Jones and Morari [2009]. For more background details the reader is referred to a recent survey [Colson et al., 2005].

Bilevel problems are hierarchical in that the optimization variables (y,z) are split into upper y and lower z parts, with the lower level variables constrained to be an optimal solution to a secondary optimization problem:

$$\begin{aligned} &\min_y V_U(y,z) \\ &\text{subject to } G_U(y,z) \leq 0 \\ &\qquad z = \arg\min_z V_L(y,z) \\ &\qquad\qquad \text{subject to } G_L(y,z) \leq 0 \end{aligned} \tag{7.2}$$

In this paper we will only consider problems where the lower-level problem has an unique optimizer. Moreover, we will usually have two low-level problems, one for the reference controller and one for the low-complexity controller.

7.3.1 Solution methods

If the lower level problem is convex and regular, then it can be replaced by its necessary and sufficient Karush-Kuhn-Tucker (KKT) conditions, yielding a standard

single-level optimization problem [Jones and Morari, 2009]:

$$\begin{aligned} \min_{y,z,\lambda} \; & V_U(y,z) \\ \text{subject to } & G_U(y,z) \leq 0 \\ & G_L(y,z) \leq 0 \\ & \lambda \geq 0 \\ & \lambda' G_L(y,z) = 0 \\ & \nabla_z \mathcal{L}(y,z,\lambda) = 0 \end{aligned} \tag{7.3}$$

where $\mathcal{L}(y,z,\lambda) := G_L(y,z) + \lambda' G_L(y,z)$ is the Lagrangian function associated with the lower-level problem. For the special case of linear constraints and a quadratic cost, all constraints of (7.3) are linear and the complimentary condition $\lambda' G_L(y,z) = 0$ is a set of disjunctive linear constraints, which can be described using binary variables, and thus leads to a mixed-integer linear problem.

7.3.2 Bilevel optimization for analysis of controllers

In this paper we use bilevel programming to find the maximal difference between a reference controller and a low-order controller. Hence, for a subset $\mathcal{X} \subset \mathbb{R}^{n_x}$, we solve

$$\begin{aligned} \max_{x \in \mathcal{X}} \; & d(u_{\text{ref}}, u_{\text{low-complexity}}) \\ \text{subject to } & \text{KKT}(\text{reference controller}) \\ & \text{KKT}(\text{low-complexity controller}) \end{aligned} \tag{7.4}$$

Typically, $\mathcal{X}$ is the intersection of the feasible states for the reference and the low-complexity controller.

Note that explicit solutions of neither the reference nor the low-complexity controllers are needed, because the solutions are implicitly given by the KKT conditions.

The distance measure $d(u_{\text{ref}}, u_{\text{low-complexity}})$ can be, for example, the difference between the next state,

$$\begin{aligned} & d(u_{\text{ref}}, u_{\text{low-complexity}}) = \\ & \| x^+{}_{\text{ref}}(x, u_{\text{ref}}) - x^+{}_{\text{low-complexity}}(x, u_{\text{low-complexity}}) \|_\infty = \\ & \| B u_{\text{ref}} - B u_{\text{low-complexity}} \|_\infty, \end{aligned} \tag{7.5}$$

but also differences between trajectories of either states or inputs.

Remark 7.1. *We observe that* (7.5) *renders* (7.4) *non-convex due to the term* $\max \|t\|_\infty$ *(where* t *is a convex function of* $(u_{ref}, u_{low\text{-}complexity})$*) . However, the problem may be converted into a mixed integer linear program (MILP) using a standard technique (e.g. [Löfberg, 2004]), in which we introduce binary variables* n_i, p_i *for each element of* t *and add the condition that the binary variable* p_i *is one if* $\|t\|_\infty = t_i$ *and* n_i *is one if* $\|t\|_\infty = -t_i$*. The method adds only linear and binary conditions to* (7.4) *and therefore the overall problem remains a MILP [Jones and Morari, 2009].*

7.4 Application on analysis of low-complexity controllers

We first present a nominal MPC policy based on optimizing a quadratic performance objective subject to a linear model of the process at and a set of polytopic constraints on both states and inputs. We thereafter present a robust MPC, where the process is subject to bounded disturbances on the states. Both these schemes fit into the bilevel problem as a reference controller.

The choice of which reference controller to use depends on the problem at hand, as this defines a benchmark for control of the process. The robust MPC scheme can be used to give a feasibility and stability certificate of the low-complexity scheme. However, in some cases the robust MPC can be quite conservative, and the nominal MPC may be a better benchmark.

Thereafter we show how several low-complexity polices can be expressed in the bilevel framework. The main "tool" we use here is to represent any logic and bilinear terms in the KKT-conditions with mixed integer linear constraints in order to let the resulting problem be a MILP.

7.4.1 Nominal MPC as reference controller

Consider the following semi-infinite horizon optimal control problem [Jones and Morari, 2009]:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \; & J(\mathbf{x},\mathbf{u}) = \frac{1}{2}x_N' P x_N + \dots \\
& \qquad + \frac{1}{2}\sum_{i=0}^{N-1} u_i' R u_i + x_i' Q x_i, \\
\text{subject to } & x_{i+1} = Ax_i + Bu_i, \quad \forall i = 0,\dots,N-1, \\
& x_i \in \mathbb{X}, \quad \forall i = 1,\dots,N-1, \\
& u_i \in \mathbb{U}, \quad \forall i = 0,\dots,N-1, \\
& x_N \in \mathbb{X}_N, \\
& x_0 = x.
\end{aligned} \tag{7.6}$$

Here $\mathbb{X}_N = \{x \mid Hx \leq h\} \subset \mathbb{X}$ is a polytopic invariant set for the system $x^+ = Ax + B\mu(x)$ for some given control law $\mu : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_u}$. Further $P \in \mathbb{R}^{n_x \times n_x}$ and $Q \in \mathbb{R}^{n_x \times n_x}$ are positive definite matrices and $R \in \mathbb{R}^{n_u \times n_u}$ is a positive semi-definite matrix. We define $\mathcal{X} \subset \mathbb{R}^{n_x}$ to be the set of states x for which there exists a feasible solution to (7.6).

If $\mathbf{u}^*(x)$ is the optimal input sequence of (7.6) for the state x, and $u_0^*(x)$ is the resulting control law, then stability of the system $x^+ = Ax + Bu_0^*(x)$ can be established under the assumption that $V_N(x) = x'Px$ is a Lyapunov function for the system $x^+ = Ax + B\mu(x)$ and that the decay rate of V_N is greater than the stage cost $l(u,x) = u'Ru + x'Qx$ within the set $\mathbb{X}_N$ [Jones and Morari, 2009].

By using $x_k = A^k x_0 + \sum_{j=0}^{k-1} A^j B u_{k-1-j}$ the MPC problem (7.6) can be rewritten as [Bemporad et al., 2002]:

$$\begin{aligned}
V(x_0) = & \frac{1}{2}x_0' Y x_0 + \dots \\
& + \min_U \{\frac{1}{2}U'HU + x_0' F U, \\
& \text{subject to } \; GU \leq W + Ex_0\},
\end{aligned} \tag{7.7}$$

where $U' = \begin{bmatrix} u_0' & u_1' & \cdots & u_{N-1}' \end{bmatrix}$.

We want to use (7.7) as a lower-level problem in bilevel programming. The

following equations define the KKT conditions for this problem:

$$\begin{aligned} HU + F'x_0 + G'\lambda &= 0 \\ GU - W - Ex_0 &\leq 0 \\ \lambda &\geq 0 \\ \lambda &\leq Ms \\ GU - W - Ex_0 &\geq -M(1-s) \end{aligned} \tag{7.8}$$

Here $s \in \{0,1\}^{n_W}$, where n_W is the number of inequality constraints in (7.7). The two last equations in (7.8) correspond to the complementary condition $\lambda' G_L(y,z) = 0$ in the general bilevel problem, here described with binary variables s. M is a constant that is large enough such that the solution to (7.8) corresponds to the solution of (7.7). (This is called a "big-M" formulation.)

7.4.2 Robust MPC as reference controller

In this subsection the results are from Mayne et al. [2005] unless otherwise noted.

Consider control of the linear system (7.1) with *additive disturbances* w on the states:

$$x^+ = Ax + Bu + w. \tag{7.9}$$

The disturbance is assumed to be bounded,

$$w \in W, \tag{7.10}$$

where W is compact and contains the origin (but may not have an interior).

Suppose $K \in \mathbb{R}^{n_u \times n_x}$ is such that $A_K \triangleq A + BK$ is stable. Let Z be a disturbance invariant set for the controlled uncertain system $x^+ = A_K x + w$ satisfying, therefore

$$A_K Z \oplus W \subseteq Z. \tag{7.11}$$

We use the following proposition as a basis for the robust MPC:

Proposition 7.1. *Suppose Z is disturbance invariant for $x^+ = A_K x + w$. If $x \in \bar{x} \oplus Z$ and $u = \bar{u} + K(x - \bar{x})$, then $x^+ \in \bar{x}^+ \oplus Z$ for all $w \in W$ where $x^+ = Ax + Bu + w$ and $\bar{x}^+ = A\bar{x} + B\bar{u}$.*

Proposition 7.1 states that the feedback policy $u = \bar{u} + K(x - \bar{x})$ keeps the states x of the uncertain system (7.9) close to the states $\bar{x}$ of the so-called nominal system $\bar{x}^+ = A\bar{x} + B\bar{u}$.

We can now define the robust MPC problem:

$$\begin{aligned}\min_{\bar{\mathbf{x}},\bar{\mathbf{u}}}\ J(\bar{\mathbf{x}},\bar{\mathbf{u}}) &= \frac{1}{2}\bar{x}_N' P \bar{x}_N + \dots \\ &\quad + \frac{1}{2}\sum_{i=0}^{N-1} \bar{u}_i' R \bar{u}_i + \bar{x}_i' Q \bar{x}_i, \\ \text{subject to } \bar{x}_{i+1} &= A\bar{x}_i + B\bar{u}_i, \quad \forall i = 0,\dots,N-1, \\ \bar{x}_i &\in \mathbb{X} \ominus Z, \quad \forall i = 1,\dots,N-1, \\ \bar{u}_i &\in \mathbb{U} \ominus KZ, \quad \forall i = 0,\dots,N-1, \\ \bar{x}_N &\in \mathcal{X}_f, \\ \bar{x}_0 &= x \oplus Z.\end{aligned} \tag{7.12}$$

In order to achieve closed loop robust stability, the terminal constraint set $\mathcal{X}_f$ must satisfy the following axioms [Mayne et al., 2005]:

$$\begin{aligned}&\text{A1} : A_K \mathcal{X}_f \subset \mathcal{X}_f,\ \mathcal{X}_f \subset \mathbb{X} \ominus Z,\ K\mathcal{X}_f \subset \mathbb{U} \ominus KZ \\ &\text{A2} : V_f(A_k x) + l(x, Kx) \leq V_f(x),\ \forall x \in X_f,\end{aligned} \tag{7.13}$$

where $V_f(v) = v'Pv$ and $l(v,z) = v'Qv + u'Ru$ in the scope of this paper.

Assume that Z is a polytopic set such that $\{v \in \mathbb{R}^{n_x} \mid H_z v \leq k_z\}$.

As for the nominal MPC, we can rewrite the robust MPC problem as:

$$\begin{aligned}\min_{(U,\bar{x}_0)}\ &\begin{bmatrix} U' \\ \bar{x}_0' \end{bmatrix}' \underbrace{\begin{bmatrix} H & F' \\ F & 2Y \end{bmatrix}}_{\tilde{H}} \begin{bmatrix} U \\ \bar{x}_0 \end{bmatrix} \\ \text{subject to } &\underbrace{\begin{bmatrix} G & -E \\ 0 & -H_z \end{bmatrix}}_{\tilde{G}} \begin{bmatrix} U \\ \bar{x}_0 \end{bmatrix} \leq \underbrace{\begin{bmatrix} W \\ k_z \end{bmatrix}}_{\tilde{W}} + \underbrace{\begin{bmatrix} 0 \\ -H_z \end{bmatrix}}_{\tilde{E}} x\end{aligned} \tag{7.14}$$

Let $v = (U, \bar{x}_0)$. The KKT-conditions corresponding to (7.14) are

$$\begin{aligned}\tilde{H}v + \tilde{G}\lambda &= 0 \\ \tilde{G}v &\leq \tilde{W} + \tilde{E}x \\ \lambda &\geq 0 \\ \lambda &\leq Ms \\ \tilde{G}v &\geq W + Ex - M(1-s)\end{aligned} \tag{7.15}$$

Note that the KKT conditions in (7.8) are a special case of the KKT-conditions above, since above $\bar{x}_0$ is included as a degree of freedom. For both nominal and

robust MPC the current state x is a parameter driving the controller, but for the nominal MPC we have substituted this with x_0, as $x_0 = x$ is a constraint in the nominal MPC formulation.

The main motivation for using robust MPC as a reference rather than nominal MPC is because the robust MPC can be used to prove feasibility and stability of the low-complexity scheme. Both properties can be established using the following proposition:

Proposition 7.2. *Consider the linear system for which robust stability and feasibility are guaranteed by the robust MPC:*

$$x^+ = Ax + Bu + w, \quad w \in W,$$

and that

$$W = \{w \in \mathbb{R}^{n_x} \mid \|w\|_\infty \leq \varepsilon\}$$

Let $u_{l\text{-}c}$ be the control input from the low-complexity controller, and u_{rMPC} the input from the robust MPC. The following holds for the system controlled by the low-complexity controller:

$$\begin{aligned} x^+ &= Ax + Bu_{l\text{-}c} \\ &= Ax + Bu_{l\text{-}c} - Bu_{rMPC} + Bu_{rMPC} \\ &= Ax + Bu_{rMPC} + B(u_{l\text{-}c} - u_{rMPC}). \end{aligned} \tag{7.16}$$

Hence, if

$$\|B(u_{l\text{-}c} - u_{rMPC})\|_\infty \leq \varepsilon, \tag{7.17}$$

the low-complexity controller is both feasible and stable.

7.4.3 Low-complexity controllers as low-level problems in bilevel programming

In this section we describe various low-complexity controllers that fit into the bilevel programming framework. Several more are possible, but not included for space restrictions.

Linear quadratic regulator with saturation

A simple low-complexity control policy is the linear quadratic regulator (LQR) with saturation. In the "unconstrained region" this is optimal, and its behaviour can be modelled using few binary variables. First, we define $\hat{u}_{\text{LQR}} = -Kx$. For simplicity we assume that the constraints on u may be written as

$$u_i^{\text{l}} \leq u_i \leq u_i^{\text{h}}, \quad i = 1, \ldots, n_u \tag{7.18}$$

Now, for each row in (7.18), we define a corresponding binary vector $d^i \in \{0,1\}^3$. The saturation can now be modelled using

$$\begin{aligned} u_i &\leq u_i^h + Md_1^i, \\ u_i &\geq u_i^l - Md_3^i, \\ d_1^i + d_2^i + d_3^i &= 1, \\ -M(1-d_k^i) \leq \mathrm{sat}(\mathrm{u_i}) - \{u_i\}_k &\leq M(1-d_k^i), \\ k &= 1,2,3, \end{aligned} \tag{7.19}$$

where $\{u_i\} = \left\{u_i^h, u_i, u_i^l\right\}$, and $\{u_i\}_k$ is the k'th element of $\{u_i\}$.

Partial enumeration (PE)

Here we follow the ideas of Pannocchia et al. [2007] and Alessio and Bemporad [2008], and we store only a subset of the possible active sets. The controller implementation is here to first locate the closest region to the current state x, and then use the control law from the corresponding region. In order to satisfy $u \in \mathbb{U}$, we saturate the input before applying the input to the plant.

Here we use the *minimal-violation distance* from Christophersen et al. [2007] to find the closest region for a set $\mathcal{L}$ of stored polytopes.

Definiton 7.1. *(Minimal-violation distance [Christophersen et al., 2007]) Let the collection $\mathcal{L}$ be the set $\mathcal{L} = \{L_i\}_{i=1}^{N_\mathcal{L}}$, where $L_i := \left\{x \in \mathbb{R}^{n_x} \mid A^i x \leq b^i\right\}$ are full-dimensional polyhedra in $\mathbb{R}^{n_x}$. We assume that $A^i x \leq b^i$ are on Hessian normal form, i.e. each row $[A^i]_r$ of A^i is normalized with $\|[A^i]_r\|_2 = 1$.*

The minimal-violation distance d_{MV} of x to $\mathcal{L}$ is given by

$$d_{MV} := \min_i \left\{\alpha_i^*(x)\right\}, \tag{7.20}$$

where

$$\alpha_i^*(x) = \arg\min \left\{\alpha_i \in \mathbb{R} \mid A^i x \leq b^i + \alpha_i \mathbf{1}\right\}, \tag{7.21}$$

for all $i = 1,\ldots,N_\mathcal{L}$ and $\mathbf{1} = [1 \ \cdots \ 1]'$.

The solution of the LP (7.21) can be found using the KKT conditions:

$$\begin{aligned} 1 - \mathbf{1}'\lambda^i &= 0, \\ 0 \leq \lambda^i &\leq Ms^i, \\ 0 \leq b + \alpha_i \mathbf{1} - A^i x &\leq M(\mathbf{1} - s^i), \end{aligned} \tag{7.22}$$

where $s^i \in \{0,1\}^{n_{b^i}}$ is a vector of binary variables of length corresponding to the number of faces in the polytope $L_i = \left\{x \in \mathbb{R}^{n_x} \mid A^i x \leq b^i\right\}$.

Let $\beta \in \{0,1\}^{n_L}$ be binary variables such that

$$\beta_i = 1 \leftrightarrow \alpha_i \leq \alpha_j \quad \forall j \neq i, \tag{7.23}$$

which implies that $\sum \beta_i = 1$. We can then define the PE control law as

$$\hat{u} = \text{sat}\left\{ \sum_{i=1}^{n_L} \beta_i \left(K^i x + c^i \right) \right\}, \tag{7.24}$$

where (K^i, c^i) is the optimal feedback in region i, and $\text{sat}\{\cdot\}$ is a normal saturation function. Equation (7.24) is bilinear in the optimization variables β_i, x, and can be implemented in the bilevel framework with the following equations (added as constraints in the problem):

$$-M(1-\beta_i) \leq \hat{u} - (K^i x + c^i) \leq M(1-\beta_i). \tag{7.25}$$

Remark 7.2. *The proposed PE-scheme, which follows from Christophersen et al. [2007], can be implemented on-line as follows:*

$$\begin{aligned} \alpha_i &= \max\left\{A^i x - b^i\right\}, \quad i = 1, \ldots, L \\ i^* &= \arg\min_i \{\alpha_i\} \end{aligned} \tag{7.26}$$

Delaunay triangulation

Assume that for some points $(x_1, \ldots, x_{n_L})$ we precompute a Delaunay triangulation. In addition we store the optimal input $(u_1^*, \ldots, u_{n_L}^*)$ at those points. A Delaunay triangulation can be understood by the empty circle method [Aurenhammer, 1991]: Consider all triangles formed by the points such that the circumcircle of each triangle is empty of other sites, where the sites in this case are the stored points $(x_1, \ldots, x_{n_L})$.

The Delaunay triangulation of the points $(x_1, \ldots, x_{n_D})$ can be used to find an interpolated control law:

- Denote the triangles from the Delaunay triangulation by $L_1, \ldots, L_{n_D}$.
- For a given state x:
  1. Find the current triangle L_i that containts x.
  2. Express x as a convex combination of the vertices of L_i, $x = \sum \lambda_k x_k^i$, where x_k^i denotes the vertices of L_i

- Implement the following interpolated control law:

$$u_{\text{Delaunay}} = \sum \lambda_k u_k^{*,i}, \tag{7.27}$$

  where $u_k^{*,i}$ are the optimal inputs corresponding to the points x_k^i.

The Delaunay triangulation itself can be implicitly defined using the following set of equations, which can be added as mixed-integer linear constraints to the overall problem:

$$\begin{aligned} x = \sum \lambda_i x_i, \quad ,\lambda_i \geq 0, \quad \sum \lambda_i = 1, \\ \lambda \leq \sigma_i, \quad \sum \sigma_i = n+1 \\ \|c - x_i\|_2^2 \leq \|c - x_j\|_2^2 + M\sigma_j + M(1-\sigma_i), \end{aligned} \tag{7.28}$$

where the last equation can be rewritten as

$$\begin{aligned} \cancel{c'c} \underbrace{-2x_i'}_{a_i'} c + \underbrace{x_i'x_i}_{b_i} \leq \cancel{c'c} \underbrace{-2x_j'}_{a_j'} c + \underbrace{x_j'x_j}_{b_j} + \dots \\ \cdots + M\sigma_j + M(1-\sigma_i) \\ a_i'c + b_i \leq a_j'c + b_j + M\sigma_j + M(1-\sigma_i) \end{aligned} \tag{7.29}$$

Here $c \in \mathbb{R}^{n_x}$ is an extra optimization variable, $\sigma \in \{0,1\}^{n_L}$ is a vector of binaries and M is a large constant.

We note that the last equation of (7.28) is an expression for the "empty-circle method".

7.5 Examples

In this section we show two examples where we use the bilevel programming to identify the worst-case distance between a reference controller and a proposed low-complexity controller. The calculations where done using ILOG CPLEX® and the problems were written in YALMIP [Löfberg, 2004]. Set calculations and explicit solution of MPC's were done using Multi-Parametric Toolbox (MPT) [Kvasnica et al., 2004].

Example 1: Double integrator with nominal MPC as reference controller and PE as low-complexity controller

In this example we consider the double integrator described in [Bemporad et al., 2002], example 7.3, but with a sample time of $T_s = 0.1$ in order to match the

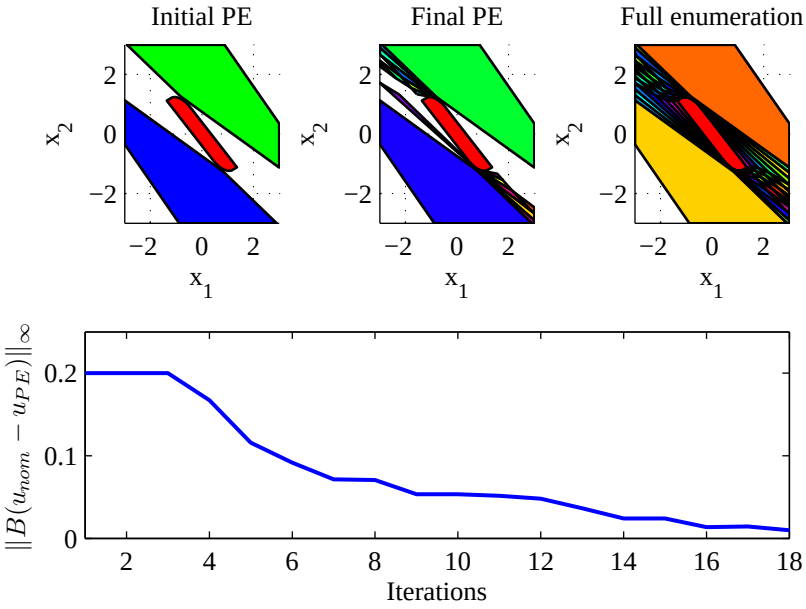

Figure 7.1: Example 1: double integrator.

conditions in [Alessio and Bemporad, 2008]. The process is hence

$$x^+ = \underbrace{\begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} 0 \\ 0.1 \end{bmatrix}}_{B} u, \quad -1 \leq u \leq 1 \tag{7.30}$$

The control parameters are $N = 8$, $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 0 \end{smallmatrix}\right]$ and $R = 0.1$. The final weight P corresponding to the LQR controller is $P = \left[\begin{smallmatrix} 8.98 & 3.59 \\ 3.59 & 2.86 \end{smallmatrix}\right]$

The nominal MPC problem is now:

$$\begin{aligned} &\min_{\mathbf{x},\mathbf{u}} x_8' P x_8 + \sum_{i=0}^{7} x_i' Q x_i + R u_i^2 \\ &\text{subject to } x_{k+1} = A x_k + B u_k, k = 0, \ldots, 7 \\ &\qquad x_0 = x \\ &\qquad -1 \leq u_k \leq 1, \quad k = 0, 1 \ldots, 7 \end{aligned} \tag{7.31}$$

We do not add any terminal constraint on x_N as we want to compare our results with [Alessio and Bemporad, 2008].

We want to compare the nominal MPC to a PE-scheme, hence we want to solve

$$\begin{aligned}
&\max_{x\in X} \|B(u^* - \hat{u})\|_\infty \\
&\text{subject to } \alpha_i = \arg\min_{\alpha} \alpha \\
&\qquad \text{subject to } A^i x \le b^i + \alpha_i \mathbf{1} \\
&\qquad \beta_i = \begin{cases} 1, & \alpha_i \le \alpha_j \ \forall j \ne i \\ 0, & \text{otherwise} \end{cases} \\
&\qquad \tilde{u} = \sum_{i=1,\cdots,L} \beta_i (K^i x + c^i), \\
&\qquad \hat{u} = \text{sat}(\tilde{u}) \\
&\qquad u^* = \arg\min (7.31)
\end{aligned} \tag{7.32}$$

This problem can be rewritten to a MILP using (7.22) for the minimal violation distance.

The main focus of this paper is to calculate the difference between two controllers, but we may also use this method for controller synthesis. This can be achieved by:

- Solve (7.32) to get the worst point in the state space x^* and the worst case norm $\|B(u^* - \hat{u})\|_\infty = \|x^{*,+} - \hat{x}^+\|_\infty$.
- Add the corresponding region and corresponding optimal control law to the PE-controller.
- Resolve (7.32) and add the corresponding worst-case region until the worst-case norm is less than a user-defined value or the number of regions in the PE is larger than a user-defined value.

This example can be solved explicitly using MPT. The full enumeration is shown in the upper right part of figure 7.1. In order to test our software we started out with an initial PE controller using the 3 largest regions, shown in the top-left part of figure 7.1. The lower part of the figure shows the maximum difference between the reference controller (nominal MPC) and the PE-controller. We then performed iterations as described above, at each iteration we added the region corresponding to the worst case point x^*. One observes that initially the difference is equal to the maximum possible difference, as $B = \left[\begin{smallmatrix} 0 \\ 0.1 \end{smallmatrix}\right]$ and $\|u\| \le 1$. However, as we add regions to the PE controller the difference decreases to quite low levels.

Note that even though the full enumeration was available for this example, we do not use this solution while solving (7.32), rather we use the KKT-conditions of the corresponding MPC problem.

Closed-loop simulations, even from the worst case points, shows very small difference between the nominal MPC and the PE, also for quite high values of the worst-case norm, and are not included here for brevity.

Example 2: Double integrator with robust MPC as reference controller

For the same process as in Example 1, with the same objectives for the controller, we designed a robust MPC using the method described in section 7.4.2, and we use this one as the reference controller. The motivation for using the robust MPC rather than the nominal MPC is because we can verify closed-loop stability of the low-complexity scheme, given that $\|B(u_{\text{robust}} - u_{\text{low-complexity}})\|_\infty \leq \|w\|_\infty$.

A box constraint on w was used such that $\|w\|_\infty \leq 0.01$, and we used the algorithm from Rakovic̀ et al. [2005] to compute Z, and in order to compute X_f we used MPT. We wanted to use this robust controller to prove closed-loop nominal stability of the PE-controller from Example 1. However, we observed that $\max_{x \in X} \|B(u_{\text{robust MPC}} - u_{\text{nominal MPC}})\|_\infty$ was growing faster than $\|w\|_\infty$, i.e. the robust MPC was very conservative with increasing $\|w\|_\infty$. Since the PE-controller from Example 1 is close to the nominal MPC, it is clear that we cannot use the robust MPC scheme to prove stability of the PE-scheme, moreover we can not even use it to prove closed-loop stability of the nominal MPC.

One reason for why $\|B(u_{\text{robust MPC}} - u_{\text{nominal MPC}})\|_\infty$ is growing faster than $\|w\|_\infty$ is that the scalar input u can only act on the process in the direction B, while the vector w is acting directly on both states (through the identity transformation I). Changing the formulation of the robust MPC to restrict w to act only in the direction B is planned as further work in this project.

7.6 Conclusions

A bilevel framework for closed loop comparison of different control schemes has been presented. Many challenges still remain, but it seems like this framework will be useful for proving stability for some “ad-hoc” low complexity control schemes, and moreover it seems to have potential in the field of model reduction.

Acknowledgements

The authors would like to acknowledge the kind help from Melanie Zeilinger for providing some of the code used for the preparation of this paper. Further we greatly acknowledge the fruitful discussions with Daniel Axehill, Thomas Besselmann, and Alexander Nölle-Fuchs.

7.7 Appendix: Further comments to problems of proving stability

It seems like our method for proving nominal stability of a given controller has a fundamental limitation that if the controller is more aggressive than the nominal controller, then we can never prove its stability.

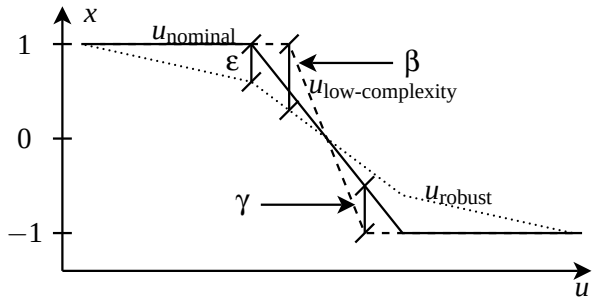

Figure 7.2: Different controllers for an example with $n_x = n_u = 1$. Full line represents the nominal MPC, dashed line is the proposed low-complexity controller and the dotted line is the robust MPC.

Consider figure 7.2 which shows an example for $n_x = n_u = 1$. The full line shows the input form a nominal MPC controller, while the dotted line shows the input from a robust MPC. We observe that the robust MPC is more conservative than the nominal MPC. The maximum difference in input occurs at ε. Now, assume that the maximum difference γ between the nominal MPC and the low-complexity controller is such that

$$\gamma \leq \varepsilon. \tag{7.33}$$

This, however, does not guarantee that the difference between the robust MPC and the low-complexity controller, β less than ε,

$$\beta \not\leq \varepsilon. \tag{7.34}$$

Chapter 8

Conclusions and suggested further work

8.1 Conclusions

In this thesis we have given several contributions to the general topic of finding simple implementations that give near-optimal operation in closed loop. In Chapters 2-4 we developed a convex approximation to the static output feedback problem, which is one of the open problems in control [Syrmos et al., 1997]. We showed by an example that the method may be useful for designing "fixed-structure" controllers, such as a multiple input – multiple output proportional-integral-derivative (MIMO-PID) controller.

In Chapter 5 we extended a method within the field of self-optimizing control, the nullspace method [Halvorsen et al., 2003], to handle changes in the active set. This was done by using results from explicit MPC [Baotič et al., 2008].

In Chapter 6 we developed a framework for analysis for different ways of speeding up model predictive control (MPC), and in Chapter 7 we used the same approach to analyse simplified controllers, such as controllers which use only parts of the whole lookup table that is used in explicit MPC.

In Appendices A-C we gave some more information about links between self-optimizing control and explicit MPC, suggested some simplifications to the point location problem in explicit MPC, and gave some more details on the static output feedback problem.

8.2 Suggested further work

8.2.1 Challenges with real-time optimization (RTO)

It is acknowledged in Chapter 5 that self-optimizing control and RTO are complementary, so in order to have good economic operation both layers need to operate as intended. By talking to practitioners we learned that RTO is in practise often not operational. It would be interesting to get updated information about what kind of challenges the practitioners are challenging and how these issues can be solved. After solving these issues one could better understand the interplay between an operational RTO and a self-optimizing control layer, a project that can have significant environmental and economic potential.

8.2.2 Mathematical transformations

In control of distillation columns it is well-known that a logarithmic transformation of the composition makes the resulting control problem linear [Skogestad, 1997]. The methods developed so far in self-optimizing control depend mostly on linear or quadratic models. Finding a procedure that can transform a nonlinear problem into a linear or quadratic problem would be very useful, as this would broaden the applicability of self-optimizing control.

8.2.3 Convex modelling

When modelling a chemical plant there is usually not much focus on if the resulting model, when used in economic optimization, results in a convex problem (interestingly, the mass and energy balances are *always* linear [Haug-Warberg, 2010], and these equations are the basis for any modelling project where one intends to use first principles). It would be nice to have a library of convex modelling methods that can be used to model a chemical system such that the resulting economic optimization is convex. Most likely, not all parts of a chemical system can be modelled as convex operations, but by paying attention to the convexity when doing the modelling one can efficiently pin-point the parts of the model that will introduce non-convexity in the resulting optimization problem. Once this insight is gained, one can choose to approximate the non-convex parts by convex approximations, or one may build the information about the *location* of the non-convexity into the optimization routine.

Hopefully such an approach would be beneficial for the application of RTO in process plants.

8.2.4 Generalization of switching scheme

The switching scheme for implementation of quadratic programs presented in Chapter 5 can easily be generalized to cover linear programs by using results from the main reference, the paper by Baotič et al. [2008]. Probably more important, one should consider to investigate further the general area of "computational algebra" in a lot more depth, perhaps starting by looking at results from explicit MPC, such as general parametric programming [Kvasnica et al., 2004]. There are already several results on explicit control of hybrid systems (see e.g. the group of Morari in ETH and the group of Bemporad in Siena, Italy) that probably can be used on the static optimization problems that are faced in self-optimizing control.

8.2.5 Improvement of numerical methods

In Chapters 2-4 we show that we can approximate the static output feedback problem to a quadratic program, but that the resulting quadratic program takes a long time to solve. In order to better investigate the properties of this method it would be useful to program the method in some language that is more optimized towards calculation time, rather than using Matlab interfaced to CPLEX as was done until now. CPLEX itself may be a suited solver, but the interface to Matlab does not allow for sparse matrices, which makes integration of the two difficult.

Once this is done it would be interesting to investigate how the method can be used on for example static decentralized control.

Appendix A

A new approach to explicit MPC using self-optimizing control

Published in "Proceedings of American Control Conference 2008, Seattle, USA."

Model predictive control (MPC) is a favored method for handling constrained linear control problems. Normally, the MPC optimization problem is solved on-line, but in 'explicit MPC' an explicit piecewise affine feedback law is computed and implemented [Bemporad et al., 2002]. This approach is similar to 'self-optimizing control,' where the idea is to find simple pre-computed policies for implementing optimal operation, for example, by keeping selected controlled variable combinations c constant. The 'nullspace' method [Alstad and Skogestad, 2007] generates optimal variable combinations, which turn out to be equivalent to the explicit MPC feedback laws, that is, $c = u - Kx$, where K is the optimal state feedback matrix in a given region. More importantly, this link gives new insights and also some new results. One is that regions changes may be identified by tracking the variables c for neighboring regions.

A.1 Introduction

Consider the general static optimization problem [Alstad and Skogestad, 2007]:

$$\begin{aligned} \min_{u_0,x} \quad & J_0(x,u_0,d) \\ \text{s.t.} \quad & f_i(x,u_0,d) = 0, \quad i \in \mathcal{E} \\ & h_i(x,u_0,d) \geq 0, \quad i \in \mathcal{I}, \end{aligned} \tag{P1}$$

where $x \in \mathbb{R}^{n_x}$ are the states, $u_0 \in \mathbb{R}^{n_{u_0}}$ are the inputs, and $d \in \mathcal{D} \subset \mathbb{R}^{n_d}$ are disturbances. By discretization and reformulation this may also represent some dynamic

optimization problems. Usually f is a model of the physical system, whilst h is a set of inequality constraints that limits the operation (e.g., physical limits on temperature measurements or flow constraints). In addition to (P1) we have measurements on the form

$$y_0 = f^y(x, u_0, d). \tag{A.1}$$

In this work the emphasis is on *implementation of the solution to* (P1). This means that the optimization problem (P1) is solved off-line to generate a 'control policy' which is suitable for on-line implementation, with particular emphasis on remaining close to optimal solution when there are unknown disturbances. That is, we search for 'control policies' such that the cost J_0 remains optimal or close to optimal when disturbances occur without the need to reoptimize.

A.1.1 Self-optimizing control

In our previous work on 'self-optimizing control' we have looked for simple control policies to implement optimal operation, and in particular 'what should we control' (choice of controlled variables (*CV*'s)). Using off-line optimization we may determine regions where different sets of active constraints are active, and implementation of optimal operation is then in each region to:

1. Control the active constraints.
2. For the remaining unconstrained degrees of freedom: Control 'self-optimizing' variables $c = Hy$ which have the property that keeping them constant ($c = c_s$) indirectly achieves close-to optimal operation (with a small loss), in spite of disturbances d. We here allow for linear measurement combinations, $c = Hy$. There are here two factors that should be considered:
   (a) Disturbances d. Ideally, we want the optimal value of c (c_{opt}) to be independent of d.
   (b) Measurements errors n^y. The loss should be insensitive to these.

A.1.2 Relationship to explicit MPC

Consider a simple static optimization problem $\min_u J(u,d)$, where u are the unconstrained degrees of freedom and the states x and the active constraints have been eliminated by substitution. For the quadratic case

$$\begin{aligned} J(u,d) &= [u \quad d]^T S [u \quad d] \\ \text{where } S &= \begin{bmatrix} J_{uu} & J_{ud} \\ J'_{ud} & J_{dd} \end{bmatrix}. \end{aligned} \tag{A.2}$$

In addition we have available 'measurements' $y = G^y u + G^d d$. A key result, which is the basis for this paper, is

For a quadratic optimization problem there exists (infinitely many) linear measurement combinations $c = Hy$ that are optimally invariant to disturbances d.

One sees immediately that there may be some link to explicit MPC, because the discrete form MPC problem can be written as a static quadratic problem. The link is: If we let y contain the inputs u and the states x, then the 'self-optimizing' variable combination $c = Hy$ is the same as the explicit MPC feedback law (control policy), i.e. $c = u - Kx$. (This is shown in section A.3.)

Based on this, we provide in this contribution some *new* ideas on explicit MPC:

1. We propose that tracking the variables c (deviation from optimal feedback law) for all regions, may be used as a local method to detect when to switch between regions.

2. We may use our results to include measurement error in y (e.g. in x and u) when deriving the optimal explicit MPC.

3. We may extend the results to output feedback ($c = u - Ky$) by including in y present and past outputs (and not present states x).

4. We can also extend the results to the case where only a subset of the states are measured (but in this case there will be a loss, which we can quantify). This may be of interest even in the unconstrained LQ case.

In this paper the basic framework and issue (1) are discussed. In [Manum et al., 2008c] it is shown how the results can be extended to handle items (2)-(4), both with theorems and examples.

A.2 Results from self-optimizing control

A.2.1 Steady state conditions

Once the set of active constraints in (P1) is known we can form the reduced problem and the unconstrained degrees of freedom u can be determined. The unconstrained measurements are

$$y = G^y u + G^y_d d, \tag{A.3}$$

and y contain information about the present state and disturbances (y may include u_0 and d, but not the active constraints.) The (measured) value of y_m available for implementation is

$$y_m = y + n^y, \tag{A.4}$$

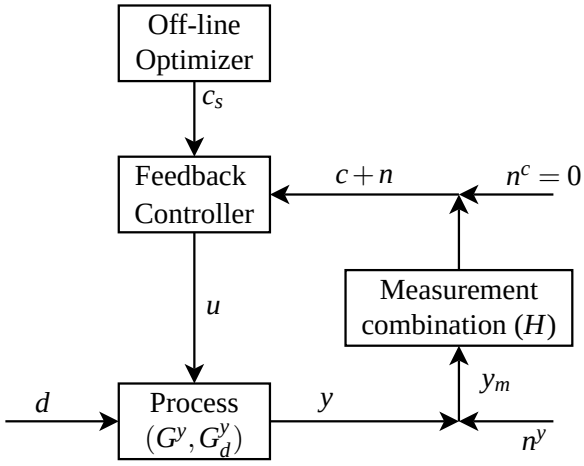

Figure A.1: Block diagram of a feedback control structure including an optimization layer [Alstad et al., 2009].

where n^y represents uncertainty in the measurement of y including uncertainty of implementation in u.

The following theorem describes a method to find linear invariants that yields zero loss from optimality when the invariants are controlled at constant setpoint. The theorem is based on the 'nullspace method' presented in [Alstad and Skogestad, 2007]. Figure A.1 illustrates how the H matrix is used to linearly combine measurements (and square down the plant).

Theorem A.1. *(Linear invariants for quadratic optimization problem [Alstad et al., 2009]) Consider an unconstrained quadratic optimization problem in the variables u (input vector of length n_u) and d (disturbance vector of length n_d)*

$$\min_u J(u,d) = \begin{bmatrix} u & d \end{bmatrix} \begin{bmatrix} J_{uu} & J_{ud} \\ J'_{ud} & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix} \tag{A.5}$$

In addition, there are 'measurement variables' $y = G^y u + G_d^y d$.

If there exists $n_y \geq n_u + n_d$ independent measurements (where 'independent' means that the matrix $\tilde{G}^y = \begin{bmatrix} G^y & G_d^y \end{bmatrix}$ has full rank), then the optimal solution to (A.5) *has the property that there exists $n_c = n_u$ linear variable combinations (constraints) $c = Hy$ that are invariant to the disturbances d. The optimal measurement combination matrix H is found by either: (1): Let $F = \frac{\partial y^{opt}}{\partial d'}$ be the optimal sensitivity matrix evaluated with constant active constraints. Under the assumptions*

stated above possible to select the matrix H in the left nullspace of F, $H \in \mathcal{N}(F')$, such that

$$HF = 0 \tag{A.6}$$

(2): If $n_y = n_u + n_d$:

$$H = M_n^{-1} \tilde{J} (\tilde{G}^y)^{-1}, \tag{A.7}$$

where $\tilde{J} = \begin{bmatrix} J_{uu}^{1/2} & J_{uu}^{-1/2} J_{ud} \end{bmatrix}$ and $\tilde{G}^y = \begin{bmatrix} G^y & G_d^y \end{bmatrix}$ is the augmented plant. M_n^{-1} may be seen as a free parameter. (Note that $M_n = J_{cc}$ is the Hessian of the cost with respect to the c-variables; in most cases we select $M_n = I$ for convenience.)

Remark A.1. *The sensitivity F matrix can be obtained from*

$$F = -\left(G^y J_{uu}^{-1} J_{ud} - G_d^y\right). \tag{A.8}$$

Remark A.2. *An equivalent formulation is: Assume that there exists a set of independent measurements y and that the (operational) constraint $c \triangleq Hy = c_s$ (where c_s is a constant) is added to the problem. Then there exists an H that does not change the solution to* (A.5)*. In terms of operation, this means that zero loss (optimal operation) is obtained by controlling $n_c = n_{u_0}$ variables $c = Hy$ with a constant set-point policy $c = c_s$, where H is selected according to theorem A.1.*

Theorem A.1 may be extended:

Lemma A.1. *(Linear invariants for constrained quadratic optimization methods) Consider an optimization problem of the form*

$$\begin{aligned} \min_{u_0, x} J_0 &= \begin{bmatrix} x & u_0 & d \end{bmatrix} S \begin{bmatrix} x \\ u_0 \\ d \end{bmatrix} \\ \text{s.t. } & Ax + Bu + Cd = 0 \\ & \tilde{A}x + \tilde{B}u + \tilde{C}d \leq 0, \end{aligned} \tag{A.9}$$

with $\det(A) \neq 0$ *and $[\tilde{A}\ \tilde{B}]$ full row rank.*

Assume that the disturbance space has been partitioned into n_a critical regions. In each region there are $n_u^i = n_{u_0} - n_A^i \geq 0$ unconstrained degrees of freedom, where $n_A^i \leq n_m$ is the number of optimally active constraints in region i.

If there exists a set of independent unconstrained measurements $y^i = (G^y)^i u^i + (G_d^y)^i d$ in each region i, such that $n_{y^i} \geq n_{u^i} + n_d$, the optimal solution to (A.9) *has the property that there exists variable combinations $c^i = H^i y^i$ that are invariant to*

the disturbances d in the critical region i. The corresponding optimal H^i may be obtained from Theorem A.1.

Within each region, optimality requires that $c^i - c_s^i = 0$ (where c_s^i is a constant). From continuity of the solution, we have that c^i is continuous across the boundary of region i. This implies that the elements in the variable vector $c^i - c_s^i$ will change sign or remain zero when crossing into or from a neighboring region.

Proof. See internal report [Manum et al., 2007]. □

A.2.2 Implementation of optimal solution

For the case of no measurement error, $n^y = 0$, Theorem A.1 shows that for the solution to quadratic optimization problems, variable combinations $c = Hy$ that are invariant to the disturbances can be found. In section A.3 this insight will be used as a new approach to the explicit MPC problem.

A.3 Application to explicit MPC

We will now look at the model predictive control problem (MPC) with constraints on inputs and outputs. For a discussion on MPC in a unified theoretical framework see Muske and Rawlings [1993].

The following discrete MPC formulation is based on [Pistikopoulos et al., 2002]. Consider the state-space representation of a given process model:

$$x(t+1) = Ax(t) + Bu(t) \tag{A.10}$$

$$y_0(t) = Cx(t), \tag{A.11}$$

subject to the following constraints:

$$y_{\min} \leq y_0(t) \leq y_{\max} \tag{A.12}$$

$$u_{\min} \leq u(t) \leq u_{\max}, \tag{A.13}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, and $y(t) \in \mathbb{R}^p$ are the state, input and output vectors, respectively, subscripts min and max denote the lower and upper bounds, respectively, and (A,B) is stabilizable. MPC problems for regulating to the origin can

then be posed as the following optimization problem:

$$\begin{aligned}
\min_U J(U,x(t)) &= x'_{t+N_y|y} P x_{t+N_y|t} + \\
&\quad + \sum_{k=0}^{N_y-1} \left[x'_{t+k|t} Q x_{t+k|t} + u'_{t+k} R u_{t+k} \right] \\
\text{s.t. } & y_{\min} \le y_{t+k|t} \le y_{\max}, \quad k = 1,\ldots,N_c \\
& u_{\min} \le u_{t+k} \le u_{\max}, \quad k = 0,1,\ldots,N_c \\
& x_{t|t} = x(t) \\
& x_{t+k+1|t} = A x_{t+k|t} + B u_{t+k}, \quad k \ge 0 \\
& y_{t+k|t} = C x_{t+k|t}, \quad k \ge 0 \\
& u_{t+k} = K x_{t+k|t}, \quad N_u \le k \le N_y
\end{aligned}$$

where $U \triangleq \{u_t,\ldots,u_{t+N_u-1}\}$, $Q = Q' \ge 0$, $R = R' > 0$, $P \ge 0$, $N_y \ge N_u$, and K is some feedback gain. Pistikopoulos et al. [2002] show that by substitution of the model equations, the problem can be rewritten on the form

$$\begin{aligned}
&\min_U \frac{1}{2} U'HU + x(t)'FU + \frac{1}{2} x(t)'Yx(t) \\
&\text{s.t. } GU \le W + Ex(t)
\end{aligned} \tag{A.14}$$

The MPC control law is based on the following idea: At time t, compute the optimal solution $U^*(t) = \{u_t^*,\ldots,u_{t+N_u-1}^*\}$ and apply $u(t) = u_t^*$ [Bemporad et al., 2002].

Remark A.3. *The trade-off between robustness and performance is included in the weights in the MPC cost function and in the constraints.*

If we let the initial state $x(t)$ be treated as a disturbance, (A.14) can be written as:

$$\begin{aligned}
&\min_U \frac{1}{2} \begin{bmatrix} U' & d' \end{bmatrix} \begin{bmatrix} H & F \\ F & Y \end{bmatrix} \begin{bmatrix} U \\ d \end{bmatrix} \\
&\text{s.t. } GU \le W + Ed,
\end{aligned} \tag{A.15}$$

and we observe that (A.15) is on the same form as (A.9), where the model equations $f(x,u_0,d) = 0$ have already been substituted into the objective function.

A property of the solution to (A.15) is that the disturbance space (initial state space) is divided into critical regions. In the i'th critical region there are $n_u^i = n_U - n_A^i$ unconstrained degrees of freedom, where n_A^i is the number of active constraints in region i.

As we discuss in section A.3.1, a possible set of measurements y is the current state and the inputs, $y' = \begin{bmatrix} x' & u' \end{bmatrix}$. We further note that causality is not an issue here, as we have the information at the current time.

A.3.1 Exact measurements of all states (state feedback)

The following theorem is well known, but we will here prove the theorem using the nullspace method.

Theorem A.2. *(Optimal state feedback [Bemporad et al., 2002]) The control law $u(t) = f(x(t))$, $f : \mathbb{R}^n \mapsto \mathbb{R}^m$, defined by the MPC problem, is continuous and piecewise affine*

$$f(x) = K^i x + g^i \quad \text{if } H^i x \leq k^i, \quad i = 1, \ldots, N_{mpc} \tag{A.16}$$

where the polyhedral sets $\left\{H^i x \leq k^i\right\}, \quad i = 1, \ldots, N_{mpc} \leq N_r$ are a partition of the given set of states X.

In this case causality is not a problem and from Theorem A.1 the optimal solution is simply $u = Kx + g$ (i.e. $c = u - (Kx - g)$). Note that $n_d = n_x$ in this case.

Proof. We consider the explicit MPC formulation as in (A.15). First we consider the unconstrained case. Let $y = (U, x)$ be the set of candidate measurements. With this choice of measurements and disturbances on the present state, we form the process model:

$$\Delta y = G^y \Delta U + G_d^y \Delta d \tag{A.17}$$

$$G^y = \begin{bmatrix} 0_{n_x \times (n_u N_u)} \\ I_{(n_u N_u) \times (n_u N_u)} \end{bmatrix} \in \mathbb{R}^{(n_x + n_u N_u) \times (n_u N_u)} \tag{A.18}$$

$$G_d^y = \begin{bmatrix} I_{n_x \times n_x} \\ 0_{(n_u N_u) \times n_x} \end{bmatrix} \in \mathbb{R}^{(n_x + n_u N_u) \times n_x}. \tag{A.19}$$

We then get the optimal sensitivity as

$$F = \frac{\partial y^{\text{opt}}}{\partial d'} = -\left(G^y J_{uu}^{-1} J_{ud} - G_d^y\right) = \tag{A.20}$$

$$-\left(\begin{bmatrix} 0_{n_x \times (n_u N_u)} \\ \left(J_{uu}^{-1} J_{ud}\right)_{(n_u N_u) \times n_x} \end{bmatrix} - \begin{bmatrix} I_{n_x \times n_x} \\ 0_{(n_u N_u) \times n_x} \end{bmatrix}\right) \tag{A.21}$$

$$= \begin{bmatrix} I_{n_x \times n_x} \\ -J_{uu}^{-1} J_{ud} \end{bmatrix} \tag{A.22}$$

We now search for a matrix H that gives a non-trivial solution to $HF = 0$:

$$\left[(H_1)_{(n_u N_u) \times n_x} (H_2)_{(n_u N_u) \times (n_u N_u)}\right] \begin{bmatrix} I_{n_x \times n_x} \\ J_{uu}^{-1} J_{ud} \end{bmatrix} = \tag{A.23}$$

$$= H_1 - H_2 \left(J_{uu}^{-1} J_{ud}\right) = 0 \tag{A.24}$$

To ensure a non-trivial solution we can for example choose $H_2 = I_{(n_u N_u)\times n_u N_u}$. Then we must have $H_1 = J_{uu}^{-1} J_{ud}$, and hence the *optimal combination c of x and U* becomes

$$c = Hy = J_{uu}^{-1} J_{ud} x + U = 0 \in \mathbb{R}^{(n_u N_u)} \tag{A.25}$$

In the internal report by Manum et al. [2007] it is shown how the affine term in (A.16) enters as a function of the active constraints. □

Remark A.4. *(Comparison with previous results on unconstrained MPC) In* (A.25) *the state feedback gain matrix is given as $J_{uu}^{-1} J_{ud}$. This is gives the same result as conventional MPC, see equation (3) in [Rawlings and Muske, 1993].*

Remark A.5. *These are not new results but the alternative proof leads to some new insights. The most important is probably that the "self-optimizing" variables $c^i = u - (K^i x + g^i)$ which are optimally zero in region i, may be used for identifying when to switch between regions (Theorem A.3) rather than using a "centralized" approach, for example based on a state tree structure search. This seems to be new. Another insight is to understand why a simple feedback solution must exist in the first place. A third is to allow for new extensions.*

Theorem A.3. *(Optimal region for explicit MPC detection using feedback law) The variables $c = u_k - (Kx_k + g)$ can be used to identify region changes.*

Proof. See report by Manum et al. [2007]. □

Remark A.6. *Neighboring regions with the same feedback law (including regions where the feedback law is to keep the input saturated) can be merged (provided that the regions remain convex or if the "crossings" inside a non-convex region due to the optimal direction of the process in closed loop only occurs in the convex part of the region). This may greatly reduce the number of regions compared to presently used enumeration schemes. Note that the number of c-variables that need to be tracked to detect region changes is only equal to the number of inputs n_{u_0} times the number of distinct merged regions. Because of the merging of regions, this may be a small number even with a large input or control horizon and with output (state) constraints.*

We present a simple example from Bemporad et al. [2002] that confirms that our switching policy based on tracking the sign of the *c*-variables works in practice.

Example A.3.1 (Optimal switching)**.** *This example is taken from Bemporad et al. [2002] (with correction), and is included here to demonstrate optimal switching using the sign change of $c = u - Kx$ as the criterion. The system is:*

$$y(t) = \frac{2}{s^2 + 3s + 2} u(t).$$

Algorithm A.1 Detect current region and calculate u_k

Require: CR_{k-1}, i.e. the region of the last sample time, and x_k

$u_k = K(CR_{k-1}) + g(CR_{k-1})$
$[\text{Regions}, \alpha] = \text{Neighbors}(CR_{k-1})$
for $i = 1$ to length(Regions) **do**
  $c_k(i) = \alpha_i \left(u_k - (K(\text{Regions}(i)) + g(\text{Regions}(i)))\right)$
end for
if sign$(c_k(i) \neq -1$) **then**
  $CR_k = \text{Regions}(i)$
else
  $CR_k = CR_{k-1}$
end if
return $u_k = K(CR_k)x_k + g(CR_k), CR_k$

With a sampling time $T = 0.1$ seconds the following state-space representation is obtained:

$$x(t+1) = \begin{bmatrix} 0.7326 & -0.0861 \\ 0.1722 & 0.9909 \end{bmatrix} x(t) + \begin{bmatrix} 0.0609 \\ 0.0064 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 0 & 1.4142 \end{bmatrix} x(t)$$

One observes that only the last state is measured, but it will be assumed that both states are known (measured) in the remainder of this example.

The task is to regulate the system to the origin while fulfilling the input constraint

$$-2 \leq u(t) \leq 2. \tag{A.26}$$

The objective function to be minimized is

$$\min x'_{t+2|t} P x_{t+2|t} + \sum_{k=0}^{1} \left[x'_{t+k|t} x_{t+k|t} + 0.01 u^2_{t+k} \right] \tag{A.27}$$

subject to the constraints and $x_{t|t} = x(t)$.

P solves the Lyapunov equation $P = A'PA + Q$, where $Q = I$ in this case. The optimal control problem can be solved for example using the MPT toolbox [Kvasnica et al., 2004]. The P-matrix is numerically:

$$P = \begin{bmatrix} 5.5461 & 4.9873 \\ 4.9873 & 10.4940 \end{bmatrix}$$

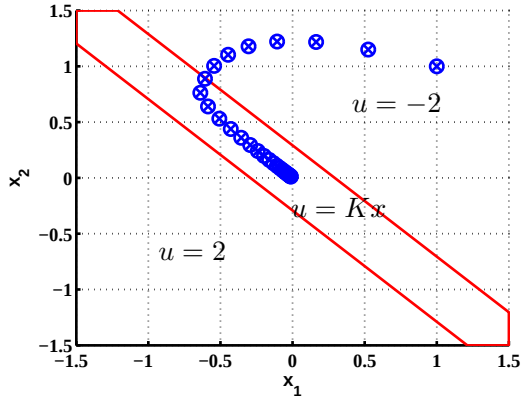

Figure A.2: Partition of state space for first input. (Example A.3.1.)

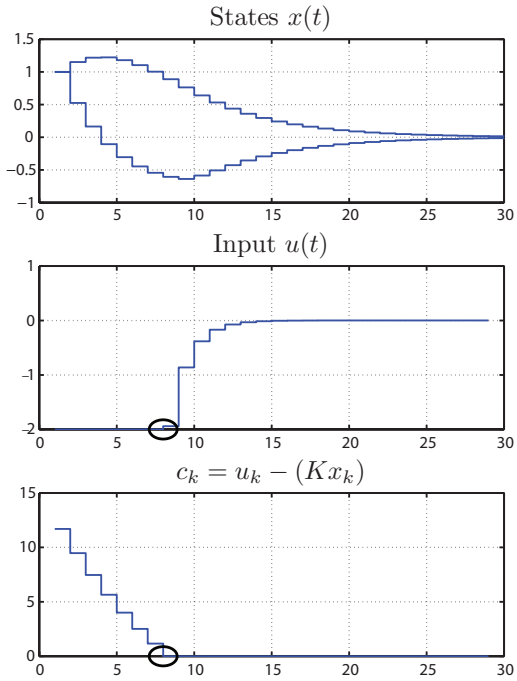

Figure A.3: Closed loop MPC with region detection using $u_k - (Kx_k)$. (Example A.3.1.)

To illustrate ideas a simulation from $x_0 = (1,1)$ was done. State space trajectories and inputs are shown in figures A.2 and A.3. As long as the state is in the input-constrained region where $u^{opt} = -2$, the linear combination $c = u_k - Kx_k$ remains positive. One chooses to leave the input-constrained region when c_k becomes zero. As one observes, this happens at time instant 8, where the process indeed is on the boundary between the input-saturated region and the center region. After the switching the controller for the center region is implemented. The state trajectory is the same as in [Bemporad et al., 2002].

The reason for why c never becomes negative is because both states are assumed measured at the present time and hence optimal switching is achieved. This can be understood from the algorithm A.1, where we show how the current critical region (CR_k) is tracked and how the current input u_k is calculated.

Example A.3.2 (Double integrator)**.** *Consider the double integrator disussed by Bemporad et al. [2002], $y(t) = 1/s^2 u(t)$, and its equivalent discrete-time state-space representation,*

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_k, \qquad y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k,$$

which is obtained by setting $\ddot{y}(t) = (\dot{y}(t+T_s) - \dot{y}(t))/T_s$, $\dot{y}(t) = (y(t+T_s) - y(t))/T_s$, $T_s = 1$. The control objective is to regulate the system to the origin while minimizing the quadratic cost funcion $J = \sum_{t=0}^{\infty} y(t)'y(t) + \frac{1}{10}u^2$ subject to the input constraint $-1 \leq u(t) \leq 1$. The infinite horizion control problem can be converted to a finite horizion problem by solving [Bemporad et al., 2002, Chmielewski and Manousiouthakis, 1996]:

$$\begin{aligned} K_{LQ} &= -(R + B'PB)^{-1}B'PA, \\ P &= (A + BK_{LQ})'P(A + BK_{LQ}) + K'_{LQ}RK_{LQ} + Q \end{aligned}$$

to obtain the unconstrained feedback gain K_{LQ} and the final state weight matrix P (see example A.3.1). In this case we get $K_{LQ} = \begin{bmatrix} 0.8166 & 1.7499 \end{bmatrix}$ and $P = \begin{bmatrix} 2.1429 & 1.2246 \\ 1.2246 & 1.3996 \end{bmatrix}$. For demonstration purposes we choose $N_u = 6$, and by solving the paramteric program we get 73 regions initially. In this case there are 11 regions of unsaturated control actions, which agrees with the general result of $(2N_u - 1)$ regions given in [Bemporad et al., 2002]. Merging all regions where the first optimal input is the same, leaves us with the 11 unsaturated regions, and two regions for which the optimal input is either at the high or low constraint. The final partitioning with 13 regions is shown in figure A.4. We note that [Bemporad et al., 2002] find 57 regions after their merging scheme.

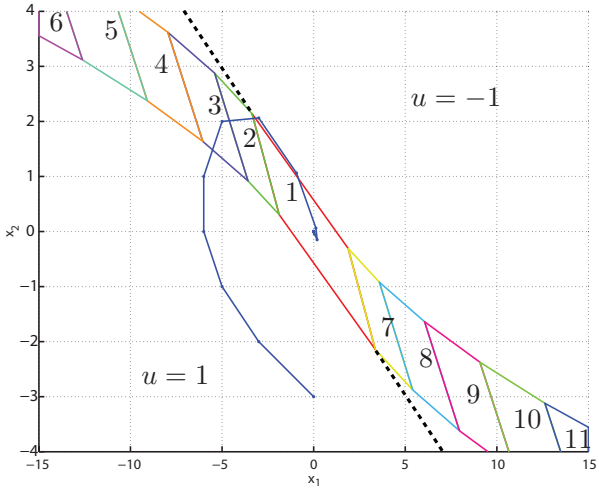

Figure A.4: Regions for double integrator example (Example 3.2).

Considering figure A.4 one observes that the input-saturated regions are non-convex. However, optimally, this process moves clockwise in the state space, and we observe that the "non-convex" crossings will not occur in practise. The remaing boundaries then form convex regions (indicated by the dashed lines in the figure.)

Figure A.5 shows the evolution of the invariants c^i in each region when we start the simulation at $x_0 = (0, -3)$ and close the loop by using the optimal inputs. We start in the input-saturated region $u = 1$, and need to track the invariants for regions 1,2,3,4,5, and 6 to determine optimal switching. We should switch to unsaturated control when one the variables c^1 to c^6 becomes zero or changes sign. As one sees, this happens for c^3 at $t = 6$, so we change to this region. After using the feedback law for region 3 for one sample time, we reach the other input constraint $u = -1$ at $t = 8$. Now, to decide when to leave this constrained region we track the invariants for regions 1,7,8,9,10,11, and we observe that at $t = 9$ the invariant for the center region becomes zero, hence we switch control to this region.

Note that in this example, where we have only input constrained regions, the challenge is to decide when to leave the input constraints. Note that the converse crossing can also be tracked using their invariants on the form $c_k = u_k - g$.

The idea of using directionality (clockwise movement in this case) to reduce the number of reigons in explicit MPC can be generalized by using the directional derivative of the process under optimal control, $(A - BK^i)$, together with the normal vectors to the boundaries of the regions, and by some normalization scheme

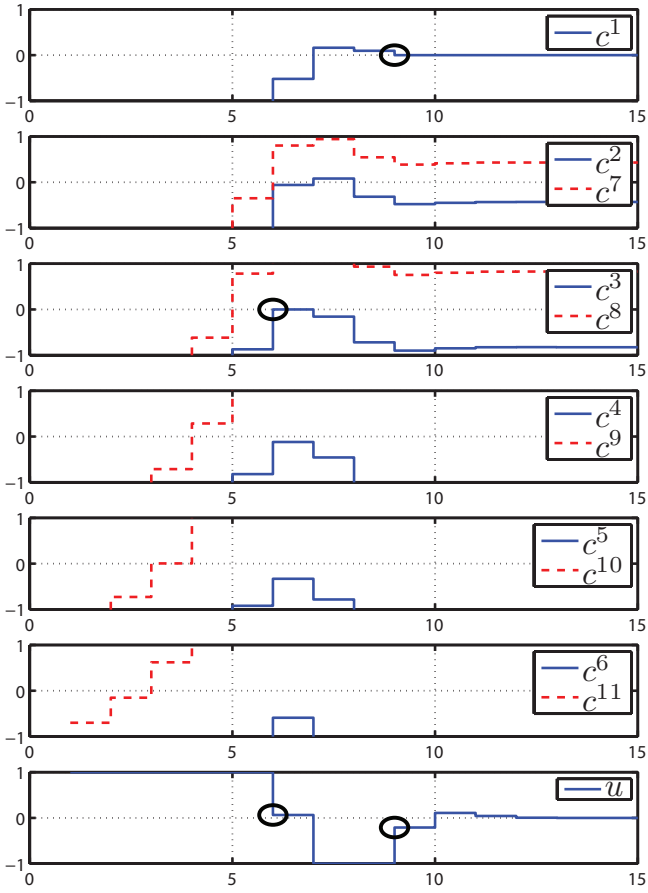

Figure A.5: Invariants for double integrator example.

we remove all boundaries for which crossings under optimal control will not occur. Here K^i is the optimal feedback gain for region i.

A.4 Discussion

In this paper we have described the link between self-optimizing control and explicit MPC. This link has been used to propose a new method for detecting region changes. This new method lets us reduce the number of regions by merging all regions for which the first input is the same. In its simple form presented in this paper, it does not handle non-convex regions, but we noted that for some processes directionality of the process in closed loops implies that the non-convex crossings may be ignored.

In a forthcoming contribution [Manum et al., 2008c] we show how the results can be extended to output feedback and how to find invariants that give minimal loss when controlled at constant set points also when we have noisy measurements. We further show how we one choose the order of the controller and we show by examples that the resulting controller will have performance in the order of magnitude of LQG controllers.

The most important problem of using results from steady state self-optimizing control is causality, in steady state optimization all measurements are available at the current time (i.e. $t \to \infty$), but in dynamic optimization we may need to find invariants between measurements at current and future times and then switch the invariants back to get a casual controller, but this controller will be non-optimal by construction. Also this is discussed in more detail in [Manum et al., 2008c].

Appendix B

Explicit MPC with output feedback using self-optimizing control

Published in "Proceedings of IFAC World Congress 2008, Seoul, South Korea."

Model predictive control (MPC) is a favored method for handling constrained linear control problems. Normally, the MPC optimization problem is solved on-line, but in 'explicit MPC' an explicit precomputed feedback law is used for each region of active constraints [Bemporad et al., 2002]. In this paper we make a link between this and the 'self-optimizing control' idea of finding simple policies for implementing optimal operation. The 'nullspace' method [Alstad and Skogestad, 2007] generates optimal variable combinations, $c = u - Kx$, which for the case with perfect state measurements are equivalent to the explicit MPC feedback laws, where K is the optimal state feedback matrix in a given region. More importantly, this link makes it possible to derive explicit feedback laws for cases with (1) state measurement error included and (2) measurement (rather than state) feedback. We further show how to generate optimal low-order controllers for unconstrained optimal control, also in the presence of noise.

B.1 Introduction

Consider the general static optimization problem [Alstad and Skogestad, 2007]:

$$\begin{aligned} \min_{u_0,x} \quad & J_0(x,u_0,d) \\ \text{s.t.} \quad & f_i(x,u_0,d) = 0, \quad i \in \mathcal{E} \\ & h_i(x,u_0,d) \geq 0, \quad i \in \mathcal{I}, \end{aligned} \tag{P1}$$

where $x \in \mathbb{R}^{n_x}$ are the states, $u_0 \in \mathbb{R}^{n_{u_0}}$ are the inputs, and $d \in \mathcal{D} \subset \mathbb{R}^{n_d}$ are disturbances. By discretization and reformulation this may also represent some dynamic optimization problems. Usually f is a model of the physical system, whilst h is a set of inequality constraints that limits the operation (e.g., physical limits on temperature measurements or flow constraints). In addition to (P1) we have measurements on the form

$$y_0 = f^y(x, u_0, d). \tag{B.1}$$

In this work the emphasis is on *implementation of the solution to* (P1). This means that the optimization problem (P1) is solved off-line to generate a 'control policy' which is suitable for on-line implementation, with particular emphasis on remaining close to optimal solution when there are unknown disturbances.

In our previous work on 'self-optimizing control' we have looked for simple control policies to implement optimal operation, and in particular 'what should we control' (choice of controlled variables (*CV*'s)). Using off-line optimization we may determine regions where different sets of active constraints are active, and implementation of optimal operation is then in each region to:

1. Control the active constraints.

2. For the remaining unconstrained degrees of freedom: Control 'self-optimizing' variables $c = Hy$ which have the property that keeping them constant ($c = c_s$) indirectly achieves close-to optimal operation (with a small loss), in spite of disturbances d.

A key result, which is the basis for this paper, is

For a quadratic optimization problem there exists (infinitely many) linear measurement combinations $c = Hy$ that are optimally invariant to disturbances d.

One sees immediately that there may be some link to explicit MPC, because the discrete form MPC problem can be written as a static quadratic problem. The link is: If we let y contain the inputs u and the states x, then the 'self-optimizing' variable combination $c = Hy$ is the same as the explicit MPC feedback law, i.e. $c = u - Kx$. (This is shown in section B.3.)

Based on this, we provide in this contribution some *new* ideas on explicit MPC:

1. We propose that tracking the variables c (deviation from optimal feedback law) for all regions, may be used as a local method to detect when to switch between regions. (This is discussed in Manum et al. [2008b].)

2. We extend the results to output feedback ($c = u - Ky$) by including in y present and past outputs.

3. For unconstrained optimal control, we show how the links can be used to give low-order controllers that give a small loss from optimality also for noisy measurements.

4. We also extend the results to the case where only a subset of the states are measured (but in this case there will be a loss, which we can quantify). This may be of interest even in the unconstrained LQ case.

B.2 Results from self-optimizing control

In this section we will present results from previous work on self-optimizing control and relate them to quadratic optimization problems.

B.2.1 Steady state conditions

Once the set of active constraints is known, we can form the reduced problem and the unconstrained degrees of freedom u can be determined. The unconstrained measurements are

$$y = G^y u + G_d^y d, \tag{B.2}$$

and y contain information about the present state and disturbances (y may include u_0 and d, but not the active constraints.) The (measured) value of y_m available for implementation is

$$y_m = y + n^y, \tag{B.3}$$

where n^y represents uncertainty in the measurement of y including uncertainty of implementation in u.

The following theorem describes a method to find linear invariants that yields zero loss from optimality when the invariants are controlled at constant setpoint. The theorem is based on the 'nullspace method' presented in Alstad and Skogestad [2007].

Theorem B.1. *(Linear invariants for quadratic optimization problem [Alstad et al., 2009]) Consider an unconstrained quadratic optimization problem in the variables u (input vector of length n_u) and d (disturbance vector of length n_d)*

$$\min_u J(u,d) = \begin{bmatrix} u & d \end{bmatrix} \begin{bmatrix} J_{uu} & J_{ud} \\ J'_{ud} & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix} \tag{B.4}$$

In addition, there are 'measurement variables' $y = G^y u + G_d^y d$.

If there exists $n_y \geq n_u + n_d$ independent measurements (where 'independent' means that the matrix $\tilde{G}^y = \begin{bmatrix} G^y & G_d^y \end{bmatrix}$ has full rank), then the optimal solution to (B.4) *has the property that there exists $n_c = n_u$ linear variable combinations (constraints) $c = Hy$ that are invariant to the disturbances d. The optimal measurement combination matrix H is found by either: (1): Let $F = \frac{\partial y^{opt}}{\partial d'}$ be the optimal sensitivity matrix evaluated with constant active constraints. Under the assumptions stated above possible to select the matrix H in the left nullspace of F, $H \in \mathcal{N}(F')$, such that*

$$HF = 0 \tag{B.5}$$

(2): If $n_y = n_u + n_d$:

$$H = M_n^{-1} \tilde{J} (\tilde{G}^y)^{-1}, \tag{B.6}$$

where $\tilde{J} = \begin{bmatrix} J_{uu}^{1/2} & J_{uu}^{-1/2} J_{ud} \end{bmatrix}$ and $\tilde{G}^y = \begin{bmatrix} G^y & G_d^y \end{bmatrix}$ is the augmented plant. M_n^{-1} may be seen as a free parameter. (Note that $M_n = J_{cc}$ is the Hessian of the cost with respect to the c-variables; in most cases we select $M_n = I$ for convenience.)

Remark B.1. *The sensitivity F matrix can be obtained from*

$$F = -\left(G^y J_{uu}^{-1} J_{ud} - G_d^y \right). \tag{B.7}$$

Theorem B.1 may be extended:

Lemma B.1. *(Linear invariants for constrained quadratic optimization methods [Manum et al., 2008b]) Consider an optimization problem of the form*

$$\begin{aligned} \min_{u_0, x} J_0 = \begin{bmatrix} x & u_0 & d \end{bmatrix} S \begin{bmatrix} x \\ u_0 \\ d \end{bmatrix} \\ s.t.\ Ax + Bu + Cd = 0 \\ \tilde{A}x + \tilde{B}u + \tilde{C}d \leq 0, \end{aligned} \tag{B.8}$$

with $\det(A) \neq 0$ *and* $[\tilde{A}\ \tilde{B}]$ *full row rank.*

Assume that the disturbance space has been partitioned into n_a critical regions. In each region there are $n_u^i = n_{u_0} - n_A^i \geq 0$ unconstrained degrees of freedom, where $n_A^i \leq n_m$ is the number of optimally active constraints in region i.

If there exists a set of independent unconstrained measurements $y^i = (G^y)^i u^i + (G_d^y)^i d$ in each region i, such that $n_{y^i} \geq n_{u^i} + n_d$, the optimal solution to (B.8) *has the property that there exists variable combinations $c^i = H^i y^i$ that for critical region i are invariant to the disturbances d. The corresponding optimal H^i may be*

obtained from Theorem B.1. Within each region, optimality requires that $c^i - c_s^i = 0$ (where c_s^i is a constant). From continuity of the solution we have that c^i is continuous across the boundary of region i. This implies that the elements in the variable vector $c^i - c_s^i$ will change sign or remain zero when crossing into or from a neighboring region.

B.2.2 Including noise

For the noise-free problem, adding the constraints $c = Hy = c_s$ does not change the optimal solution (Theorem B.1). However with measurement noise there will be some loss, which can be minimized if H is selected as given in Theorem B.2.

Theorem B.2. *(Loss by introducing linear constraint for noisy quadratic optimization problem [Alstad et al., 2009]) Consider the unconstrained quadratic optimization problem in Theorem B.1:*

$$\min_u J(u,d) = \begin{bmatrix} u & d \end{bmatrix} \begin{bmatrix} J_{uu} & J_{ud} \\ J_{ud}' & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix}$$

and a set of noisy measurements $y_m = y + n^y$. Assume that $n_c = n_u$ constraints $c = Hy_m = c_s$ are added to the problem, which will result in a non-optimal solution with loss $L = J(u,d) - J_{\text{opt}}(d)$. Consider the disturbances d and the noise n^y with magnitudes:

$$d = W_d d'; \quad n^y = W_{n^y} n^{y'}; \quad \| \begin{bmatrix} d' \\ n^{y'} \end{bmatrix} \| \leq 1. \tag{B.9}$$

Then, for a given H, the worst-case loss is $L_{\text{wc}} = \bar{\sigma}(M)^2/2$, where $M = [M_d \; M_{n^y}]$ is given by

$$M_d = -J_{uu}^{1/2} (HG^y)^{-1} HFW_d, \tag{B.10}$$

$$M_{n^y} = -J_{uu}^{1/2} (HG^y) HW_{n^y}., \tag{B.11}$$

and the optimal H that minimizes $\bar{\sigma}(M)$ is given by

$$H' = \left(\tilde{F}\tilde{F}'\right)^{-1} G^y (G^{y\prime} (\tilde{F}\tilde{F}')^{-1} G^y)^{-1} J_{uu}^{1/2}, \tag{B.12}$$

where $\tilde{F} = [FW_d \; W_{n^y}]$. This solution also minimizes the average loss $\|M\|_F$.

Remark B.2. *The optimal H can also be found by solving the constrained optimization problem*

$$H = \arg\min_H \bar{\sigma}(H\tilde{F}) \text{ subject to } HG^y = J_{uu}^{1/2} \tag{B.13}$$

B.3 Application to explicit MPC

Pistikopoulos et al. [2002] show that by substitution of the model equations, the linear MPC problem can be rewritten to the form

$$\begin{aligned} &\min_U \frac{1}{2}U'HU + x(t)'FU + \frac{1}{2}x(t)'Yx(t) \\ &\text{s.t. } GU \leq W + Ex(t) \end{aligned} \tag{B.14}$$

The MPC control law is based on the following idea: At time t, compute the optimal solution $U^*(t) = \{u_t^*, \ldots, u_{t+N_u-1}^*\}$ and apply $u(t) = u_t^*$ [Bemporad et al., 2002].

If we let the initial state $x(t)$ be treated as a disturbance, (B.14) can be written as:

$$\begin{aligned} &\min_U \frac{1}{2}\begin{bmatrix} U' & d' \end{bmatrix}\begin{bmatrix} H & F \\ F & Y \end{bmatrix}\begin{bmatrix} U \\ d \end{bmatrix} \\ &\text{s.t. } GU \leq W + Ed, \end{aligned} \tag{B.15}$$

and we observe that (B.15) is on the same form as (B.8), where the model equations $f(x, u_0, d) = 0$ have already been substituted into the objective function.

A property of the solution to (B.15) is that the disturbance space (initial state space) will be divided into critical regions. In the i'th critical region there will be $n_u^i = n_U - n_A^i$ unconstrained degrees of freedom, where n_A^i is the number of active constraints in region i.

As we will discuss in section B.3.1, a possible set of measurements y is the current state and the inputs, $y' = \begin{bmatrix} x' & u' \end{bmatrix}$. We further note that causality is not an issue here, as we have the information at the current time.

B.3.1 Exact measurements of all states (state feedback)

The following theorem is well known, but we shown in [Manum et al., 2008b] that it can be derived using the nullspace method. The proof is left out here due to space limitations.

Theorem B.3. *(Optimal state feedback [Bemporad et al., 2002]) The control law $u(t) = f(x(t))$, $f : \mathbb{R}^n \mapsto \mathbb{R}^m$, defined by the MPC problem, is continuous and piecewise affine*

$$f(x) = K^i x + g^i \quad \text{if } H^i x \leq k^i, \quad i = 1, \ldots, N_{mpc} \tag{B.16}$$

where the polyhedral sets $\left\{H^i x \leq k^i\right\}$, $i = 1, \ldots, N_{mpc} \leq N_r$ are a partition of the given set of states X.

Remark B.3. *(Comparison with previous results on unconstrained MPC) In the proof shown in Manum et al. [2008b] the state feedback gain matrix is given as* $J_{uu}^{-1}J_{ud}$*. This is gives the same result as conventional MPC, see equation (3) in Rawlings and Muske [1993].*

Remark B.4. *Our alternative proof of Theorem B.3 leads to some new insights. The most important is probably that the 'self-optimizing' variables* $c^i = u - (K^i x + g^i)$ *which are optimally zero in region i, may be used for identifying when to switch between regions (Theorem B.4) rather than using a 'centralized' approach, for example based on a state tree structure search. This seems to be new. Another insight is to understand why a simple feedback solution must exist in the first place. A third is to allow for new extensions.*

Theorem B.4. *(Optimal region for explicit MPC detection using feedback law [Manum et al., 2008b]) The variables* $c = u_k - (Kx_k + g)$ *can be used to identify region changes.*

An algorithm for implementing the region detection scheme is presented in Manum et al. [2008b].

We present a simple example from Bemporad et al. [2002] that confirms that our switching policy based on tracking the sign of the *c*-variables works in practice.

Example B.3.1 (Optimal switching)**.** *This example is taken from Bemporad et al. [2002] (with correction), and is included here to demonstrate optimal switching using* $c = u - Kx$ *as criterion. For more details on this example see [Manum et al., 2008b]. The system is:*

$$y(t) = \frac{2}{s^2 + 3s + 2} u(t).$$

With a sampling time $T = 0.1$ *seconds the following state-space representation is obtained:*

$$x(t+1) = \begin{bmatrix} 0.7326 & -0.0861 \\ 0.1722 & 0.9909 \end{bmatrix} x(t) + \begin{bmatrix} 0.0609 \\ 0.0064 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 0 & 1.4142 \end{bmatrix} x(t)$$

One observes that only the last state is measured, but it will be assumed that both states are known (measured) in the remainder of this example.

The task is to regulate the system to the origin while fulfilling the input constraint

$$-2 \leq u(t) \leq 2. \tag{B.17}$$

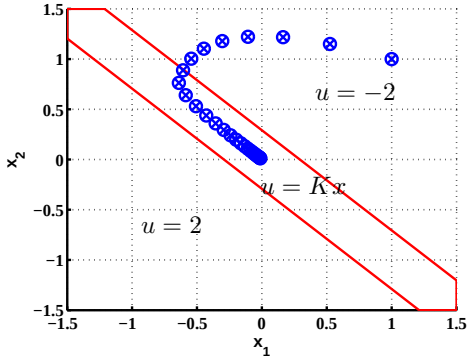

Figure B.1: Partition of state space for first input. (Example B.3.1.)

The objective function to be minimized is

$$\min x'_{t+2|t} P x_{t+2|t} + \sum_{k=0}^{1} \left[x'_{t+k|t} x_{t+k|t} + 0.01 u^2_{t+k} \right] \tag{B.18}$$

subject to the constraints and $x_{t|t} = x(t)$.

P solves the Lyapunov equation $P = A'PA + Q$*, where* $Q = I$ *in this case. The P-matrix is numerically* $P = \left[\begin{smallmatrix} 5.5461 & 4.9873 \\ 4.9873 & 10.4940 \end{smallmatrix}\right]$*. The optimal control problem can be solved for example using the MPT toolbox [Kvasnica et al., 2004].*

To illustrate the ideas, we show a simulation where the control objective is to bring the process from $x_0 = (1,1)$ *and back to* $x = (0,0)$*. State space trajectories and inputs are shown in figures B.1 and B.2 (dotted line). As long as the state is in the input-constrained region where* $u^{opt} = -2$*, the linear combination* $c = u_k - Kx_k$ *remains positive. One chooses to leave the input-constrained region when c becomes zero. The state trajectory is the same as in Bemporad et al. [2002].*

B.3.2 Output feedback with no noise

Consider now the case where all the states x are not measured. The objective is to find linear combinations $c = Hy$ that are optimally constant in each optimal region. From the nullspace method, this requires that we have as many independent measurements y as there are inputs and disturbances.

With no measurement error, the optimal combination $c = Hy$ can be obtained from the nullspace method. This requires that $\tilde{G}_y$ has full rank, which again implies that all d's can be observed from the outputs y. Because of causality, $\tilde{G}_y$ will not be full rank initially (just after the disturbance occurs), but the rank condition will

be satisfied if we consider a disturbance entering sufficiently long ($n_x - 1$ steps) back in time. From this time and on the solution is the same as the state feedback solution.

In terms of detecting region changes, we suggested for the state feedback case to use the deviation c from the optimal feedback laws $c = u - Kx$ as tracking variables. This simple strategy may not work as well with output feedback, partly because output feedback is not truly optimal, and partly because the outputs do not contain accurate information about the present state. (It can however be applied in the following example.)

Example B.3.2 (Output feedback)**.** *Consider the same model and optimal control problem as in example B.3.1, but assume that only the output $y(t)$ is available (and not both states). Recall from figure B.1 that the state space is optimally partitioned into 3 regions with 3 different state feedback laws. As before, let $d = x_k$.*

One approach is to find the optimal sensitivity F from $F = -(G^y J_{uu}^{-1} J_{ud} - G_d^y)$, where $y = (y_k, y_{k+1}, U)$, and

$$y = \begin{bmatrix} y_k \\ y_{k+1} \\ u_k \\ u_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 \\ CB & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{G^y} \begin{bmatrix} u_k \\ u_{k+1} \end{bmatrix} + \underbrace{\begin{bmatrix} C \\ CA \end{bmatrix}}_{G_d^y} x_k \tag{B.19}$$

By finding an H such that $HF = 0$, this method yields feedback gains from the outputs to the inputs. Note that we can always 'decouple' the invariants in the inputs u when all inputs are included in the candidate vector y. This is because $n_c = n_u$ and we have a degree of freedom in H such that multiplying by a non-singular $n_c \times n_c$ matrix on the left yields the same loss as before. Write $H = [H^y \; H^u]$, then a combination matrix that is decoupled in u is $\hat{H} = (H^u)^{-1} H$.

We here get two invariants, one between (u_{k+1}, y_{k+1}, y_k), and one between u_k, y_{k+1}, y_k, where only the first one is implementable because of causality.

The controller gains for the central region are $(k_1, k_2) = (-16.7, 13.7)$, with control equation $u_k = -(k_1 y_k + k_2 y_{k-1})$.

Another approach for finding F and H is to use the optimal solution $u_k = -Kx_k$ a priori*, which we did not do above. If we use the knowledge of the optimal feedback law, we can for example find that*

$$F'' = \left(\frac{\partial [u_k \; y_k \; y_{k+1}]'}{\partial x_k} \right)^{\text{opt}} = \begin{bmatrix} K' & C' & (C(A+BK))' \end{bmatrix}, \tag{B.20}$$

and by solving $H'F' = 0$ we get an invariant between (u_k, y_k, y_{k+1}). This invariant is not implementable, but by using the same idea we can find another invariant

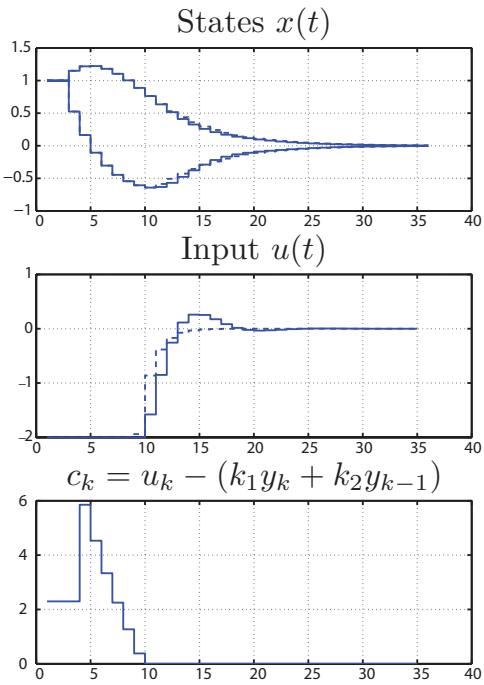

Figure B.2: Simulation of output feedback configuration, where the output feedback law is used both for switching and control. Dotted line is optimal switching and control when both states assumed measured. (Examples B.3.1 and B.3.2.)

between (y_k, y_{k+1}, y_{k+2})*, shift this invariant one time-step back and then combine with the first one. The resulting output feedback law becomes the same as for the method above, where we did not use the optimal state feedback law in the derivations.*

Figure B.2 shows the result of a simulation of the output feedback control from $x_0 = (1,1)$*. Note that we use the output feedback control law for the unconstrained region to decide when to leave the constrained region. The previous optimal control with both states assumed measured is shown as the dotted line. One observers that the optimal control scheme leaves the constrained region one time instant before the output feedback scheme. This is expected from the discussion above. The 'discontinuity' in* c_k *at sample* 1 *is due to initialisation issues.*

B.4 Low-order controllers for optimal control in the presence of noise

In example B.3.2 we used Theorem B.1 to derive an output feedback law, and moreover this feedback law could be used for region detection. We will now focus on unconstrained optimization problems and show, by using Theorem B.2, how we can find optimal invariants between *noisy* measurements y. To achieve this we use the weights W_d and W_{n^y}, see (B.9). The approach we use is summarized in algorithm B.1. The algorithm can easily be extended to cover a non-stable system matrix A by not setting $u_k = 0$ for $k \geq N$.

Algorithm B.1 Finding low-order controllers. (For stable system matrix A)

1: Define cost function $\hat{J}(u,x) = \sum_{k=0}^{\infty} x_k' Q x_k + u_k' R u_k$.
2: Choose N_y, where $u_k = 0,\ k \geq N_y$.
3: Rewrite cost function to $J(u,x) = \sum_{k=0}^{N_y-1} \left(x_k' Q x_k + u_k' R u_k \right) + x_{N_y} P x_{N_y}$, where $P = A'PA + Q$.
4: Treat x_0 as a disturbance d and find J_{uu} and J_{ud}.
5: Decide candidate variables y, for example $y = (y_k, u_{k+1}, u_k, \ldots, u_{k+N-1})$ and form the "open-loop" model $y = G^y U + G_d^y d$, $U = (u_k, \ldots, u_{k+N-1})$.
6: Decide disturbance weight W_d and noise weight W_{n^y}.
7: Find $\tilde{H}$ by solving either the optimization problem (B.13) or use (B.12).
8: Decouple the inputs in $\tilde{H} = [\tilde{H}_x\ \tilde{H}_u]$ by setting $H = \tilde{H}_u^{-1}[\tilde{H}_x\ \tilde{H}_u] = [\tilde{H}_u^{-1}\tilde{H}_x\ I]$.

In step 2 one has to choose the input horizon N_y. In practical applications we found that this should be set rather high to give good performance in the low-order controllers. In the following example, where we focus on the central region (unconstrained) for example B.3.2, we had to increase N_y from 1 (as it was in example B.3.2) to 10 to get acceptable performance. To reduce complexity in constrained explicit MPC one can decrease the input horizon to get less number of regions, but the resulting controller will have a poorer performance, so obviously there is a trade-off. In future work we will investigate the possibility of using different input horizons for solving the parametric program and for deriving the controllers in each region.

The disturbance and noise weights W_d, W_{n^y} in step 6 should contain information about the expected variation in disturbances versus measurement noise.

In the following example we compare different low-order controllers found by using algorithm B.1 with optimal LQG controllers for the central region of example B.3.2 using output feedback.

Example B.4.1. *(Low-order controllers and comparison with LQG: output feedback) In this example we investigate the same process as before, but with noisy*

Table B.1: Simulated costs for example B.4.1. Noise levels for w_k, v_k: $(\alpha, \beta) = (0.8, 1)$.

Number	Control equation	J_1	J_2
0	$u_k = -[6.08\ 6.07]x_k$ (noise free, perfect measurement)	2.86	0.284
1	$u_k = -[6.08\ 6.07]\hat{x}_k$	**3.40**	0.400
2	$u_k = -(3.25y_k)$	5.27	0.569
3	$u_k = -(1.54y_k + 0.5y_{k-1})$	3.88	0.401
4	$u_k = -(0.78y_k + 0.44y_{k-1} - 0.03y_{k-2})$	3.88	**0.394**
5	$u_k = -(0.39y_k + 0.28y_{k-1} + 0.12y_{k-2} - 0.09y_{k-3})$	4.11	0.416

measurements, i.e.

$$x_{k+1} = \begin{bmatrix} 0.7326 & -0.0861 \\ 0.1722 & 0.9909 \end{bmatrix} x_k + \begin{bmatrix} 0.0609 \\ 0.0064 \end{bmatrix} u_k + w_k$$
$$y_k = \begin{bmatrix} 0 & 1.4142 \end{bmatrix} x_k + v_k, \tag{B.21}$$

where the process noise w_k are two uniformly distributed random numbers drawn from a uniform distribution on a $[-\beta, \beta]$ interval, and the measurement noise v_k is a uniformly distributed random number drawn from a uniform distribution on a $[-\alpha, \alpha]$ interval. There is no correlation between the noises. This implies variances $\mathrm{var}(w_k) = \frac{\beta^2}{3} I$, *and* $\mathrm{var}(v_k) = \frac{\alpha^2}{3}$.

The objective is to find low-order controllers that can give comparable performance with the well-known LQG controller.

In this example we investigate the following controllers for controlling the noisy process:

1. *LQG from y_k to u_k.*
2. *Invariant (u_k, y_k).*
3. *Invariant (u_k, y_k, y_{k-1}).*
4. *Invariant $(u_k, y_k, \ldots, y_{k-2})$.*
5. *Invariant $(u_k, y_k, \ldots, y_{k-3})$.*

Algorithm B.1 can directly be applied to find invariants between inputs u_k and output $y_k, y_{k-1}, \ldots$, also when there is noise on the measurements. Here we choose $N_y = 10$, which will give a good performance in the resulting controllers. Apart from this, the cost function is the same as in (B.18).

In Manum et al. [2007] analytical expressions for the derivatives J_{uu} and J_{ud} are given. These can be derived by substituting the state space model into the

objective function to get an unconstrained optimization problem as a function of (U, x_k)*, where again we treat* x_k *as a disturbance.*

The open-loop model follows from the model equations. For example, for $y = (y_k, y_{k+1,U})$*, where* $U = (u_0, \ldots, u_9)$*, we establish the model:*

$$y = \begin{bmatrix} y_k \\ y_{k+1} \\ U \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 \\ CB & 0 \\ I & 0 \\ 0 & I \end{bmatrix}}_{G_y} U + \underbrace{\begin{bmatrix} C \\ CA \\ 0 \end{bmatrix}}_{G_d^y} x_k. \tag{B.22}$$

The disturbance weight W_d *should reflect the variation in disturbances, whilst the noise weight* W_{n^y} *the noises on measurements and inputs. In [Manum et al., 2008a] it is shown that the resulting controllers are not affected by the noise on the inputs using the current formulation. We therefore choose:*

$$W_d = \frac{\beta^2}{3} I_{2\times 2} \qquad\qquad W_{n^y} = \begin{bmatrix} \frac{\alpha^2}{3} I_{\tilde{n}_y \times \tilde{n}_y} & 0 \\ 0 & I_{N_y \times N_y} \end{bmatrix}, \tag{B.23}$$

where $\tilde{n}_y \leq N_y$ *is the number of measurements we want to include in the implementation (i.e. the order of the resulting controller).*

This framework was used to generate the controllers shown in Table B.1. The LQR and LQG controllers were designed using standard software, and the tuning was based on the known distributions of the process and measurement noises.

The reference controller is an LQR using full state information (available in Matlab).

The LQG controller (from y_k *to* u_k*) is implemented as:*

$$\begin{aligned} \hat{x}_{n+1|n} &= A\hat{x}_{n|n-1} + Bu_n + L(y_n - C\hat{x}_{n|n-1}) \\ \hat{x}_{n|n} &= \hat{x}_{n|n-1} + M(y_n - C\hat{x}_{n|n-1}) \\ u_k &= -K\hat{x}_{n|n} \end{aligned} \tag{B.24}$$

with $L' = [0.04\ 0.59]$ *and* $M' = [0.12\ 0.57]$.

The simulated costs for the different controllers are shown in Table B.1. We investigate two cases, one where the process noise (i.e. disturbances) occurs at all time instants (J_1*) and one where the process noise occurs only every tenth instant (*J_2*). The simulated costs are the values of the objective function divided by the simulation length.*

When the process noise is occurring at all time instants (see J_1*), the LQG controller is optimal. The best variable combination between the present input and the outputs back in time, controller no. 4, has a simulated cost* 13% *higher*

than the LQG controller. However, if the process noise occurs only every tenth time instant (see J_2), a simple combination between y_k, y_{k-1}, y_{k-2} actually yields slightly better performance than the LQG controller.

As we increase the order of the controller we will reduce the noise sensitivity but we will be more sensitive to startup problems. The control law using $y_k, \ldots, y_{k-3}$ is only optimal 3 time-steps after the disturbance occurs, this is the reason why it has a higher simulated cost than controller number 4.

This example shows that our approach for deriving low-order controllers has some inherent problems regarding causality; to achieve optimal operation in the noise-free case we need at least $n_y = n_u + n_d$ measurements, and in the presence of noise we should include even more to reduce the sensitivity of noise. However, increasing the number of y's in the control law makes the causality problem more significant as we need to 'wait' until the rank conditions from the disturbance to the measurements becomes fulfilled.

The example further shows that the method works, and we get controllers comparable with the LQG controller. For disturbances occurring at every time instant the LQG controller will be optimal at all times. However, in most practical cases we do not expect that the disturbances will change in a random manner from one time step to the next, so the assumption of d_k changing for example only every tenth time step may not be too wrong. Further, if we are allowed to change the sample time we can always increase it to be faster than the dynamics of the disturbances and our method can be applied.

B.5 Discussion and extensions

In this paper we have discussed that feedback laws may be viewed as additional constraints (invariants) to the original optimization problem, and based on this, we have shown that optimal linear feedback laws can be derived for quadratic optimization problems.

Further, we have presented a mathematical framework, Theorem B.2 that gives optimal invariants of *noisy* measurements. This theorem can also be used in the case of too few measurements, which can be of interest even for the unconstrained LQ case.

Currently we are working on how to determine changes in the active set for noisy measurements and how to optimally include integral action in the low-order controllers.

Appendix C

Convex initialization of the $\mathcal{H}_2$-optimal static output feedback problem

Published in "Proceedings of American Control Conference 2009, St. Louis, USA."

Recently we have established a link between invariants for quadratic optimization problems and linear-quadratic (LQ) optimal control [Manum et al., 2008b]. The link is that for LQ control one invariant is $c_k = u_k - Kx_k$, which yields zero loss from optimality when controlled to a constant setpoint $c = c_s = 0$. In general there exists infinitely many such invariants to a quadratic programming (QP) problem. In [Manum et al., 2008c] we show how the link can be used to generate output feedback control by using current and old measurements. In this paper we extend this approach by considering in more detail some interesting examples, and the use of additional (old) measurements. In particular, we show that if the number of measurements is less than the number of disturbances (initial states) plus independent inputs, we can not with this method find a policy $u_k = -K^y y_k$ that minimizes the original problem, because K^y is not optimally constant. However, this method may be used to find initial values for $\mathcal{H}_2$-optimal static output feedback synthesis.

C.1 Introduction

Consider a finite horizon LQ problem of the form

$$\begin{aligned} \min_{u_0,u_1,\ldots,u_{N-1}} J(u,x(0)) &= E\{x_N' P x_N + \\ &\quad + \sum_{k=0}^{N-1} \left[x_k' Q x_k + u_k' R u_k\right]\} \\ \text{subject to } x_0 &= x(0) \\ x_{k+1} &= Ax_k + Bu_k, \quad k \geq 0 \\ y_k &= Cx_k + n_k^y, \end{aligned} \tag{C.1}$$

where $x_k \in \mathbb{R}^{n_x}$ are the states, $u_k \in \mathbb{R}^{n_u}$ are the inputs and $y_k \in \mathbb{R}^{n_y}$ are the measurements. Further $P = P' > 0$, $Q \geq 0$, and $R > 0$ are matrices of appropriate dimensions, and $E\{\cdot\}$ is the expectation operator.

It is well-known that if $C = I$ and $n^y = 0$, such that $y_k = x_k$, the solution to (C.1) is state feedback $u_k = -Kx_k$, where the gain matrix K can be found by solving an iterative Riccati equation. For the case with white noise assumption on x_0 and y (n^y), the optimal solution is $u_k = -K\hat{x}_k$, where $\hat{x}_k$ is a state estimate from a Kalman filter [Åström and Wittenmark, 1984], which in effect gives a dynamic compensator K^{lqg} (from y to u) of same order n_x as the plant.

In this paper we consider the static output feedback problem, $u_k = -K^y y_k$, where K^y is a static gain matrix. Note that the case with a fixed-order controller of order less than n_x may also be brought back to the static output feedback problem. A particular controller considered in this paper is the multi-input multi-output proportional-integral-derivative controller (MIMO-PID) where we have as many controlled outputs y^c as there are inputs u. The "allowed" measurements y_k in the formulation in (C.1) are the present value of the controlled output y_k^c (P), the integrated value $\sum_{i=0}^{k} y_i^c$ (I) and the derivative $\frac{\partial y_k^c}{\delta t}$ (D).

This optimal solution to this problem is unsolved [Syrmos et al., 1994] so one cannot expect to find an analytic or convex numerical solution. The contribution of this paper is therefore to propose a convex approach to find a good initial estimate for K^y, as a starting point for a numerical search.

C.1.1 Notation

Notation adopted from self-optimizing control is summarized in figure C.1. Typically, $u = (u_0, u_1, \ldots, u_{N-1})$, $d = x_0$ and $y = (x_0, u)$ or $y = (y_0, \frac{\partial y_0}{\partial t}, \ldots, u)$, but also other variables y will be considered.

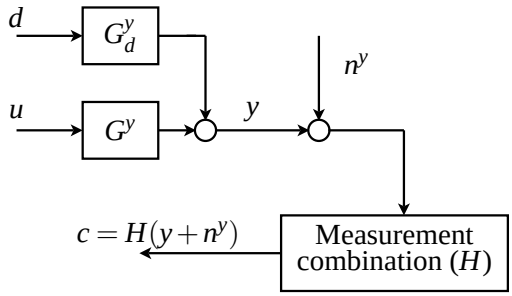

Figure C.1: Notation for self-optimizing control.

C.2 Main results

C.2.1 Results from self-optimizing control

Nullspace method

From [Alstad et al., 2009] we have the following theorem:

Theorem C.1. *(Nullspace theorem = Linear invariants for quadratic optimization problem) Consider an unconstrained quadratic optimization problem in the variables u (input vector of length n_u) and d (disturbance vector of length n_d)*

$$\min_u J(u,d) = \begin{bmatrix} u \\ d \end{bmatrix}' \begin{bmatrix} J_{uu} & J_{ud} \\ J_{ud}^T & J_{dd} \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix}. \tag{C.2}$$

In addition, there are "measurement" variables $y = G^y u + G_d^y d$. If there exists $n_y \geq n_u + n_d$ independent measurements (where "independent" means that the matrix $\tilde{G}^y = \begin{bmatrix} G^y & G_d^y \end{bmatrix}$ has full rank), then the optimal solution to (C.2) *has the property that there exists $n_c = n_u$ linear variable combinations (constraints) $c = Hy$ that are invariant to the disturbances d. The optimal measurement combination matrix H is found by selecting H such that*

$$HF = 0, \tag{C.3}$$

where $F = \frac{\partial y^{\text{opt}}}{\partial d}$ is the optimal sensitivity matrix which can be obtained from

$$F = -(G^y J_{uu}^{-1} J_{ud} - G_d^y), \tag{C.4}$$

(That is, H is in the left nullspace of F.)

Generalization: Exact local method

A generalization of Theorem C.1 is the following:

Theorem C.2. *(Exact local method = Loss by introducing linear constraint for noisy quadratic optimization problem [Alstad et al., 2009]) Consider the unconstrained optimization problem in Theorem C.1,* (C.2)*, and a set of noisy measurements $y_m = y + n^y$, where $y = G^y u + G_d^y d$. Assume that $n_c = n_u$ constraints $c = Hy_m = c_s$ are added to the problem, which will result in a non-optimal solution with a loss $L = J(u,d) - J_{\text{opt}}(d)$. Consider disturbances d and noise n^y with magnitudes*

$$d = W_d d'; \quad n^y = W_{n^y} n^y; \quad \left\| \begin{bmatrix} d' \\ n^{y'} \end{bmatrix} \right\|_2 \le 1.$$

Then for a given H, the worst-case loss introduced by adding the constraint $c = Hy$ is $L_{wc} = \bar{\sigma}^2(M)/2$, where M is

$$\begin{aligned} M &\triangleq \begin{bmatrix} M_d & M_n \end{bmatrix} \\ M_d &= -J_{uu}^{1/2} (HG^y)^{-1} HFW_d \\ M_n &= -J_{uu}^{1/2} (HG^y)^{-1} HW_{n^y}, \end{aligned} \tag{C.5}$$

and $\bar{\sigma}(\cdot)$ is the maximum singular value. The optimal H that minimizes the loss can be found by solving the convex *optimization problem*

$$\begin{aligned} &\min_H \|H\tilde{F}\|_F \\ &\text{subject to } HG^y = J_{uu}^{1/2} \end{aligned} \tag{C.6}$$

Here $\tilde{F} = [FW_d \;\; W_{n^y}]$.

The reason for using the Frobenius norm is that minimization of this norm also minimizes $\bar{\sigma}(M)$ [Kariwala et al., 2007].

Remark C.1. *From [Alstad et al., 2009] we have that any optimal H premultiplied by a non-singular matrix $n_c \times n_c$ D, i.e. $H_1 = DH$ is still optimal. One implication of this is that for a square plant, $n_c = n_u$, we can write $c = H_1 y = H_1^{y_m} y_m + Iu$. To see this, assume $y = (y_m, u)$, so $H = [H^{y_m} \; H^u]$, where H^u is a non-singular $n_u \times n_u$ matrix. Now, $H_1 = (H^u)^{-1}[H^{y_m} \; H^u] = [(H^u)^{-1} H^{y_m} \; I]$.*

Remark C.2. *More generally, for the case when $\tilde{F}\tilde{F}'$ is singular, we can solve the convex problem* (C.6) *using for example* CVX*, a package for specifying and solving convex programs [Grant and Boyd, 2008], with the following code:*

```
cvx_begin
   variable H(N*nu,ny+nu*N);
   minimize norm(H*Ftilde,'fro')
   subject to
      H*Gy == sqrtm(Juu);
cvx_end
```

Remark C.3. *Noise will not be further discussed in this paper, but is covered in [Manum and Skogestad, 2009].*

C.2.2 Some special cases

Some special cases will now be considered where explicit expressions can be found.

Full information

No new results are represented here, but we show the matrices G^y, G_d^y, J_{uu}, and J_{ud} for LQ-optimal control.

Assume that noise-free measurements of all the states are available. It is well known that the LQ problem (C.1) can be rewritten on the form in (C.2) (see for example [Rawlings and Muske, 1993]) by treating x_0 as the disturbance d, and letting $u = (u_0, u_1, \ldots, u_{N-1})$. Thus, from Theorem C.1 we know that for the LQ problem there exists *infinitely* many invariants (but only one of these involves only present states).

Without loss of generality consider the case when the model in (C.1) is stable.

Let $y = (x_0, u_0, u_1, \ldots, u_{N-1}) = (x_0, u)$. Note that this includes also future inputs, but we will use the normal "trick" in MPC of implementing only the present (first) input change u_0. Since we have $n_y = n_d + n_u$ and no noise, we can use Theorem 1. The open loop model becomes:

$$\begin{aligned} y &= G^y u + G_d^y d \\ G^y &= \begin{bmatrix} 0_{n_x \times (n_u N)} \\ I_{n_u N} \end{bmatrix} \in \mathbb{R}^{(n_x + n_u N) \times (n_u N)} \\ G_d^y &= \begin{bmatrix} I_{n_u N} \\ 0_{(n_u N) \times n_x} \end{bmatrix} \in \mathbb{R}^{(n_x + n_u N) \times n_x} \end{aligned} \tag{C.7}$$

Here I_m is an $m \times m$ identity matrix and $0_{m \times n}$ is a $m \times n$ matrix of zeros.

The matrices J_{uu} and J_{ud} are the derivatives of the linear quadratic objective function.For the objective and process model in (C.1) we show in [Manum et al.,

2007] that

$$\frac{J_{uu}}{2} = \begin{bmatrix} B'PB+R & B'A'PB & \dots & B'(A^{N-1})'PB \\ B'PAB & B'PB+R & \dots & B'(A^{N-2})'PB \\ \vdots & \vdots & \ddots & \vdots \\ B'PA^{N-1}B & B'PA^{N-2}B & \dots & B'PB+R \end{bmatrix} \tag{C.8}$$

and

$$\frac{J_{ud}}{2} = \begin{bmatrix} B' & & & \\ & B' & & \\ & & \ddots & \\ & & & B' \end{bmatrix} \begin{bmatrix} P \\ PA \\ \vdots \\ PA^{N-1} \end{bmatrix} A \tag{C.9}$$

The sensitivity matrix (optimal change in y when d is perturbed) becomes:

$$F = \frac{\partial y^{\text{opt}}}{\partial d'} = -(G^y J_{uu}^{-1} J_{ud} - G_d^y) = \begin{bmatrix} I_{n_x} \\ -J_{uu}^{-1} J_{ud} \end{bmatrix} \tag{C.10}$$

We can use Theorem C.1 to get the combination matrix H, i.e. find an H such that $HF = 0$:

$$\begin{bmatrix} H_1 & H_2 \end{bmatrix} \begin{bmatrix} I_{n_x} \\ J_{uu}^{-1} J_{ud} \end{bmatrix} = H_1 - H_2 (J_{uu}^{-1} J_{ud}) = 0 \tag{C.11}$$

To ensure a non-trivial solution we can choose $H_2 = I_{n_u N}$ and get the following optimal combination of x_0 and u:

$$c = Hy = J_{uu}^{-1} J_{ud} x_0 + u, \tag{C.12}$$

which can be interpreted as:

$$\begin{aligned} &\text{Invariant 1: } u_0 = K_0 x_0 \\ &\text{Invariant 2: } u_1 = K_1 x_0 \\ &\qquad\vdots \\ &\text{Invariant } N\text{: } u_{N-1} = K_{N-1} x_0 \end{aligned} \tag{C.13}$$

From Theorem C.1 implementation of (C.13) give zero loss from optimality, i.e. they correspond to the optimal input trajectory $u_0^*, u_1^*, \dots, u_{N-1}^*$ from the solution of (C.1). Moreover, since the states capture all information, we must have that

$$u_1 = K_1 x_0 = \underbrace{K_1 (A + BK_0)^{-1}}_{=K_0} x_1. \tag{C.14}$$

From this we deduce that the solution to (C.1) can be implemented as $u_k = K_0 x_k$, $k = 0, 1, \dots$.

In [Manum et al., 2007] we prove that this gives the same result as conventional linear quadratic control, by conventional meaning for example equation (3) in Rawlings and Muske [1993].

Output feedback

In this section we will show how Theorem C.2 can be used for the special (but common) case when $y_k = Cx_k + 0 \cdot u_k,\ k = 0,1,\ldots,N$ and we look for controllers on the form $u_k = -K^y y_k$. If C is full column rank, then we have full information (state feedback), but we here consider the general case where C has full row rank (independent measurements), but not full column rank.

Let $y = (y_0, u)$ and as before $u = (u_0, u_1, \ldots, u_{N-1})$. The disturbance $d = x_0$. The open loop model is now

$$y = \begin{bmatrix} y_0 \\ u \end{bmatrix} = \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{G^y} u + \underbrace{\begin{bmatrix} C \\ 0 \end{bmatrix}}_{G_d^y} d, \tag{C.15}$$

and the sensitivity matrix F is

$$F = -(G^y J_{uu}^{-1} J_{ud} - G_d^y) = \begin{bmatrix} C \\ -J_{uu}^{-1} J_{ud} \end{bmatrix} \tag{C.16}$$

Since we now have that $n_y = n_{\tilde{y}} + n_u < n_u + n_d$, where $n_{\tilde{y}}$ are the number of measurements from the plant, $n_{\tilde{y}} < n_d$, we cannot simply set $HF = 0$, but we need to solve (C.6).

Let us analyze this problem. For $G^{y\prime} = [0\ I]$, $HG^y = J_{uu}^{1/2}$ is equivalent to $H_2 = J_{uu}^{1/2}$, where

$$H_{n_c \times (n_{\tilde{y}} + n_u)} = \begin{bmatrix} H_{1 n_c \times n_{\tilde{y}}} & H_{2 n_c \times n_u} \end{bmatrix} \tag{C.17}$$

With this partitioning we get that

$$H\tilde{F} = H[FW_d\ W_{n^y}] = [HFW_d\ HW_{n^y}], \tag{C.18}$$

and for $W_{n^y} = 0$, i.e. the noise-free case,

$$H\tilde{F} = [HFW_d\ 0]. \tag{C.19}$$

We want to minimize the Frobenius-norm of this matrix and we have that

$$\|[HFW_d\ 0]\|_F = \|HFW_d\|_F + \|0\|_F \tag{C.20}$$

Assume without loss of generality that $W_d = I$, and let $\tilde{J} = -J_{uu}^{-1} J_{ud}$. With $F' = [C'\ J']$ we have that

$$HF = H_1 C + H_2 \tilde{J}\big|_{H_2 = J_{uu}^{1/2}} = H_1 C - J_{uu}^{-1/2} J_{ud} \tag{C.21}$$

We want to minimize $\|H_1C - J_{uu}^{-1/2}J_{ud}\|$, hence we look for a H_1 such that

$$H_1C = J_{uu}^{-1/2}J_{ud}. \tag{C.22}$$

Using the pseudo-inverse, we find that

$$H_1 = J_{uu}^{-1/2}J_{ud}C^\dagger, \tag{C.23}$$

and we get that the optimal H is

$$H = [J_{uu}^{-1/2}J_{ud}C^\dagger \; J_{uu}^{1/2}]. \tag{C.24}$$

In the final implementation we can decouple the invariants in the inputs by

$$\tilde{H} = J_{uu}^{-1/2}H = [-J_{uu}^{-1}J_{ud}C^\dagger \; I]. \tag{C.25}$$

This means that the open-loop optimal "output feedback" is

$$u_k = -\underbrace{J_{uu}^{-1}J_{ud}}_{K^{\text{state feedback}}} C^\dagger y_k = -K^y y_k, \tag{C.26}$$

that is, for an optimal state feedback K, the optimal "output feedback" is $KC^\dagger$.

This means that for this case we have

$$\begin{aligned} &\text{"Invariant" 1: } u_0 = K_0C^\dagger y_0 \\ &\text{"Invariant" 2: } u_1 = K_1C^\dagger y_0 \\ &\qquad\vdots \\ &\text{"Invariant" } N\text{: } u_{N-1} = K_{N-1}C^\dagger y_0 \end{aligned} \tag{C.27}$$

We have called these variable combinations "invariants" in quotation marks because they are not invariant to the solution of the original problem, but rather the variable combinations that minimize the (open-loop) loss. Indeed, the non-negative loss is

$$\begin{aligned} \|HF\| &= \|J_{uu}^{-1/2}J_{ud}C^\dagger C - J_{uu}^{-1/2}J_{ud}\| \\ &= \|J_{uu}^{-1/2}J_{ud}(C^\dagger C - I)\| \\ &\leq \|J_{uu}^{-1/2}J_{ud}\|\|C^\dagger C - I\| \end{aligned} \tag{C.28}$$

For output feedback we have in the least squares sense

$$u_1 = \underbrace{K_1C^\dagger C(A + BK_0C^\dagger C)^{-1}C^\dagger}_{K_1} y_1. \tag{C.29}$$

Unfortunately, in general $K_1 \neq K_0$ and hence the open loop solution (C.27) cannot be implemented as a constant feedback $u_k = K_0C^\dagger y_k$, as was the case for state feedback, see (C.14).

C.3 Main algorithm

We now propose an algorithm for finding output feedback controllers. This is a two-step procedure where we first find initial values using Theorem C.2. These initial values correspond to a controller that in the open-loop sense is closest to the optimal state feedback LQ controller. Thereafter we improve this controller by solving a closed-loop optimization problem where the controller parameters are the degrees of freedom.

In the previous section we showed that if $y = (Cx_0, u_0, \ldots, u_{N-1})$ Theorem C.1 gives $u_0 = -K^y y_0 = K^{\text{state feedback}} C^\dagger y_0$. The algorithm presented here is more general in the sense that it handles "measurements" such as $y = (y_0, y_1, \ldots, y_M, u_0, \ldots, u_{N-1})$. (In the latter case a casual controller is $u_M = -K^y [y_0' \ \ldots \ y_M']'$.)

Algorithm C.1 Low-order controller synthesis

1: Define a finite-horizon quadratic objective $J(u,x) = x_N' P x_N + \sum_{i=0}^{N-1} x_i' Q x_i u_i' R u_i + 2 x_i' N u_i$.
2: Calculate J_{uu} and J_{ud} as in (C.8), (C.9).
3: Define candidate variables $y = G^y u + G_d^y x_0$, $u = (u_0, u_1, \ldots, u_{N-1})$.
4: Decide weights W_d and W_n^y (Default: $W_d = I$, $W_n^y = 0$)
5: Find $\tilde{H}$ by solving the convex optimization problem (C.6)
6: Optional: Improve control by closed-loop optimization. (Section C.3.1.)

C.3.1 Relationship between LQ-control and $\mathcal{H}_2$ optimal control

It is well-known that the LQG problem may be cast into the $\mathcal{H}_2$ framework and that a class of $\mathcal{H}_2$ optimal controllers may be implemented in an LQG-scheme with a Kalman estimator and a constant feedback gain from the estimated states [Doyle et al., 1989]. In this paper we propose to improve the solution from the open-loop control by minimizing the $\mathcal{H}_2$ norm

$$\min_K \|F_l(P,K)\|_2. \tag{C.30}$$

In this context $\min_K$ means minimizing over the *parameters* in K. The lower-fractional transform $F_l(P,K) = P_{11} + P_{12}(I - P_{22}K)^{-1}P_{21}$ for a $P = \left[\begin{smallmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{smallmatrix}\right]$ [Skogestad and Postlethwaite, 2005]. The interconnection structure we use for P is shown in figure C.2.

The last row of Algorithm C.1 consists of solving (C.30) with initial values as $\tilde{H}$ on step 5 in algorithm C.1, which is the solution to (C.6).

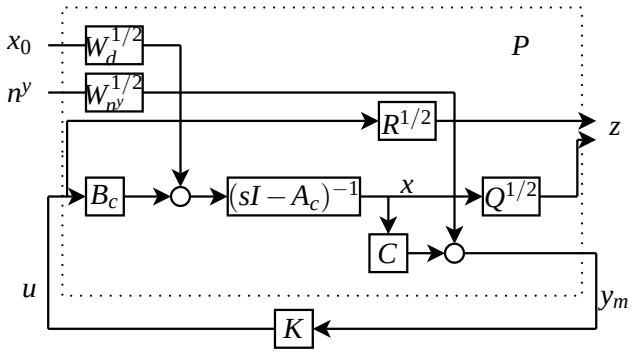

Figure C.2: Interconnection structure for closed-loop optimization of K.

C.4 Examples

In this section two examples will be considered. First we discuss P-control of a second order plant, then MIMO-PID control of a model of a distillation column.

Example C.4.1. *(P-control of second order plant) Consider the plant* $g(s) = \frac{2}{s^2+3s+2}$. *The plant is sampled with* $T_s = 0.1$ *to get*

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 0.7326 & -0.1722 \\ 0.0861 & 0.9909 \end{bmatrix} x_k + \begin{bmatrix} 0.1722 \\ 0.0091 \end{bmatrix} u_k \\ y_k &= \begin{bmatrix} 0 & 1 \end{bmatrix} x_k \end{aligned} \tag{C.31}$$

The objective is to derive two LQ-optimal controllers for this process, one P-controller on the form $u_k = -K^y y_k$ *and a PD-controller* $u_k = -(K_1^y y_k + K_2^y y_{k-1})$.

In the synthesis of the controllers we use algorithm C.1. The open loop objective to be minimized is $\tilde{J}(u,x) = \sum_{i=0}^{\infty} x_i' Q x_i + u_i' R u_i$ *with* $Q = \left[\begin{smallmatrix} 0 & 0 \\ 0 & 1 \end{smallmatrix}\right]$ *and* $R = 1$. *The infinite horizon objective can be approximated by the following objective:*

$$J(u,x) = x_N' P x_N + \sum_{i=0}^{N-1} x_i' Q x_i + u_i' R u_i, \tag{C.32}$$

with $P = \left[\begin{smallmatrix} 0.8333 & 2.4917 \\ 2.4917 & 9.6667 \end{smallmatrix}\right]$ *and* $N = 10$. *(*P *is a solution to the discrete Lyapunov equation* $P = A'PA + Q$.*) The objective is now on the form of step 1 in the algorithm.*

P-control: For the P-controller, the variables to combine are $y^1 = (y_0, u_0, \ldots, u_{N-1})$. *The matrices* J_{uu} *and* J_{ud} *are the same as those reported in equations* (C.8) *and* (C.9). *Since we do not consider noise* $W_d = I$ *and* $W_{n^y} = 0$. *We can now find* $\tilde{H}$ *either by solving the convex problem in* (C.6)*, or we can simply use the explicit formula in* (C.23)*, i.e.* $H_1 = J_{uu}^{-1} J_{ud} C^{\dagger}$. *As shown in section C.2.2 (see equation*

Controller	$\|H\tilde{F}\|_F$	$\|F_l(P,K)\|_2$	
1: $u_k = -0.404y_k$	0.390	0.2993	First invariant using Theorem C.2 for $y^1 = (y_0, u_0, \ldots, u_{N-1})$
2: $u_k = -0.313y_k$	–	0.2981	Closed-loop optimal P-controller
3: $u_k = -(1.49y_k - 1.11y_{k-1})$	0	0.3176	Second invariant using Theorem C.2 for $y^2 = (y_0, y_1, u_0, \ldots, u_{N-1})$
4: $u_k = -(0.416y_k - 0.109y_{k-1})$	–	0.2979	Closed-loop optimal controller PD-controller
5: $u_k = -[0.131\ 0.396]x_k$	0	0.2972	LQR

Table C.1: Controllers for example C.4.1.

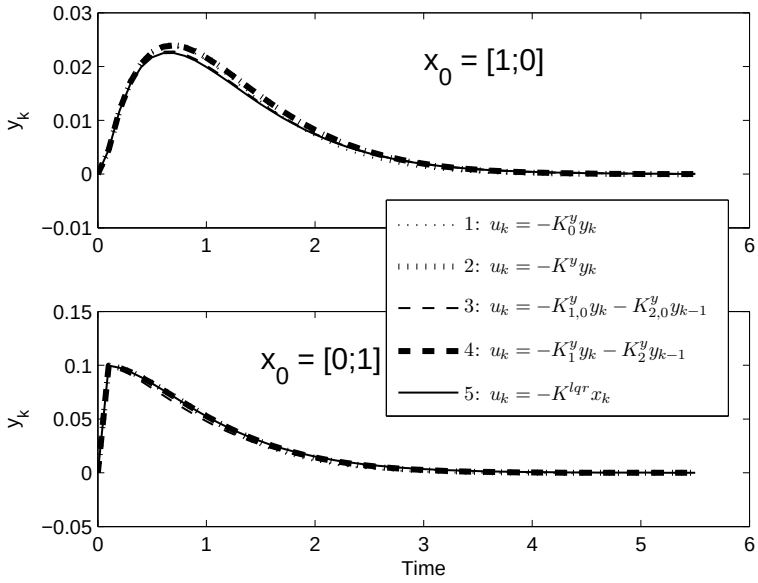

Figure C.3: Simulation results for disturbances in initial conditions, example C.4.1.

C.27) we now get $N = 10$ "invariants" to the solution to the original optimization problem in (C.32)*. The first one of these invariants is reported as controller 1 in table C.1. We observe that $\|H\tilde{F}\| > 0$, which is expected from Theorem 1, as $n_y < n_u + n_d$ in this case ($n_y = 1+10, n_u = 10, n_d = 2$).*

Using this P-controller ($u_k = -0.404 y_k$) as the initial estimate, the $\mathcal{H}_2$-optimal closed-loop controller K in Figure C.2 is obtained numerically. Note from row 2 in Table I that the $\mathcal{H}_2$-norm is only reduced slightly (from 0.2993 to 0.2981), although K changes from -0.404 to -0.313.

PD-control: For the synthesis of a PD controller we again use algorithm C.1. The variables to combine are now $y^2 = (y_0, y_1, u_0, \ldots, u_{N-1})$. The objective function and the matrices J_{uu} and J_{ud} remain the same. The open-loop model $y = G^y u + G^y_d x_0$ is now:

$$y^2 = \begin{bmatrix} y_0 \\ y_1 \\ u \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ CB & 0 \\ I & 0 \\ 0 & I \end{bmatrix} u + \begin{bmatrix} C \\ CA \\ 0 \end{bmatrix} x_0. \tag{C.33}$$

For this particular variable combination (C.23) *cannot be used, as the variables occur on different instances in time. We therefore solve the optimization problem* (C.6) *using* `cvx`*, as shown in remark C.2. The solution is again on the form of* (C.27)*, for which the second invariant is reported as controller 3 in table C.1. The solution (all the invariants) gives $\|H\tilde{F}\| = 0$, which is expected from Theorem C.1, as $n_y = n_u + n_d$ and no noise is present. Further numerical optimization reduces the $\mathcal{H}_2$ norm from 0.3176 to 0.2979.*

It can be verified that the variable combination is indeed optimal after one step with the following calculations:

$$\begin{aligned} u_0 = -Kx_0, \quad u_1 = -K_1^y y_1 - K_2^y y_0 \\ \Rightarrow u_1 = -\underbrace{(K_1^y C + K_2^y C(A-BK)^{-1})}_{=K} x_1, \end{aligned} \tag{C.34}$$

where K is the LQR controller. For implementation some sub-optimality must be expected since we are not starting the control with LQR, rather we use the PD controller at all time instances.

Simulations: From the closed-loop norms reported in table C.1 the controllers are expected to perform similarly in closed loop. This is confirmed in the closed loop simulations of disturbances in initial states, see figure C.3.

Example C.4.2. *(Linear dynamic model of distillation column.) In this example we consider MIMO-PI and -PID control of "column A" in [Skogestad, 1997]. The model is used as an example for offset-free control in [Muske and Badgwell, 2002]. The model is based on the following assumptions:*

- *binary separation,*
- *41 stages, including reboiler and total condenser,*
- *each stage is at equilibrium, with constant relative volatility* $\alpha = 1.5$*,*
- *linearized liquid flow dynamics,*
- *negligible vapor holdup,*
- *constant pressure.*

The feed enters on stage 21. $u = \left[\begin{smallmatrix} L \\ V \end{smallmatrix}\right]$ *and* $y = \left[\begin{smallmatrix} x_D \\ x_B \end{smallmatrix}\right]$.

We here consider the LV-configuration, where D and B are used to control the levels. With level controllers implemented (P-control with $K_c = 10$*) the rest of the column is stable.*

Description	Control equation	$\|H\tilde{F}\|_F$	$\|F_l(P,K)\|$
"First-move" PI	$u_k = -\left(\begin{bmatrix} 5.316 & 0.25664 \\ -3.1953 & -3.3371 \end{bmatrix} y_k^P + \begin{bmatrix} 2.6897 & -0.5975 \\ -0.13498 & -2.5939 \end{bmatrix} y_k^I\right)$	4.28	3.99
Closed-loop optimal PI	$u_k = -\left(\begin{bmatrix} 16.0156 & -5.17125 \\ 0.541199 & -9.57775 \end{bmatrix} y_k^P + \begin{bmatrix} 2.7148 & -0.715 \\ 0.33949 & -2.7672 \end{bmatrix} y_k^I\right)$	–	3.65
"First-move" PID	$u_k = -\left(\begin{bmatrix} 9.9305 & -0.96741 \\ -5.2025 & -3.5369 \end{bmatrix} y_k^P + \begin{bmatrix} 2.6891 & -0.62581 \\ -0.13969 & -2.6454 \end{bmatrix} y_k^I \dots + \begin{bmatrix} 1.0724 & -0.22799 \\ -0.53974 & -0.43514 \end{bmatrix} y_k^D\right)$	3.44	3.78
Closed-loop optimal PID	$u_k = -\left(\begin{bmatrix} 17.5043 & -8.22394 \\ 3.48592 & -17.7333 \end{bmatrix} y_k^P + \begin{bmatrix} 2.743 & -1.3167 \\ 0.15148 & -4.3547 \end{bmatrix} y_k^I \dots + \begin{bmatrix} -1.78427 & -6.07614 \\ -9.79446 & -13.3285 \end{bmatrix} y_k^D\right)$	–	3.63
LQR	$u_k = -\begin{bmatrix} -0.0022 & 0.0002 & -0.0004 & -0.0007 & 0.0016 & -0.0097 \\ 0.0008 & 0.0015 & -0.0016 & -0.0037 & 0.0079 & -0.0074 \end{bmatrix} x_k(1:6) \dots - \begin{bmatrix} -0.0036 & 0.0048 & 0.0116 & -0.0011 & -0.0213 & 0.0305 \\ -0.0066 & 0.0262 & 0.0610 & 0.0044 & 0.0093 & -0.0148 \end{bmatrix} x_k(7:12) \dots - \begin{bmatrix} 0.0149 & 0.0521 & 0.1349 & 0.1034 & 2.6897 & -0.5975 \\ 0.0233 & -0.0372 & -0.1607 & 0.0895 & -0.1350 & -2.5939 \end{bmatrix} x_k(13:18)$	–	3.61

Table C.2: Controllers for example C.4.2.

	Algorithm C.1	$K^0 = 0$	SIMC-tuned PI controllers
PI	44	–	71
PID	91	–	123

Table C.3: Iteration count using `fminunc` (Matlab© R2008a) with different initializations

Balanced reduction is used to reduce the number of states to 16. Then integrated outputs are added to the model, resulting in a model with 18 states. If we let the outputs of the model be P, I, and D, we get a model with the following structure:

$$\begin{bmatrix} \dot{x} \\ \dot{\sigma} \end{bmatrix} = \begin{bmatrix} a & 0 \\ c & 0 \end{bmatrix} \begin{bmatrix} x \\ \sigma \end{bmatrix} + \begin{bmatrix} b \\ d \end{bmatrix} u$$
$$\begin{bmatrix} y^P \\ y^I \\ y^D \end{bmatrix} = \begin{bmatrix} c & 0 \\ 0 & I \\ ca & 0 \end{bmatrix} \begin{bmatrix} x \\ \sigma \end{bmatrix} + \begin{bmatrix} d \\ 0 \\ cb \end{bmatrix} u \quad . \tag{C.35}$$

This model is sampled with $T_s = 1$ to get a discrete time model. Again we set up an infinite time objective function, with $Q = C' \begin{bmatrix} 0 & & \\ & I & \\ & & 0 \end{bmatrix} C$, and $R = 0.1 \cdot I$, and for intermediate calculations we approximate this by a finite horizon objective with $N = 150$ and $P = Q$.

We now look for controllers on the form

$$u_k = -\left(K^P y_k^P + K^I y_k^I + K_D y_k^D\right) \tag{C.36}$$

and we assume measurements of the compositions with a sample time of 1 minute is available.

Table C.4.2 shows "first-move" PI (= first move invariant realized as feedback), closed loop PI and PID controllers, and the LQR controller for reference. In addition to the initialziation proposed in this paper we tried to initialize the numerical search with $K^0 = 0$ and two SIMC-tuned [Skogestad, 2003a] PI controllers with $\tau_c = 10$ minutes, leading to $K^0_{SIMC} = \left[\begin{smallmatrix} 14.63 & 0 & 0.37 & 0 & 0 & 0 \\ 0 & -10.91 & 0 & -0.27 & 0 & 0 \end{smallmatrix}\right]$. As reported in table C.4.2 did $K^0 = 0$ not converge, whereas intializing with two SIMC-tuned PI controllers converged in both cases (both for PI and PID design), though with some more iterations than the method proposed in this paper.

Figure C.4 shows simulation results where we at $t = 0$ introduce a step in the feed rate and at $t = 70$ a step in the feed composition. 'PI' and 'PID' refers to the closed-loop optimal controllers. As one observes is the MIMO-PID controller quite close in performance to the LQR controller.

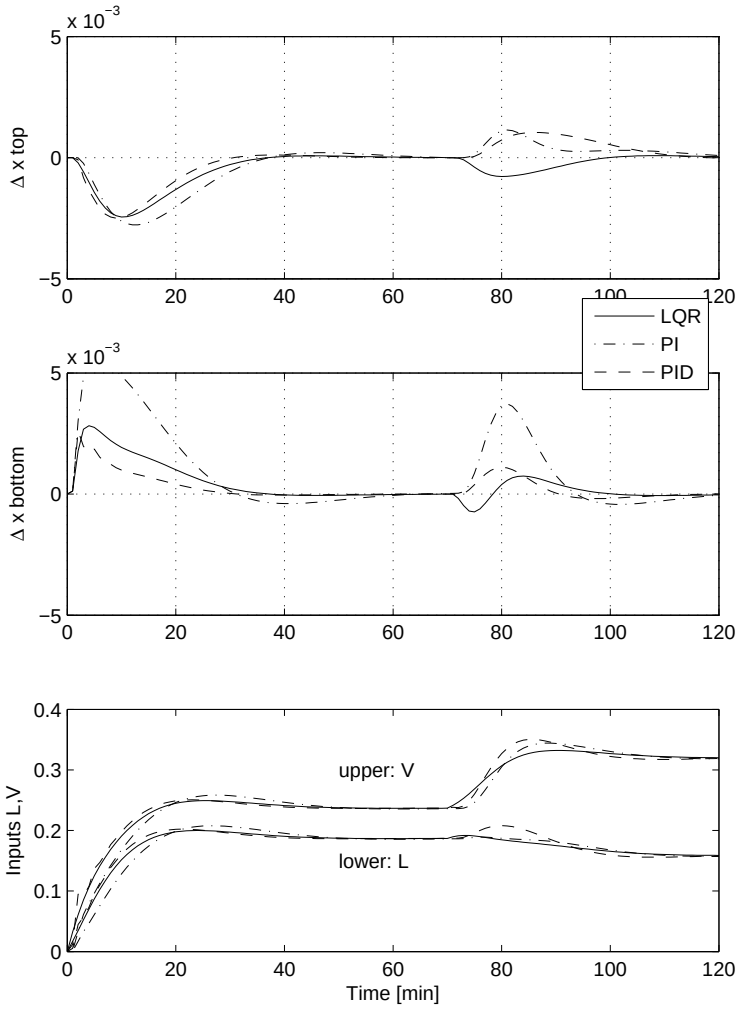

Figure C.4: Simulation results for example C.4.2. At $t = 0$ a step-change of 0.1 in F occurs, and at $t = 70$ z_F is changed from 0.5 to 0.6.

C.5 Conclusions

In this paper we have discussed synthesis of $\mathcal{H}_2$-optimal static output feedback, and in particular the MIMO-PID. We have shown that initial conditions for closed loop optimization can be found by solving a convex program, and that the resulting closed loop optimization problem converges for some interesting cases.

C.6 Acknowledgments

The authors gratefully acknowledge the comments from Bjarne Grimstad.

Bibliography

MATLAB version r2009b. Computer program.

A. Alessio and A. Bemporad. A survey on explicit model predictive control. In *Proc. Int. Workshop on Assessment and Future Directions of Nonlinear Model Predictive Control*, Pavia, Italy, 2008.

V. Alstad and S. Skogestad. Null space method for selecting optimal measurement combinations as controlled variables. *Ind. Eng. Chem. Res*, 46:846–853, 2007.

V. Alstad, S. Skogestad, and E.S. Hori. Optimal measurement combinations as controlled variables. *Journal of Process Control*, 19(1):138 – 148, 2009.

Karl J. Åström and Bjørn Wittenmark. *Computer Controlled Systems*. Prentice-Hall, 1984.

F. Aurenhammer. Voronoi diagrams - A Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys*, 23(3), September 1991.

M. Baotiċ, F. Borrelli, A. Bemporad, and M. Morari. Efficient On-Line Computation of Constrained Optimal Control. *SIAM Journal on Control and Optimization*, 47(5):2470–2489, 2008.

G.I. Bara and M. Boutayeb. Static output feedback stabilization with $\mathcal{H}_\infty$ performance for linear discrete-time systems. *IEEE Transactions on Automatic Control*, 50(2):250–254, February 2005.

J.F. Bard. *Practical Bilevel Optimization*. Kluwer Academic Publishers, Dodrecht/Boston/London, 1998.

J.F. Bard and J.T. Moore. A branch and bound algorithm for the bilevel programming problem. *SIAM Journal of Scientific Computing*, 11(2):281–292, March 1990.

R. Bellman. The Theory of Dynamic Programming. *Bull. Amer. Math. Soc.*, 60: 503–515, November 1954.

A. Bemporad. Reducing conservativeness in predictive control of constrained systems with disturbances. In *Proceedings of the IEEE Conference on Decision and Control*, Tampa, Florida USA, December 1998.

A. Bemporad and M. Morari. Robust Model Predictive Control: A Survey. *Robustness in Identification and Control*, 245:207–226, 1999.

A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38:3–20, 2002. See also corrigendum 39(2003), pages 1845-1846.

A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit solution of model predictive control via multiparametric quadratic programming. In *Proceedings of the American Control Conferene*, June 2006.

V. Blondel and J.N. Tsitsikilis. NP-Hardness of some linear control design problems. *Siam Journal of Control and Optimization*, 35(6):2118–2127, November 1997.

T. Bouarar, K. Guelton, and N. Manamanni. Static Output Feedback Controller Design for Takagi-Sugeno Systems – A Fuzzy Lyapunov LMI Approach. In *Proceedings of IEEE Conference on Decision and Control*, pages 4150–4155, Shanghai, China, 2009.

A.E. Bryson and Y.C. Ho. *Applied Optimal Control*. Hemisphere Publishing, 1975.

R. Cagienard, P. Grieder, E.C. Kerrigan, and M. Morari. Move blocking strategies in receding horizon control. *Journal of Process Control*, 17:563–570, 2007.

J.D. Camm, A.S. Raturi, and S. Tsubakitani. Cutting Big M down to Size. *Interfaces*, 20(5):61–66, Sep.-Oct. 1990.

D. Chmielewski and V. Manousiouthakis. On constrained infinite-time linear quadratic optimal control. *Systems and Control Letters*, 29:121–129, 1996.

F. J. Christophersen, M. N. Zeilinger, C. N. Jones, and M. Morari. Controller Complexity Reduction for Piecewise Affine Systems Through Safe Region Elimination. In *Proc. of the Conf. on Decision & Control*, New Orleans, USA, December 2007.

B. Colson, P. Marcotte, and G. Savard. Bilevel programming: A survey. *4OR: A Quarterly Journal of Operations Research*, 3(2):87–107, June 2005.

C.R. Cutler and B.L. Ramaker. Dynamic Matrix Control – A Computer Control Algorithm. In *Proceedings of American Control Conference*, San Francisco, USA, 1980.

D.-W. Ding and G.-H. Yang. $\mathcal{H}_\infty$ Static Output Feedback Control for Discrete-time Switched Linear Systems with Average Dwell Time. In *Proceedings of American Control Conference*, pages 2356–2361, St. Louis, MO, USA, 2009a.

D.-W. Ding and G.-H. Yang. Static Output Feedback Control for Discrete-time Switched Linear Systems under Arbitrary Switching. In *Proceedings of American Control Conference*, pages 2385–2390, St. Louis, MO, USA, 2009b.

J.C. Doyle. Guaranteed Margins for LQG Regulators. *IEEE Transactions on Automatic Control*, AC-23(4):756–757, August 1978.

J.C. Doyle, K. Glover, P.P. Khargonekar, and B.A. Francis. State-space solutions to standard $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems. *IEEE Transactions on Automatic Control*, 34(8):831–847, 1989.

X. Du and G.-H. Yang. LMI Characterizations of Positive Realness and Static Output Feedback Positive Real Control of Discrete-time Systems. In *Proceedings of American Control Conference*, pages 5132–5137, St. Louis, MO, USA, 2009.

W. Findeisen, F.N. Bailey, M. Bryds, K. Malinowski, P. Tatjewski, and A. Wozniak. *Control and Coordination in Hierarchical Systems*. John Wiley & Sons, 1980.

A. Fujimori. Optimization of Static Output Feedback Using Substitutive LMI Formulation. *IEEE Transactions on Automatic Control*, 49(6):995–999, June 2004.

J. Gadewadikar, F.L. Lewis, K. Subbarao, K. Peng, and B.M. Chen. H-Infinity Static Output-feedback Control for Rotorcraft. *Journal of Intelligent Robot Systems*, 54:629–646, 2009.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming (web page and software), August 2008. URL `http://standford.edu/~boyd/cvx`.

P. Grieder, F. Borrelli, F. Torrisi, and M. Morari. Computation of the constrained infinite time linear quadratic regulator. *Automatica*, 40:701–708, 2004.

A. Haidar, E.K. Boukas, S. Xu, and J. Lam. Exponential Stability and Static Output Feedback Stabilization of Singular Time-Delay Systems with Saturating Actuators. In *Proceedings of American Control Conference*, pages 4945–4950, St. Louis, MO, USA, 2009.

U. Halldorsson, M. Fikar, and H. Unbehauen. Nonlinear predictive control with multirate optimisation. In *IEE Proc., Control Theory Applications*, volume 152, pages 273–284, 2005.

I. J. Halvorsen, S. Skogestad, J. C. Morud, and V. Alstad. Optimal selection of controlled variables. *Ind. Eng. Chem. Res.*, 42:3273–3284, 2003.

Tore Haug-Warberg. Personal Communication, 2010.

E.S. Hori and S. Skogestad. Selection of controlled variables: Maximum gain rule and combination of measurements. *Ind. Eng. Chem. Res.*, 47:9465–9471, 2008.

M. Hovd and S. Skogestad. Control of Symmetrically Interconnected Plants. *Automatica*, 30(6):957–973, 1994.

S. Hovland and J.T. Gravdahl. Complexity Reduction in Explicit MPC through Model Reduction. In *Proceedings of the 17th World Congress*, pages 7711–7716, Seoul, Korea, July 2008. The International Federation of Automatic Control.

S. Hovland, K. Willcox, and J.T. Gravdahl. MPC for Large-Scale Systems via Model Reduction and Multiparametric Convex Programming. In *Proceedings of the IEEE Conference on Decision and Control*, San Diego, USA, Dec 2006.

C.N. Jones and M. Morari. The Double Description Method for the Approximation of Explicit MPC Control Laws. In *Conference on Decision and Control, CDC*, Cancun, Mexico, December 2008.

C.N. Jones and M. Morari. Approximate Explicit MPC using Bilevel Optimization. In *European Control Conference*, Budapest, Hungary, August 2009. URL `http://control.ee.ethz.ch/index.cgi?page=publications;action=details;id=3339`.

R.E. Kalman. Contributions to the Theory of Optimal Control. *Boletin de la Sociedad Matematica Mexicana*, 5:102–119, 1960.

V. Kariwala, Y. Cao, and S. Janardhanan. Local self-optimizing control with average loss minimization. *Ind. Eng. Chem. Res*, 46:3629–3634, 2007.

V. Kariwala, Y. Cao, and S. Janardhanan. Local self-optimizing control with average loss minimization. *Industrial and Engineering Chemistry Research*, 47: 1150–1158, 2008.

M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004. URL `http://control.ee.ethz.ch/ mpt/`.

H. Kwakernaak and R. Sivan. *Linear optimal control systems*. Wiley, 1972.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL `http://control.ee.ethz.ch/ joloef/yalmip.php`.

D. G. Luenberger. Observers for multivariable systemts. *IEEE Transactions on Automatic Control*, AC-II(2):190–197, April 1966.

P. Lundström and S. Skogestad. Non-uniqueness of robust $\mathcal{H}_\infty$ decentralized PI-control. In *Proceedings of the American Control Conference*, Boston, USA, 1991.

D.H.S. Maithripala, J.M. Berg, and W.P. Dayawansa. Control of an Electrostatic Microelectromechanical System Using Static and Dynamic Output Feedback. *Journal of Dynamic Systems, Measurement and Control*, 127:443–450, September 2005.

H. Manum and S. Skogestad. Convex initialization of the $\mathcal{H}_2$-optimal static feedback problem with noise. Submitted to Conference on Decision and Control, 2009.

H. Manum, S. Narasimhan, and S. Skogestad. A new approach to explicit MPC using self-optimizing control. Avaliable at: http://www.nt.ntnu.no/users/skoge/publications/2007/, 2007.

H. Manum, S. Narasimhan, and S. Skogestad. A new approach to explicit MPC based on self-optimizing control. Manuscript in preparation, 2008a.

H. Manum, S. Narasimhan, and S. Skogestad. A new approach to explicit MPC using self-optimizing control. In *Proceedings of American Control Conference*, Seattle, USA, 2008b.

H. Manum, S. Narasimhan, and S. Skogestad. Explicit MPC with output feedback using self-optimizing control. In *Proceedings of IFAC World Conference*, Seoul, Korea, 2008c.

T. Marlin and A.N. Hrymak. Real-time operations optimization of continuous processes. In *Proceedings of Chemical Process Control-V*, Tahoe City, Nevada, USA, 1996.

I. Maruta, T.-H. Kim, and T. Sugie. Fixed structure $\mathcal{H}_\infty$ controller synthesis: A meta-heuristic approach using simple constrained particle swarm optimization. *Automatica*, 45:553–559, 2009.

D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

D.Q. Mayne, M.M. Seron, and S.V. Raković. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41:219–224, 2005.

P.P. Menon and C. Edwards. Decentralized static output feedback stabilization and synchronization of networks. *Automatica*, 45:2910–2916, 2009.

B. Moore. Principal Component Analysis in Linear Systems: Controllability, Observability and Model Reduction. *IEEE Transactions on Automatic Control*, 26 (1):17–32, 1981.

M. Morari and J. H. Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23:667–682, 1999.

H. Mukadidani and H. Xu. Guaranteed Cost Control for Uncertain Stochastic Systems with Multiple Decision Makers via Static Output Feedback. In *Proceedings of IEEE Conference on Decision and Control*, pages 2917–2922, Shanghai, China, 2009.

K. R. Muske and T. A Badgwell. Disturbance modeling for offset-free linear model predictive control. *Journal of Process Control*, 12:617–632, 2002.

K. R. Muske and J. B. Rawlings. Model predictive control with linear models. *AIChE Journal*, 39:262–287, February 1993.

T. Nagashio and T. Kida. An Optimal Design of Symmetric $\mathcal{H}_\infty$ Static Output Feedback Controller using LMI for Collocated Second-Order Linear System. In *Proceedings of IEEE Conference on Decision and Control*, pages 6177–6182, Shanghai, China, 2009.

S. Narasimhan and S. Skogestad. Implementation of optimal operation using offline calculations. In *Dycops*, 2007.

S. Narasimhan and S. Skogestad. Control structure adaptation for linear constrained systems. Preprint submitted to Elsevier Science, 2010.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research, 1999.

P. Pakshin and D. Peaucelle. LQR parameterization of static output feedback gains for linear systems with Markovian switching and related robust stabilization and passification problems. In *Proceedings of IEEE Conference on Decision and Control*, pages 1157–1162, Shanghai, China, 2009.

G. Pannocchia and J. B. Rawlings. Disturbance models for offset-free model-predictive control. *AIChE Journal*, 49(2):426–437, February 2003.

G. Pannocchia, N. Laachi, and J.B. Rawlings. A Candidate to Replace PID Control: SISO-Constrained LQ Control. *AIChE Journal*, 51(4):1178–1189, April 2005.

G. Pannocchia, J. B. Rawlings, and S. J. Wright. Fast, large-scale model predictive control by partial enumeration. *Automatica*, 43:852–860, May 2007.

E. N. Pistikopoulos, V. Dua, N. A. Bozinis, A. Bemporad, and M. Morari. On-line optimization via off-line parametric optimization tools. *Computers and Chemical Engineering*, 26:175–185, 2002.

S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practise*, 11:733–764, 2003.

J. Qiu, G. Feng, H. Gao, and Y. Fan. Exponential $\mathcal{H}_\infty$ Static Output Feedback Control of Switched Systems with Average Dwell-Time and Time-Varying Uncertainties. In *Proceedings of IEEE Conference on Decision and Control*, pages 6383–6388, Shanghai, China, 2009.

S.V. Rakovič, E.C. Kerrigan, K.I. Kouramas, and D.Q. Mayne. Invariant approximations of the minimal robust positively invariant set. *IEEE Transactions on Automatic Control*, 50(3):406–410, March 2005.

C.V. Rao, J. Wright, and J.B. Rawlings. Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications*, 99 (3):723–757, December 1998.

J. B. Rawlings and K. R. Muske. The stability of constrained receding-horizon control. In *IEEE Transactions on Automatic Control*, volume 38, pages 1512–1516, 1993.

J.B. Rawlings. Tutorial overview of model predictive control. *IEEE Control Systems Magazine*, 20(3):38–52, June 2000.

J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison, Wisconsin, first printing edition, 2009.

J. Richalet, A. Rault, J. L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14:413–428, 1978.

J.A. Rossiter, M.J Rice, and B. Kouvaritakis. A robust state-space approach to stable predictive control strategies. In *Proceedings of the American Control Conference*, pages 1640–1641, Albuquerque, New Mexico, USA, 1997.

P. O. M. Scokaert and J. B. Rawlings. Constrained linear quadratic regulation. *IEEE Transactions on automatic control*, 43(8):1163–1169, 1998.

Z. Shu, J. Lam, and J. Xiong. Static output-feedback stabilization of discrete-time Markovian jump linear systems: A system augmentation approach. *Automatica*, 46:687–694, 2010.

S. Skogestad. Dynamics and control of distillation columns - a tutorial introduction. *Trans IChemE, Part A*, 75:539–562, September 1997.

S. Skogestad. Plantwide control: the search for the self-optimizing control structure. *Journal of Process Control*, 10:487–507, 2000a.

S. Skogestad. Self-optimizing control: the missing link between steady-state optimization and control. *Computers & Chemical Engineering*, 24:569–575, 2000b.

S. Skogestad. Simple rules for model reduction and pid controller tuning. *Journal of Process Control*, 2003a.

S. Skogestad. *Prosessteknikk*. Tapir Akademisk Forlag, 2nd edition, 2003b.

S. Skogestad. Near-optimal operation by self-optimizing control: From process control to marathon running and business systems. *Computers and Chemical Engineering*, 27:127–137, 2004.

S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control*. Wiley, 2005.

Jørgen Spjøtvold, Eric. C Kerrigan, Colin N. Jones, Petter Tøndel, and Tor A. Johansen. One the facet-to-facet property of solutions to convex parametric quadratic programs. Preprint submitted to Automatica, May 2006.

V.L. Syrmos, C. Abdallah, and P. Dorato. Static output feedback: a survey. *Decision and Control, 1994., Proceedings of the 33rd IEEE Conference on*, 1: 837–842 vol.1, Dec 1994. doi: 10.1109/CDC.1994.410963.

V.L. Syrmos, C.T. Abdallah, P. Dorato, and K. Grigoriadis. Static Output Feedback – A Survey. *Automatica*, 33(2):125–137, 1997.

S.H van der Meulen, R.L. Tousain, and O.H. Bosgra. Fixed Structure Feedforward Controller Design Exploiting Iterative Trials: Application to a Wafer Stage and a Desktop Printer. *Journal of Dynamic Systems, Measurement, and Control*, 130(5):051006, September 2008.

Y. Wang and S. Boyd. Fast Model Predictive Control Using Online Optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, March 2010.

C. Wen, X. Ma, and B.E. Ydstie. Analytical expression of explicit mpc solution via lattice piecewise-affine function. *Automatica*, (45):910–917, 2009.

H.-N. Wu. An LMI approach to robust $\mathcal{H}_2$ static output feedback fuzzy control for uncertain discrete-time nonlinear systems. *Automatica*, 44:2333–2339, 2008.

R. Yelchuru. Personal communication, 2010.

C.M. Ying and B. Joseph. Performance and Stability Analysis of LP-MPC and QP-MPC Cascade Control Systems. *AIChE Journal*, 45(7):1521–1534, 1999.