NTNU
Norwegian University of
Science and Technology

Lagrangian Decomposition for Dynamic Real-Time Optimization: Applied to Production Optimization Network

Ingvild Sørlie

Chemical Engineering and Biotechnology
Submission date: July 2017
Supervisor: Sigurd Skogestad, IKP
Co-supervisor: Dinesh Krishnamoorthy, IKP

Norwegian University of Science and Technology
Department of Chemical Engineering

Preface

This thesis was written as the final work of the master program in Chemical Engineering at the Norwegian University of Science and Technology NTNU.

First, I would like to express my deepest gratitude towards my supervisor professor Sigurd Skogestad for his guidance and inquisitive attitude to my work. I would also like to thank my co-supervisor PhD candidate Dinesh Krishnamoorthy for his sharing of knowledge and help in troubled times and on weekends.

My fellow master's students at the Process Systems Engineering group also deserves acknowledgment. The open and sharing environment with you has helped me get through this thesis. And all of my other friends in Trondheim for supporting me and making me years at NTNU what it was.

Thanks to my parents for their proof reading, even when on vacation in the Seychelles. And also for their encouragement throughout my six years as a master students.

Last, but not least, I send my love to popcorn-kollektivet, for all the laughter and cozy Sundays.

Declaration of Compliance

I hereby declare that this thesis is an independent work in agreement with the exam rules and regulations of the Norwegian University of Science and Technology.

Ingvild Marie Sørlie
July 12, 2017, Trondheim

Abstract

The scope of this thesis was to investigate the use of lagrangian decomposition for a dynamic real-time optimization(RTO) problem. The decomposition was performed on a gas lifted two-well network, controlled by a nonlinear model predictive controller(NMPC). Even though decomposition is a well known and utilized method, it has not been investigated much for the upstream production of oil and gas. The dynamic lagrangian dual decomposition in this thesis is an extension of previous work with steady state decomposition for gas lifted well networks.

The lagrangian dual decomposed NMPC was modeled and simulated in Matlab using CasADi v3.1.0, the NLP problem was solved with the IPOPT solver. The base case refers to the lagrangian decomposition simulations, which were performed for two different constraints becoming active. The simulation results were compared to a centralized NMPC and a steady state optimization for the same system. A successful decomposed NMPC should yield the same converged simulation values as the centralized NMPC and steady state optimization. Numerical methods were applied to improve the base case decomposed NMPC. The line search method, adjusting the step length α for each update of the lagrangian multiplier λ, was included to reach a better convergence of λ and lower the number of iterations needed. The augmented lagrangian, penalizing constraint violations, was included to improve the start-up dynamics of the base case decomposed NMPC.

The base case decomposition simulation results show that the decomposed NMPC converges to the same results as the centralized NMPC and a steady state optimization performed for the same system. However, in certain cases, the dynamics in the initial part of the simulation could be problematic. For the case where the gas rate capacity is the active constraint, the decomposed simulation oscillates around the trajectory of the centralized NMPC, even violating the constraint. When the available gas lift is the active constraint, the start-up dynamics is not a problem. The line search method successfully brings down the number of iterations, but at the cost of a larger constraint violation than in the base case decomposition. The augmented lagrangian decomposition decreases the constraint violations and converges faster than the base case decomposition. Combined with the fact that λ converges and that the simulation needs few iterations, it makes this the best method for dynamic decomposition of this system.

Since the system used in this problem is very simple, the results are only of academic interest. No conclusion can be made on the realistic uses of dynamic decomposition for control within the oil and gas industry. However, the results show a possibility for this type of control system to be suitable for use in more complex systems.

Sammendrag

Hensikten med denne masteravhandlingen har vært å undersøke bruken av "lagrangian" dekomponering for en dynamisk sanntidsoptimalisering Dekomponeringen ble utført på et nettverk av to brønner med gasløft, regulert av en ikke-linear modellprediktiv kontroller(NMPC). Selv om dekomponering er en mye brukt metode, har den ikke vært undersøkt så mye for bruk i oppstrøms olje og gass produksjon. Den dynamiske dekomponeringen i denne avhandlingen er en forlengelse av et arbeid gjort med stasjonærtilstand dekomponering for et nettverk av brønner med gasløft.

Den "lagrangian" dekomponerte NMPCen ble modellert og simulert i Matlab med bruk av CasADi v3.1.0, NLP-problemet ble løst ved hjelp av IPOPT-solveren. "Base case" refererer til simuleringene med "lagrangian" dekomponering, som ble simulert med to forskjellige aktive beskrankninger. Resultatene fra simuleringene ble sammenlignett med simuleringer gjort med en sentralisert NMPC og en statisk optimalisering av det samme systemet. En vellykket dynamisk dekomponering burde gi de samme konvergerte simuleringsverdiene som den sentraliserte NMPCen og statisk optimalisering. Numeriske metoder ble anvendt for å forbedre "base case" dekomponeringen. Linjesøkmetoden, som tilpasser steglengden α for hver oppdatering av "lagrangian" multiplikator λ, ble brukt for å oppnå en bedre konvergert verdi av λ og minke det nødvendige antallet iterasjoner. "Augmented lagrangian" dekomponering, som straffer brudd på beskrankningene, ble inkludert for å forbedre dynamikken i oppstartsfasen for "base case" dekomponerte NMPC problemet.

Simuleringsresultatene av "base case" dekomponeringen viser at den dekomponerte NMPCen konvergerer til det samme resultatet som den sentraliserte NMPCen og statisk optimalisering. Men for noen systemer kan dynamikken i oppstartsfasen av simuleringen være problematisk. Når total gasskapasitet er den aktive beskrankningen, så oscillerer den dekomponerte NMPCen rundt den sentraliserte NMPCen og bryter beskrankningen. Dynamikken er imidlertid ikke et problem når tilgjengelig gasløft er den aktive beskrankningen. Linjesøkmetoden lykkes i å redusere antallet iterasjoner, men på bekostning av et større brudd på beskrankningene. Augmented lagrangian dekomponering reduserer bruddet på beskrankningene og konvergerer til optimum raskere enn "base case". Ettersom denne metoden også har få iterasjoner i simuleringen er "augmented lagrangian" den beste metoden for dynamisk dekomponering av dette systemet.

Ettersom systemet brukt i dette problemet er veldig forenklet, så er resultatet av akademisk interesse. Ingen konklusjoner kan trekkes om den praktiske innføringen av dynamisk dekomponerte reguleringssystemer i olje og gass-sektoren. Likevel viser resultatetene at slike typer reguleringssystemer er mulige og fordelaktige for komplekse systemer.

Contents

List of Figures

List of Tables

List of Code Snippets

Abbreviations

Aug.x	Augmented Lagrangian Runs (x=[1-6])
BC.x	Base Case Runs (x=[1-2])
DAE	Differential Algebraic Equations
DOF	Degrees of Freedom
LD	Lagrangian Dual
IPOPT	Interior Point Optimization
LLBP	Lagrangian Lower Bound Program
LS.x	Line Search Runs (x=[1-9])
MILP	Mized Integer Linear Programming
MPC	Model Predictive Control
MV	Manipulated Variable
NLP	Nonlinear Programming
NMPC	Nonlinear Model Predictive Control
OCP	Optimal Control Problem
ODE	Ordinary Differential Equations
RTO	Real-time Optimization
RTPO	Real-time Production Optimization
SFI	Norwegian Center for Research Based Innovation
ss-opt	Steady State Optimization
SUBPRO	Subsea Production and Processing

List of Symbols

Roman letters:

Symbol	*Description*	*Units*
A_a	Cross-section area of annulus	$[m^2]$
A_{bh}	Cross-section area of well at bottom hole	$[m^2]$
A_w	Cross-section area of well	$[m^2]$
C_{11}	Control volume of annulus in well 1	[-]
C_{12}	Control volume of tubing in well 1	[-]
C_{21}	Control volume annulus in well 2	[-]
C_{22}	Control volume of tubing in well 2	[-]
C_{iv}	Flow coefficient for downhole injection valve	$[m^2]$
C_{pc}	Flow coefficient for production valve	$[m^2]$
c	The constraint function in Augmented Lagrangian Method	[-]
c	Some chosen constant in line search method	[-]
D_a	Diamater of annulus	[m]
D_{bh}	Diamater of well at bottom hole	[m]
D_w	Diamater of well	[m]
F	Dynamics constraint function in NLP	[-]
f	Set of differential equations in DAE	[-]
f	Objective function in decomposition	[-]
f_1	Objective function for subproblem 1	[-]
f_2	Objective function for subproblem 2	[-]
G	Algebraic constraint function in NLP	[-]
GOR	Gas-Oil ratio	$[kg/kg]$
$\Delta_{max}GL$	Maximum change in gas lift over one oscillation	[-]
g	Acceleration of gravity constant	$[m/s^2]$
g	Set of algebraic equations in DAE	[-]
g(w)	Constraint matrix in the NLP problem	[-]
g(λ)	Decomposed objective functions	[-]
g_i	Set of algebraic equations for well i	[-]

Roman letters:

Symbol	*Description*	*Units*
H_a	Height of annulus	[m]
H_{bh}	Height of well at bottom hole	[m]
H_w	Height of well	[m]
h	The constraint function in decomposition	[-]
h_1	Decomposed constraint in subproblem 1	[-]
h_2	Decomposed constraint in subproblem 2	[-]
J	Cost function	[-]
L_a	Length of annulus	[m]
L_{bh}	Length of well at bottom hole	[m]
L_w	Length of well	[m]
$\mathscr{L}$	Lagrangian	[-]
M_w	Molecular weight of gas	$[g/mol]$
m_{ga}	Mass of gas in annulus	[kg]
m_{gt}	Mass of gas in tubing	[kg]
m_{ot}	Mass of oil in tubing	[kg]
n_w	Number of wells	[-]
PI	Reservoir productivity index	$[kg/s \cdot bar]$
p_a	Annulus pressure	[bar]
p_{bh}	Bottom hole pressure	[bar]
p_k	Direction in line search method	[-]
p_m	Manifold pressure	[bar]
p_r	Reservoir pressure	[bar]
p_{wh}	Wellhead pressure	[bar]
p_{wi}	Well injection point pressure	[bar]
Q	Penalizing function	[-]
$Q_{g_{max}}$	Maximum gas production rate	$[kg/s]$
$Q_{GL_{max}}$	Maximum available gas lift	$[kg/s]$
R	Gas constant	$[J/molK]$
T_a	Temperature in annulus	$[K]$
T_w	Temperature in well tubing	$[K]$
t	Time	[min]
$t_{K,i}$	Points on the collocation time interval	[-]

Roman letters:

Symbol	*Description*	*Units*
u	Control variables	[-]
$\underline{u}$	Lower limit on the control variables	[-]
$\overline{u}$	Upper limit on the control variables	[-]
u^*	Optimal control input	[-]
V_a	Volum of gas in annulus	$[m^3]$
w	set of decision variables for NLP	[-]
w_{gl}	Gas lift rate and control input	$[kg/s]$
w_{iv}	Well injection flow rate	$[kg/s]$
w_{pc}	Wellhead total produciton rate	$[kg/s]$
w_{pg}	Wellhead gas production rate	$[kg/s]$
w_{po}	Wellhead oil production rate	$[kg/s]$
w_{ro}	Oil rate from reservoir	$[kg/s]$
w_{rg}	Gas rate from reservoir	$[kg/s]$
w_{tg}	Total gas production rate	$[kg/s]$
w_{to}	Total oil production rate	$[kg/s]$
x	Differential state variable set	[-]
x_0	Intital value of the state variable	[-]
$\underline{x}$	Lower limit on the differential states	[-]
$\overline{x}$	Upper limit on the differential states	[-]
$\dot{x}$	Derivative of differential states	[-]
z	Algebraic states variable set	[-]
$\underline{z}$	Lower limit on the algebraic states	[-]
$\overline{z}$	Upper limit on the algebraic states	[-]

Greek letter:

Symbol	*Description*	*Units*
α	step length in updating of Lagrange multiplier	[-]
β	Updating parameter in Augmented Lagrangian method	[-]
λ	Lagrange multiplier	[-]
$\Delta_{max}\lambda$	Maximum change in λ over one oscillation	[-]

Greek letter:

Symbol	*Description*	*Units*
$\Delta_{min}\lambda$	Minimum change in λ over one oscillation	[-]
μ	Penalty parameter in Augmented Lagrangian method	[-]
Φ	The objective function in an OCP	[-]
ρ	Contracting factor in line search method	[-]
ρ_a	Density of gas in annulus	$[kg/m^3]$
ρ_o	Density of oil	$[kg/m^3]$
ρ_w	Density of fluid mixture in well tubing	$[kg/m^3]$
θ	Interpolation parameters	[-]

Chapter 1

Introduction

This work is the author's final thesis and represents the conclusion of a M.Sc in Chemical Engineering with the Norwegian University of Science and Technology NTNU. This work is a part of a Norwegian Center for Research Based Innovation (SFI) named Subsea Production and Processing (SUBPRO) which was started up in the third quarter of 2015. This Masters' thesis is organized as follows: The model is presented and assumptions are made in chapter 2, the basic theory of optimization and control is presented in chapter 3 and the method to decomposition used in this work is discussed in chapter 4. The simulations performed are explained in chapter 5 and the results are presented and discussed in chapter 6 before the work is concluded in chapter 7.

Subsea technology is increasingly used, replacing the tradition of building platforms. This opens more possibilities for smaller and difficultly located fields to be economically profitable. Technology development makes it more profitable and safer to extract oil and gas. Prevention of maintenance on platforms were previously an after thought but with new subsea solutions the planning and design phase must entail methods to avoid unexpected maintenance and operation failure. Analysis of real-time data and a good control strategy of the system can help avoid costly unplanned down-time[1]. As failure and rehabilitation becomes increasingly expensive with subsea operations the focus on predicting system behavior becomes more crucial and the cost and time consuming development of such tools become more profitable for subsea fields[2]. The subsea fields are also high in complexity where removing the human factor in favor of full automation of production control can be very profitable. Another challenge can be that fields have different characteristics, hence each case might have several technological challenges and need tailored solutions[3][1]. These tailored solutions need to be included in the decision making process which can be quite complex.

The complex decision making process of the offshore production of oil and gas can be divided in a hierarchal way depending on the time-horizon of the decision making. This hierarchy is shown in figure fig. 1.1. Long-term decision making involves asset management, infrastructure and investment strategy, while the medium decision making refers to reservoir management, involving drilling and production scheduling and strategy. "Daily Production Optimization" decisions are made with intervals of a few hours to a couple of days. Below that, on the lowest level of the control hierarchy, is the control and automation layer that continuously accounts and correct for changes in the operating conditions[4][5]. The work done in this thesis is located on this level.

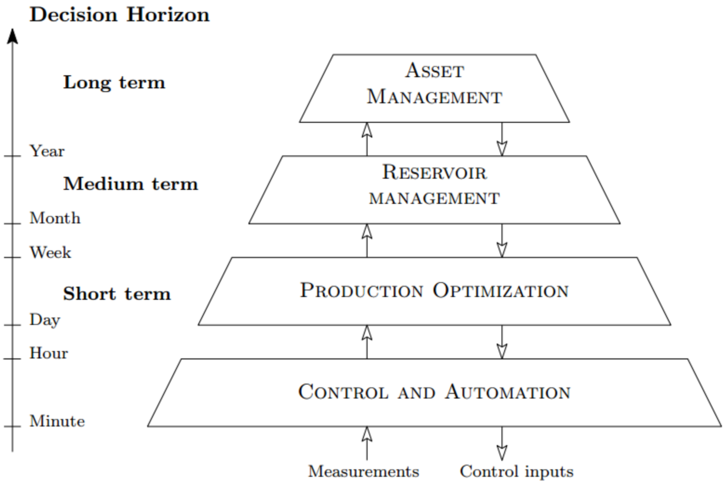


Figure 1.1: Typical decision making hierarchy for the control process of offshore oil and gas production. Figure taken from [5] p.2 which was adapted from [4].

A controller in the lowest level could be an on-line real time optimization(RTO) controller, that continuously manipulates the control input to adapt to changes in the system. A highly successful RTO is the Model Predictive Controller(MPC), that uses a prediction of the controlled system for a certain time horizon and optimizes that prediction. The optimized solution for the next time step determines the set point for the controller and the measurements from the plant are given as input for the next prediction optimization. It is a widely used tool for control in the process industry and further research is very relevant[6].

There are two ways of separating a control system[7]. The vertical hierarchical decomposition refers to a decoupling in time, normally with a sequential control design. The hierarchical system above in fig. 1.1 is a vertical decomposition of plant control. The other way of partitioning a control system is horizontal hierarchical decomposition. This is a decoupling in space, where the plant is divided into decentralized subproblems. The decomposition in space is the main topic in this thesis.

Gas lift is a complicated manipulated variable, the correct amount of lift gas increases the production of oil, too much gas lift can decrease the gas and oil rate[8]. With limited amount of lift gas available the correct allocation of the gas between wells can be complicated. Previously the two methods used to optimize gas lift allocation were the equal-slope method, where the gas lift performance curves were controlled to be equal, and the Quasi-Newton method[9]. Both methods verify optimality by using derivative information and therefore problems tended to get stuck in local optima[8].

Usually the optimization problems within oil and gas are large and complex with hundreds of decision variables. This is also the case with the Troll field, explained in [10]. When two or more wells act together this affects the optimum allocation of gas lift, making it complex to control. In a well network the gas lift optimization is a nonlinear and constrained problem that is in need of good nonlinear optimization control structures. Finding the optimal control structure for gas lifted oil well networks is what has been attempted in this thesis by using decomposition methods for a dynamic system. In this work only two wells are included in the network. If the dynamic decomposition can be performed for such a simple and small network, then the method might be feasible for larger complicated networks where the benefits from decomposition are larger. This is therefore an interesting academic problem, that maybe in the future can be evolved to practical application.

1.1 Motivation for Performing a Decomposition on the Dynamic Control Problem

The idea behind decomposition is to separate a large and complex optimization problem, like gas lifted oil well networks, into smaller subproblems, solving them separately either in parallel or sequentially[11]. There are multiple methods of decomposition described in literature and different applications of the method. The method investigated in this work is the Lagrangian Dual decomposition, which is described in detail in chapter 4.

Lagrangian relaxation of optimization problems and other decomposition method has been used in a wide range of applications for a long time[12]. In process optimization decomposition can be used in any level of the hierarchy. The approach has also been successfully performed for downstream industries [13]. A more aggressive decomposition method that builds on the lagrangian dual method is the augmented lagrangian decomposition that penalizes constraint violations. This method has previously also been applied to the downstream section of oil and gas production, for example on production planning and scheduling[14] or supply chain planning for multiple companies[15].

Even though decomposition is a well known and utilized method, it has not been investigated much for the upstream production of oil and gas, or the optimization of allocating gas lift. Work on a steady state control structure with decomposition of a large oil field network has previously been performed, where the optimization gives the control set points[16][17]. The studies showed that there is a limit to how complex and large the control systems can become, before a global control structure has difficulties finding an optimal solution. The theoretical work on steady state decomposition was modeled for the Troll field, and the decomposed control structures outperformed the global controllers when the composed subproblems were many enough. The work on steady state decomposition is further discussed in section 4.1. The low level of usage of this method for oil and gas production might be because of the efforts it demands to implement. However, with more complex fields and larger networks of wells being utilized, the method might have an economical gain. Challenges to operating and maintain 115 subsea wells on the troll field[10] motivates research into finding new and better methods to control production from subsea oil fields. In addition, there are drawbacks to using a single MPC on a large, complex system[18]. Splitting the MPC into smaller subproblems might make the control structure more transparent and easier to understand and maintain.

What this thesis wants to examine is a dynamic decomposition, optimizing over a time-horizon and implementing the decomposed solutions into a NMPC for the whole two well system. The dynamic lagrangian dual decomposition is performed as an extension of previous work with steady state decomposition. Numerical methods were added to improve the dynamic decomposition. The line search method adapts the step length for each iteration and the augmented lagrangian decomposition adds a penalty for violating the constraints. The methods are explained in more detail in chapter 4. In this work, a simplified model with two gas lifted wells is used to demonstrate the concept of dynamic decomposition methods for a production.

Chapter 2

Problem Formulation

The gas lifted oil well network is set up with a model predictive controller to study the different improvement possibilities. In this chapter, the assumptions, equations and Differential Algebraic Equations(DAE) system for the model will be presented. The model of the network is based on a common gas lifted well model used in the paper *Stabilization of Gas Lifted Wells Based on State Estimation* by Eikrem et al[19]. The equations are written for a network with n_w number of wells. In this project two wells make up the network in a very simple model of a gas lifted well network. The complete system that is used for this project is illustrated below in fig. 2.1.

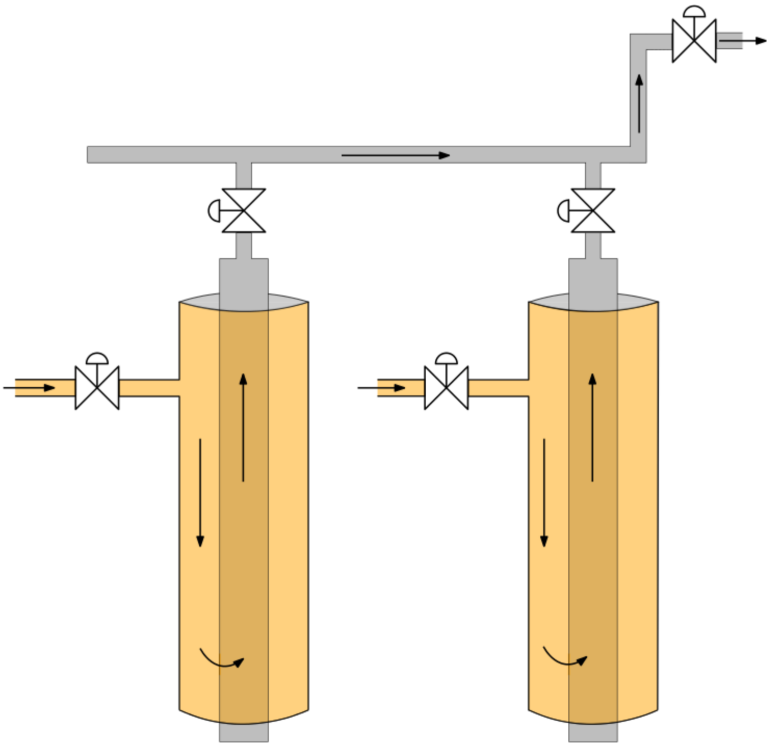


Figure 2.1: Gas lifted well network consisting of two identical gas lifted wells connected to a common manifold. The manifold leads to a separator not included in the model.

2.1 Description of System and Assumptions

The set up of the well system is illustrated below in figure fig. 2.1. The gas lifted well system consists of two wells connected through a manifold that mixes the fluid produced from both wells, before the fluid travels up through the riser. The riser is connected to a separator that is not included in the model, but the separator pressure is given as a parameter to the model. Both wells are identical, one well is illustrated below in fig. 2.2. The wells consist of two inlets, one from a reservoir and one gas lift injection. The gas lift enters the annulus surrounding the well tubing through the gas lift injection valve, with a flow controller to adjust the inlet flow. The gas lift mixes with the gas-oil mixed fluid from the reservoir through the injection valve at the bottom of the annulus, which is also referred to as the downhole inlet. The production choke valve at the top of each well controls the outlet flow from each well. The riser head valve at the top of the system has a gas capacity constraint that is used in the control as the active constraint. The valve positions at the gas lift inlet is the control input set to maximize total flow production of the gas lifted oil well system.

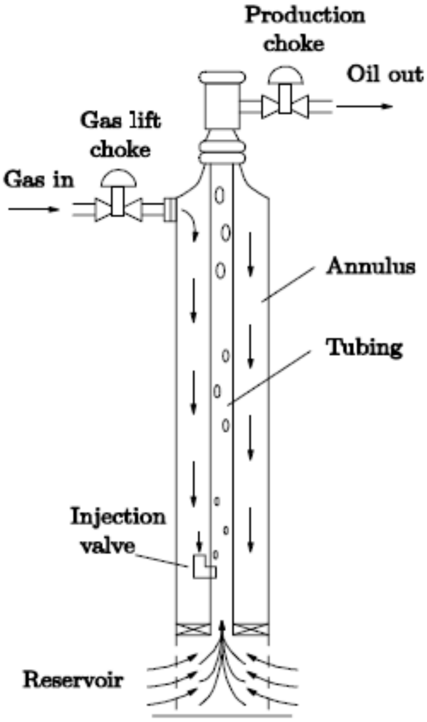


Figure 2.2: The gas lifted oil well that is used for this network. Figure taken from [19].

2.1.1 Assumptions

For the modeling of the gas lifted oil well network, multiple assumptions were made. The assumptions are described below.

Reservoir properties

The reservoir properties are assumed to vary from well to well, and only the reservoir inflow model is included in this model. The wells can be producing from different sections of the reservoir, or different reservoirs all together. The reservoir model is not crucial for the network model.

Gas lift injection

The gas lift injection is adjusted with a flow controller on the gas lift injection valve. For this model, the gas lift injection is assumed to be perfectly controlled to give the desired flow rate.

Manifold dynamics

The n_w number of wells all produce to a common manifold. Compared to the length of the wells, the manifold's length is negligible and therefor also the manifold dynamics is assumed negligible. The pressure in the manifold is assumed to be the same for all wells and at any given time.

Manifold pressure

It can be assumed that the riser dynamics are negligible, so the manifold is connected directly to the separator. The separator is not included in the model. The manifold is assumed to have perfect pressure controller. Any changes in the manifold pressure is assumed to be instantaneous.

Chokes

To avoid energy losses over the chokes, they should be kept as open as possible and therefor the wellhead choke valves are assumed to be kept at a fully open position. This assumption coincides with multiple other gas lifted wells where the wellhead choke is kept at a constant position and the well production is manipulated by gas lift inlet rate[7]. Another assumption is that the valve size has been chosen to be valid for most operating conditions by choosing an appropriate valve parameter C_v. The downhole gas lift injection valve is modeled without a choke opening, this is because the downhole gas lift injection valve usually is a mandrel with fixed opening.

2.2 The Model Equations

This differential algebraic equation system (DAE) is built up of 3 differential equations and 12 algebraic equations. The $\dot{m}$ denotes the change in mass over time for a control volume for both gas and oil. A flow sheet that depicts the control volumes with inlet and outlet flows of oil and gas is included in fig. 2.3. The control volumes C11 and C12 make up well 1 with inlet from the reservoir and gas lift, the control volumes C21 and C22 refer to well 2 also with inlet from gas lift and a second reservoir. The outlet flows from the two wells are combined in the manifold and sent to the separator.

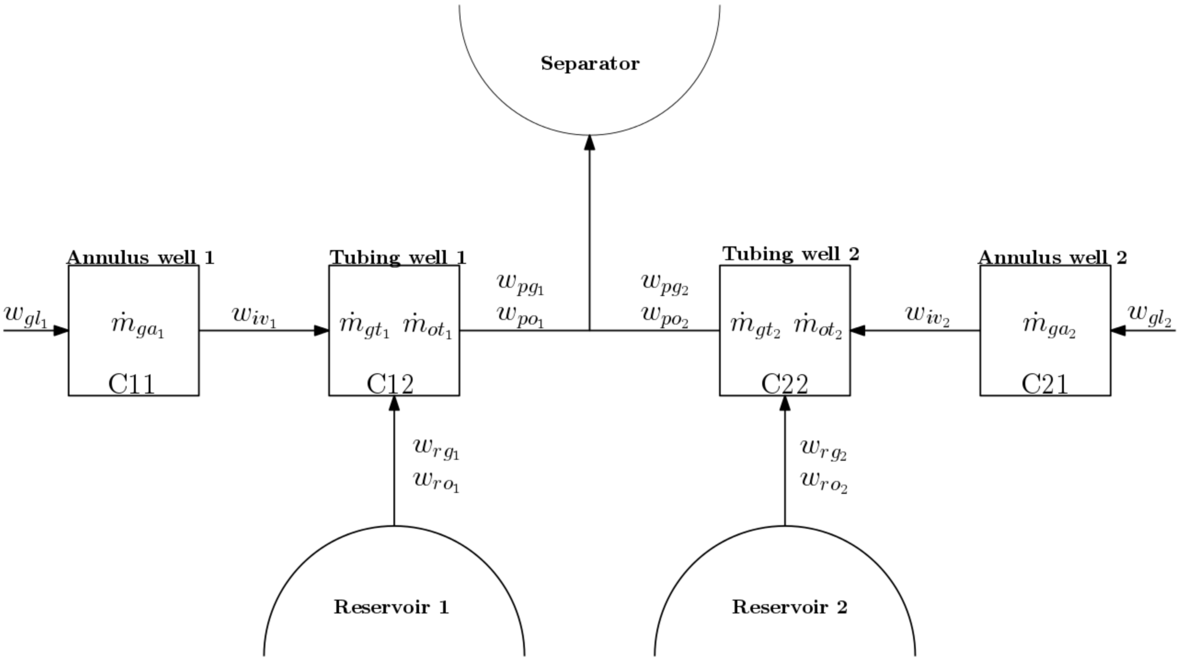


Figure 2.3: A flow sheet depicting how the control volumes of the system is connected. For each control volume the $\dot{m}$ represents the change in mass over time. w denotes the flows in and out of the control volumes. C11 and C12 make up well 1 and C21 and C22 make up well 2.

2.2.1 Differential equations

Mass Balances

For each well i the mass balances are given in eq. (2.1)-eq. (2.3).

$$\dot{m}_{ga_i} = w_{gl_i} - w_{iv_i} \tag{2.1}$$

$$\dot{m}_{gt_i} = w_{iv_i} - w_{pg_i} + w_{rg_i} \tag{2.2}$$

$$\dot{m}_{ot_i} = w_{ro_i} - w_{po_i} \tag{2.3}$$

Here m_{ga_i} is the mass of gas in the annulus, m_{gt_i} is the mass of gas in the tubing and m_{ot_i} is the mass of oil in the tubing. w_{gl_i} denotes the gas lift injection rate into well i, w_{iv_i} is the gas flow into the tubing, w_{rg_i} and w_{ro_i} is the gas and oil rate in from the reservoir, w_{pg_i} and w_{po_i} is the production rate of gas and oil from each well.

2.2.2 Algebraic equations

The algebraic part of the model consists of the equation for pressure, density, and flow for a well system of n_w number of wells. In addition the equations for the cross-sectional area calculations are needed.

Pressure Equations

The bottom hole pressure p_{bh_i}, well injection pressure p_{wi_i}, the annulus pressure p_{a_i} and the wellhead pressure p_{wh_i} for each well is given in eq. (2.4) - eq. (2.7).

$$p_{bh_i} = p_{wi_i} + \rho_{w_i} g H_{bh_i} \tag{2.4}$$

$$p_{wi_i} = p_{wh_i} + \frac{g}{L_{w_i} A_{w_i}} \left(m_{ot_i} + m_{gt_i} - \rho_o L_{bh_i} A_{bh_i} \right) H_{w_i} \tag{2.5}$$

$$p_{a_i} = \left(\frac{T_{a_i} R}{V_{a_i} M_w} + \frac{g L_{a_i}}{L_{a_i} A_{a_i}} \right) m_{ga_i} \tag{2.6}$$

$$p_{wh_i} = \frac{T_{w_i} R}{M_w} \left(\frac{m_{gt_i}}{L_{w_i} A_{w_i} + L_{bh_i} A_{bh_i} - \frac{m_{ot_i}}{\rho_o}} \right) \tag{2.7}$$

Here H_{bh_i} and H_{w_i} refers to the respective heights of each well below and above the injection point, L_{bh_i}, L_{w_i} and L_{a_i} is the length of the well below and above the injection point and of the annulus,

A_{bh_i}, A_{w_i} and A_{a_i} refers to the cross-sectional areas of the tubing below and above the injection point as well as the annulus. The temperatures in the tubing and the annulus is denoted by T_{w_i} and T_{a_i}, ρ_o and ρ_{w_i} are respectively the density of oil and density of the oil-gas mixture in the tubing, M_w is the molar weight of the gas, R is the gas constant and g is the acceleration of gravity constant.

Density Equations

The density of the gas in the annulus ρ_{a_i} and the fluid mixture in the tubing ρ_{w_i} for each well i is given by eq. (2.8) and eq. (2.9).

$$\rho_{a_i} = \frac{M_w p_{a_i}}{T_{a_i} R} \tag{2.8}$$

$$\rho_{w_i} = \frac{m_{gt_i} + m_{ot_i} - \rho_o L_{bh_i} A_{bh_i}}{L_{w_i} A_{w_i}} \tag{2.9}$$

Flow

The flow through each of the downhole gas lift injection valves w_{iv_i}, the flow through each of the production chokes w_{pc_i}, the gas and oil production rate from each well w_{pg_i} and w_{po_i}, and the gas and oil flow rates from the reservoir into each well w_{rg_i} and w_{ro_i} is given in equations eq. (2.10) - eq. (2.15).

$$w_{iv_i} = C_{iv_i}\sqrt{max(0, \rho_{a_i}(p_{a_i} - p_{wi_i}))} \tag{2.10}$$

$$w_{pc_i} = C_{pc_i}\sqrt{max(0, \rho_{w_i}(p_{wh_i} - p_m))} \tag{2.11}$$

$$w_{pg_i} = \frac{m_{gt_i}}{m_{gt_i} + m_{ot_i}} w_{pc_i} \tag{2.12}$$

$$w_{po_i} = \frac{m_{ot_i}}{m_{gt_i} + m_{ot_i}} w_{pc_i} \tag{2.13}$$

$$w_{rg_i} = GOR_i w_{ro_i} \tag{2.14}$$

$$w_{ro_i} = PI_i(p_{r_i} - p_{bh_i}) \tag{2.15}$$

Here C_{iv_i} and C_{pc_i} are the injection valves and production valves characteristics, GOR_i is the Gas-Oil ratio in each well and PI_i and p_{r_i} are the productivity indexes and pressure of the reservoir supplying each well. The manifold where the two wells mix together has a pressure denoted by p_m.

Cross-sectional Area Calculations

The cross-sectional area for the annulus A_{a_i} and tubing A_{w_i} for each well are calculated by using the respective diameters D_{a_i} and D_{w_i} in eq. (2.16) and eq. (2.17).

$$A_{a_i} = \frac{\pi D_{a_i}^2}{4} - \frac{\pi D_{w_i}^2}{4} \tag{2.16}$$

$$A_{w_i} = \frac{\pi D_{w_i}^2}{4} \tag{2.17}$$

2.2.3 DAE Formulation

This system is modeled as a semi-explicit index-1 DAE system that has the form shown in eq. (2.18).

$$\dot{x} = f(x, z, u) \tag{2.18a}$$

$$g(x, z, u) = 0 \tag{2.18b}$$

Here $f(x, z, u)$ denotes the set of the differential equations given in eq. (2.1) - eq. (2.3) and $g(x, z, u)$ denotes the set algebraic equations given in eq. (2.4) - eq. (2.15). This gives a set of differential states, x, algebraic states, z, and the decision variables, u, that are shown below in eq. (2.19a) - eq. (2.19c).

$$x = [m_{ga_i} \quad m_{gt_i} \quad m_{ot_i}]^T \tag{2.19a}$$

$$\begin{aligned} z = [&p_{bh_i} \quad p_{wi_i} \quad p_{a_i} \quad p_{wh_i} \quad \rho_{a_i} \quad \rho_{m_i} \\ &w_{iv_i} \quad w_{pc_i} \quad w_{pg_i} \quad w_{po_i} \quad w_{ro_i} \quad w_{rg_i}] \end{aligned} \tag{2.19b}$$

$$u = w_{gl_i} \tag{2.19c}$$

2.3 Problem Formulation

The system used in this work has 4 manipulated variables (MVs), and these are the degrees of freedom (DOF) in the control system. The DOF are two well head choke positions and the two gas lift injection rates. To avoid production losses, the two chokes are positioned fully open. The two remaining DOF, the objective is to maximize the total produced oil, subject to total gas capacity constraint and total available gas lift constraint. The problem formulation is shown below in eq. (2.20e).

$$
\begin{aligned}
\min \quad & -\sum_{i=1}^{n_w} w_{po_i} && (2.20a)\\
s.t. \quad & \sum_{i=1}^{n_w} w_{pg_i} \leq Q_{g_{max}} && (2.20b)\\
& \sum_{i=1}^{n_w} w_{gl_i} \leq Q_{GL_{max}} && (2.20c)\\
& \dot{x} = f(x,z,u) && (2.20d)\\
& g(x,z,u) = 0 && (2.20e)
\end{aligned}
$$

Where the objective function is the sum of oil produce from each well w_{po_i}, subject to two different constraints in eq. (2.20b) and eq. (2.20c). The objective function is also subject to the system equations, consisting of the differential and algebraic equations from eq. (2.18). $Q_{g_{max}}$ and $Q_{GL_{max}}$ denote the maximum gas capacity constraint and the available gas lift constraint.

2.4 The Decomposed Model

In this study, the main model is decomposed into subproblems connected by a common constraint to investigate this control structure. The model with two gas lifted well systems is separated into two subproblems, one problem for each well. The manifold pressure p_m is given as an input to the subproblems. The decomposed DAE structure for each subproblem i is shown in eq. (2.21) where f_i and g_i are the separated differential and algebraic equations for each subproblem, each well, denoted by i.

$$
\begin{aligned}
& \dot{x}_i = f_i(x,z,u) && (2.21a)\\
& g_i(x,z,u) = 0 && (2.21b)
\end{aligned}
$$

For the decomposed system there is a set of differential states, x, algebraic states, z, and decision variables u for each subproblem. The new sets for the first subproblem, well 1, is shown below in eq. (2.22a) - eq. (2.22c) and the new sets for the second subproblem, well 2, is shown below in eq. (2.23a) - eq. (2.23c).

$$
\begin{aligned}
x_1 &= [m_{ga_1} \quad m_{gt_1} \quad m_{ot_1}]^T && (2.22a)\\
z_1 &= [p_{bh_1} \quad p_{wi_1} \quad p_{a_1} \quad p_{wh_1} \quad \rho_{a_1} \quad \rho_{m_1} \quad w_{iv_1} \quad w_{pc_1} \quad w_{pg_1} \quad w_{po_1} \quad w_{ro_1} \quad w_{rg_1}] && (2.22b)\\
u_1 &= w_{gl_1} && (2.22c)
\end{aligned}
$$

$$
\begin{aligned}
x_2 &= [m_{ga_2} \quad m_{gt_2} \quad m_{ot_2}]^T && (2.23a)\\
z_2 &= [p_{bh_2} \quad p_{wi_2} \quad p_{a_2} \quad p_{wh_2} \quad \rho_{a_2} \quad \rho_{m_2} \quad w_{iv_2} \quad w_{pc_2} \quad w_{pg_2} \quad w_{po_2} \quad w_{ro_2} \quad w_{rg_2}] && (2.23b)\\
u_2 &= w_{gl_2} && (2.23c)
\end{aligned}
$$

Subproblem 1 is formulated below in eq. (2.24d), where the objective function is the oil produced from well 1, w_{po_1} subjected to either of the constraint. Only one of the constraints can be active, depending on the values of the maximum gas capacity and the available gas lift. The objective function is also subjected to the decomposed DAE from eq. (2.21).

$$
\begin{aligned}
\min \quad & -w_{po_1} && (2.24a)\\
s.t. \quad & \sum_{i=1}^{n_w} w_{pg_i} \leq Q_{g_{max}} && (2.24b)\\
\text{or} \quad & \sum_{i=1}^{n_w} w_{gl_i} \leq Q_{GL_{max}} && (2.24c)\\
& eq.\ (2.21) && (2.24d)
\end{aligned}
$$

The second subproblem 2 is formulated similar to the first subproblem. The oil produced from well 2, w_{po_2}, is maximized subjected to either of the constraints. The system equations for well 2 are defined from eq. (2.21), where $i = 2$, and the objective function is subjected to these differential and algebraic equations.

$$
\begin{aligned}
\min \quad & -w_{po_2} && \text{(2.25a)}\\
s.t. \quad & \sum_{i=1}^{n_w} w_{pg_i} \leq Q_{g_{max}} && \text{(2.25b)}\\
\text{or} \quad & \sum_{i=1}^{n_w} w_{gl_i} \leq Q_{GL_{max}} && \text{(2.25c)}\\
& eq.\ (2.21) && \text{(2.25d)}
\end{aligned}
$$

Chapter 3

Control and Optimization Theory

Optimization is an important tool in any decision making, whether it happens naturally as a physical system seeking the state of minimum energy, or the engineer seeking to optimize production in a plant. In science, optimization started with early mathematics and research on the topic was mostly done in theory, until the development of the simplex method for linear programming in 1947 by Dantzig [20]. After this work, it started to become a focus to research and optimization was implemented within real-world problems. Today, controlling operations to it's optimum is a vital part for the industry to save both time and money.

Identifying the objective, variables and constraints of an optimization problem is called modeling. This model can be written as an optimal control problem(OCP), which aims to find the path of the systems control variable to minimize the cost function[21]. Optimization and control is widely used in chemical engineering, especially within the process system engineering, where problems often can be complex, both in economics and performance of a large composed system. The need for a common and automatic approach of finding the optimal solutions, are the basis for control strategies. There is always a need for new and better solutions.

For a long time, the engineers' intuition was the guidelines for controlling a chemical process system. However, with more and more complex systems and plants, the theoretical work on MPC has been trying to catch up to the long tradition of engineering use of the model predictive controller method[22]. However, there are many large progresses being made within the field. An overview of methods on Robust MPC up till 2006 was performed by Jalali et al and is presented in [23].

In this chapter, first an Optimal Control Problem(OCP) is defined, the concept of Real-Time Optimization(RTO) and Model Predictive Control(MPC) is explained and then the collocation method to solving Nonlinear MPC is presented. This follows the method of this thesis, starting with an OCP, the collocation method is utilized to turn the problem into an NLP. Then the NLP is solved using an Interior-Point Optimization approach in CadADi.

In mathematics and engineering, optimization is searching for the best solution of a problem, finding a maximum or minimum of an objective function defined as f(x,z,u) depending on the variables x, z, u, subjected to constraints defined by the system. For this work, the nonlinearity in the state equations for the system classifies this as a nonlinear programming(NLP) problem. NLP is one of the methods to solving an optimization problem, when the constraint equations or the objective function of an optimization problem have one or more nonlinear equations. An algorithm class to solving NLP problems is the interior-point methods, it is considered one of the best solution tools for large-scale nonlinear programming[21]. Nonlinear programming and the interior-point methods are explained further in "Numerical Optimization" by Nocedal and Wright[21]. The interior-point method has been adapted to a programming algorithm and implemented as an open source NLP solver named IPOPT(Interior-Point Optimization)[24] which is the solver used in this thesis.

3.1 Optimal Control Problem (OCP)

Optimal control is choosing the optimal values of input to a dynamic system. A standard optimization problem using a semi-explicit DAE is presented in eq. (3.1) where x(t) are the differential states, z(t) are the algebraic states, f(x,z,u,) are the differential equations and g(x,z,u,) are the algebraic equations. More on DAE and numerical methods for solving can be found in "Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations" by Ascher and Petzold [25].

$$
\begin{aligned}
&\dot{x} = f(x(t), z(t), u(t)) \\
&x(0) = x_0 \\
&g(x(t), z(t), u(t)) = 0
\end{aligned}
\tag{3.1}
$$

The state variables and the control variables have lower and upper constraints for the operating conditions. The underline of the variables denotes the lower limit, $\underline{x}$, $\underline{z}$, $\underline{u}$. The bar over the variables denote the upper limit of the states and control variables, $\overline{x}$, $\overline{z}$, $\overline{u}$. The constraints of the states and control variables are presented in eq. (3.4). The constraints are modeled from limitations of the system, depending on for instance capacity constraints over valves or control volumes.

$$\underline{x} \leq x \leq \overline{x} \tag{3.2}$$

$$\underline{z} \leq z \leq \overline{z} \tag{3.3}$$

$$\underline{u} \leq u \leq \overline{u} \tag{3.4}$$

For this semi-explicit constrained DAE the OCP can be written eq. (3.5).

$$\min_{u(t),x(t),z(t)} \quad \Phi(x(t), z(t), u(t), p) \tag{3.5a}$$

$$s.t \quad \dot{x} = f(x(t), z(t), u(t), p) \tag{3.5b}$$

$$x(0) = x_0 \tag{3.5c}$$

$$g(x(t), z(t), u(t), p) = 0 \tag{3.5d}$$

$$\underline{x} \leq x \leq \overline{x} \tag{3.5e}$$

$$\underline{z} \leq z \leq \overline{z} \tag{3.5f}$$

$$\underline{u} \leq u \leq \overline{u} \tag{3.5g}$$

3.2 Model Predictive Controller (MPC)

The Model Predictive Controller (MPC) uses a dynamic model to predict the system behavior for a finite time-horizon and generate process input for a real-time optimization problem. The predicted input variable is optimized to obtain the optimal solution of the cost function. The optimized value of the input for the current time step is implemented into the simulation of the plant to develop a closed loop control system. For a detailed theory and examples of MPC design see "Model Predictive Control: Theory and Design" by Rawlings and Mayne [26].

Traditionally, MPC consists of an optimization problem to achieve certain control objectives, such as set point control, state and input constraints, rate of change constraints etc. The behavior of the control variable is determined from an optimization starting at the current position and predicted for a given time-horizon into the future. The measured variables for the states of the system are given as new initial conditions for the next horizon optimization. The Receding horizon strategy that define MPCs is illustrated in fig. 3.1 where the measured variables and the control variables are plotted both in the past and in the predicted future. The measured and the predicted output is compared to a reference trajectory.

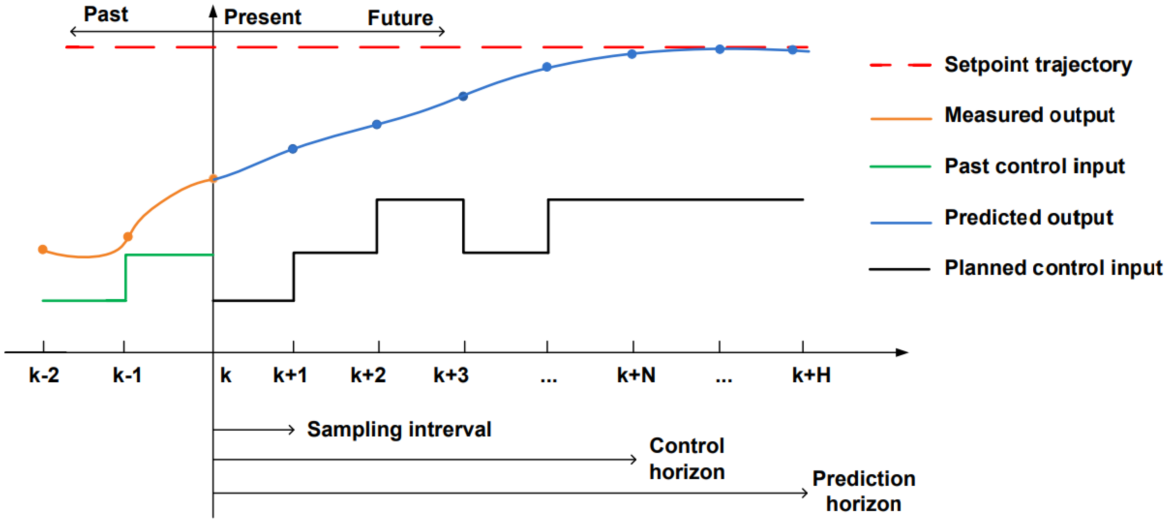


Figure 3.1: Receding horizon strategy used in an MPC. Where the predicted output and the corresponding planned control input is shown in the future trying to reach the setpoint. The measured output and past control input in the past is connected to the predicted values in the present position k. The figure is taken from page 4 in [27].

The advantage to the MPC method is that it can anticipate future changes and change the control variable accordingly, hence diminishing the impact of changes on the operating conditions. For a plant with an implemented MPC, the benefits could be less down time, better performance of control and improved flexibility. [22]

3.2.1 Nonlinear Model Predictive Control(NMPC)

NMPC is an OCP that is constantly solved on-line, usually for a DAE model. It is a MPC with a nonlinear cost function or constraints equations. The cost function should penalize the distance from an arbitrary state to an optimal solution of that state. NMPC is explained more in detail in "Nonlinear Model Predictive Control: Theory and Algorithm" by Gründe et. al. [28]. A study reviewing the recent advanced within dynamic RTOs like NMPC[29] concludes that advances in large-scale NLP solvers opens for NMPC requiring very little on-line computations.

In this thesis the control system is an NMPC with a cost function of eq. (3.6) with dynamic parts of the state equations. The mathematical model of the NMPC can be solved using the collocation method described below to write the problem as an NLP.

$$J = -\sum_{i=1}^{n_w} w_{po_i} \tag{3.6}$$

3.3 Real-time Optimization(RTO)

On line optimization constantly updates the control system according to measurements and the prediction of the mathematical model of the system. The purpose is to operate a plant as close to optimum as possible at all times. Traditionally, Real-time optimization(RTO) employs static models and provides set points to lower layer MPC controllers. Static RTO is is often performed open-loop and there is limited Feedback to the RTO and for this reason, it is not very good [7].

Dynamic RTO, on the other hand, uses a dynamic model on line and computes set points at each sampling time. Increasingly, there has been a steady interest in tightly integrating the RTO and MPC layers by combining economic and control objectives in the framework of the so-called Economic MPC. In this work, the optimization problem is formulated as an economic MPC problem.

An economical model has an objective function that includes a cost or production value that is maximized or minimized. The different uses and methods to RTO, as well as challenges to using RTO, is discussed in [30]. In this work, the on line optimization problem is defined as a Real-time production optimization(RTPO).

3.4 Collocation Method

For dynamic systems, a higher-order method is required for solving. The collocation method is a higher-order implicit Runge-Kutta method often used to solve OCP. For this chapter, "Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes" by Biegler [31], is used as a central source. Also the method in this thesis is adapted from the lecture notes by Sébastian Gross for "Winter School on Numerical Optimal Control with Differential Algebraic Equations" [32] and [33]. The large-scale NLP formulation that results from the collocation of the OCP, produces a large, structured and sparse matrix which makes the problem more efficient to solve. In addition to this advantage, DAE solvers with convergence difficulties can be avoided with better NLP solvers.

3.4.1 Polynomial integration

In a time interval split into time grids, a function f can approximate over the time interval. Considering a time grid $t_{k,0} \ldots, t_{k,K} \in [t_k, t_{k+1}]$ a Lagrange polynomial can be defined for each $t_{k,i}$ on the time interval to approximate f. The degree of the polynomial is the same as the number of points on the time interval. The lagrangian polynomial $\mathscr{L}_{k,i}(t)$ is defined in eq. (3.7) with the order of K.

$$\mathscr{L}_{k,i}(t) = \prod_{j=0,j\neq i}^{K} \frac{t - t_{k,j}}{t_{k,i} - t_{k,j}} \tag{3.7}$$

Here $t_{k,i}$ with $i \in (0,4)$ with collocation points that are between t_k and t_{k+1}. The properties of the Lagrange polynomial is defined in eq. (3.8).

$$\mathscr{L}_{k,i}(t_{k,l}) = \begin{cases} 1 & \text{if } l = i \\ 0 & \text{if } l \neq i \end{cases} \tag{3.8}$$

Then interpolation with the parameters $\theta_{k,i} \in \mathbb{R}^n$ and $\mathscr{L}_{k,i}(t)$ for each collocation point approximates approximates a function variable x.

$$x(\theta_k, t) = \sum_{i=0}^{k} \underbrace{\theta_{k,i}}_{parameters} \cdot \underbrace{\mathscr{L}_{k,i}(t)}_{polynomials} \tag{3.9}$$

The selection of the collocation points can be done with either a gauss-legendre or a radau roots method. The collocation point location varies with the degree of the Lagrange polynomial. The values of the collocation point for gauss-legendre and radau roots are included in table B.1 in appendix B.

For DAEs, the Radau collocation points are better suited, while the Gauss-Legendre collocation points are better suited for ODEs. The Radau method for finding the collocation point, applies the interpolation at every time-step and this discretization method has been found to behave much better than collocation at Gauss-Legendre points. The Radau method is less exposed to oscillations and is stable for stiff differential system equations and for large time-steps[34].

To illustrate the polynomial integration over an interval [k,k+1], the scheme representing a third order direct collocation using Radau collocation points is included below in fig. 3.2. The polynomial approximation is shown for the state and control variables x,z and u.

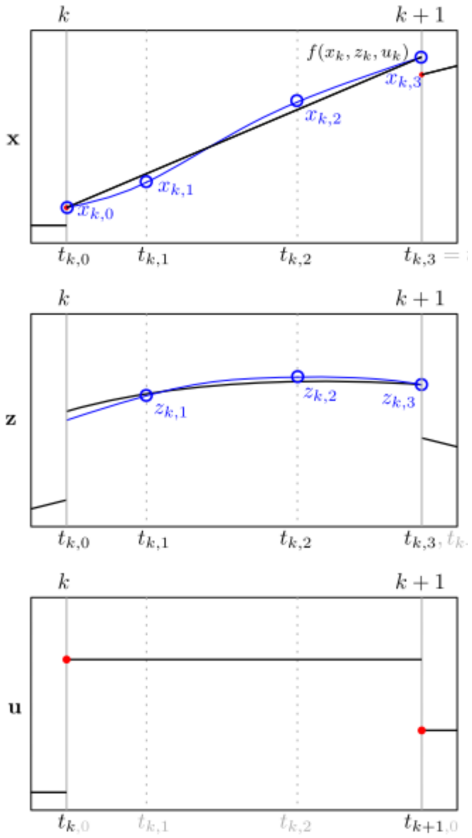


Figure 3.2: Polynomial approximation for a third order direct collocation using Radaus points, on a single sampling interval [k,k+1]. The differential states x has one more point than the algebraic states z, which is located at $t_{k,0}$ and assures the continuity with the shooting gap constraint. The control input u is piecewise constant over the interval. Taken from[5] p.7.

3.4.2 Direct Collocation Method for a Constrained DAE problem

Direct collocation is a method within collocation in addition to single and multiple shooting collocation. It is the most preferred method when it comes to handling DAE systems. The direct collocation method is the approach that generates the biggest and the most sparse NLP. The method for direct collocation is implemented for the given the semi-explicit DAE with constraints in eq. (3.1), turning this DAE-problem into a NLP-problem. The definition of interpolation with the Lagrange polynomials applied to the states of the semi-explicit DAE results in the interpolations for differential states x and the algebraic states z that are given in eq. (3.10a) and eq. (3.10b).

$$x(\theta_k, t) = \sum_{i=0}^{k} \overbrace{\theta_{k,i}}^{parameters} \cdot \overbrace{\mathscr{L}_{k,i}(t)}^{polynomials} \tag{3.10a}$$

$$z(z_k, t) = \sum_{i=0}^{k} \underbrace{z_{k,i}}_{parameters} \cdot \underbrace{\mathscr{L}_{k,i}(t)}_{polynomials} \tag{3.10b}$$

For a point $t_{k,i}$ on the time interval, the approximations have the definition $x(\theta_k, t_{k,i}) = \theta_{k,i}$ and $z(z_k, t_{k,i}) = z_{k,i}$ where the differential states have one higher degree of the polynomial, K+1, than the algebraic state which have K degrees of freedom. This is because the algebraic states can be discontinuous and does not need to close the shooting gap.

With the interpolations presented above, the constraints for the NLP problem from direct collocation for semi-explicit DAE become as shown in eq. (3.11a) for continuity constraint, eq. (3.11b) for the dynamics constraint and eq. (3.11c) for the algebraic constraint.

$$0 = \theta_{k,K} - \theta_{k+1,0} \tag{3.11a}$$

$$\frac{\partial}{\partial t} x(\theta_k, t_{k,i}) = F(\theta_{k,i}, z_{k,i}, u_k) \tag{3.11b}$$

$$0 = G(\theta_{k,i}, z_{k,i}, u_k) \tag{3.11c}$$

Then the complete NLP problem transformed from DAE with direct collocation can be written as an OCP as shown in eq. (3.12b).

$$\min_{w} \quad \Phi(w) \tag{3.12a}$$

$$s.t \quad g(w) = \begin{pmatrix} \theta_{0,0} - x_0 \\ \theta_{0,K} - \theta_{1,0} \\ F(\theta_{k,0}, z_{k,0}, u_k) - \frac{\partial}{\partial t} x(\theta_k, t_{k,0}) \\ G(\theta_{k,0}, z_{k,0}, u_k) \\ \dots \\ \theta_{k,K} - \theta_{k+1,0} \\ F(\theta_{k,K}, z_{k,K}, u_k) - \frac{\partial}{\partial t} x(\theta_k, t_{k,K}) \\ G(\theta_{k,K}, z_{k,K}, u_k) \\ \dots \end{pmatrix} \tag{3.12b}$$

Where the first row of the NLP constraints g(w) represents the initial conditions x_0, second row is the shooting gap constraints for continuity, third and fourth row are the integration constraints for differential and algebraic states at k=0. The second to fourth rows are repeated for k=0,...,N+1, as shown for a middle point k=K. The decision variables for this problem are collected in $w = \{\dots, \theta_{k,0}, \theta_{k,1}, z_{k,1}, \dots, \theta_{k,K}, z_{k,K}, u_k, \dots\}$ for k=0$,\dots,N-1$.

Chapter 4

Decomposition

The motivation for performing a decomposition of a large scale problem is presented in the introduction of this thesis in chapter 1. In this chapter, the previous work on decomposition is reviewed before the general theory and idea of decomposition and the Lagrangian Dual(LD) decomposition method is presented. Then, the decomposition methods are adapted to fit the model described in chapter 2. This gives algorithms for the decomposition that are used in the programming and simulations of the decentralized solver of the system shown in chapter 5. Lastly, theory of numerical methods for improving the results of the decompositions is presented. A very central source used in this chapter is "Notes on Decomposition" by Boyd et al[11].

Decomposition is the method of solving a major problem by splitting it into multiple subproblems and solve the minor problems either sequential or in parallel[11]. The decentralized optimization problem can yield an optimal solution for each subproblem that is connected by a master problem for the main control system.

When finding a good solution to a complicated problem using combinatorial optimization, two criteria must be considered[35]. Each optimization problem must calculate:

- a lower bound that is as close to the optimum solution as possible
- an upper bound that is as close to the optimum solution as possible

The upper bound is in this case given by a maximum constraint on the total gas production capacity. The objective is to find a lower bound that is as close to the optimum solution as possible by using the different decomposition methods for the optimization problem. The Lagrange Dual(LD) decomposition uses the Lagrange relaxation to enable a splitting of the main problem.

4.1 Previous Work on Decomposition

Dynamic decomposition is a well known method, but has not yet been investigated and implemented into the field of oil and gas production. There is, however, work that has been done on steady state decomposition for usage within oil and gas, with the Troll field as an example field structure. The method and conclusion of this work is explained below.

4.1.1 Decomposition in Oil and Gas

Two papers covering the use of decomposition within an oil and gas field was reviewed. The first study compared two decomposition methods where Lagrangian decomposition was one method [17]. The decomposition with Lagrangian relaxation was performed for a mixed integer linear program (MILP) model of a gas lifted oil well system based on a general network topology, very similar to the Troll field structure. The nonlinearity of the model was piecewise linearized before the decentralized model was developed with a LD decomposition. The decomposed model consist of subproblems, each cluster containing one manifold. The subproblems are controlled by a master problem, with the common constraint also decomposed and added to the objective functions.

The second paper investigated only LD decomposition for the Troll field with larger clusters, each cluster containing two manifolds[16]. As in the first paper, the model was piecewise linearized so that each subproblem becomes an MILP problem before the LD decomposition was performed. Then, the Lagrangian decomposition was tested against a global strategy with all the clusters in one MILP, and a with a preallocation of gas to each cluster for the sum of the gas reaching the gas constraint.

The results and the conclusion from these papers favor the decomposition method for finding the optimal set points for steady state optimization. They both argue that decomposition is well suited for an oil and gas optimization problem for such a complex field structure as that of the Troll field. The papers conclude that the decentralized control structure outperforms a global control structure in efficiency and that it can be further improved by parallel solving of the decomposed optimization problems. It is also noted in both papers that the decomposed structures can solve larger problems more easily than the global method. In [17], the global method was not able to solve a problem with eight clusters(subproblems) and struggled with six clusters. In [16], the global approach was not able to handle more than two clusters, with six wells in each cluster. From this it can be assumed that the more complex system, the more beneficial can decomposition become. The paper comparing Danzig-Wolfe decomposition to LD decomposition, prefers the first method because of a more efficient updating algorithm of Lagrange multipliers. In this paper, line search method and an augmented lagrangian method are applied to improve the updating algorithm of the lagrangian decomposition.

Since these are stationary problems, they are re-optimized frequently, typically once a day, which means they can be defined as quasi-dynamic controls. One of the papers[17] points out that dynamic models being decomposed might be beneficial for certain model were the dynamics are important to the optimal performance, for example long pipe models. But this might be a challenge during start-up of wells, which can become a problem if shut down is frequently caused by maintenance or other reasons.

4.2 Decomposition with Constraints

When the problem considered is not completely separable because of a common constraint, the subproblems must be coupled through this constraint. A simple example where the subproblems are coupled by a common constraint is shown below in eq. (4.1).

$$\begin{aligned} min \quad & f_1(x_1) + f_2(x_2) \\ s.t. \quad & x_i \in C_1, x_2 \in C_2 \\ & h_1(x_1) + h_2(x_2) \leq 0 \end{aligned} \tag{4.1}$$

Where $f_1(x_1)$ and $f_2(x_2)$ are two subproblems coupled by a constraint that $h_1(x_1)$ and $h_2(x_2)$ collectively has an upper limit of 0. In this study the Lagrange Dual decomposition has been used, which is described below.

4.3 Dual decomposition - Lagrangian Relaxation

As mentioned above, a decomposition problem needs to generate a good lower and upper bound for the problem. To generate a good lower bound, a Lagrangian relaxation of the original problem can be generated. The generating and application of the lagrangian relaxation is explained in "Modern Heuristics Techniques for Combinatorial Problems" by Beasley[35]. The steps to Lagrangian relaxation for an integer program involves:

1. moving any number of the constraint into the objective functions adding a Lagrange multiplier to relax the constraints

2. finding an exact solution for the new integer program

The exact solution for the relaxed problem gives a lower bound to the optimum of the original problem.

There are multiple advantages to solving a relaxed integer program, instead of directly finding an optimal solution for the original integer program. Two of the reasons for using Lagrangian relaxation is that it has been found to generate good lower bounds at low computational costs. In addition, for a simple problem with complication of extra constraints, a solution where the constraints are absorbed into the objective function can simplify the total problem. In this problem the Lagrange multiplier is variable coupling the common constraints of the two wells.

4.3.1 Lagrange Multiplier and Finding the Maximum Lower Bound

For an optimization problem given in eq. (4.2), the constraints can be relaxed using Lagrange multiplier λ to generate a lower bound for the optimal solution.

$$\begin{aligned} max \quad & f(x,y) \\ s.t. \quad & h(x,y) = a \end{aligned} \tag{4.2}$$

This optimization problem can be rewritten with the Lagrangian relaxation by introducing the Lagrange multiplier $\lambda \geq 0$. Bringing the constraint into the objective function gives a new problem in eq. (4.3). The new problem can be referred to as the Lagrangian Lower Bound Program (LLBP).

$$\mathscr{L}(x,y,\lambda) = f(x,y) - \lambda \cdot (h(x,y) - a) \tag{4.3}$$

To generate a lower bound as close to the optimal solution as possible, the choosing of the numerical value of λ is very important, in other words the ideal solution is the maximum lower bound. This problem is formulated by maximizing λ over the problem, 4.4, this gives the Lagrangian dual program.

$$\max_{\lambda \geq 0} \begin{cases} \min & f(x) + \lambda(a - h_1(x)) \\ s.t. & x \in (0,1) \end{cases} \tag{4.4}$$

4.3.2 Dual Decomposition for the well system with coupled constraints

Using the Lagrangian relaxation explained above, an optimization problem with a constraint can be split into two subproblems with a common constraint[11]. The decomposed model is explained in previous chapter 2. The constraint is included in the optimization problem first with the use of the Lagrange multiplier λ. The partial Lagrangian is then written as in eq. (4.5).

$$\begin{aligned} \mathscr{L}(x_1, x_2, \lambda) &= f_1(x_1) + f_2(x_2) + \lambda^T(h_1(x_1) + h_2(x_2)) \\ &= (f_1(x_1) + \lambda^T h_1(x_1)) + (f_2(x_2) + \lambda^T h_2(x_2)) \end{aligned} \tag{4.5}$$

This partial Lagrangian can be split into two subproblems connected by a common λ to not violate the constraint while optimizing both subproblems. When the constraint is included in the optimization problem it can be written as the dual function $g(\lambda)$, shown in eq. (4.7). The decomposed functions $g_1(\lambda)$ and $g_2(\lambda)$ can be evaluated fully independently, in parallel or sequential.

$$g(\lambda) = g_1(\lambda) + g_2(\lambda) \tag{4.6}$$

$$\text{where}: g_i(\lambda) = (f_i(x_i) + \lambda^T h_i(x_i)) \tag{4.7}$$

The first subproblem is written in eq. (4.8).

$$\begin{aligned} min \quad & f_1(x_1) + \lambda^T h_1(x_1) \\ s.t. \quad & x_1 \in C_1 \end{aligned} \tag{4.8}$$

The second subproblem for this study of two well gas lifted system is written in eq. (4.9).

$$\begin{aligned} min \quad & f_2(x_2) + \lambda^T h_2(x_2) \\ s.t. \quad & x_2 \in C_2 \end{aligned} \tag{4.9}$$

The master algorithm for the Lagrange Dual decomposition updates the Lagrange multiplier λ for each iteration until it reaches a given termination criteria. The master algorithm is illustrated below in fig. 4.1.

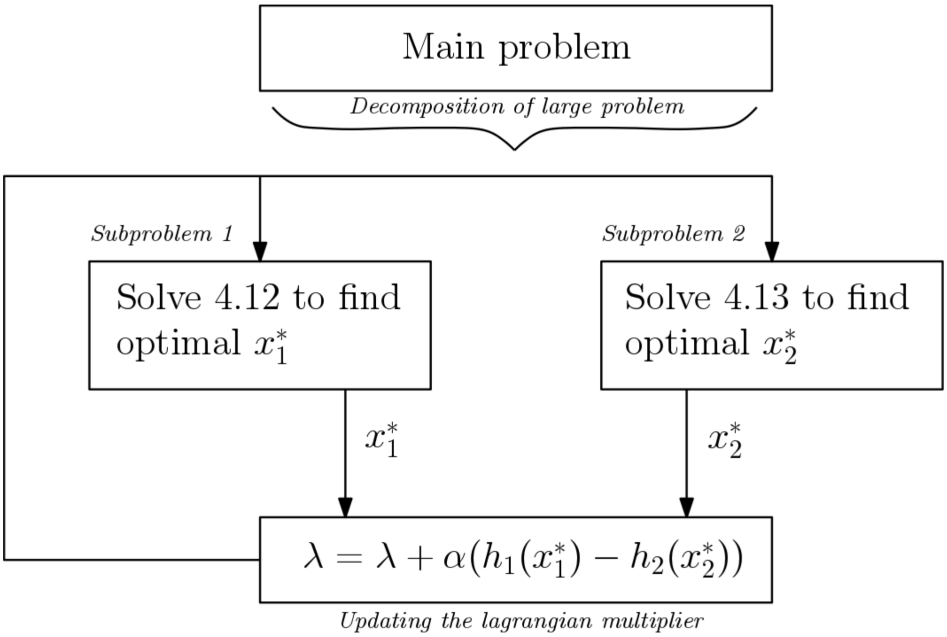

Main problem
Decomposition of large problem
Subproblem 1
Subproblem 2
Solve 4.12 to find optimal x_1^*
Solve 4.13 to find optimal x_2^*
x_1^*
x_2^*
$\lambda = \lambda + \alpha(h_1(x_1^*) - h_2(x_2^*))$
Updating the lagrangian multiplier

Figure 4.1: The general algorithm for LD decomposition with eq. (4.8) and eq. (4.9) as the two decomposed optimization problems. The lagrange multiplier, the coupled variable in this decomposition, is updated for each loop through the step length α until it converges to an optimal value. The tolerance is set to be an acceptable value and is measured by the change in λ for each iteration.

4.3.3 Formulating the Lagrangian Dual Decomposition for the Model

For this study, the optimization objective for the first subproblem(eq. (4.11)) $f_1(x_1)$ is the oil production from well 1 and the decomposed constraint function $h_1(x_1)$ presented in eq. (4.10) is the gas produced from well 1. Through the Lagrange multiplier and the decomposed constraint function this subproblem has a common constraint for the total gas production capacity.

$$h_1(x_1) = w_{pg_1} \tag{4.10}$$

$$\max \quad w_{po_1} + \lambda^T w_{pg_1} \tag{4.11}$$

In the same form as well 1, the decomposed optimization objective for the second subproblem eq. (4.13) $f_2(x_2)$ is the oil production from well 2. The decomposed constraint function $h_2(x_2)$ presented in eq. (4.12) is the gas produced from well 2 with the common constraint for the total gas production capacity for both wells combined added.

$$h_2(x_2) = w_{pg_2} - Q_{g_{max}} \tag{4.12}$$

$$\max \quad w_{po_2} + \lambda^T(w_{pg_2} - Q_{g_{max}}) \tag{4.13}$$

The flow sheet for solving the dual decomposition for this model is illustrated in fig. 4.2, following the master algorithm given in fig. 4.1. The calculated gas production from each well is used to update the Lagrange multiplier λ to maintain the active constraint of the total gas capacity. The error from the active constraint corrects the λ with an appropriate step length α_k.

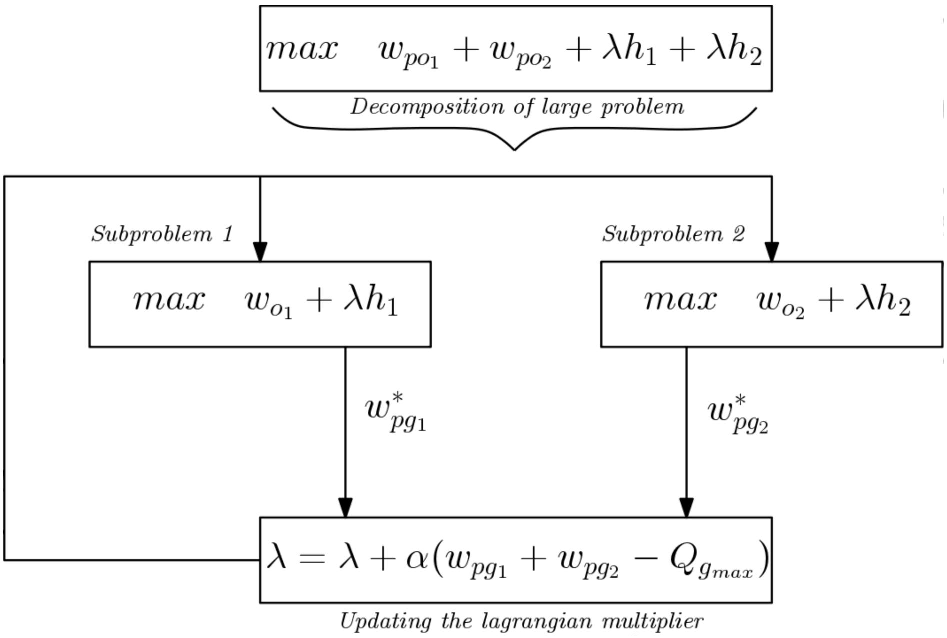

$max \quad w_{po_1} + w_{po_2} + \lambda h_1 + \lambda h_2$
Decomposition of large problem
Subproblem 1
Subproblem 2
$max \quad w_{o_1} + \lambda h_1$
$max \quad w_{o_2} + \lambda h_2$
$w^*_{pg_1}$
$w^*_{pg_2}$
$\lambda = \lambda + \alpha(w_{pg_1} + w_{pg_2} - Q_{g_{max}})$
Updating the lagrangian multiplier

Figure 4.2: The algorithm for the lagrangian dual decomposition solution adapted to fit the model used in this thesis. The two subproblems' objective functions are given in eq. (4.11) and eq. (4.13). The optimal gas production $w^*_{pg_i}$ for each subproblem and the step length α_k is used to update the Lagrange multiplier λ.

Since λ >0 the implementation of the error is written as in eq. (4.14).

$$\lambda = \max(0, \lambda + \alpha_k(w_{pg_1} + w_{pg_1} - Q_{g_{max}})) \tag{4.14}$$

4.4 Solving the dynamic master problem

After optimizing the decomposed OCPs and finding the correct value of the lagrangian multiplier λ for the optimal solution, the next step is implementing the optimal value of the control value at the first time step into the simulation of the plant. The simulation of the plant is for the combined model of the two wells, with the optimum of the two control variable found from the decentralized model prediction. After the simulation of the plant, new initial values for the state variables are given as input for the next optimization of each of the decomposed OCPs. The master problem structure is shown in fig. 4.3.

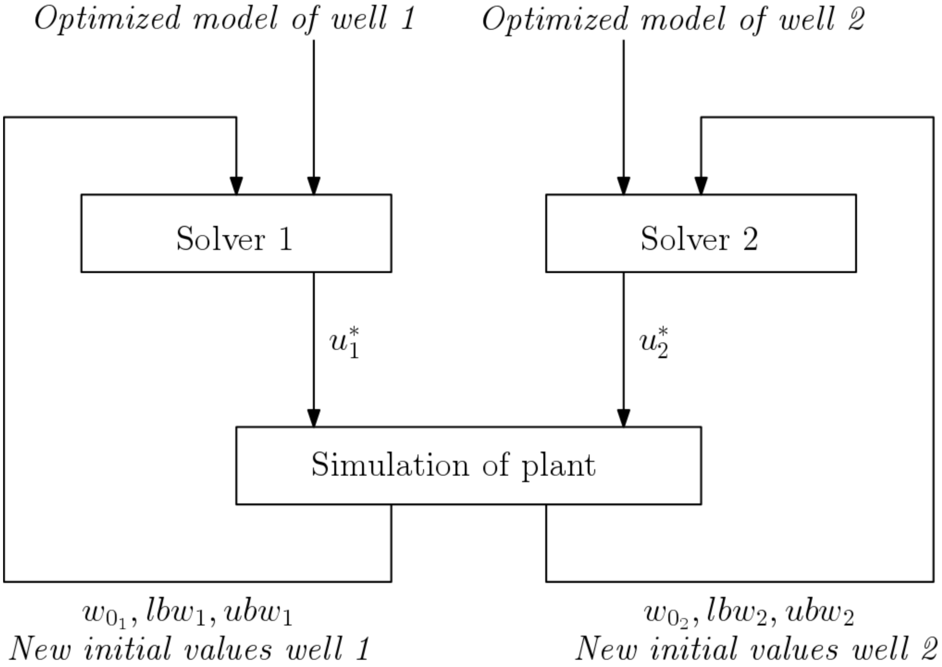

Optimized model of well 1
Optimized model of well 2
Solver 1
Solver 2
u_1^*
u_2^*
Simulation of plant
w_{0_1}, lbw_1, ubw_1
New initial values well 1
w_{0_2}, lbw_2, ubw_2
New initial values well 2

Figure 4.3: The master algorithm for solving the closed-loop optimal control problem. The algorithms for each solver, decomposed optimization, is shown in fig. 4.2. The optimal input value of the MVs are given as input for simulation of the plant. The simulated measurements are given as initial and boundary values to the solvers for the next optimization.

4.5 Line Search Methods

The line search method is a procedure to find a good step length, α, for each iteration as the value of x_k gets closer to the optimum solution. The algorithm searches along the direction p_k for the new iterate with a lower value than the current iterate value x_k. From this search, the distance to the next iteration point is determined, also known as the step length α_k. The efficiency of the method is very dependent on the choice of the step length α_k and direction p_k. The background for the line search method is taken from "Numerical Optimization" by Nocedal and Wright [21].

The trick is to find the right balance between choosing a large enough α_k, to give a sufficient reduction in the function value f, as well as not spending too much time searching for the best α_k. Therefore, the line search method is performed in two stages, where the first stage is identifying intervals for suitable values of α_k, and the second stage is estimating a good step within this specific interval. A common line search method outline of the first stage is to try out multiple values of α_k until the step length satisfies certain conditions. There are multiple different termination conditions[21]. For this study, the sufficient decrease condition is chosen as the terminating definition and backtracking is used to compute a new value of α_k for each iteration. This is described below.

4.5.1 Sufficient Decrease Condition and Backtracking

Defining α_k from satisfying the condition $f(x_k + \alpha_k p_k) < f(x_k)$, which implies a decrease in the function value for each iteration, is a simple assumption to make. However, this assumption is not sufficient for a certain convergence to the optimal solution of f. To account for this insufficient reduction, the reduction of the function value f should correspond to the directional derivative as well as the step length α_k. $\bigtriangledown f_k^T p_k$ is added to the decrease condition to give the sufficient decrease condition which is presented in eq. (4.15).

$$f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \bigtriangledown f_k^T p_k \tag{4.15}$$

Here f is the objective function, p_k is the direction of the line search and c is a chosen variable. c is defined on the interval [0,1], and is usually quite small in practice ($c = 10^{-4}$)[21]. The method of finding an acceptable interval for α_k for both the decrease in f and the derivative of the direction is illustrated in fig. 4.4.

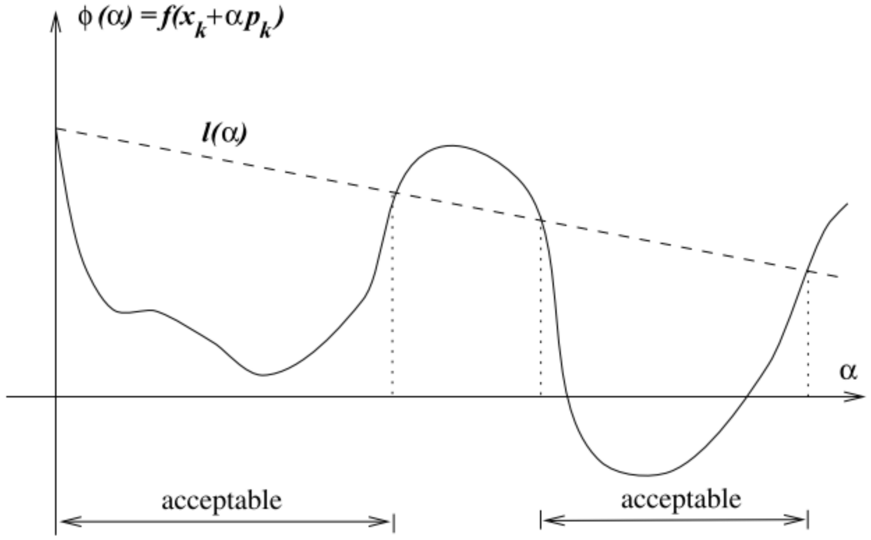


Figure 4.4: An example of the intervals that are acceptable with the sufficient decrease condition. Taken from page 33 in [21].

The sufficient decrease condition is not enough in itself, so other conditions can be added to the problem to ensure that a good value is computed for α_k. However, a backtracking can also be used to avoid adding extra conditions to the line search. The algorithm of sufficient decrease and backtracking taken from [21] is presented below in fig. 4.5.

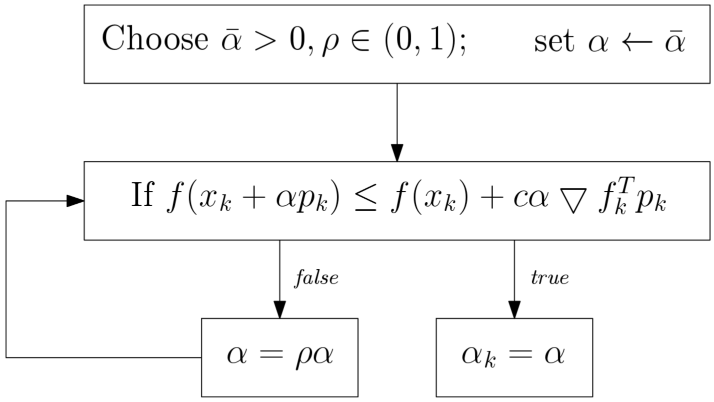

Choose $\bar{\alpha} > 0, \rho \in (0,1)$; set $\alpha \leftarrow \bar{\alpha}$
If $f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \bigtriangledown f_k^T p_k$
false
true
$\alpha = \rho\alpha$
$\alpha_k = \alpha$

Figure 4.5: The algorithm for the line search method. As long as the current step length α does not satisfy the sufficient decrease condition α is updated. When the condition is satisfied the step length for the next iteration α_k is defined as the α from the last iteration.

The backtracking is simply a constant decrease in α for each iteration. The decrease in α is defined by a contraction factor ρ. This means that the appropriate step length will be computed within a finite number of iterations, since the α will eventually decrease to a small enough value. When the step length fulfills the sufficient decrease condition, the current value of α is set as the step length for the next optimization iteration, $\alpha_k = \alpha$

4.6 Augumented Lagrangian

Augmented Lagrangian method is a penalizing method to counter constraint violations. The penalizing function Q is shown in eq. (4.16) where μ is the penalty parameter. Here, f(x) is the original objective function of the OCP and c_i is the i-th constraint function. The method for using the augmented lagrangian is explained in detail in "Numerical Optimiaztion" by Nocedal and Wright in [21].

$$Q(x;\mu) = f(x) + \frac{\mu}{2}\sum_{i\in\epsilon} c_i^2(x) \tag{4.16}$$

The second term of eq. (4.16) penalizes constraint violations by a weight of $\mu/2$, where μ is a chosen parameter that can be tuned to fit the control problem. For the total augmented lagrangian method shown in eq. (4.17), the lagrangian decomposition approach of moving a constraint c_i into the objective function is included as well as the new penalizing term.

$$\mathscr{L}_A(x,\lambda;\mu) = f(x) - \sum_{i\in\epsilon} \lambda_i c_i(x) + \frac{\mu}{2}\sum_{i\in\epsilon} c_i^2(x) \tag{4.17}$$

For the work in this thesis, the constraint function is divided into two separate functions, as described in chapter 3, h_1 from eq. (4.10) and h_2 from eq. (4.12). There is only one constraint function to include in the objective function, so the constraint function c becomes as defined below in eq. (4.18).

$$c = h_1 + h_2 = w_{pg_1} + w_{pg_2} - Q_{g_{max}} \tag{4.18}$$

This yields that the optimization problem with the augmented lagrangian method becomes max $\mathscr{L}_A(x,\lambda;\mu)$. For this model the Augmented Lagrangian becomes eq. (4.19).

$$\mathscr{L}_A(x,\lambda;\mu) = w_{po_1} + w_{po_2} + \lambda(h_1 + h_2) + \frac{\mu}{2}(h_1 + h_2)^2 \tag{4.19}$$

The subproblems are no longer separable due to the last term. Stephen Boyd et. al.[11] proposes to solve this using the Alternating Direction Method of multipliers (ADMM), where each of the subproblems are solved sequentially. In other words, x2 is fixed, then the decomposed system is solved for x1. In the second subproblem, x1 is fixed and the subproblem is solved for x2 beore finally updating the master problem. The method to solving this is explained further in chapter 5. The algorithm for solving the dual decomposition with augmented lagrangian is illustrated below in fig. 4.6. The lagrange multiplier is here updated with the penalty parameter and for each iteration the penalty parameter μ with a scalar β.

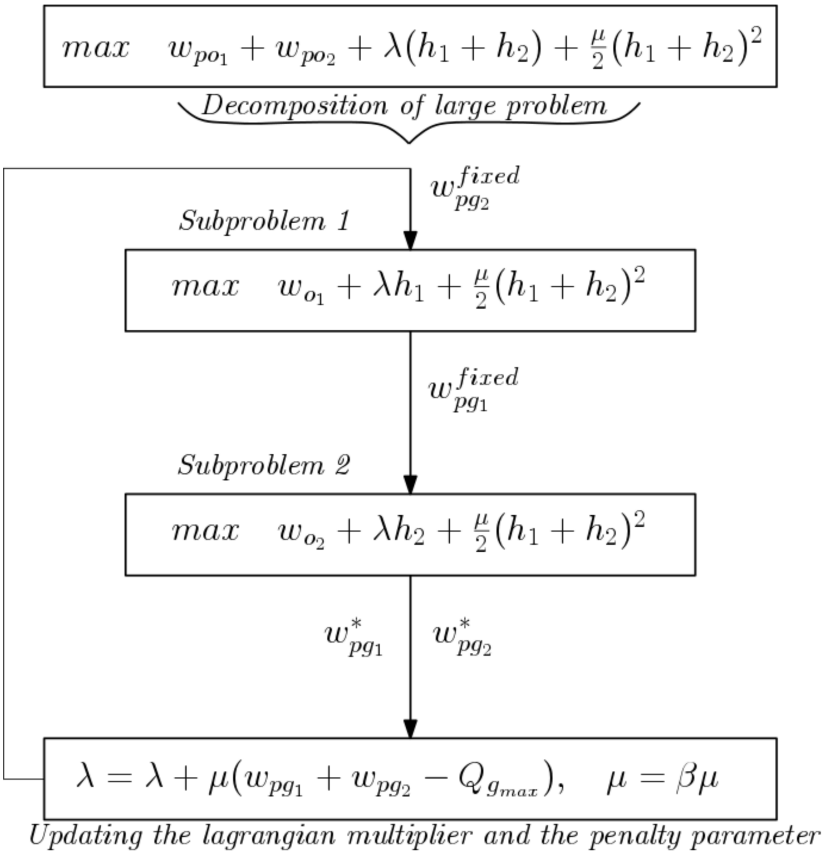

$max \quad w_{po_1} + w_{po_2} + \lambda(h_1 + h_2) + \frac{\mu}{2}(h_1 + h_2)^2$
Decomposition of large problem
$w_{pg_2}^{fixed}$
Subproblem 1
$max \quad w_{o_1} + \lambda h_1 + \frac{\mu}{2}(h_1 + h_2)^2$
$w_{pg_1}^{fixed}$
Subproblem 2
$max \quad w_{o_2} + \lambda h_2 + \frac{\mu}{2}(h_1 + h_2)^2$
$w_{pg_1}^*$
$w_{pg_2}^*$
$\lambda = \lambda + \mu(w_{pg_1} + w_{pg_2} - Q_{g_{max}}), \quad \mu = \beta\mu$
Updating the lagrangian multiplier and the penalty parameter

Figure 4.6: The algorithm for the augmented lagrangian method. The penalty parameter is used to update λ for each iteration. The subproblems are not entirely separable so optimal solution of $w_{pg_1}^*$ and $w_{pg_2}^*$ are given as input for the other decomposition problem in the next iteration.

Chapter 5

Simulation cases

This chapter gives details to how the methods explained previously were implemented for the simulations. The simulations were performed in Matlab using CasADi v3.1.0. For more details into the functions and applications see the CasADi user guide [36]. Some snippets from the Matlab codes are presented in appendix appendix C.

5.1 Centralized NMPC

The model was written as explained in chapter 2 and for use in CasADi. Collocation was used to make the state equations and the constraints into a NLP matrix, that can be solved using IPOPT. The behavior of the system was predicted for 60 sample instants and then optimized. The optimal control inputs u_1^* and u_2^* for the first sample instant were implemented to the plant simulator. The simulator yields measurements of the model states, which were given as input for the next optimization of the prediction horizon. This loop was repeated for a time-horizon, where the simulation result were the past measured behavior of the plant. The NMPC in this thesis has 60 time steps, each representing 5 minutes of operating time. The simulation scheme of the NMPC is illustrated below in fig. 5.1.

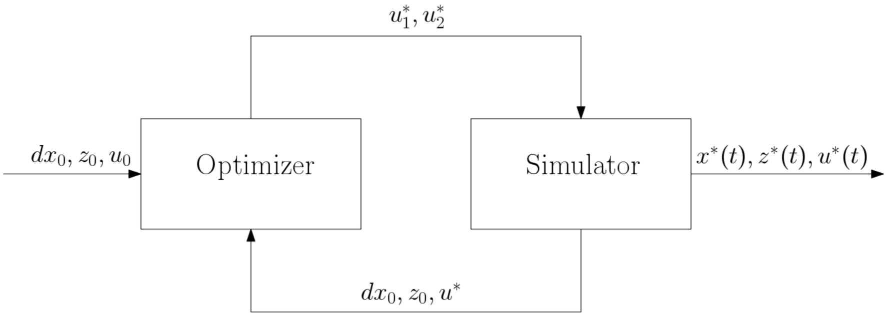


Figure 5.1: The simulation scheme of the NMPC in this thesis. The loop of optimizing the prediction horizon and simulating the plant make up the closed-loop simulation of the NMPC. The initial values dx0, z0, u0 were given as input for the optimizer, and the optimal control input was used to simulate the plant. The simulation measurements were given as initial values for the next optimization. This loop continuous for the given time horizon.

The simulation scheme described above is the same for all the decomposed NMPCs, however, with some modification. The cost function and structure of the initial conditions and bounds matrices are different with the LD decomposition.

5.2 Steady statesimulations

The steady state optimization was simulate to check the results of the decomposed NMPC. The converged values of the control input and the gas and oil production from each well should be to the same values for a decomposed NMPC as found from the steady state optimization of the same system.

5.3 Lagrange Dual Decomposition of the NMPC

The separated wells were modeled as explained in section 2.4. When building the decomposed NLP matrix the largest difference was including the Lagrangian relaxed constraint in the cost function. With the decomposed simulations the updating and iterations to find the correct Lagrangian multiplier λ was added. The algorithm for this loop to find the correct λ is shown in fig. 4.2. This algorithm must be repeated for each optimization of the prediction horizon as shown in fig. 5.1, iterating until reaching a tolerance=0.15 or the maximum iterations cap at 5 iterations. The initial values of the Lagrange multiplier λ and the step length α were chosen for the first iteration. For each prediction optimization, the initial λ was set to be the past value optimal value.

5.3.1 Testing the Decomposed NMPC

The centralized and the decomposed NMPC were simulated for two different constraints being active, the maximum gas capacity and the available gas lift. This was to verify the decentralized model with the centralized model for two different operating conditions. The two runs are presented in table 5.1. Note that when the gas lift available was the active constraint, λ was updated using the gas lift and the gas lift constraint was added to the objective function.

Table 5.1: Overview of the runs performed for the decomposed and centralized NMPC. For run BC.1 and BC.2 the active constraint were respectively the gas capacity and the available gas lift.

Run #	Gas Lift capacity($Q_{GL_{max}}$)	Gas production capcity($Q_{g_{max}}$)
BC.1	5	8
BC.2	3	8

5.4 Line Search Method

The sufficient decrease condition in eq. (4.15) was calculated for each update of λ. The algorithm shown in fig. 4.5 was repeated as long as the condition was not satisfied. The step length α was decreased with a parameter ρ for each iteration. For the practical simulation, the tolerance of the condition was not set to be exactly zero but rather close to zero. The steady state values from the prediction horizons were used to calculate the condition. For each time step implementation α was set back to it's initial value. This was done for α to be large enough to push λ to the optimum for each optimization of the prediction horizon.

5.4.1 Tuning the Parameters c and ρ

Initially the values of c and ρ were guessed with some background in literature[21]. The updating parameter ρ is defined to be on the interval [0,1] and c is said to be quite small($c = 10^{-4}$). The tuning of the parameters were then done by trial and error. The different runs to tune the parameters are shown in table 5.2 For the initial tuning of the parameters, the updating of λ was set to run until the norm of the change in the λ-vector was lower than the tolerance=0.15. In addition a maximum for the update was set to 10 iterations. Three runs were performed without this iteration cap, where two had a new measuring method of the tolerance. For run LS.8 and LS.9 the tolerance was only calculated for change in λ at the first sample instant, tol=$\lambda(1)_{k+1} - \lambda(1)_k$.

Table 5.2: Overview of runs performed to tune parameters in line search method. *The tolerance (change in λ) was calculated using only the first sample instant.

Run #	ρ	c	Tolerance	Iteration Cap
LS.1	0.99	0.1	0.15	10
LS.2	0.99	0.001	0.15	10
LS.3	0.99	0.5	0.15	10
LS.4	0.9	0.1	0.15	10
LS.5	0.7	0.1	0.15	10
LS.6	0.5	0.1	0.15	10
LS.7	0.7	0.1	0.15	–
LS.8*	0.7	0.1	0.15	–
LS.9*	0.7	0.1	0.001	–

5.5 Augmented Lagrangian

The augmented Lagrangian was implemented by adding an augmented Lagrangian part to the cost function and slightly changing the updating method for each optimization iteration. The new cost function for this method is given in eq. (4.19). This objective function is not separable with h_1 and h_2. Because of this it was necessary to make a guess of h_2 for the optimization of well 1, this guess was set to half of the maximum gas production capacity (h_{2_0}=4). For the optimization of well 2 the optimal value of h_1^* from subproblem 1 was given as an input. For the next iterations, the previous optimal solutions h_2^* and h_1^* were given as input for respectively optimization of well 1 and well 2. For the augmented lagrangian method, the step length for updating the Lagrange multiplier λ is the penalty parameter μ. For each update of λ, the penalty parameter is also updating by $\mu = \beta\mu$, where $\beta \geq 1$.

5.5.1 Tuning the Parameters μ and β

The tuning of the parameters μ and β was performed by a trial and error approach. The different runs performed to test the parameters are presented below in table 5.3.

Table 5.3: Overview of runs performed to tune parameters in augmented lagrangian method.

Run #	μ	β	Tolerance	Iteration Cap
Aug.1	0.5	1	0.15	5
Aug.2	1	1	0.15	5
Aug.3	2	1	0.15	5
Aug.4	2	1.01	0.15	5
Aug.5	2	1.03	0.15	5
Aug.6	2	1.05	0.15	5

Chapter 6

Results and Discussion

In this thesis the objective has been to investigate the possibility of a dynamic decomposition for a gas lifted oil well network. The two well system model is described in chapter 2, and the method and theory behind the lagrangian dual decomposition method used for this work is explained in chapter 4. The different simulations performed are outlined in chapter 5, describing the methods and numerical data for each run included in this work.

In this chapter the results from the LD decomposition are presented and discussed. The decomposed NMPC is first numerically validated up against the centralized NMPC for two different active constraints, before the deviation from the decentralized control structure is discussed in detail. When presenting the results, only the most central data is presented in tables and plots as a background for discussion. Two numerical methods were applied for improving the results of the decomposition, line search method and augmented lagrangian decomposition. The results of the best run for each numerical method are first presented and the run is compared to the base case decomposition and the centralized NMPC. Then the results from all runs tuning the parameters in the numerical methods are presented and the reason for choosing the best run is explained. Finally, the two numerical methods are compared and discussed up against each other.

When interpreting the results, two major parts of the simulation results that are analyzed. The first one is the past time horizon simulations, the simulated behavior of the system. The second is the optimized prediction horizon, the part of the NMPC that attempts to predict future behavior of the system. This predicted future is optimizes the measured variables by adjusting the prediction horizon of the controlled variables. For this whole chapter it is important to understand the theory behind NMPC described chapter 3 and how this theory has been applied to the model in this system.

6.1 The Lagrange Dual Decomposition Simulations

When determining the success of the decomposed control system, it needs to be compared to a centralized control system. The aim of using the method is to obtain the same result, just with a more decentralized structure with subproblems that are coupled by a master control. Therefore, the results of the decomposed NMPC are compared to a centralized NMPC and a steady-state optimization for the same two well system. The difficult part of the decomposed simulations were expected to be the initial dynamics, so this is studied in detail. The prediction horizon that is optimized for each time step is also studied for each decomposed run. For each prediction horizon the lagrange multiplier is updated until reaching convergence, changing less than a certain tolerance for each update. This iteration of the prediction horizon with λ is only performed for the decomposed solutions.

The base case decomposition simulation is explained in chapter 5 and the results are shown below in fig. 6.1. The total gas production rate and gas rate for each well are plotted in the top left plot, and the oil production rate for the same combination are shown in the top right plot. The gas lift inlet to each well is shown in the bottom left plot and the gas-oil ratio (GOR) for each well are shown on the bottom right. For all the plots the blue lines represents well 1, while the red lines represents well 2. The black line represent the total of both wells. The simulation results will be further investigated and discussed below.

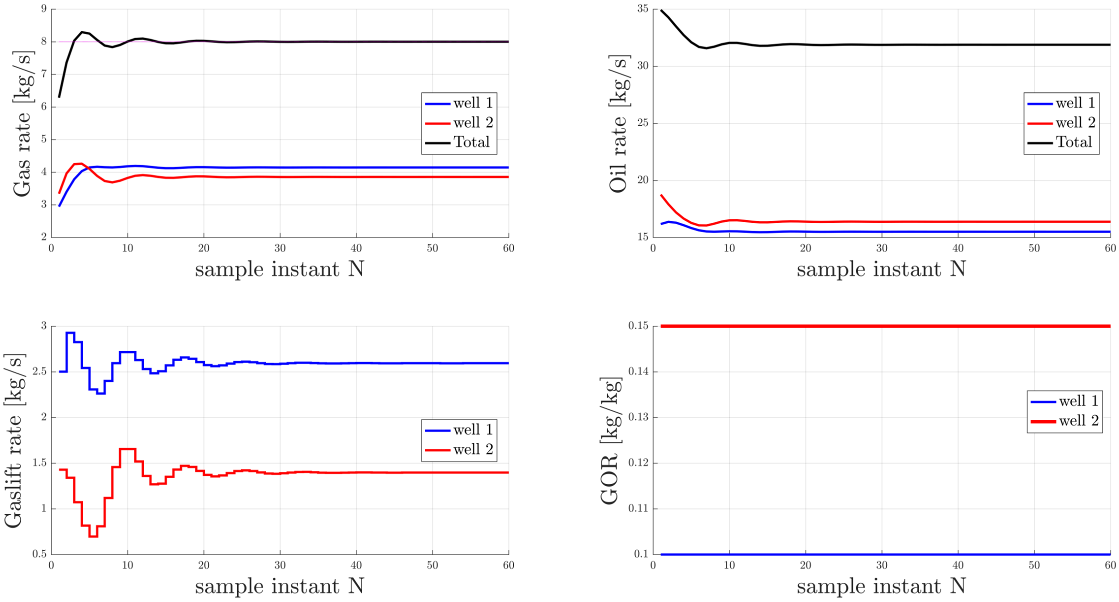


Figure 6.1: The general simulation results for the base case of the decomposed NMPC. The gas rate and oil rate for each well, the gas lift injection rate for both wells and the gas-oil ratio(GOR) are plotted for each time step simulated. The blue and red lines refer to well 1 and well 2, while the black lines are the total of both wells.

An important thing to note is that the gas production rate of the combined wells has reached maximum capacity, making the common constraint active and the solution optimal. From the plots in fig. 6.1, it is observed that the simulation oscillates before finding it's stable steady state value. The oscillation is caused by the dynamics in the start up of the simulation, this is one of the challenges mentioned in the steady state decomposition paper[17] and discussed in section 4.1. This initial dynamics part is of interest when investigating the decomposed NMPC further. But first the steady state values of the decomposed NMPC needs to be validated to assure that the simulation yields the same optimal solution as the centralized NMPC.

6.1.1 Validating with Gas Production Rate Capacity as Active Constraint (Run BS.1)

When validating the decomposition method, the first confirmation is if the decomposed NMPC converges to the same values as the equivalent centralized NMPC. The values found for the gas lift rate and the production rates of gas and oil for both wells are checked. As shown below in table 6.1, the results from the base case decomposed NMPC coincide with the results for a centralized NMPC and the steady state optimization. From the table it can be concluded that the decomposed NMPC simulates the same gas lift allocation as the centralized solution, and therefore decomposition might be possible for the two-well gas lifted system modeled in this work.

Table 6.1: The converged simulation values for the base case decomposed NMPC, the centralized NMPC and the steady state optimization of the model used in this work. Here the gas capacity constraint is active.

	w_{gl_1}	w_{gl_2}	w_{pg_1}	w_{pg_2}	w_{po_1}	w_{po_2}
Decomposed NMPC	2.595	1.398	4.145	3.855	15.50	16.38
Centralized NMPC	2.595	1.398	4.145	3.855	15.50	16.38
Steady State Optimization	2.595	1.398	4.145	3.855	15.50	16.38

From this it can be confirmed that the dynamic decomposition converges to the same result as the global controller. However, the interesting part of the decomposed NMPC is the initial dynamics and the λ-prediction horizon coupling the constraints. The fact that the steady state decomposition can be performed for this type of system has been confirmed before in previous work[16][17]. The dynamic part is, as expected, the difficult part and to illustrate this the decomposed case is plotted against the centralized case below.

The decomposed NMPC is plotted against the simulation of the centralized NMPC in fig. 6.2. The simulations are plotted over the time-horizon, consisting of past measured behavior in negative values from -300 to 0, and the predicted optimized behavior 300 minutes into the future. The gas lift injection rate and the gas production rate from the wells are both plotted for this time horizon. For the decomposed NMPC the lines are solid, where well 1 is blue, well 2 is red and the total gas rate is black. For the centralized NMPC the lines are dashed and the orange represents well 1, the light blue represents well 2 and the gray dashed line is the total gas rate. For the predicted future values the colors are the same but the lines overlap so much that there is very little visible differences.

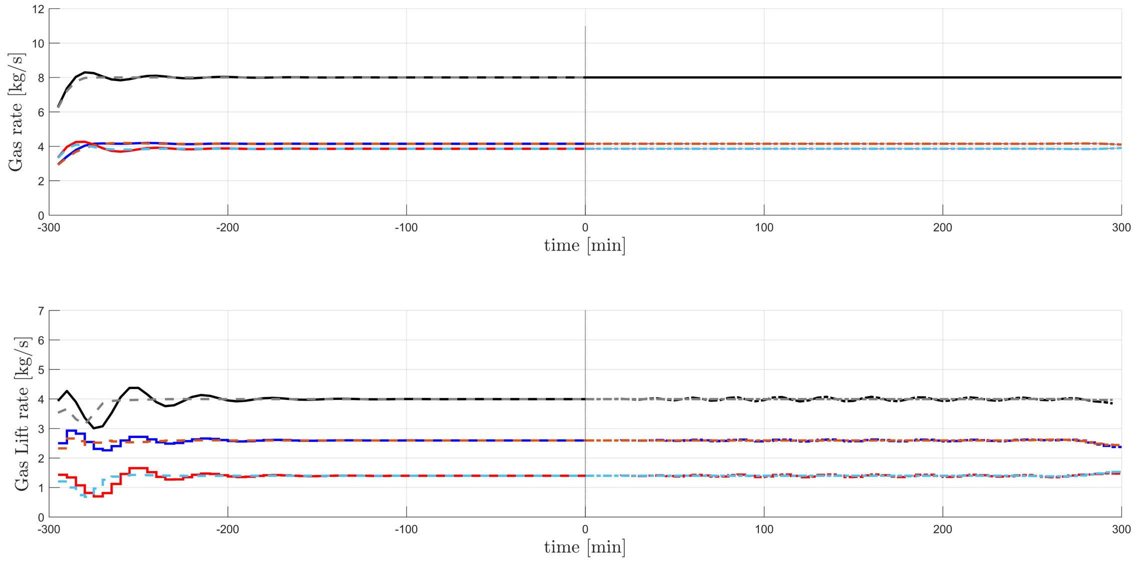


Figure 6.2: The simulation of the base case of the decomposed NMPC plotted against the centralized NMPC for comparison. The past simulation results are plotted from time -300 minutes to the present value of the system. The prediction horizons are plotted 300 minutes into the future. The solid lines represent the decomposed NMPC, where the blue, red and black are respectively well 1, well 2 and both wells combined. The dashed lines represent the centralized NMPC, where the orange, light blue and gray are respectively well 1, well 2 and both wells combined.

Figure 6.2 is included to illustrate how much the two control systems overlap. After the first 150 minutes, there is almost no visible deviation and that coincides with the converged numerical values being the same, as shown in table 6.1. However, it is also clear that the trajectory for the base case decomposed NMPC is not overlapping with the centralized NMPC for the initial part of the simulation. This is due to the fact that the gas capacity constraints is not active during the initial time steps. This will be further discussed in section 6.1.3.

6.1.2 Validating with Gas Lift Available as Active Constraint (Run BS.2)

The same base case decomposition can be validated a second way, by changing what the active constraint is. This was done by changing the maximum available gas lift value, so that this constraint is reached before the constraint on gas rate capacity. This gives another assurance that the decomposition was performed correctly and that it produces the same optimal value as the centralized NMPC would do. In this case, the input constraints are always active and therefore the trajectory should be the same.

The values are presented in table 6.2, and the comparison of the simulation values show that the decomposed NMPC has converged to the same result as the two other methods. The steady state optimization and the centralized NMPC simulations are performed with the gas lift as active constraint. There are some insignificant differences, but the conclusion is that the decomposed NMPC can replace the centralized NMPC and produce the same results.

Table 6.2: The converged simulation values for the base case decomposed NMPC, the centralized NMPC and the steady state optimization of the model used in this work. Here the available gas lift constraint is active.

	w_{gl_1}	w_{gl_2}	w_{pg_1}	w_{pg_2}	w_{po_1}	w_{po_2}
Decomposed NMPC	2.087	0.913	3.623	3.348	15.35	16.23
Centralized NMPC	2.088	0.912	3.623	3.348	15.35	16.23
Steady State Optimization	2.087	0.913	3.623	3.348	15.35	16.23

To compare all of the simulation values, the decomposed NMPC is plotted against the simulation of the centralized NMPC in fig. 6.3. For these two solutions the gas lift is both the constraint and the manipulated variable(MV), so the gas production rate is not included. The only simulation included in this plot is the gas lift injection rate to each well and the total gas lift allocated. The colors are the same as previously used, where blue, red and black represent the simulations of well 1, well 2 and total gas lift for the decomposed NMPC. The dashed lines represent the centralized NMPC, where orange is well 1, light blue is well 2 and gray is the total gas lift.

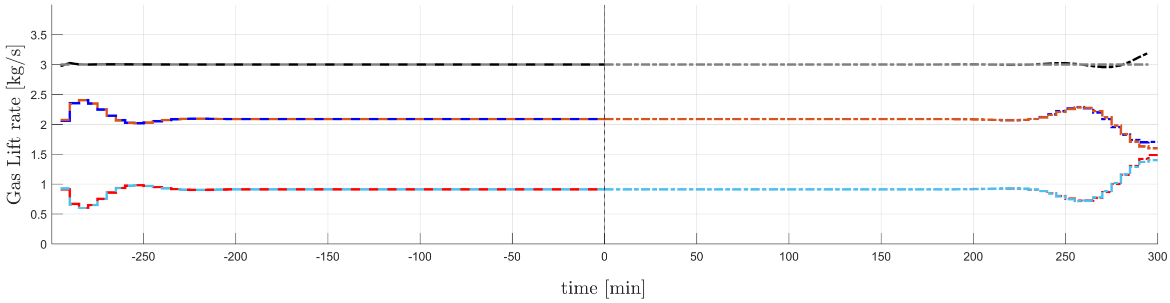


Figure 6.3: The decomposed NMPC where the gas lift available is the active constraint is plotted with the centralized NMPC. The past time horizon, from -300 to 0 minutes, and the predicted future horizon, from the present 0 minutes until 300 minutes, for the gas lift for each well is presented. The solid lines represent the decomposed NMPC, where the blue, red and black are respectively well 1, well 2 and both wells combined. The dashed lines represent the centralized NMPC, where the orange, light blue and gray are respectively well 1, well 2 and both wells combined.

The simulations have an almost perfect overlap, as expected the initial part of the decomposed simulation does not deviate from the centralized simulation. It is not necessary to study the initial trajectory for this simulation, as they overlap. However, it is interesting to study the prediction horizons over different time steps to see what is different for this active constraint versus the previous gas rate active constraint. This is done below.

Because the overlap is so good with available gas lift as the active constraint, the dynamic decomposition might be a very suitable solution for some optimal control problems. For example, an optimization problem where there is a limitation in gas lift capacity.

6.1.3 The Base Case Decomposition Results

After having validated the converged value, the results of the base case decomposition are presented and discussed further. The most central numerical results are presented in table 6.3. As seen above, the total gas production rate is observed to oscillates around the constraint value, before stabilizing completely at the constraint value around N=27. For each oscillation the constraint violation decreases significantly, the second violation is only a third of the first violation value. The maximum constraint violation is given in the table below.

The other values given in table 6.3 refer to the optimized prediction horizon and it's properties for the last time step. For each time step the prediction horizon was optimized by updating λ. The base case λ never reaches the set tolerance, therefore the updating of λ was capped for each time step at five iterations. The total number of iterations are given as a reference in the table, kept in mind that this was capped without convergence of the λ prediction horizon.

Table 6.3: The central results from the base case decomposed NMPC. The maximum constraint violations and the oscillation values of the last prediction horizon are presented, as well as the total iterations for updating λ.

Constraint violations	$\Delta_{max}\lambda$	$\Delta_{min}\lambda$	$\Delta_{max}GL$	Total iterations
0.295	16	5	0.09	300

The difference in λ-values over the prediction horizon are given as the maximum ($\Delta_{max}\lambda$) and minimum ($\Delta_{min}\lambda$) difference for one oscillation. The largest oscillation over the gas lift prediction horizon ($\Delta_{max}GL$) is also given. Even though the λ and gas lift rate prediction horizons have not converged, the gas rate prediction for both wells has converged to optimal solution.

For steady state optimization, the lagrange multiplier for the coupling constraint is a scalar, where as for dynamic decomposition the lagrange multiplier $\lambda(k)$ is a vector over the prediction horizon. For the prediction horizon of the MVs to converge, λ needs to converge. As long as the λ prediction horizon oscillates the control system has not reached a stable solution.

The prediction horizon attempts to predict the trajectory of the control input needed to reach an optimal solution. For each time step, the predicted future is optimized for a certain horizon.

For the decomposed NMPC the optimization of the prediction horizon is done by updating λ. The update of λ and the prediction horizon is explained in more detail in chapter 4, but briefly explained: the λ for each iteration is updated with the error between the predicted values and the optimal solution. The prediction horizon should converge to a stable solution when optimum is reached in the simulation, however, for the base case this does not happen.

To further illustrate the oscillations, the last optimized prediction horizon is shown in fig. 6.4. The predicted behavior of the gas lift injection rate and the gas rate from each well and λ are presented for N sample instant into the future. The prediction horizon in fig. 6.4 clearly shows that λ oscillates.

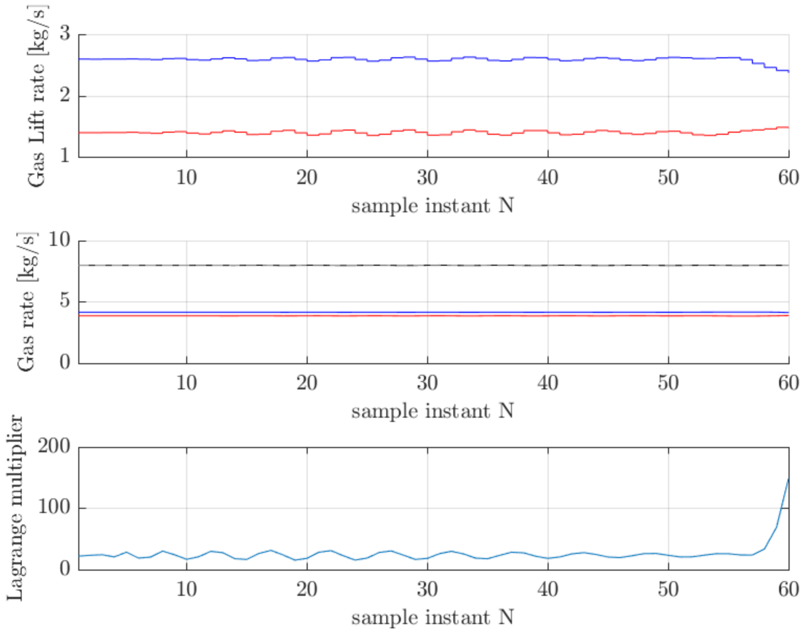


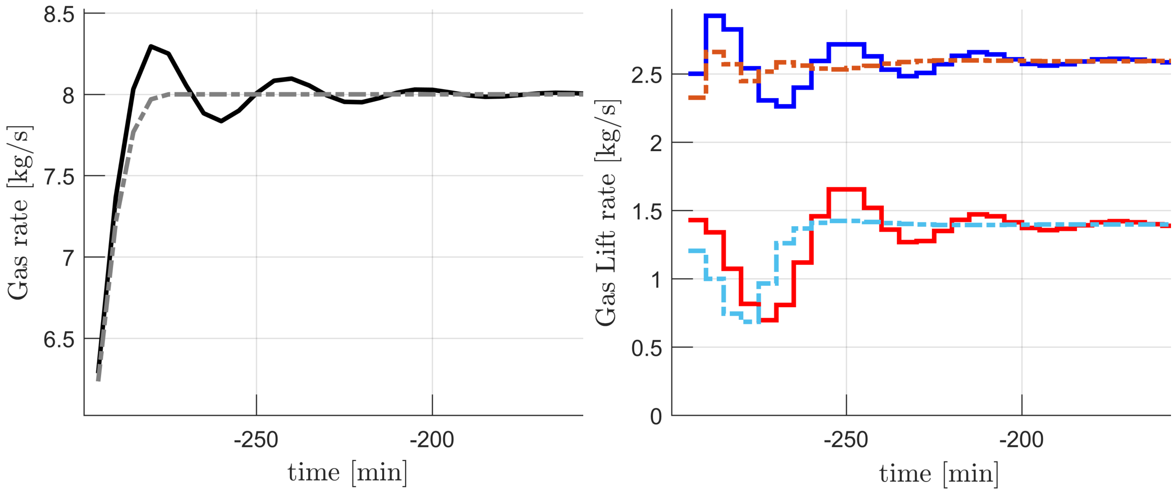
Figure 6.4: The last optimized prediction horizon for the base case decomposition. The prediction horizon of gas lift, gas rate and λ are plotted over the sample instants N.

The fact that λ does not converge to approximately the same value over the prediction horizon shows that the control system is not fully stable. The line search method, which decreases the step length as λ gets closer to the optimal solution, might help improve this oscillation and help λ converge to a more stable value. This will be the focus when tuning the line search.

At the end of each prediction horizon the values have a sudden increase. This can be seen for all prediction horizons in this work and the cause is the same. This increase could easily be removed by adding a terminal constraint to the simulation constraints, setting the last value to be the converged value of the prediction horizon. This is a normal constraint to add in a predictive controller to achieve full closed-loop stability, however, in this work, close-loop stability is ensured by using a long prediction horizon[37].

6.1.4 The Initial Dynamics

As mentioned above, the difficult part of a dynamic decomposition is the initial start-up of the simulations. In this section the initial dynamics of the base case decomposed NMPC is compared to of the centralized NMPC. Since the constraint is on the total gas rate, this is the only part focused on when comparing the initial trajectory of the dynamic decomposition. The MV is the gas lift rate, therefore the total gas lift used is not included in the comparison. The plotted values of the initial dynamics is shown in fig. 6.5, on the left side in fig. 6.5a the black trajectory represents the base case decomposition, while the gray line represents the centralized NMPC. The right fig. 6.5b represent the gas lift rate trajectories, where the blue and red are well 1 and well 2 in the base case decomposed NMPC, and the orange and light blue are well 1 and well 2 for the centralized NMPC.



(a) The initial dynamics simulation of the gas production rate for the first time steps.

(b) The initial dynamics simulation of the gas lift input to each well for the first time steps.

Figure 6.5: The first part of the simulations up-close, to illustrate the deviation between the simulation trajectories of the decomposed NMPC and the centralized NMPC. The solid lines represent the decomposed NMPC, where the blue, red and black are respectively well 1, well 2 and both wells combined. The dashed lines represent the centralized NMPC, where the orange, light blue and gray are respectively well 1, well 2 and both wells combined.

It can be observed that the decomposed NMPC oscillates around the trend of the centralized solution follows. This shows that the simulation is just unstable and not incorrect. The trajectory for the decomposed solution violates the constraint with a peak at time step four, with a maximum value of 0.295 kg/s above the total gas rate constraint. For applications where the constraints are soft, this is no problem. However, if the constraint coupling the subproblems is a hard constraint this might be an issue for the dynamic decomposition. In the simulations with the numerical methods the constraint violations were sought decreased. However, there will always be an evaluation of the cost of constraint violations against the saved cost of the decomposed control structure.

When comparing the initial dynamics to the predicted λ there is an obvious correlation. For the first four time-step, the prediction horizon of λ is zero for the first sample instant, meaning that the λ is not able to "push" for the optimal solution and the constraint is not coupling the decomposed subproblems. As λ is giving the subproblems a common master constraint, it is natural that the simulation solution is not pushed towards the active constraint when λ is equal to zero. It is clear that λ pushes the convergence of the simulations and that the improvement of the λ-prediction could improve the dynamics of the decomposed NMPC. This is illustrated by the optimized prediction horizon presented and discussed for different time steps below.

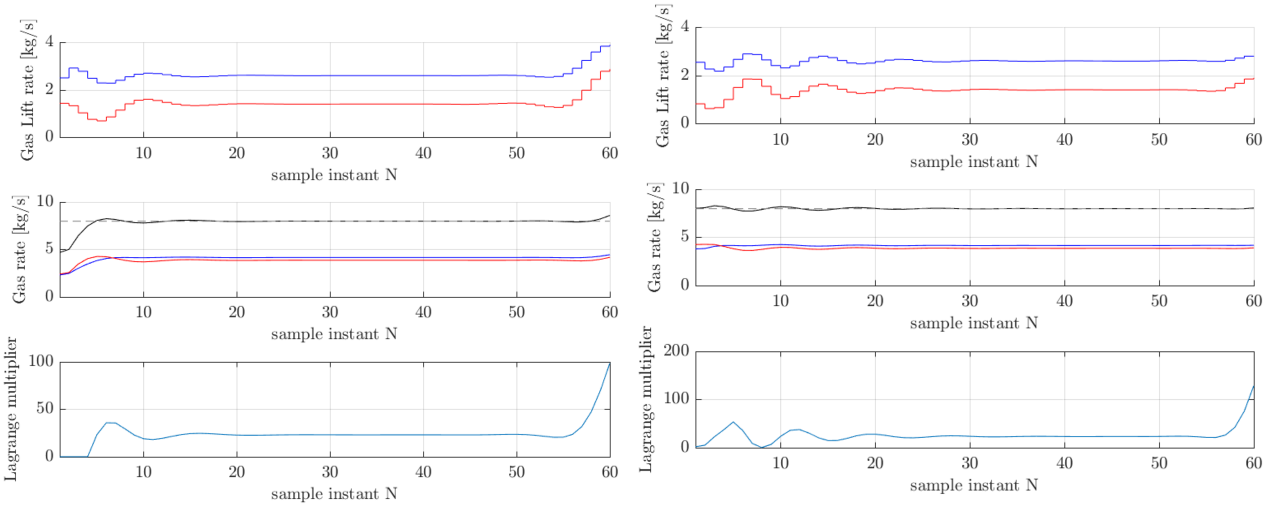
Therefore, the conclusion is that, when the coupling constraint is not active during a short period, the lagrange multiplier is zero and is unable to coordinate the subproblems during these time steps.

6.1.5 Optimization of the Prediction Horizon for Gas Capacity as Active Constraint

When looking at the prediction horizon, the important things to notice are the first sample instant of the horizon and the general trend. When the simulations has reached it's optimum and is stable, the prediction horizon should be converged to the steady state value and not oscillate.

The left fig. 6.6a pictures the first optimized prediction horizon after updating λ 5 times. The value of the lagrange multiplier does not converge, and for the first few sample instants the predicted values of λ are equal to zero. When λ is zero the common constraint is not active, therefore not pushing the gas rate to reach the active constraint. The lack of push from the lagrange multiplier can also be seen in the prediction horizon, as the total gas rate is less than the constraint value for the first few sample instants.

The first predicted value of λ is zero for the first three time optimized prediction horizons, or in other word first three time steps. At the fourth prediction horizon, included on the right in fig. 6.6b, the λ prediction is non-zero for all sample instants. At this point, the decomposed NMPC starts to push the simulation against the active constraint. For the fourth optimization of the prediction horizon λ is active and so the predicted horizon of the total gas rate is much closer to the optimal value.



(a) The optimized prediction horizon for the first time step in the base case decomposed NMPC.

(b) The optimized prediction horizon for the fourth time step in the base case decomposed NMPC.

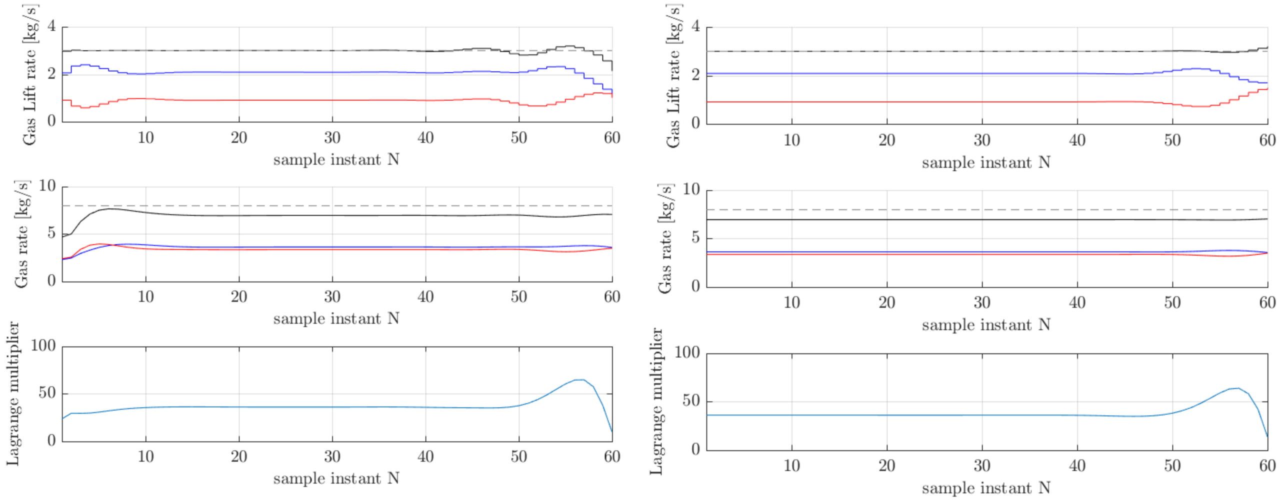
Figure 6.6: The prediction horizons for the base case of gas lift rate in each well, the gas production rate for each well and the collective gas rate, and also the optimized lagrange multiplier after 5 updates are plotted over the horizon of sample instants N. The red lines represent well 1 and the blue lines represent well 2.

As mentioned, the fact that the constraint is violated during the initial period of the simulation is due the coupling constraint not being active for the first few time steps. And the oscillations in the optimized value of the λ can be connected to the oscillations in the simulation of the total gas rate. The observation is that when the coupling constraints are not active, the master problem in unable to co-ordinate the sub problems.

Although the lagrange multiplier has become active after the fourth time step, the value of λ still doesn't converge. This is an issue for the stabilization of the solution. The difference between the λ values vary a lot and does not stabilize, the oscillation values are shown in table 6.3. The gas lift also oscillates with a large difference and does not converge to one value, although the produced gas for both wells w_{pg_1} and w_{pg_2} does converge to the correct value.

6.1.6 Optimization of the Prediction Horizon for Available Gas Lift as Active Constraint

Figure 6.7 shows a comparison of the first and the last prediction horizon for the decomposed NMPC when the available gas lift is the active constraint. Here it can be observed in fig. 6.7a that λ is active and at a value close to optimum at the last prediction in fig. 6.7b. The fact that λ pushes the solution towards the constraint from the beginning and does not oscillate is consistent with the results from the simulation. The total gas lift rate of both wells combined is also close to it's optimum solutions for the first optimized prediction horizon, and the first gas lift rate prediction is the same as the trajectory in the simulations.



(a) The optimized prediction horizon for the first time step in the decomposed NMPC.

(b) The optimized prediction horizon for the last time step in the decomposed NMPC.

Figure 6.7: The optimized prediction horizon for the first and last time steps in the simulation of the decomposed NMPC with the gas lift available as the active constraint. The gas lift rates, the gas production rates and the coupling variable λ are optimized for a future prediction horizon. The red lines represent well 1 and the blue lines represent well 2.

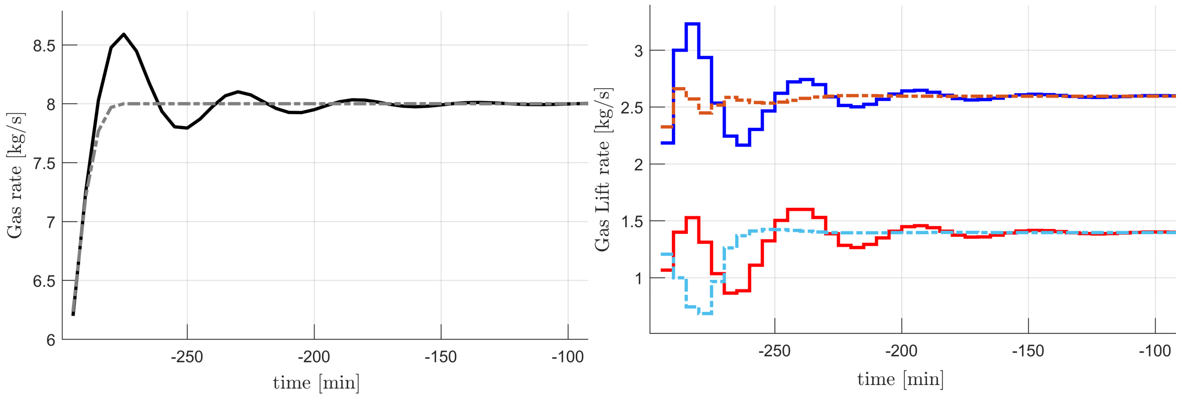
6.2 Implementing the Line Search Method

The line search method is described in chapter 4, in short the method finds a suitable step length for each iteration as the solution gets closer to the optimum. Adding the line search method to the base case decomposed NMPC has two advantages. The λ-prediction horizon is able to converge to a certain tolerance, and therefore there is no need to take a short-cut in capping the iterations of updating λ. The line search method also makes it possible to iterate closer to the optimal solution for each time step, in effect removing the oscillations seen for the λ-horizon in the base case. For the right tuning, the iterations needed to obtain a converged λ can be decreased. The saved computational time to obtain a good solution might be economical for some systems. However, the constraint violations increase as the iterations needed decreases. When discussing computational time in this work, it is measured in the amount of total iterations.

In the base case of the decomposed NMPC, the λ is each time updated with the same step length, $\alpha = 10$. For the line search decomposed NMPC the sufficient decrease method combined with the backtracking method was used to update the step length for each update of λ. The runs performed for the line search methods are presented in table 5.2 and described in chapter 5. The results from the best run using the line search method is first presented and compared to the centralized NMPC, then compared to the base case decomposition. After the results from all the runs are presented in three parts, the two first discusses the tuning of the parameters in the line search method. The last part presents runs performed without an iteration cap and with a different measurement of tolerances for updating of the λ-prediction.

6.2.1 Comparing the Line Search Decomposed NMPC with the Centralized NMPC

The best tuned line search method is found to be run LS.8 and it is compared to the centralized NMPC below. The interesting part to study is the initial dynamics of both NMPCs, shown together in fig. 6.8. As the objective of the line search is to decrease the needed iterations and converge the λ prediction horizon, a successful result is not measured in the constraint violations. However, this merged plot is included to illustrate how much more the decomposed NMPC violates the constraint when the line search has been used. The colors and line types used for the comparison are the same as used above, where the decomposed NMPC with the best line search run is represented by the solid lines and the centralized NMPC is represented with the dot-dashed lines.



(a) Total gas rates compared for Line Search method and the centralized NMPC.

(b) Gas lift rates compared for Line Search method and the centralized NMPC.

Figure 6.8: The first part of the simulations up-close, to illustrate the deviation between the simulation trajectories of the line search method and the centralized NMPC. The solid lines represent the best line search run LS.8, where the black is the total gas rate, the blue and the red represents the gas lift rate in well 1 and well 2. The dot-dashed lines represent the centralized NMPC, where the gray is the total gas rate, the orange and the light blue represents the gas lift rate in well 1 and well 2.

For both the total gas rate and the gas lift inlet rate, the trajectories of the decomposed NMPC with line search is still different from that of the centralized solution. It can be concluded that the deviation from the centralized NMPC are larger with the line search method. It is therefore more interesting to look at the improvements of line search, by comparing the best run of the line search method to the base case decomposed NMPC.

The converged values for including the line search method is not presented in a table, however, the values for the line search decomposed NMPC were confirmed with the centralized NMPC and the steady state optimization values presented in table 6.1.

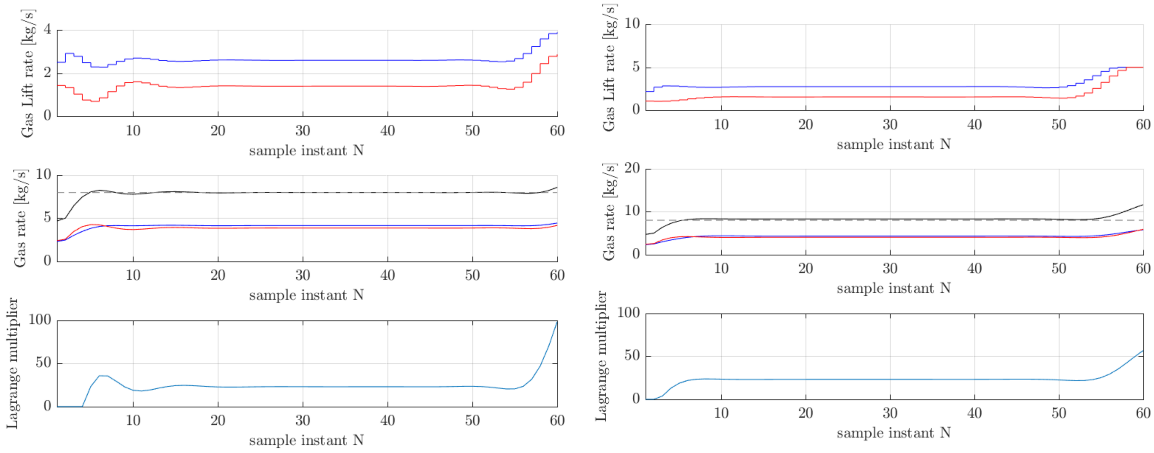
6.2.2 Comparing the Line Search Decomposed NMPC with Base Case Decomposition

The central results to compare for the line search method are presented below in table 6.4. As well as the total iterations and the convergence of λ, the constraint violations are also compared. One of the biggest issues with the base case decomposed NMPC is that the prediction horizon of λ oscillates, not able to reach the tolerance criteria. With the line search method this is not the case. λ converges and the method is also able to decrease the total needed iterations to get to that convergence. When using the line search method to save computational time, the cost is in a larger constraint violation.

Table 6.4: The central results from using the line search method compared to the base case decomposition. The maximum constraint violations and the oscillations of the last prediction horizons are presented, as well as the total iterations for updating λ.

Sim	Constraint violations	$\Delta_{max}\lambda$	$\Delta_{max}GL$	Total iterations
Run LS.8	0.590	0	0.01	158
Base case	0.295	16	0.09	300

To illustrate the differences in convergence, the optimized prediction horizons for the first time step are included in fig. 6.9. On the left in fig. 6.9a, the first prediction horizon for the base case decomposed NMPC is included. On the right in fig. 6.9b, the first prediction of the best line search run LS.8 is included. The obvious difference between the two prediction horizons is that the one oscillation in the base case is not present for the line search method. Another difference is the prediction of the gas rates for the line search, the combined produced gas rate is a little bit above the constraint value, marked with a dashed line. This does not effect the converged simulation result but might be a sign of why the constraint violation increases for the line search method.

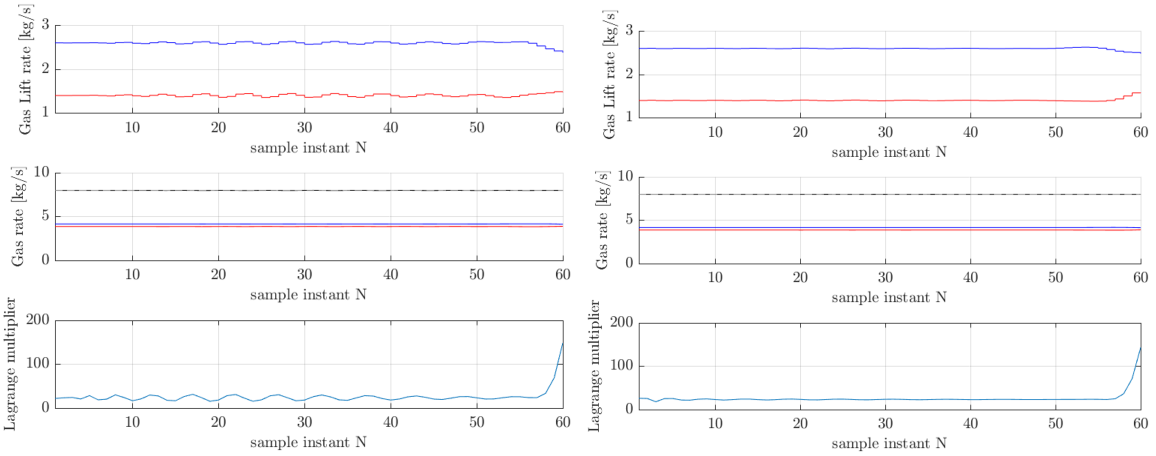


(a) The optimized prediction horizon for the first time step in the base case decomposed NMPC.

(b) The optimized prediction horizon for the first time step in the line search decomposed NMPC.

Figure 6.9: The first optimized prediction horizon of the decomposed NMPC with and without the line search method. The red lines represent well 1 and the blue lines represent well 2. The black lines represent the total gas rate.

The last optimized prediction horizon is more interesting. Here it can be seen that the oscillations in the lagrange multiplier have diminished when using the line search method. The oscillations in the gas lift rate prediction are also almost gone. The difference is a key advantage to including the line search method in the decomposition.



(a) The optimized prediction horizon for the last time step in the base case decomposed NMPC.

(b) The optimized prediction horizon for the first time step in the line search decomposed NMPC.

Figure 6.10: The last optimized prediction horizon of the decomposed NMPC with and without the line search method. The red lines represent well 1 and the blue lines represent well 2. The black lines represent the total gas rate.

This lack of oscillation comes from the tuning of the parameter c for the line search method. When adjusting the line search method this parameter is first tuned, and then the optimal value found is kept constant while the parameter ρ is tuned. The tuning of line search that affects the number of iterations needed is the tuning of the updating parameter ρ.

The line search lessens the iterations needed and the oscillations in λ are reduced, but at the cost of a larger constraint violation. The advantages and disadvantages are discussed more below, when presenting the results of the parameter tuning.

6.2.3 Tuning the Parameter c

The parameter c is a part of the inequality that must be fulfilled in the sufficient decrease method presented in chapter 4. The criteria for ending the update of the step length α is very dependent on the value of this parameter. The different values of c for the tuning runs are presented in table 5.2 in chapter 5. When tuning the parameter c the prediction horizon, in particular the oscillations of λ, is the part of the result presented and discussed. The constraint violations are presented numerically and the number of iterations and change in the step length α is presented in the appendix D for more details.

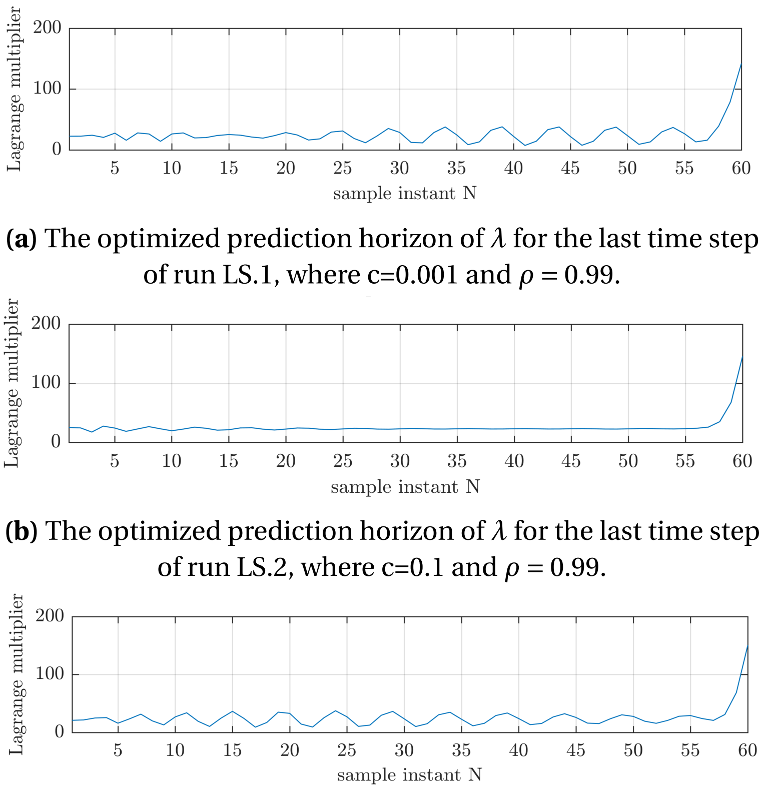
The most important results from the tuning of the parameter are investigated and summed up in table 6.5. Three different runs are performed where the parameter c is varied, while the other parameter ρ is kept constant. The oscillations of the lagrange multiplier is presented in both the $\Delta_{max}\lambda$, the maximum change in the λ-values over one of the oscillations, and $\Delta_{min}\lambda$ the minimum change of λ over an oscillation. The maximum change in the prediction horizon of the MVs, gas lift rates for each wells, is also presented for the three different runs. The constraint violations for each run is also included, as this can be the trade off for converging λ.

Table 6.5: The most central results for the runs performed for tuning the parameter c in the decomposed NMPC with the line search method. The parameter c is varied with the values of 0.001, 0.1 and 0.5 for respectively LS.1, LS.2 and LS.3. The other parameter ρ is kept constant at 0.99.

Run #	$\Delta_{max}\lambda$	$\Delta_{min}\lambda$	Δ_{max}GL	Constraint violations
LS.1	31	5	0.16	0.21
LS.2	10	0(converged)	0.02	0.36
LS.3	27	10	0.13	0.50

From the results presented in table 6.5 it is clear that run LS.2, shown in fig. 6.11b, has the best value for the parameter, c=0.1. Run LS.2 is the only of these three simulations that has a lagrange multiplier and MVs that converges for the last optimized prediction horizon. Although both run LS.1 and run LS.3 don't converge for λ, the constraint violation are very different. From the results it seems that the constraint violations increase with the increase of c. This can be connected to the small step length value that α iterates to, when the parameter c has a large value. When α decreases fast, the update of λ for each iterations is smaller and therefore slower. When λ is slower in getting to the optimal value, the constraint violation becomes larger. The updated value of α for each time step is shown in appendix D, and are consistent with the trend explained above.

To get a better idea of the difference in λ prediction horizons of each run, they are included below in fig. 6.11. The last optimized prediction horizons of λ is presented for each of the three runs. The prediction horizon of the gas lift rate and the gas production rate are not included, seeing that λ is the best representation for the convergence. From these plots, it is clear that the differences when tuning the parameter c are large. The oscillations are large for both in run LS.1 and run LS.3, but for the second run the horizon of λ has converged.



(a) The optimized prediction horizon of λ for the last time step of run LS.1, where c=0.001 and $\rho = 0.99$.

(b) The optimized prediction horizon of λ for the last time step of run LS.2, where c=0.1 and $\rho = 0.99$.

(c) The optimized prediction horizon of λ for the last time step of run LS.3, where c=0.5 and $\rho = 0.99$.

Figure 6.11: The comparison of the λ prediction horizon over sample instants N for the last time step. The runs are performed for tuning the parameter c.

Runs were also performed for the same values of c but with a smaller value of $\rho = 0.9$. These showed no difference in performance from the results presented above and are therefore not included in this work. It does however, support the result that $c = 0.1$ is the correct value of the parameter.

From this it can be concluded that with the right tuning, the oscillations observed for the lagrange multiplier in the prediction horizon can be removed by using the line search method. The method adjusts the time step as the solution nears the optimum, making it easier to reach the optimal solution. However, as the correct value of the parameter c is in between two values, it can be assumed that the parameter can be hard to tune correctly. For a system with hard constraints this is not a good method, as also the constraint violation increases compared to the base case decomposition.

Since only run LS.2 converges, the number of iterations are not discussed. Run LS.1 and run LS.3 always iterate until reaching the cap set to 10 iterations. Therefore the total number of iterations will only be discussed for the next runs in the line search method. When tuning the updating parameter ρ, the main measure of good tuning is the total number of iterations needed for reaching the optimal solution. The simulation trajectory is also not focused on, because the simulations with the line search method are similar to the base case decomposition and each other.

6.2.4 Tuning the Parameter ρ

The parameter ρ is the updating parameter of the step length α, explained in chapter 4. The value of the parameter determines how fast the step length decreases when trying to satisfy the sufficient decrease method. The larger ρ is, the more aggressive is the line search method. When tuning the updating parameter ρ, the interesting result to look at is the iterations used for each optimization of the prediction horizon over the entire simulation. However, there are disadvantages to tuning the method too aggressively, like larger constraint violations.

The runs where the parameter c is kept constant and the ρ is varied are presented in table 5.2 in chapter 5. The parameter c is set to 0.1, the optimal value determined above. As mentioned previously, ρ affects the number of iterations needed for λ to converge. When presenting the most important results, this is the main focus. The rest of the results are included in appendix D for more details.

The most central results are summed up in table 6.6, showing the constraint violations for each run, as well as the number of iterations for the last few time steps. The number of times steps it takes before the iterations to not reach the cap at 10 iterations and the total number of iterations for the whole are also included. The three last values mentioned are included to compare iterations of the different runs.

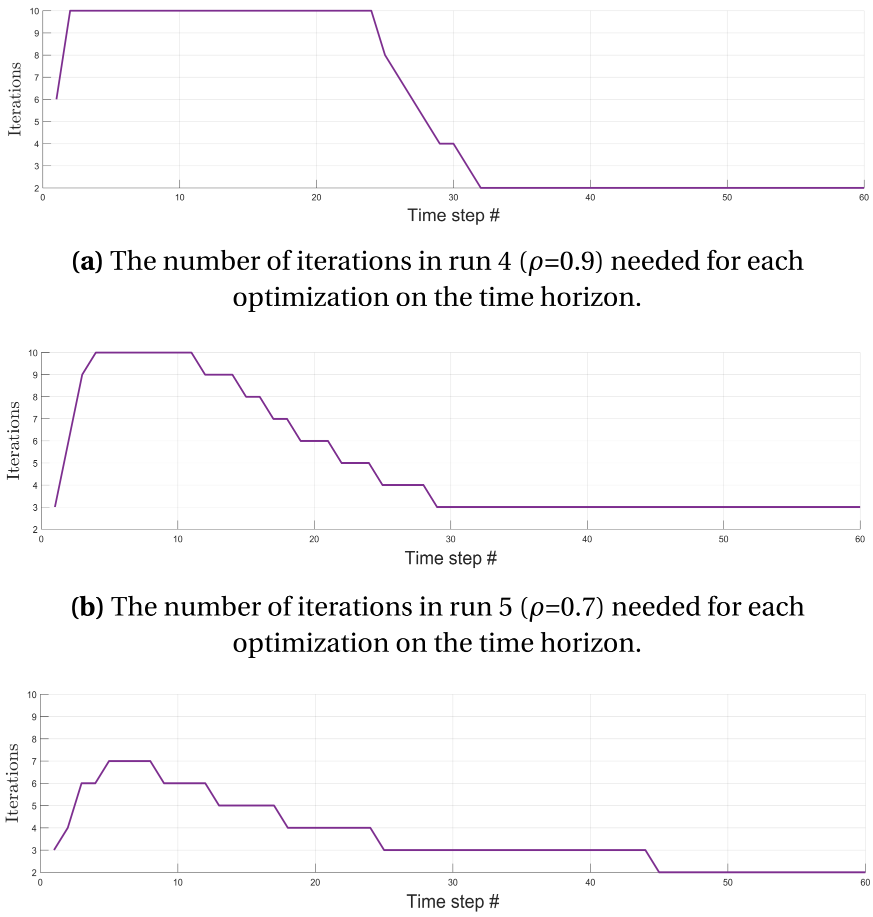
Table 6.6: The most central results for the runs performed for tuning the parameter ρ in the decomposed NMPC with the line search method. The parameter ρ is varied with the values of 0.9, 0.7 and 0.5 for respectively run LS.4, LS.5 and LS.6. The other parameter c is kept constant at 0.1.

Run #	Constraint violations	iterations convergence	iterations<10 sample time N	total iterations
LS.4	0.408	2	24	331
LS.5	0.453	3	11	300
LS.6	0.513	2	Never reaches 10 (max 7)	216

As seen from the table, as the parameter ρ decreases the constraint violation increases, while the iterations are respectively decreasing. Therefore this is a method of using fewer iterations to obtain the same results, at a cost of a larger constraint violation. The threshold for when the cost of violating the constraint is too large depends on the constraint and the consequences of violating that constraint. For soft constraints, the time saved using fewer iterations to get equally good control might be profitable. As mentioned above, a smaller step length will make the λ-prediction horizon take longer to reach optimum. Therefore the constraint violation keeps becoming larger as the updating parameter decreases. However, once the λ is close to optimum, the line search method makes it possible to converge completely.

From the results, run LS.6 is the least time consuming run with 84 fewer iterations or more. The small increase in the constraint violation is a good trade off for the saved cost in computational time. As the λ-prediction in the base case decomposition does not converge, the only other suitable run to compare run LS.6 with is run LS.2. Run LS.2 converges to the optimal value with a total of 360 iteration, and with a constraint violation that is 0.25 kg/s smaller. The cost of decreasing the iterations will always be a trade off against the accuracy of the simulation.

When discussing the iteration convergence and the sample time where the iterations no longer reach the cap of 10 iterations, it is easier to do so with a graphical presentation given in fig. 6.12. There is a clear trend that at one point the iterations starts decreasing at an almost steady pace, until it settles to a constant value of iterations needed. This is because of the warm start used for the λ optimization, meaning that the previous optimized value of the λ-vector is the starting point for the optimization for the next time step. Therefore when λ converges to an optimal value, over time fewer and fewer iterations are needed to find the new optimized value.



(a) The number of iterations in run 4 (ρ=0.9) needed for each optimization on the time horizon.

(b) The number of iterations in run 5 (ρ=0.7) needed for each optimization on the time horizon.

(c) The number of iterations in run 6 (ρ=0.5) needed for each optimization on the time horizon.

Figure 6.12: The overview of how many iterations that were needed for λ to converge for each time step. At each time step the optimization of the prediction horizon iterates until the convergence of λ reaches the tolerance=0.15.

All of the runs seem to stabilize at two or three iterations per time step for the last part of the simulations. The difference is in how long it takes the iterations to start to decrease. The first run reaches the cap of 10 iterations for almost half of the simulation, while the iterations of the second run starts to decrease after only a sixth of the simulation's time steps. For the last run, the iterations never reach the cap, leading to this run having much fewer total iterations. It has a maximum iteration per time step of only 7 iterations.

Since this work does not entail a hard constraint, and since constraints on capacity usually uses some safety off-push from the real constraint, the saved cost with fewer iterations is desirable. For the fewest iterations, run LS.6 and $\rho = 0.5$ would be the best choice.

6.2.5 Line Search method Without Maximum Iterations

After tuning the two parameters, it is of interest to investigate how removing the cap on the maximum number of iterations will affect the simulation. The three runs discussed below were performed without a cap on the maximum number of iterations and with two different methods of measuring the tolerance. The first run has the same parameters as run LS.5 but without iteration cap, this is because run LS.6 already does not reach the cap. The two last runs have a different way of measuring the tolerance, change in λ for each update, by only looking at the change in λ at the first sample instant of the prediction horizon. As the new tolerance measurement makes it possible to reach a smaller tolerance, the last run LS.9 has a different tolerance criteria. The runs are described in more detail in chapter 5 and the values of the runs are shown in table 5.2.

For these runs, again the interesting results are the iterations and the constraint violations. In table 6.7 the maximum iterations per optimization of prediction horizon and the total number of iterations for each run are presented, along with the constraint violations and the convergence value of the step length α.

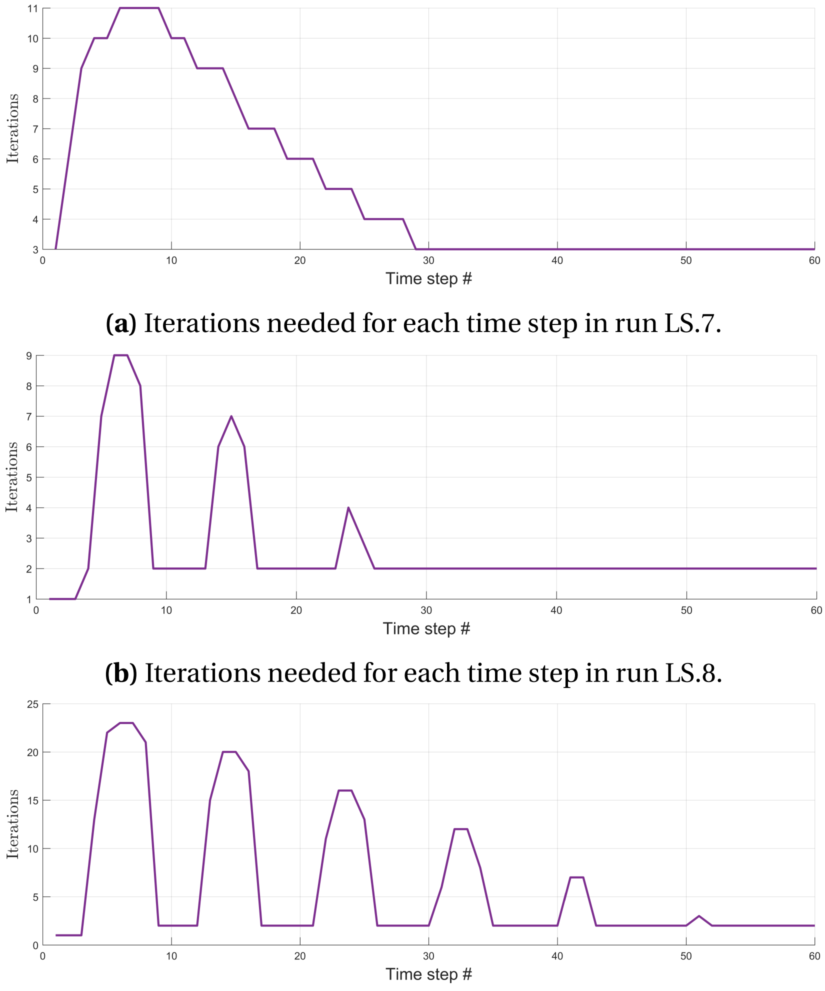
Table 6.7: Overview of runs performed without maximum iterations in line search method. For run LS.8 and LS.9 the tolerance was calculated using only the first sample instant.

Run #	Constraint violations	α convergence	max iterations	Total # iterations
LS.7	0.468	3.43	11	303
LS.8	0.590	4.9	9	158
LS.9	0.549	4.9	23	363

As run LS.7 is the same as run LS.5 without the iteration cap, the first comparison should be made between these two. When removing the iteration cap the total iteration increases with three iterations, while the constraint violation also increases. This is not a good trade off, and might point to it not being efficient for the solution to iterate a lot in the early stage of the simulations. It might be wiser to skip some iterations when the first few λs of the prediction horizon are zero and wait with the accurate line search approach for later time steps. This can be obtained by only measuring the tolerance for the first sample instant of λ on the prediction horizon. If the first sample instant of λ has stopped updating, the iteration loop will be stopped even though the rest of the horizon might still be changing.

To look at the effect of this new tolerance measurement, run LS.7 and run LS.8 are compared. It is observed that the number of iterations is almost cut in half, while the constraint violations increases a lot. The trade off is fewer iterations against a larger constraint violation, but in this case it seems economical. The difference in constraint violation from run LS.8 to run LS.9 is a decrease. Then again, the total amount of iterations increase because an even more precise convergence is reached. The added accuracy in run LS.9 does not pay off as much in constraint violations as it loses in iteration efficiency.

As seen in table 6.7, there is just a small difference in the maximum iterations for one optimization of the prediction horizon for run LS.7 and LS.8. However, run LS.8 has a lot more peaks than run LS.7, which can be seen in fig. 6.13. Since the tolerance is lowered significantly for run LS.9, there is also respectively more iterations need to find a more exact solution, this is shown in the large number of peaks and total amount of iterations need.



(a) Iterations needed for each time step in run LS.7.

(b) Iterations needed for each time step in run LS.8.

(c) Iterations needed for each time step in run LS.9.

Figure 6.13: The development of the iterations needed to reach convergence of λ for each time step in the simulation.

The same oscillation in number of iterations are seen for both run LS.8 in fig. D.6b and run LS.9 in fig. D.6c. Therefore it can be concluded that the reason for these oscillations is the method of measuring the tolerance, which is only done in this way for these two runs. The time steps for the peaks of iterations observed in fig. D.6b and fig. D.6c, overlap with the time step for the oscillations seen in the simulations. It seems that when an update of λ is needed the iterations increase only for a short while. This makes the second method of measuring tolerance more economical in iterations, but less tight when it comes to the constraint violations.

To further affirm the conclusion that the second measuring method is more economical, a run with this method was performed with the parameters from run LS.6. The results were 136 total iterations, with a maximum constraint violation of 0.598 kg/s above the total gas rate capacity. This is the lowest number of iterations and the largest constraint violation, however only 0.008 kg/s more than for run LS.8. The saved cost in iterations make up for that increase. If the aim of this method is saving time at a cost in performance, the last run would be the ideal. The results from this run is included in appendix E.

6.3 Implementing the Augmented Lagrangian

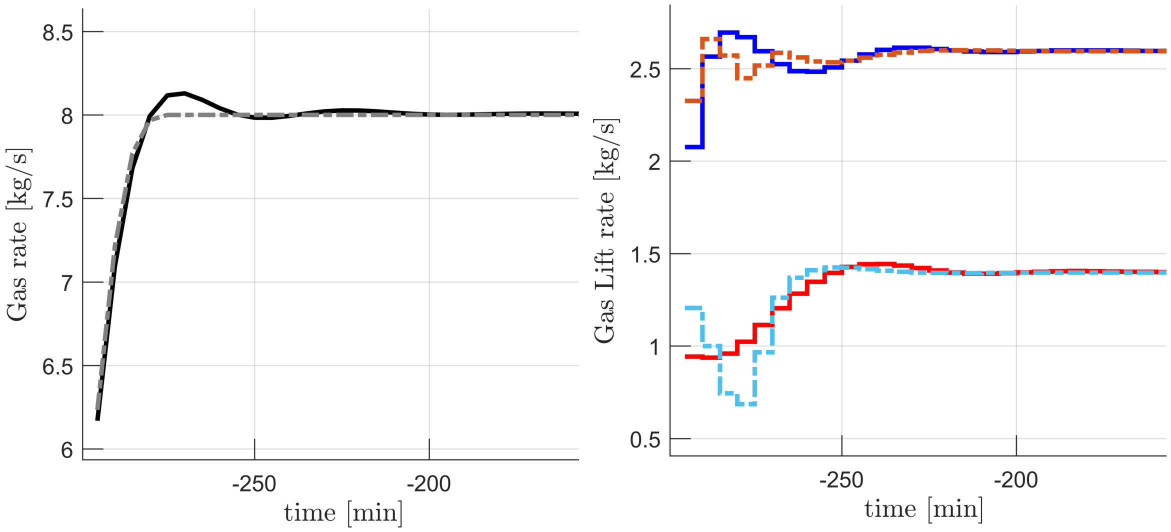
The augmented lagrangian decomposition penalizes the constraint violations and has an even tighter control than the regular lagrangian dual decomposition. It is explained in more details in chapter 4. In short, an extra penalizing term is added to the objective function of the decomposed problem, the penalizing parameter μ deciding the weight of the penalty. The penalty parameter is updated for each iteration by the updating parameter β.

The augmented lagrangian implementation diminishes the oscillations in the standard lagrangian decomposition dynamics. The constraint violations are still present, varying with the different values of the tuning parameters. Therefore the initial dynamics for the best augmented lagrangian run are compared and discussed against the centralized NMPC. Then the augmented lagrangian decomposition is compared to the base case decomposed NMPC. After this, the tuning of the two parameters for augmented lagrangian decomposition is presented. The results and discussion around the different runs conclude the best suited value for the parameters.

6.3.1 Comparing the Augmented Lagrangian Decomposed NMPC with the Centralized NMPC

The augmented lagrangian affect the dynamics and constraint violations of the simulations, so the interesting part to compare with the centralized NMPC is the initial dynamics. As run Aug.3 is found to be the best tuned augmented lagrangian decomposition, it is plotted together with the centralized NMPC. As for the previous investigation on the initial dynamics, the total gas rate is presented on the left in fig. 6.14a and the gas lift inlet to each well is shown on the right in fig. 6.14b.

When compared to the initial dynamics for the base case, it is observed that the augmented lagrangian decomposition gives a trajectory that is closer to the trajectory of the centralized NMPC. The violation of the constraint also seems to have decreased but it is still present. The colors and line types used for the comparison are the same as used above, where the best augmented lagrangian decomposed NMPC run Aug.3 is represented by the solid lines and the centralized NMPC is represented with the dot-dashed lines.



(a) The total gas production rate compared. **(b)** The gas lift inlet to each well compared.

Figure 6.14: The initial dynamics for the augmented lagrangian decomposed NMPC are compared to the simulation of the centralized NMPC. The black line represent the total gas rate for the best run Aug.3, the blue and the red represent the gas lift rate in well 1 and 2. The gray line represent the total gas rate for the centralized NMPC, the orange and light blue represent gas lift rate for well 1 and well 2.

It seems that the total gas rate in fig. 6.14a violates the constraint only twice, and the second time is much less than the first initial peak of the augmented lagrangian decomposition simulation. The augmented lagrangian decomposed NMPC trajectory does not follow that of the centralized NMPC, however it does not oscillate much around the solution. The fact that the oscillations have diminished is supported by the trajectory of the gas lift inlet rates shown in fig. 6.14b. The simulations don't have the same path but quickly reach the same values. If the constraint is a hard one, the augmented lagrangian method might be not enough to make the use of decomposition suitable. If the constraint is soft and it is profitable to have a small constraint violation as possible, then the augmented lagrangian method can be used for a dynamic decomposition control system. To compare the effect of using the augmented lagrangian decomposition method, it is compared to the base case decomposition.

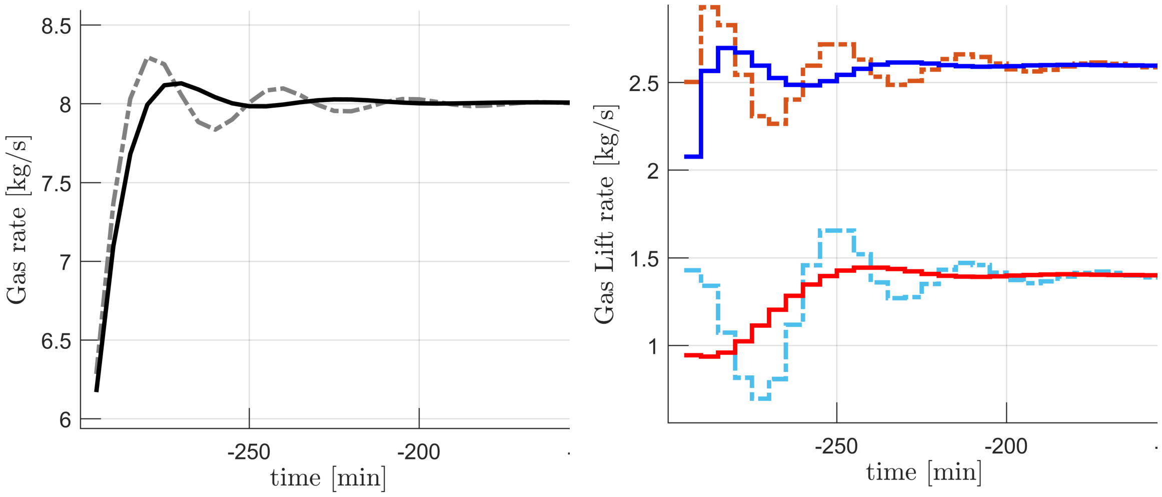
6.3.2 Comparing the Augmented Lagrangian Decomposed NMPC with the Base Case Decomposition

To compare the best run(Aug.3) of the augmented lagrangian decomposition to the base case decomposition, some central results are presented in table 6.8. The most interesting is the comparison of the constraint violations and the time until convergence. Also the first λ-value in the last prediction horizon is presented as $\lambda(1)$, and the time step at which λ becomes active for the entire prediction horizon is also included for comparison.

Table 6.8: The central results from using the augmented lagrangian method compared to the base case decomposition. The maximum constraint violations and the convergence time are included, as well as the converged λ at the first sample instant and the interval on the time horizon where λ is larger than zero.

Sim	Constraint violations	Convergence (time step #)	$\lambda(1)$	λ active (time step #)
Run Aug.3	0.129	18	5.34	6
Base case	0.295	27	21.65	4

It is clear from the table above that the constraint violations and the convergence towards optimum improve when the augmented lagrangian method is used. The time step at which λ becomes active for the entire prediction horizon is later than for the base case. However, this also coincides with the peak of the maximum constraint violation being a few steps later for the augmented lagrangian method. Although it takes longer for λ to begin to push, once the constraint is active the added penalty pushes harder, and therefore lessens the constraint violation. As a natural effect of adding a penalty to the constraint, the implemented value of λ is a lot less for the augmented lagrangian decomposition. Below, the trajectories of the augmented lagrangian and the base case decomposed NMPC are plotted together with respectively solid and dot-dashed lines. The trend explained above is clearly observed, and the results support the conclusion that the trajectory is better with the augmented lagrangian decomposed NMPC. The colors and line types used for the comparison are the same as used above. The best augmented lagrangian decomposed NMPC is represented by the solid lines, but here, the base case decomposed NMPC is represented with the dot-dashed lines.



(a) The total gas production rate compared.

(b) The gas lift inlet to each well compared.

Figure 6.15: The initial dynamics for the augmented lagrangian decomposed NMPC are compared to the simulation of the base case decomposed NMPC. The solid lines represent the best run Aug.3 for the augmented lagrangian method, where the black is total gas rate, the blue and the red represent the gas lift rate in well 1 and 2. The dot-dashed lines represent the base case, with the gray line representing the total gas rate, the orange and light blue represent gas lift rate for well 1 and well 2.

Another advantage to using the augmented lagrangian decomposed NMPC over the base case, is the fact that the iterations are lessened, also by using this method. The best run with the augmented lagrangian decomposition only iterates a total of 192 times, this is less than for the base case decomposition. As opposed to with the base case decomposition, the λ prediction horizon does converge with the augmented lagrangian. The iterations were, as in the base case, capped at 5 iterations. However, after the 24th time step, the iterations settle at only two iterations. The development of iterations for the best run of augmented lagrangian is shown below in fig. 6.16.

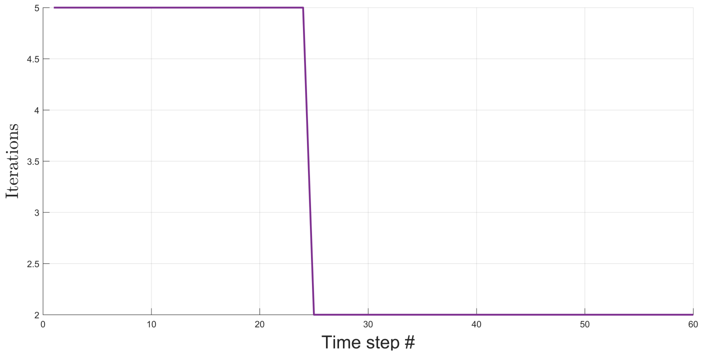


Figure 6.16: The iterations for each time step in the best run with augmented lagrangian decomposition Aug.3. Max 5 iterations for each time step.

The converged values for the augmented lagrangian decomposition are not completely consistent with the reference values from the centralized NMPC and the steady state optimization(ss-opt). The deviation is shown in table 6.9, and for run Aug.5 and run Aug.6 the deviation is considered to be too large for the runs to be optimal decomposed NMPCs. For the rest of the runs, the error is not considerer too significant for the solution to be valid.

Table 6.9: The converged values for each run performed with augmented lagrangian of the gas lift and gas rate for each well. The reference values from the steady state optimization are also presented to illustrate the deviation.

Run #	w_{gl_1}	w_{gl_2}	w_{pg_1}	w_{pg_2}
Aug.1	2.598	1.407	4.148	3.864
Aug.2	2.597	1.404	4.147	3.861
Aug.3	2.596	1.400	4.146	3.857
Aug.4	2.595	1.401	4.145	3.858
Aug.5	2.573	1.389	4.135	3.857
Aug.6	2.568	1.391	4.118	3.848
ss-opt	2.595	1.398	4.145	3.855

This error from the reference simulations appears to be due to the extra penalty constraint added into the cost function. When the penalty parameter is too large, the decomposed control structure does not seem to converge to the correct optimum. This will be a part of the discussion for tuning the parameters below. When tuning the parameters the constraint violation and the convergence time are the most central results. For the augmented lagrangian it is hard to compare the prediction horizon of λ, seeing that the value of λ is dependent on the value of μ and it changes for each run. But from looking at the prediction horizons it is apparent that oscillations in λ are not present for any of the runs. The prediction horizons for the augmented lagrangian are included in the appendix E

6.3.3 Tuning the Parameter μ

The parameter μ is the weight given to the penalty of the quadratic constraint violations. Therefore this parameter is referred to as the penalty parameter. The simulation runs for tuning the penalty parameter is explained in chapter 5 and the different values are presented in table 5.3.

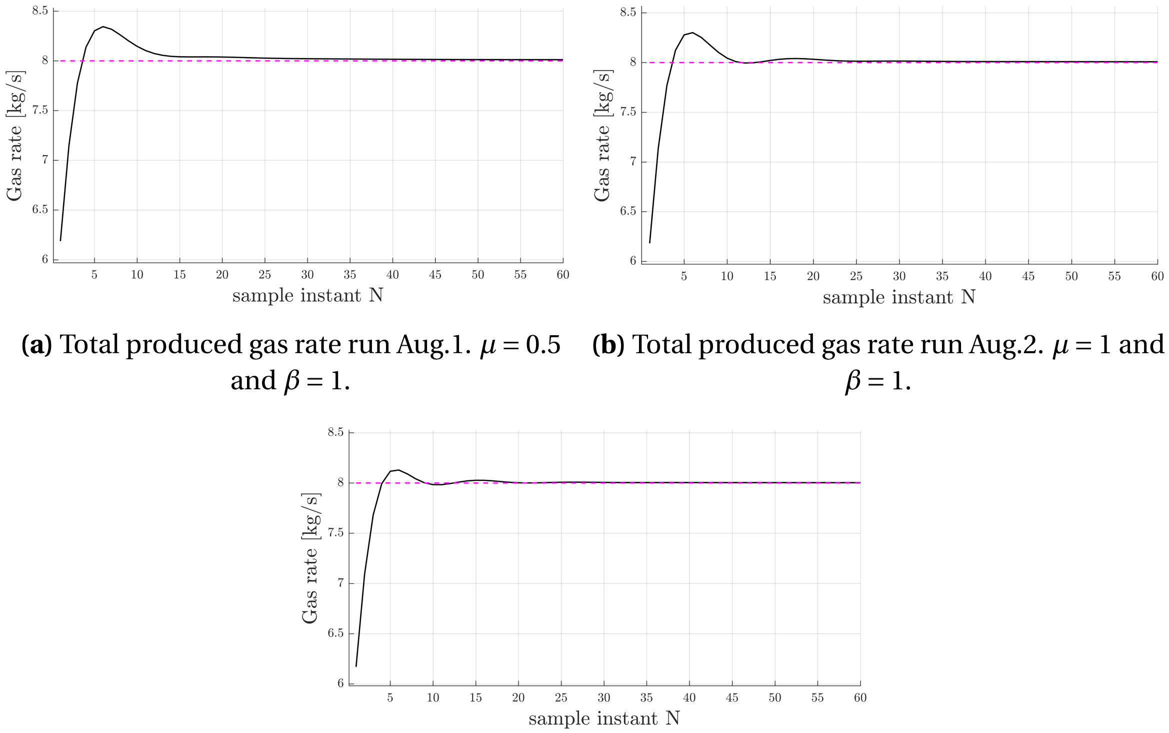
The central values for the three first runs tuning the parameter μ are presented below in table 6.10. The constraint violation decreases with the increase in μ, so does the time steps it takes for the solutions to converge. This because the penalty of the constraint violations pushes harder for a larger penalty parameter. The λ-value both at convergence and at the first time step decreases as μ increases, due to the adding the penalty part to the cost function forcing λ to decrease.

Table 6.10: The most central results for the runs performed for tuning the parameter μ in the augmented lagrangian decomposed NMPC. The parameter μ is varied with the values of 0.5, 1 and 2 for respectively run Aug.1, Aug.2 and Aug.3. The other parameter β is kept constant at 1.

Run #	Constraint violations	convergence (time step #)	$w_{pg_{tot}}$ (N=1)	Time-step opt-sol found
Aug.1	0.344	48	7.918	3
Aug.2	0.301	25	7.690	5
Aug.3	0.129	18	7.474	4

From the results presented above, it can be observed that both the constraint violations and the time until the simulation has converge decreases with increase in the penalty parameter. However, the first optimization of the prediction horizon does not reach the active constraint. The larger μ is, the further from the optimized value is the prediction horizon for the first time step. Although by the fifth time step all of the runs have reached the active constraint.

To illustrate the different trajectory of each of the runs, the initial dynamics are plotted for run Aug.1-Aug.3 in fig. 6.17. The total gas production rates are plotted over the time horizon.



(a) Total produced gas rate run Aug.1. $\mu = 0.5$ and $\beta = 1$.

(b) Total produced gas rate run Aug.2. $\mu = 1$ and $\beta = 1$.

(c) Total produced gas rate run Aug.3. $\mu = 2$ and $\beta = 1$.

Figure 6.17: The total gas rate from the augmented lagrangian decomposed NMPC is shown up-close, for run Aug.1-Aug.3 to illustrate the trajectory of the simulation.

For the first run with the lowest value of μ, shown in fig. 6.17a, the solution takes a long time to converge towards the constraint value, but it does not oscillate. For the second run, shown in fig. 6.17b, the constraint violation is a little bit less and it is converging much faster however, there is more oscillation. The best run is clearly the third run shown in fig. 6.17c, with the smallest constraint violation and the quickest convergence. Although, for this run the optimal solution is not obtained in the prediction horizon until the fourth time step.

The gas lift rate simulations are not included, as they are not very visibly different. When studied closely, the difference in the trajectory of the gas lift simulations is that the larger μ is, the steeper is the gas lift rate trajectory. The gas lift rate trajectories are included in the appendix E. The optimized prediction horizons are also included in the appendix for the first and the last time step.

Run Aug.3 is the only run that improves the constraint violation of the base case, so the augmented lagrangian method needs correct tuning to improve the decomposition. This might cause difficulty with a hard constraint, however, when tuned correctly the method can eliminate more than half of the constraint violations.

6.3.4 Tuning the Parameter β

The parameter β is the update parameter increasing the penalty parameter for each update of λ. The different runs for tuning the updating parameter β are explained in chapter 5 and the different values are presented in table 5.3.

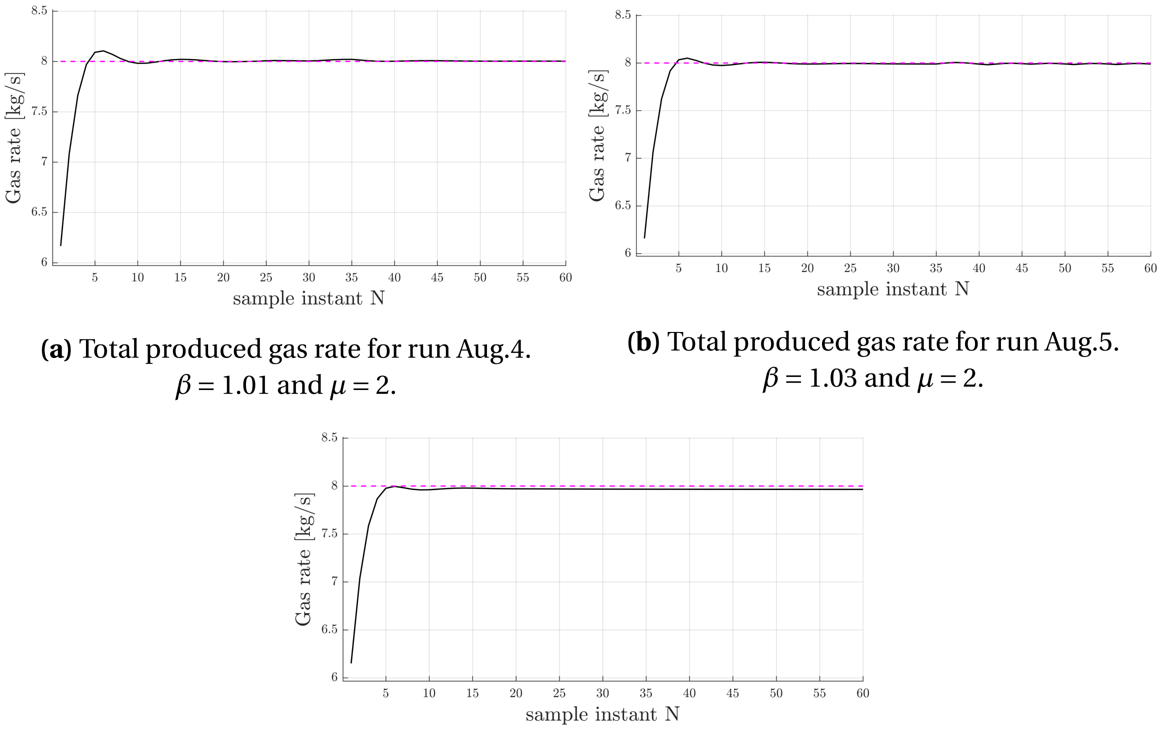
With augmented lagrangian decomposition the lagrange multiplier is updated with the step length of the penalty parameter μ. As with the line search method and the step length α, μ is reset to the initial value before each optimization of the prediction horizon. This is necessary for the penalty parameter to not become to large and push the prediction horizon away from the optimal solution. As the updating parameter increases the penalty parameter, β must be greater or equal to one. For the runs shown above $\beta = 1$, meaning the penalty parameters is the same throughout the simulations.

The central results of the simulations for the tuning of β are presented in table 6.11. When attempting to set β to a larger value than one, the optimization of the prediction horizon struggles to reach the active constraint. This can be observed by looking at the intervals where the first sample instant of λ ($\lambda(1)$) is active and the last predicted value of the total gas rate. For run Aug.5 and Aug.6, the prediction horizon does not reach the active constraint for all the time steps of the horizon. In the simulation however, run Aug.5 reaches the optimum total gas rate. For more details the last prediction horizons can be found in appendix E.

Table 6.11: The most central results for the runs performed for tuning the parameter β in the augmented lagrangian decomposed NMPC. The parameter β is varied with the values of 1.01, 1.03 and 1.05 for respectively run Aug.4, Aug.5 and Aug.6. The other parameter μ is kept constant at 2.

Run #	Constraint violations	convergence (time step #)	last predicted $w_{pg_{tot}}$	$\lambda(1)$ active interval
Aug.4	0.105	40	8.00	N=[6-60]
Aug.5	0.051	25	7.914	N=[7-11]
Aug.6	-0.031	20	7.86	Never active

In addition to the numerical results in the table above, the total gas production rates are presented in fig. 6.18, in the same way as the previous three runs. The observed trends of the gas lift matches the numerical values presented in the table above.



(a) Total produced gas rate for run Aug.4. $\beta = 1.01$ and $\mu = 2$.

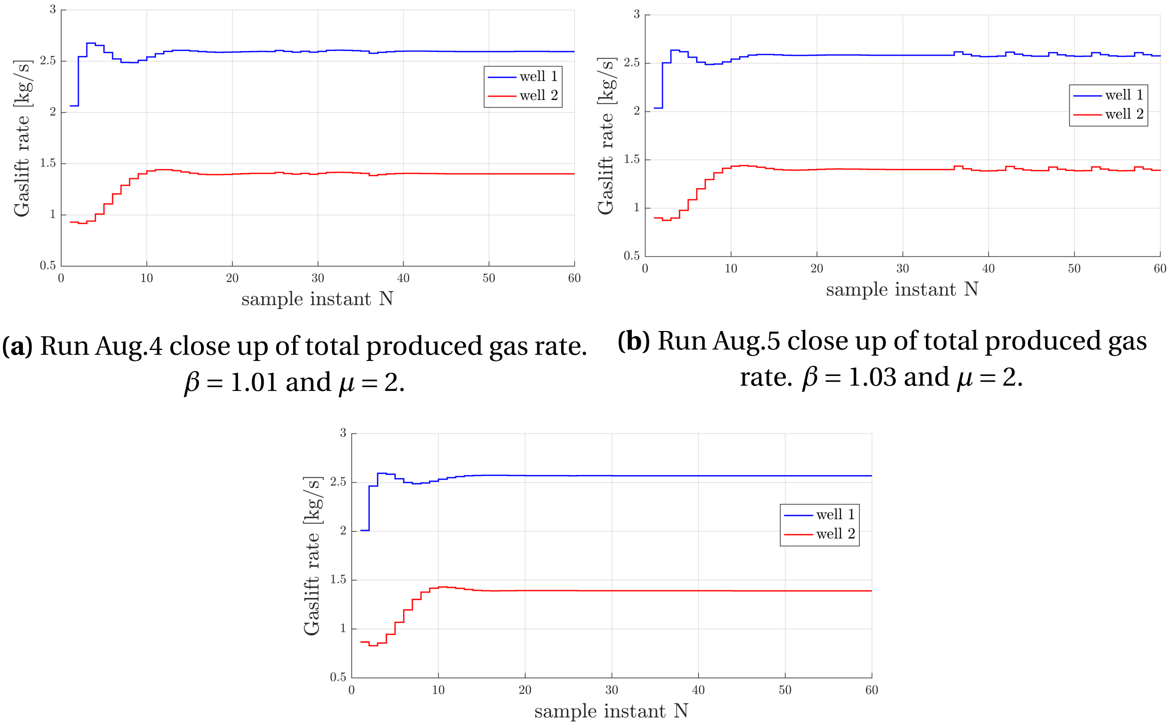
(b) Total produced gas rate for run Aug.5. $\beta = 1.03$ and $\mu = 2$.

(c) Total produced gas rate for run Aug.6. $\beta = 1.05$ and $\mu = 2$.

Figure 6.18: The total gas rate from the augmented lagrangian decomposed NMPC.

For run Aug.4 the constraint violation is lessened but the time until convergence is a lot longer. For run Aug.5 the constraint violations and the convergence is less, however the λ is only active for a short interval on the time horizon. This paired with the fact that the prediction horizon does not fully converge to the optimum value of the total gas rate, makes run Aug.5 not a valid run. This is also the case for run Aug.6, seeing that for this run the first sample instant on the λ prediction horizon is never non-zero. The simulation in run Aug.6 never reaches the active constraint, and is therefore not optimal and not a valid run. Since λ is never active for run Aug.6, the subproblems are never coupled and therefore this is just two separate control structures, not a decomposed NMPC.

For run Aug.4, the constraint violation has decrease compared to the best value found in run Aug.3. Although the convergence takes longer, as can be seen in fig. 6.18a. The simulation is also observed to oscillates after the optimal solution is reached. This tuning of the control system therefore seems to be more instable than for that in run Aug.3. Run Aug.5 in fig. 6.18b also seems to be instable, and oscillates at the end of the simulated horizon, after having reached the steady state value. The last run, shown in fig. 6.18c, has the largest value of the update parameter β and never reaches the optimal solution for the entire simulation. Hence the common constraint for the subproblems is not active.The trends of instability can also be observed in the simulation results of the gas lift injection rate shown in fig. 6.19.



(a) Run Aug.4 close up of total produced gas rate. $\beta = 1.01$ and $\mu = 2$.

(b) Run Aug.5 close up of total produced gas rate. $\beta = 1.03$ and $\mu = 2$.

(c) Run Aug.6 close up of total produced gas rate. $\beta = 1.05$ and $\mu = 2$.

Figure 6.19: The gas lift rate for each well from the augmented lagrangian decomposed NMPC.

The gas lift rates for run Aug.4(fig. 6.19a) and for run Aug.5(fig. 6.19b), show the same oscillations at the end of the simulation as they did for the total gas rates. This supports the conclusion that these runs are not stable, and that increase in the penalty parameter can make the augmented lagrangian decomposed NMPC unstable. There are no oscillations observed for run Aug.6, then again the constraint was never active for the subproblems in this run, making it stabilize at solutions that are separately optimal.

As a conclusion the updating parameter β should be equal to one, in other words μ was the same throughout the simulations. None of these runs are stable enough for the decomposed NMPC, and therefore run Aug.3 was the best run for the augmented lagrangian decomposition. As for the line search method, augmented lagrangian method must be carefully tuned to give an advantage. With the wrong tuning, the decomposition might produce a worse result.

6.4 Comparing the Numerical Methods

The two numerical methods have different improvements and purposes, and are therefore hard to directly compare. However, their total effects can be compared and discussed. The comparison of these methods are compared for the gas lifted well network system used in this thesis.

Line search method is mostly suitable for a system with soft constraints, and where lowering the number of iterations can be profitable. As it increases the constraint violations, it is not a good method for a tighter control of the decomposed system. The line search is also fairly hard to tune, and can therefore be hard to operate. The tuning does not send the system outside of specs, but it needs correct tuning to save computational time. For this system, where λ does not converge for the base case, a numerical method was needed for the prediction horizon to be optimized properly. Since the constraints in this work are soft, the line search method was eligible for making this improvement.

Unlike the line search method the constraint violations are lessened when using the augmented lagrangian decomposition. The prediction horizon of λ also converges using the augmented lagrangian method. Although the total number of iterations are not the focus when tuning the augmented lagrangian method, the best case has relatively few iterations. However, this method is also hard to tune correctly. A wrongly tuned augmented lagrangian decomposed NMPC can actually end up without a proper decomposition, where the constraint coupling the two subproblems is not active. This is a major issue that might cause economical losses as a result. Still, tuned correctly the lagrangian method is the best in terms of the dynamics of the initial simulations.

Chapter 7

Conclusion

The purpose of this thesis was to investigate the possibility of using dynamic decomposition as a control tool within the oil and gas sector. As todays industry seeks better optimization and utilization of oil production and profit, the control systems are becoming more complex. With the complexity of systems becoming larger, the control structure needs more automation and better solutions for solving the large optimization problems. Introducing a dynamic decomposition of the large control structures can make it possible to have a global constraint for a larger problem and it can simplify the operating work on the total system. An additional advantage of a dynamic decomposed control structure is the possibility of shutting down one subsystem for maintenance, while still having an active control optimizing for the remaining subproblems.

The previous work on decomposition for oil and gas well network has only been investigating a steady state control system. The previously published work found the controller input from steady state subproblems connected by a common constraint, making it necessary for operators to manually reset the controller input with a chosen interval. The dynamic decomposition in this work looks at optimizing a prediction horizon for a NMPC, updating the controller set point automatically as the operating conditions changes. If it is possible to obtain a dynamic decomposed NMPC that controls the system satisfactory, this can remove the human factor of resetting the set point from the steady state decomposition.

The base case decomposition in this thesis refers to the lagrangian dual decomposition performed for a gas lifted two-well network system. The simulation results show that the decomposed NMPC converges to the same results as the centralized NMPC and a steady state optimization performed for the same system. However, in certain cases, the dynamics in the initial part of the simulation could be problematic. For the case where the gas rate is the active constraint, the decomposed simulation oscillates around the trajectory of the centralized NMPC, even violating the constraint. This validates the concerns about the start-up in the dynamic decomposition made in [17], however, this is just a problem for some cases. In addition to the dynamic issues, the λ prediction horizon does not converge throughout the optimization, so closed-loop stability is not obtained. These problems with the base case decomposition are attempted improved by applying numerical solutions to the decomposed NMPC.

The line search method successfully brings down the number of iterations, but at the cost of a larger constraint violation than in the base case decomposition. With a converged λ prediction horizon and saved computational time, this is an improvement to the base case decomposed NMPC simulation. Although hard to tune, with the correct parameters the line search method can cut two-thirds of the computational time, measured in iterations. From comparing two different ways of measuring the change in λ an important finding was made. If λ is not active, in other words zero, for the first sample instant in the prediction horizon, the optimization should be stopped after few iterations. This is to save computational time for the optimization of λ, as the total amount of iterations become considerable lower and the change in simulation results are minor. Although this is a result found for the line search method, this rule applies to all dynamic decomposition with a lagrangian relaxation of the constraints.

The augmented lagrangian decomposition decreases the constraint violations and converges faster than the base case decomposition. Combined with the fact that λ converges and that the simulation needs few iterations, it makes this the best method for dynamic decomposition of this system. However, there are some difficulties to tuning this method, in the worst case not being able to reach optimal solution for certain parameters. Still, with the best tuned run the augmented lagrangian decomposed NMPC has a correct solution with the least amount of constraint violation and few total iterations.

Since the system used in this problem is very simple, the results are only of academic interest. No conclusion can be made on the realistic uses of dynamic decomposition for control within the oil and gas industry. However, the results show a possibility for this type of control system to be suitable for use in more complex systems. In theory, the simple model is easily scaled up, but dynamics might increase for a larger system. Some systems might not be suitable for decomposed control structures, like some cases with hard constraints. However, control problems like the available gas lift constraint can handle hard constraints with a decomposed control system. When the controlled system has a soft constraint the dynamic decomposition could be a good control method.

7.1 Further Work

As mentioned above, this dynamic decomposition was applied to a very simple model, so no final conclusions can be made about the use of dynamic decomposition from this work alone. Therefore, further work should focus on testing dynamic decomposition for a more complex and larger model. Adding more wells or manifolds to the model, as well as including uncertainties in the simulations, will make it easier to judge the potential of the method.

Bibliography

[1] Jorge Moreno-trejo and Tore Markeset. Identifying Challenges in the Maintenance of Subsea Petroleum Production Systems. *IFIP International Conference on Advances in Production Management Systems*, pages 251–259, 2012.

[2] C Roberts and T Lazng. Achieving Reliability Improvement for Subsea Challenges. *In: The Proceedings of the Subsea Controls and Data Acquisition Conference (SCADA 2002), 13-14 June, Paris, France*, pages 101–110, 2002.

[3] Jorge Moreno-trejo Rajesh Kumar and Tore Markeset. Mapping factors influencing the selection of subsea petroleum production systems : a case study. *International Journal of System Assurance Engineering and Management*, 3(1):6–16, 2012.

[4] Bjarne Foss and John Petter Jensen. Performance Analysis for Closed-Loop Reservoir Management. *SPE Journal*, 16(1):183–190, 2011.

[5] Dinesh Krishnamoorthy, Bjarne Foss, and Sigurd Skogestad. Real-Time Optimization under Uncertainty Applied to a Gas Lifted Well Network. *Processes*, 4(4):52—-70, 2016.

[6] David Q Mayne. Model predictive control : Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

[7] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. John Wiley & Sons, Ltd, 2nd edition, 2005.

[8] Pengju Wang. *Development and Applications of Production Optimization Techniques for Petroleum Fields*. PhD thesis, 2003.

[9] H Nishikiori, R A Redner, D R Doty, and Z Schmidt. An Improved Method for Gas Lift Allocation Optimization. *Journal of Energy Resources Technology*, 117(2):87–92, 1995.

[10] J Hauge, T Horn, and Norsk Hydro Asa. OTC 17111 The Challenge of Operating and Maintaining 115 Subsea Wells on the Troll Field. *Offshore Technology Conference, Houston, Texas*, pages 1–14, 2005.

[11] Stephen Boyd, Lin Xiao, Almir Mutapcic, and Jacob Mattingley. Notes on Decomposition Methods. 2008. `https://pdfs.semanticscholar.org/d746/f6b5f69401b147996d21a4b5940f6dd40020.pdf`.

[12] Marshall L Fisher. An Applications Oriented Guide to Lagrangian Relaxation. *Interfaces*, 15(2):10–21, 1985.

[13] Jennifer R. Jackson and Ignacio E. Grossmann. Temporal Decomposition Scheme for Nonlinear Multisite Production Planning and Distribution Models. *Industrial & Engineering Chemistry Research*, 42(13):3045–3055, 2003.

[14] Zukui Li and Marianthi G Ierapetritou. Production planning and scheduling integration through augmented Lagrangian optimization. *Computers and Chemical Engineering*, 34(6):996–1006, 2010.

[15] Tatsushi Nishi, Ryuichi Shinozaki, and Masami Konishi. An Augmented Lagrangian Approach for Distributed Supply Chain Planning for Multiple Companies. *IEEE Transactions on Automation Science and Engineering*, 5(2):259–274, 2008.

[16] Bjarne Foss, Vidar Gunnerud, and Marta Dueñas Díez. Lagrangian Decomposition of Oil-Production Optimization Applied to the Troll West Oil Rim. *SPE Journal*, 14(4):646–652, 2009.

[17] Vidar Gunnerud and Bjarne Foss. Oil production optimization — A piecewise linear model , solved with two decomposition strategies. *Computers and Chemical Engineering*, 34:1803–1812, 2010.

[18] Michael G Forbes, Rohit S Patwardhan, Hamza Hamadah, and R Bhushan Gopaluni. Model Predictive Control in Industry: Challenges and Opportunities. *9th International Symposium on Advanced Control of Chemical Processes, The International Federation of Automatic Control*, pages 532–539, 2015.

[19] Gisle Otto Eikrem, Lars Imsland, and Bjarne Foss. Stabilization of Gas Lifted Wells Based on State Estimation. *IFAC Proceedings Volumes*, 37(1):323–328, 2004.

[20] Philip E Gill, Walter Murray, Michael A Saunders, John A Tomlin, and Margaret H Wright. George B . Dantzig and systems optimization. *Discrete Optimization*, 5:151–158, 2008.

[21] Stephen J. Nocedal, Jorge; Wright. *Numerical Optimization.* Springer US, 2006.

[22] Michael Nikolaou. Model Predictive Controllers: A Critical Synthesis of Theory and Industrial Needs. In *Advances in Chemical Engineering*, volume 26, pages 131–204. 1998.

[23] A. A. Jalali and V. Nadimi. A Survey on Robust Model Predictive Control from 1999-2006. *2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, (1):207–207, 2006.

[24] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematcal Programming*, 106(1):25–57, 2006.

[25] Uri M. Ascher and Linda R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. 1998.

[26] David Q. Rawlings, James B.; Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, LLc, Madison, Wisconsin, 2009.

[27] Interconnected Power Systems. Anticipating and Coordinating Voltage Control for Interconnected Power Systems. *Energies*, 7(2):1027–1047, 2014.

[28] Lars Grüne and Jürgen Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Springer London, 2010.

[29] Lorenz T Biegler. A Survey on Sensitivity-based Nonlinear Model Predictive Control. In *IFAC Proceedings Volumes*, volume 46, pages 499–510. IFAC, 2013.

[30] H P Bieker, O Slupphaug, and T A Johansen. Real - Time Production Optimization of Offshore Oil and Gas Production Systems : A Technology Survey. In *SPE Intelligent Energy Conference and Exhibition*, 2006.

[31] Lorenz T. Biegler. *Nonlinear Programming: Concepts, Algorithms, and Appliactions to Chemical Processes*. Society for Industrial and Applied Mathematics, Philadelphia, 2010.

[32] Sébastian Gros. Numerical Optimal Control with DAEs. Lecture 8: Direct Collocation. *System Control and Optimization Laboratory, IMTEK, Faculty of Engineering, University of Freiburg*. https://www.syscop.de/files/2015ws/noc-dae/lectureslides/08-Collocation.pdf.

[33] Sébastian Gros. Numerical Optimal Control with DAEs. Lecture 12: Optimal Control with DAEs. https://www.syscop.de/files/2015ws/noc-dae/lectureslides/12-DAEOptimalControl.pdf.

[34] Jens Bausa and Georgios Tsatsaronis. Dynamic Optimization of Startup and Load-Increasing Processes in Power Plants — Part I : Method. *Journal of Engineering for Gas Turbines and Power*, 123(January 2001):246–250, 1999.

[35] John E Beasley. *Modern Heuristics Techniques for Combinatorial Problems*. Blackwell Scientific Publications, 1993.

[36] Joel Andersson, Joris Gillis, and Moritz Diehl. User Documentation for CasADi v3.1.0. 2016. http://casadi.sourceforge.net/users_guide/casadi-users_guide.pdf.

[37] Jan Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 1st edition, 2002.

Appendix A

Model Parameters and Initialization Values

Table A.1: List of parameters given as input to the NMPC for the gas lifted oil well system.

Parameter	**Value[Well1;Well2]**	**Unit**
L_w	[1500;1500]	[m]
H_w	[1000;1000]	[m]
D_w	[0.121;0.121]	[m]
L_{bh}	[500;500]	[m]
H_{bh}	[100;100]	[m]
D_{bh}	[0.121;0.121]	[m]
L_a	[1500;1500]	[m]
H_a	[1000;1000]	[m]
D_a	[0.189;0.189]	[m]
ρ_o	$[9;8]\cdot10^2$	$[kg/m^3]$
C_{iv}	$[0.1;0.1]\cdot10^{-3}$	$[m^2]$
C_{pc}	$[1;1]\cdot10^{-3}$	$[m^2]$
p_{res}	[150;155]	[bar]
p_m	[20;18]	[bar]
PI	[2.2;2.2]*	$[kg/s\cdot bar]$
T_a	[28;28]	[°C]
T_w	[32;32]	[°C]
Parameter	**Value[Well1;Well2]**	**Unit**
M_w	$20\cdot10^{-3}$	[g/mol]
Active constraints:		
qGLMax	5	[kg/s]
QgMax	8	[kg/s]

Table A.2: List of initial values of the state variables and the control variables. These initial values are taken from previously simulated data.

Parameter	**Value**	**Unit**
m_{ga_0}	[9.34;10.20]	[kg]
m_{gt_0}	[1.88;1.56]	[kg]
m_{ot_0}	[11.95;12.59]	[kg]
p_{ai_0}	[80.65;88.08]	[bar]
p_{wh_0}	[25.68;28.50]	[bar]
p_{wi_0}	[74.87;82.79]	[bar]
p_{bh_0}	[83.70;90.64]	[bar]
ρ_{ai_0}	[6.93;7.57]	$[kg/m^3]$
ρ_{m_0}	[5.01;5.53]	$[kg/m^3]$
w_{iv_0}	[2.00;2.00]	[m/s]
w_{pc_0}	[16.88;21.69]	[m/s]
w_{pg_0}	[2.29;2.39]	[m/s]
w_{po_0}	[14.59;19.31]	[m/s]
w_{ro_0}	[14.58;19.31]	[m/s]
w_{rg_0}	[2.29;2.39]	[m/s]
w_{gl_0}	[1;1]	[m/s]
GOR	[0.1;0.15]	[kg/kg]

Appendix B

Collocation Points

Table B.1: Shifted Gauss-Legendre and Radau roots as collocation points. [31]

Polynomial Degree	Legendre Roots	Radau Roots
1	0.500000	1.000000
2	0.211325	0.333333
	0.788675	1.000000
3	0.112702	0.155051
	0.500000	0.644949
	0.887298	1.000000
4	0.069432	0.088588
	0.330009	0.409467
	0.669991	0.787659
	0.930568	1.000000
5	0.46910	0.057104
	0.230765	0.057104
	0.230765	0.276843
	0.500000	0.583590
	0.769235	0.860204
	0.953090	1.000000

Appendix C

Matlab Code Snippets: NMPC

Code Snippets C.1: NMPC modelling

```
clear all
clc
%% Import CasADi
addpath ('C:\Users\IngvildMarie\Documents\CasADi\casadi-matlabR2014b-v3.1.0')
import casadi.*
%% Set parameters, initial values, upper and lower bounds
% All the parameter values are defined inside this function
GasLiftSystem_Param;
n_w = par.n_w;  % no. of wells;
N = 60;         % no. of samples
tf = 300;       % sampling time
T = tf*N;
% The initial values, upper and lower bounds are defined here:
[dx0,z0,u0,lbx,lbz,lbu,ubx,ubz,ubu] = GasLiftSystem_Initialization_bounds(par);

%% Modelling

L_w = par.L_w;
H_w = par.H_w;
D_w = par.D_w;

L_bh = par.L_bh;
H_bh = par.H_bh;
D_bh = par.D_bh;

L_a = par.L_a;
H_a = par.H_a;
```

```
D_a = par.D_a;

rho_o = par.rho_o;
C_iv = par.C_iv;
C_pc = par.C_pc;

A_w = pi.*(D_w/2).^2;
A_bh = pi.*(D_bh/2).^2;
V_a = L_a.*(pi.*(D_a/2).^2 - pi.*(D_w/2).^2);

Zc = 1;
% differential states
m_ga = MX.sym('m_ga',n_w);
m_gt = MX.sym('m_gt',n_w);
m_ot = MX.sym('m_ot',n_w);
% Algebraic states
p_ai = MX.sym('p_ai',n_w);          % 1-2
p_wh = MX.sym('p_wh',n_w);          % 3-4
p_wi = MX.sym('p_wi',n_w);          % 5-6
p_bh = MX.sym('p_bh',n_w);          % 7-8
rho_ai = MX.sym('rho_ai',n_w);      % 9-10
rho_m = MX.sym('rho_m',n_w);        % 11-12
w_iv = MX.sym('w_iv',n_w);          % 13-14
w_pc = MX.sym('w_pc',n_w);          % 15-16
w_pg = MX.sym('w_pg',n_w);          % 17-18
w_po = MX.sym('w_po',n_w);          % 19-20
w_ro = MX.sym('w_ro',n_w);          % 21-22
w_rg = MX.sym('w_rg',n_w);          % 23-24
% control input
w_gl = MX.sym('w_gl',n_w);

% parameters
p_res = MX.sym('p_res',n_w);
PI = MX.sym('PI',n_w);
GOR = MX.sym('GOR',n_w);
T_a = MX.sym('T_a',n_w);
T_w = MX.sym('T_w',n_w);
p_m = MX.sym('p_m',1);
```

```
% algebraic equations
f1 = -p_ai.*1e5 + ((R.*T_a./(V_a.*M) + 9.81.*H_a./V_a).*m_ga.*1e3);
f2 = -p_wh.*1e5 + ((R.*T_w./M).*(m_gt.*1e3./(L_w.*A_w + L_bh.*A_bh - m_ot.*1e3./rho_o)));
f3 = -p_wi.*1e5 + (p_wh.*1e5 + 9.81./(A_w.*L_w).*max(0,(m_ot.*1e3+m_gt.*1e3-rho_o.*L_bh.*A_bh)).*H_w);
f4 = -p_bh.*1e5 + (p_wi.*1e5 + rho_o.*9.81.*H_bh);
f5 = -rho_ai.*1e2 +(M./(R.*T_a).*p_ai.*1e5);
f6 = -rho_m.*1e2 + (m_gt.*1e3+m_ot.*1e3 - rho_o.*L_bh.*A_bh)./(L_w.*A_w);
f7 = -w_iv + C_iv.*sqrt(rho_ai.*1e2.*(p_ai.*1e5 - p_wi.*1e5));
f8 = -w_pc + C_pc.*sqrt(rho_m.*1e2.*(p_wh.*1e5 - p_m.*1e5));
f9 = -w_pg + (m_gt.*1e3./max(1e-3,(m_gt.*1e3+m_ot.*1e3))).*w_pc;
f10 = -w_po + (m_ot.*1e3./max(1e-3,(m_gt.*1e3+m_ot.*1e3))).*w_pc;
f11 = -w_ro + PI.*1e-6.*(p_res.*1e5 - p_bh.*1e5);
f12 = -w_rg.*1e-1 + GOR.*w_ro;  % 23 - 24
% differential equations
df1 = (w_gl - w_iv).*1e-3;
df2 = (w_iv + w_rg.*1e-1 - w_pg).*1e-3;
df3 = (w_ro - w_po).*1e-3;

% Form the DAE system
diff = vertcat(df1,df2,df3);
alg = vertcat(f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12);

% give parameter values
alg = substitute(alg,p_res,par.p_res);
alg = substitute(alg,PI,par.PI);
alg = substitute(alg,T_a,par.T_a);
alg = substitute(alg,T_w,par.T_w);
alg = substitute(alg,p_m,par.p_m(1));

% concatenate the differential and algebraic states
x_var = vertcat(m_ga,m_gt,m_ot);
z_var = vertcat(p_ai,p_wh,p_wi,p_bh,rho_ai,rho_m,w_iv,w_pc,w_pg,w_po,w_ro,w_rg);
p_var = vertcat(w_gl,GOR);

```

```
% Define Steady state objective
L = -sum(w_po);
% Define the DAE system
dae = struct('x',x_var,'z',z_var,'p',p_var,'ode',diff,'alg',alg,'quad',L); % der(m_tot) = w_in - w_out;
opts = struct('tf',T/N);

% create IDAS integrator - Simulator Model
F = integrator('F','idas',dae,opts);

% Create the semi-explicit DAE function of the form:
% x_dot = f(x,z,u)
% g(x,z,u) = 0
f = Function('f',{x_var,z_var,p_var},{diff,alg,L},{'x','z','p'},{'xdot','zj','qj'});
```

Code Snippets C.2: The collocation method used for both centralized and decomposed NMPC

```
%% Direct Collocation

% Degree of interpolating polynomial
d = 3;

% Get collocation points
tau_root = [0, collocation_points(d, 'radau')];

% Coefficients of the collocation equation
C = zeros(d+1,d+1);

% Coefficients of the continuity equation
D = zeros(d+1, 1);

% Coefficients of the quadrature function
B = zeros(d+1, 1);

% Construct polynomial basis
for j=1:d+1
    % Construct Lagrange polynomials to get the polynomial basis at the collocation point
    coeff = 1;
    for r=1:d+1
        if r ~= j
            coeff = conv(coeff, [1, -tau_root(r)]);
            coeff = coeff / (tau_root(j)-tau_root(r));
        end
    end
    % Evaluate the polynomial at the final time to get the coefficients of the continuity equation
    D(j) = polyval(coeff, 1.0);

    % Evaluate the time derivative of the polynomial at all collocation points to get the coefficients of the continuity equation
    pder = polyder(coeff);
    for r=1:d+1
        C(j,r) = polyval(pder, tau_root(r));
    end

    % Evaluate the integral of the polynomial to get the coefficients of the quadrature function
    pint = polyint(coeff);
    B(j) = polyval(pint, 1.0);
end
```

Code Snippets C.3: Building the nlp for centralized NMPC

```
%% Optimization and Simulation

nu = 2;
nz = length(z_var); % no. of alg  states
nx = length(x_var); % no. of diff states
GOR_val=par.GOR;
%% Build NLP
% empty nlp
w = {};
w0 = [];
lbw = [];
ubw = [];
J = 0;
```

```
g = {};
lbg = [];
ubg = [];

% initial conditions
X0 = MX.sym('X0',nx);
Z0 = MX.sym('Z0',nz);
w = {w{:}, X0,Z0};
lbw = [lbw; dx0;z0];
ubw = [ubw; dx0;z0];
w0 = [w0; dx0;z0];

U0 = MX.sym('U0',2);
GOR0 = MX.sym(['GOR0'],nu);
Upar0 = vertcat(U0,GOR0);
w = {w{:}, Upar0};
lbw = [lbw; u0;GOR_val];
ubw = [ubw; u0;GOR_val];
w0 = [w0; u0;GOR_val];

% Formulate NLP
Xk = X0;
Xkj = {};
Zkj = {};
Uk_prev = U0;

for k = 0:N-1

    % PWC control input at time interval k
    Uk = MX.sym(['U_' num2str(k)],nu);
    GOR_k = MX.sym(['GOR_' num2str(k)],nu);
    Upar = vertcat(Uk,GOR_k);
    w = {w{:},Upar};
    lbw = [lbw;lbu;GOR_val];
    ubw = [ubw;ubu;GOR_val];
    w0 = [w0;u0;GOR_val];

    % state at collocation points = s(\theta)
    Xkj = {};
    Zkj = {};

    % build diff and alg state vectors for each collocation point at time k
    for j = 1:d
        Xkj{j} = MX.sym(['X_' num2str(k) '_' num2str(j)],nx);
        Zkj{j} = MX.sym(['Z_' num2str(k) '_' num2str(j)],nz);
        w = {w{:},Xkj{j},Zkj{j}};
        lbw = [lbw;lbx;lbz];
        ubw = [ubw;ubx;ubz];
        w0 = [w0; dx0;z0];
    end

    % Loop over collocation points
    Xk_end = D(1)*Xk;

    for j = 1:d
        % Expression for the state derivative at the collocation point
        % \dot s(\theta_k,t) = \sum \theta_k,j*\dot P_k,j

        xp = C(1,j+1)*Xk;  % helper state

        for r = 1:d
            xp = xp + C(r+1,j+1)*Xkj{r};
        end

        [fj,zj,qj] =  f(Xkj{j},Zkj{j},Upar);

        g = {g{:},tf*fj-xp,zj};  % dynamics and algebraic constraints
        lbg = [lbg;zeros(nx,1);zeros(nz,1)];
        ubg = [ubg;zeros(nx,1);zeros(nz,1)];

        % Gas capacity constraints on all the collocation points
        g = {g{:},(Zkj{j}(17)+Zkj{j}(18))};
        lbg = [lbg;0];
        ubg = [ubg;QgMax];

        % Add contribution to the end states
        Xk_end = Xk_end + D(j+1)*Xkj{j};

        % Add contribution to the cost function
        J = J +(B(j+1)*qj*tf ) + 10.*sum((Uk_prev - Uk).^2);
        % second term penalizes MV moves
    end

    % control input at prev sampling time u_{k-1}
    Uk_prev = MX.sym(['Uprev_' num2str(k+1)],nu);
    Uk_prev = Uk; % set u_{k-1} = u_k before incrementing k

    % New NLP variable for state at end of interval
    % Additional DOF for shooting gap constraints
    Xk = MX.sym(['X_' num2str(k+1)], nx);

    w = {w{:},Xk};
    lbw = [lbw;lbx];
    ubw = [ubw;ubx];
    w0 = [w0; dx0];

    % Shooting Gap constraint
    g = {g{:},Xk_end-Xk};
    lbg = [lbg;zeros(nx,1)];
    ubg = [ubg;zeros(nx,1)];

    % constraint on the total gas lift rate
    g = {g{:},sum(Uk)};
    lbg = [lbg;0];
    ubg = [ubg;qGLMax];
end

% create and solve NLP solver
nlp = struct('x',vertcat(w{:}),'f',J,'g',vertcat(g{:}));

% Assign IPOPT solver
solver = nlpsol('solver','ipopt',nlp);
```

Code Snippets C.4: The closed-loop simulations of the NMPC

```
for sim_k = 1:nIter
    % Solve NLP
    sol = solver('x0',w0,'lbx',lbw,'ubx',ubw,'lbg',lbg,'ubg',ubg);

    % Extract Solution
    w_opt = full(sol.x);

    % extract solution from the Big solution array
    n_w_i = nx+nz+nu+nu+((nx+nz)*d+nx+nu+nu)*N; % size of solution vector
    u_opt1 = [w_opt((nx+nz+nu+nu+1):d*nz+(d+1)*nx+nu+nu:n_w_i);NaN];
    u_opt2 = [w_opt((nx+nz+nu+nu+2):d*nz+(d+1)*nx+nu+nu:n_w_i);NaN];

    % implement the first sample on the simulator
    u_in_1 = u_opt1(1,1);
    u_in_2 = u_opt2(1,1);
    u_in = [u_in_1;u_in_2;GOR_val];

    %% Simulator using IDAS integrator
    % Simulate Plant
    Fk = F('x0',dx0,'z0',z0,'p',u_in);

    % set new initial values for the next iteration
    dx0 =  full(Fk.xf);
    z0 = full(Fk.zf);
    u0 = [u_in_1;u_in_2;GOR_val];
    J_real(sim_k) = full(Fk.qf);

    %% Set new initial values for next iteration
    w0 = [];
    lbw = [];
    ubw = [];

    w0 = [w0;dx0;z0];
    lbw = [lbw;dx0;z0];
    ubw = [ubw;dx0;z0];
    w0 = [w0;u0];
    lbw = [lbw;u0];
    ubw = [ubw;u0];
    for i = 1:N
        w0 = [w0;u0];
        lbw = [lbw;lbu;GOR_val];
        ubw = [ubw;ubu;GOR_val];
        for d = 1:3
            w0 = [w0;dx0;z0];
            lbw = [lbw;lbx;lbz];
            ubw = [ubw;ubx;ubz];
        end
        w0 = [w0;dx0];
        lbw = [lbw;lbx];
        ubw = [ubw;ubx];
    end

end
```

Code Snippets C.5: Updating lambda for the decomposed NMPC

```
alpha =10;
lambda = 20.*ones(par.N,1);
for sim_k=1:N
t0 = 0.*lambda;
tol =1;
iter = 0;
while tol > 0.0001 && iter<5

        [sol1,par] = WellOptDyn(par,1,OptCase,lambda,init1);
        w_opt1 = full(sol1.x);
        w_pg1 = w_opt1([(nx+nz-3),(nx+nz-3)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);

        [sol2,par] = WellOptDyn(par,2,OptCase,lambda,init2);
        w_opt2 = full(sol2.x);
        w_pg2 = w_opt2([(nx+nz-3),(nx+nz-3)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);

        lambda = max(0,lambda + alpha*(w_pg1(1:end-1) + w_pg2(1:end-1) - 8.*ones(par.N,1)));
        tol = norm(abs(lambda - t0));
        iter = iter + 1;
        t0 = lambda;
end
...
```

Code Snippets C.6: Bulding of the nlp for the decomposed NMPC

```
function [sol,par] = WellOptDyn(par,wellNo,OptCase,t,init)
addpath ('C:\Users\IngvildMarie\Documents\CasADi\casadi-matlabR2014b-v3.1.0')
import casadi.*


[diff,alg,L,var] = WellModel(par,wellNo);
x_var = var.x;
z_var = var.z;
p_var = var.p;
dae = struct('x',x_var,'z',z_var,'p',p_var,'ode',diff,'alg',alg,'quad',L);

f = Function('f',{x_var,z_var,p_var},{diff,alg,L},{'x','z','p'},{'xdot','zj','qj'});

%% Optimization and Simulation
 it = 1;

    % empty nlp
    w = {};
    w0 = [];
    lbw = [];
    ubw = [];
    J = 0;
    g = {};
    lbg = [];
    ubg = [];

    for js = 1:nS
        GOR_val = GOR_case(wellNo,js);

        % initial conditions for each scenario
        X0 = MX.sym('X0',nx);
        Z0 = MX.sym('Z0',nz);
        w = {w{:}, X0,Z0};
        lbw = [lbw; dx0;z0];
        ubw = [ubw; dx0;z0];
        w0 = [w0; dx0;z0];

        % Formulate NLP
        Xk = X0;
        Xkj = {};
        Zkj = {};
        Uk_prev = u0;
        for k = 0:N-1

            Uk = MX.sym(['U_' num2str(k) '_' num2str(js)],nu);
            GOR_k = MX.sym(['GOR_' num2str(k) '_' num2str(js)],nu);
            Upar = vertcat(Uk,GOR_k);
            w = {w{:},Upar};
            lbw = [lbw;lbu;GOR_val];
            ubw = [ubw;ubu;GOR_val];
            w0 = [w0;u0;GOR_val];

            % state at collocation points = s(\theta)
            Xkj = {};
            Zkj = {};

            for j = 1:d
                Xkj{j} = MX.sym(['X_' num2str(k) '_' num2str(j) '_' num2str(js)],nx);
                Zkj{j} = MX.sym(['Z_' num2str(k) '_' num2str(j) '_' num2str(js)],nz);
                w = {w{:},Xkj{j},Zkj{j}};
                lbw = [lbw;lbx;lbz];
                ubw = [ubw;ubx;ubz];
                w0 = [w0; dx0;z0];
            end

```

```
% Loop over collocation points
Xk_end = D(1)*Xk;

for j = 1:d
    % Expression for the state derivative at the collocation point
    % \dot s(\theta_k,t) = \sum \theta_k,j*\dot P_k,j
    xp = C(1,j+1)*Xk;  % helper state

    for r = 1:d
        xp = xp + C(r+1,j+1)*Xkj{r};
    end

    [fj,zj,qj] =  f(Xkj{j},Zkj{j},Upar);

    g = {g{:},tf*fj-xp,zj};  % dynamics and algebraic constraints
    lbg = [lbg;zeros(nx,1);zeros(nz,1)];
    ubg = [ubg;zeros(nx,1);zeros(nz,1)];

    if wellNo == 1
        qgWell = 0;
    else
        qgWell = 8;
    end
    % Add contribution to the end states
    Xk_end = Xk_end + D(j+1)*Xkj{j};
    if k == N-1
         J = J + t(k+1)*(Zkj{j}(9) - qgWell) + wS(js).*(B(j+1)*qj*tf )+ 10.*sum((Uk_prev - Uk).^2) ;
    else
        J = J + t(k+1)*(Zkj{j}(9) - qgWell) + wS(js).*(B(j+1)*qj*tf )+ 10.*sum((Uk_prev - Uk).^2)  ;
    end

end

Uk_prev = MX.sym(['Uprev_' num2str(k+1)],nu);
```

```matlab
            Uk_prev = Uk;

            % New NLP variable for state at end of interval
            Xk = MX.sym(['X_' num2str(k+1) '_' num2str(js)], nx);
            w = {w{:},Xk};
            lbw = [lbw;lbx];
            ubw = [ubw;ubx];
            w0 = [w0; dx0];

            % Shooting Gap constraint
            g = {g{:},Xk_end-Xk};
            lbg = [lbg;zeros(nx,1)];
            ubg = [ubg;zeros(nx,1)];

            g = {g{:},sum(Uk)};
            lbg = [lbg;0];
            ubg = [ubg;par.qGLMax];

            if k < nR
                u_ant{js,k+1} = MX.sym(['u_ant_' num2str(k) '_' num2str(js)],nu);
                u_ant{js,k+1} = Uk;
            end
        end
    end

    % Add Non-anticipativity constraints
    for k = 1:nR
        for js = 1:nS-1
            if ~isnan(nonant(js,k))
                if nonant(js) == nonant(js+1)
                    g = {g{:},(u_ant{js,k} - u_ant{js+1,k})};
                    lbg = [lbg;zeros(nu,1)];
                    ubg = [ubg;zeros(nu,1)];
                end
            end
        end
    end

    % create and solve NLP solver
    nlp = struct('x',vertcat(w{:}),'f',J,'g',vertcat(g{:}));
    solver = nlpsol('solver','ipopt',nlp);
    sol = solver('x0',w0,'lbx',lbw,'ubx',ubw,'lbg',lbg,'ubg',ubg);
```

Code Snippets C.7: Updating the lambda with the line search method

```
lambda = 20.*ones(par.N,1);
alpha =10;
c=0.1.*ones(par.N,1);
rho=0.7;
for sim_k=1:N
t0 = 0.*lambda;
tol =1;
iter = 0;
alpha =10;
while tol > 0.15 && iter<10

    [sol1,par] = WellOptDyn(par,1,OptCase,lambda,init1);
    w_opt1 = full(sol1.x);
    u_opt1 = [w_opt1((nx+nz+1):d*nz+(d+1)*nx+nu:n_w_i);NaN];
    w_po1 = w_opt1([(nx+nz-2),(nx+nz-2)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);
    w_pg1 = w_opt1([(nx+nz-3),(nx+nz-3)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);

    [sol2,par] = WellOptDyn(par,2,OptCase,lambda,init2);
    w_opt2 = full(sol2.x);
    u_opt2 = [w_opt2((nx+nz+1):d*nz+(d+1)*nx+nu:n_w_i);NaN];
    w_po2 = w_opt2([(nx+nz-2),(nx+nz-2)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);
    w_pg2 = w_opt2([(nx+nz-3),(nx+nz-3)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);

    lambda = max(0,lambda + alpha*(w_pg1(1:end-1) + w_pg2(1:end-1) - 8.*ones(par.N,1)));
    tol = norm(abs(lambda - t0));
    %tol=lambda(1)-t0(1);
    iter = iter + 1;
    t0 = lambda;

    %Updating alpha
    wpo=w_po1(1:end-1,1) + w_po2(1:end-1,1);
    wpg=(w_pg1(1:end-1,1) + w_pg2(1:end-1,1) - 8.*ones(par.N,1));
    Jm=wpo + lambda.*wpg;
    pk=-1*wpg;
    gradJm=wpg';
    Jmp=wpo + (lambda+alpha.*pk).*wpg;
    rhs=Jm + c * alpha.*gradJm*pk;
    lstol=Jmp-rhs;
    lstol=sum(lstol(30));
    while lstol>0
        alpha=rho*alpha;
        Jmp=wpo + (lambda+alpha.*pk).*wpg;
        rhs=Jm + c * alpha.*gradJm*pk;
        lstol=Jmp-rhs;
        lstol=round((lstol(30)));
    end
end
...
```

Code Snippets C.8: Updating the lambda with the augmented lagrangian method

```
lambda = 20.*ones(par.N,1);
alpha =10;
my=2;
beta=1.03;
for sim_k=1:N
t0 = 0.*lambda;
tol =1;
iter = 0;
my=2;
while tol > 0.15 && iter<5

    [sol1,par] = WellOptDyn_Aug(par,1,OptCase,lambda,my,h2,init1);
    w_opt1 = full(sol1.x);
    w_pg1 = w_opt1([(nx+nz-3),(nx+nz-3)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);
    h1=w_pg1(1:end-1);

    [sol2,par] = WellOptDyn_Aug(par,2,OptCase,lambda,my,h1,init2);
    w_opt2 = full(sol2.x);
    w_pg2 = w_opt2([(nx+nz-3),(nx+nz-3)+(nx+nz+nu):d*nz+(d+1)*nx+nu:n_w_i]);
    h2=w_pg2(1:end-1);

    lambda = max(0,lambda + my.*(w_pg1(1:end-1) + w_pg2(1:end-1) - 8.*ones(par.N,1)));
    my=beta*my;
     tol = norm(abs(lambda - t0));
     iter = iter + 1;
     t0 = lambda;
end
...
```

Code Snippets C.9: The cost function with the added penalty from the augmented lagrangian

```
if wellNo == 1
    qgWell = 0;
    h2=h(k+1)-qgWell;
else
    qgWell = 8;
    h2=h(k+1);
end
% Add contribution to the end states
Xk_end = Xk_end + D(j+1)*Xkj{j};
if k == N-1
      J = J + t(k+1)*(Zkj{j}(9) - qgWell) + wS(js).*(B(j+1)*qj*tf )+ (m/2)*((Zkj{j}(9)+h2).^2) + 10.*sum((Uk_prev - Uk).^2) ;
else
    J = J + t(k+1)*(Zkj{j}(9) - qgWell) + wS(js).*(B(j+1)*qj*tf ) + (m/2)*((Zkj{j}(9)+h2).^2) + 10.*sum((Uk_prev - Uk).^2)  ;
end
```

Appendix D

Line Search Method Plots

D.1 Tuning the Parameter c

The development of the iterations and α over the time horizon for run LS.1 is shown below in fig. D.1. The iterations run until the change in λ reaches a set tolerance=0.15 or the number of iterations exceed the maximum iteration cap of ten iterations. Run LS.1 below never converges to the set tolerance, and therefore the iterations for all time steps are 10. The α is the step length, which is adjusted by the line search method. α varies sporadically for each run, with no visible trend.

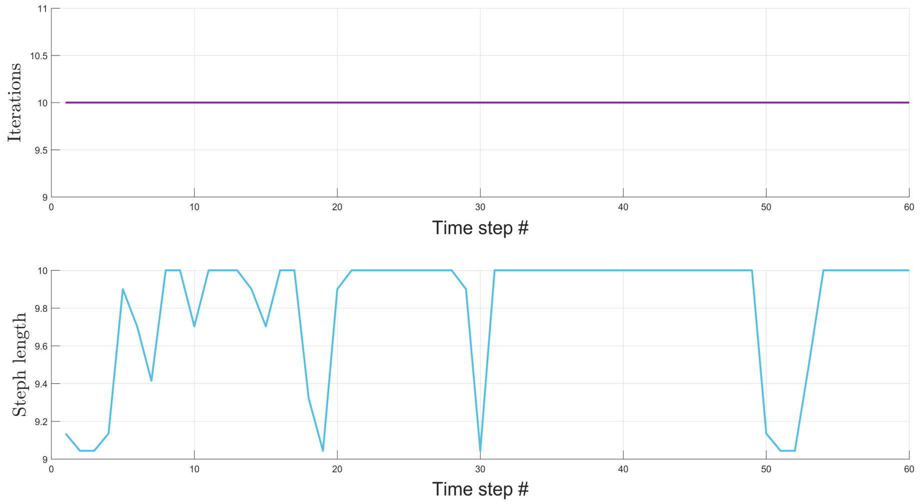


Figure D.1: The development of the iterations and the step length α over the time horizon for run LS.1.

The development of the iterations and α over the time horizon for run LS.2 is shown below in fig. D.2. The iterations reach the maximum iterations for the first half of the time horizon, when the number of iterations steadily decrease until stabilizing on two iterations needed for each time step. The step length α mirrors the iterations trend, being small at first, then not updating a lot. This is natural, as when the total gas lift approaches the optimal solution, α does not need to update a lot for the set value of 10.

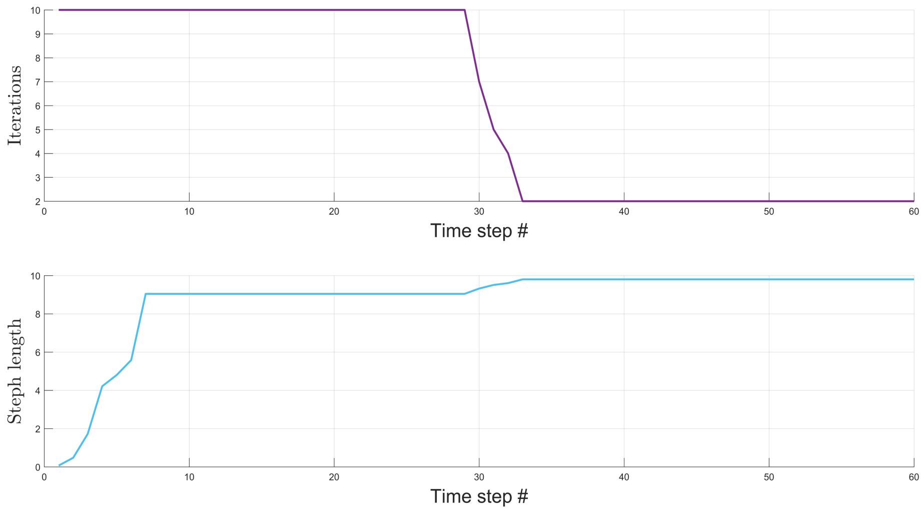


Figure D.2: The development of the iterations and the step length α over the time horizon for run LS.2.

The development of the iterations and α over the time horizon for run LS.3 is shown below in fig. D.3. The iterations reach maximum for all time steps, and the α updates less after the first 10 time steps.

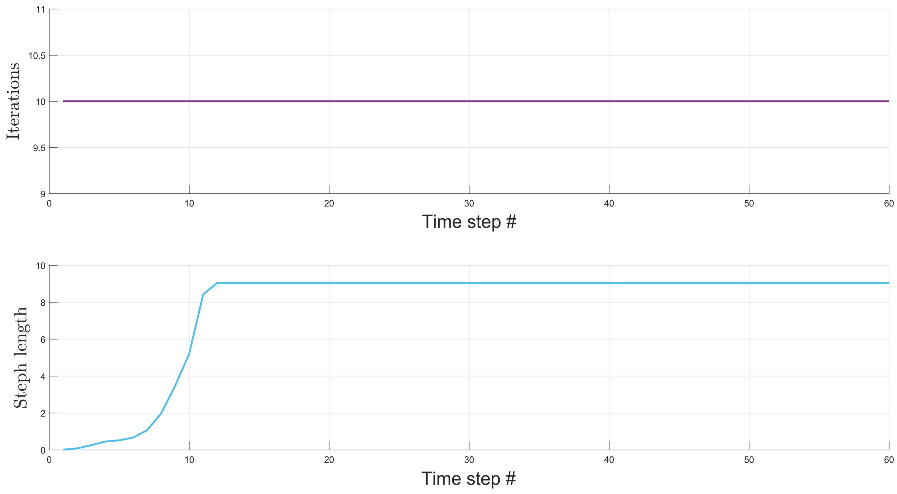
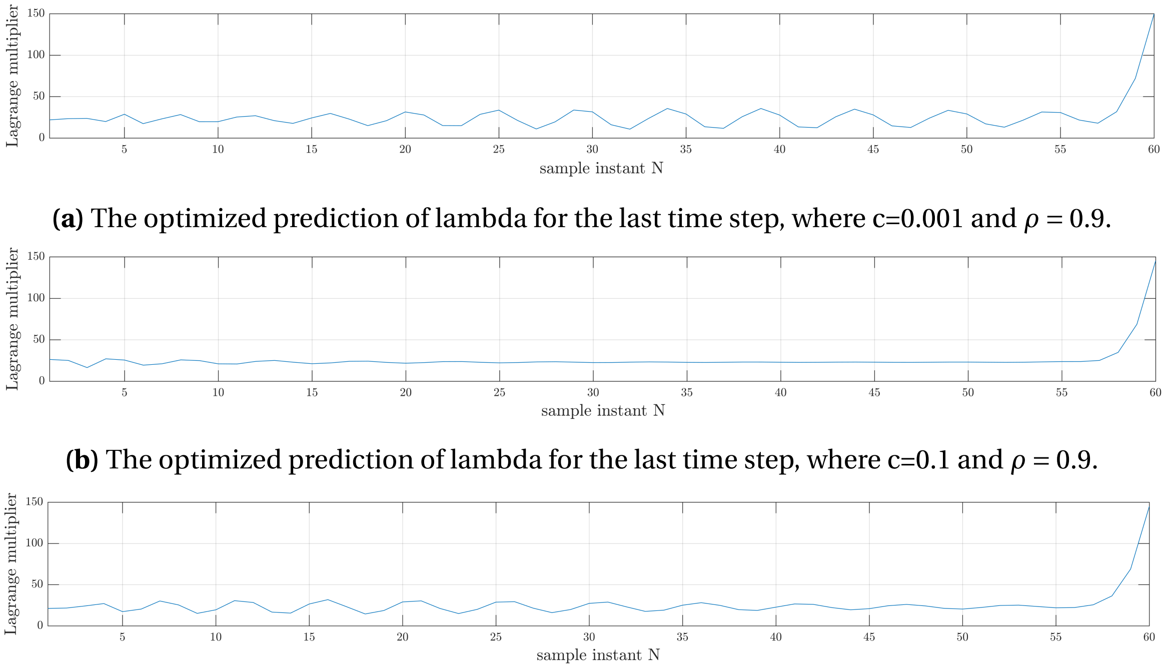


Figure D.3: The development of the iterations and the step length α over the time horizon for run LS.3.

From this it can be seen that run LS.1 and run LS.2 reach the maximum iteration cap each time step.

D.1.1 Tuning Parameter c with $\rho = 0.9$

When checking the tuning of c, three runs with the same values of c where performed with a different value of ρ. For the runs shown in fig. D.4 below $\rho = 0.9$, while c was varied with the values of 0.001, 0.1 and 0.5. This tuning of c with a different ρ yields the same results, and the same conclusion as for the other runs LS.1, LS.2 and LS.3 tuning the parameter c.



(a) The optimized prediction of lambda for the last time step, where c=0.001 and $\rho = 0.9$.

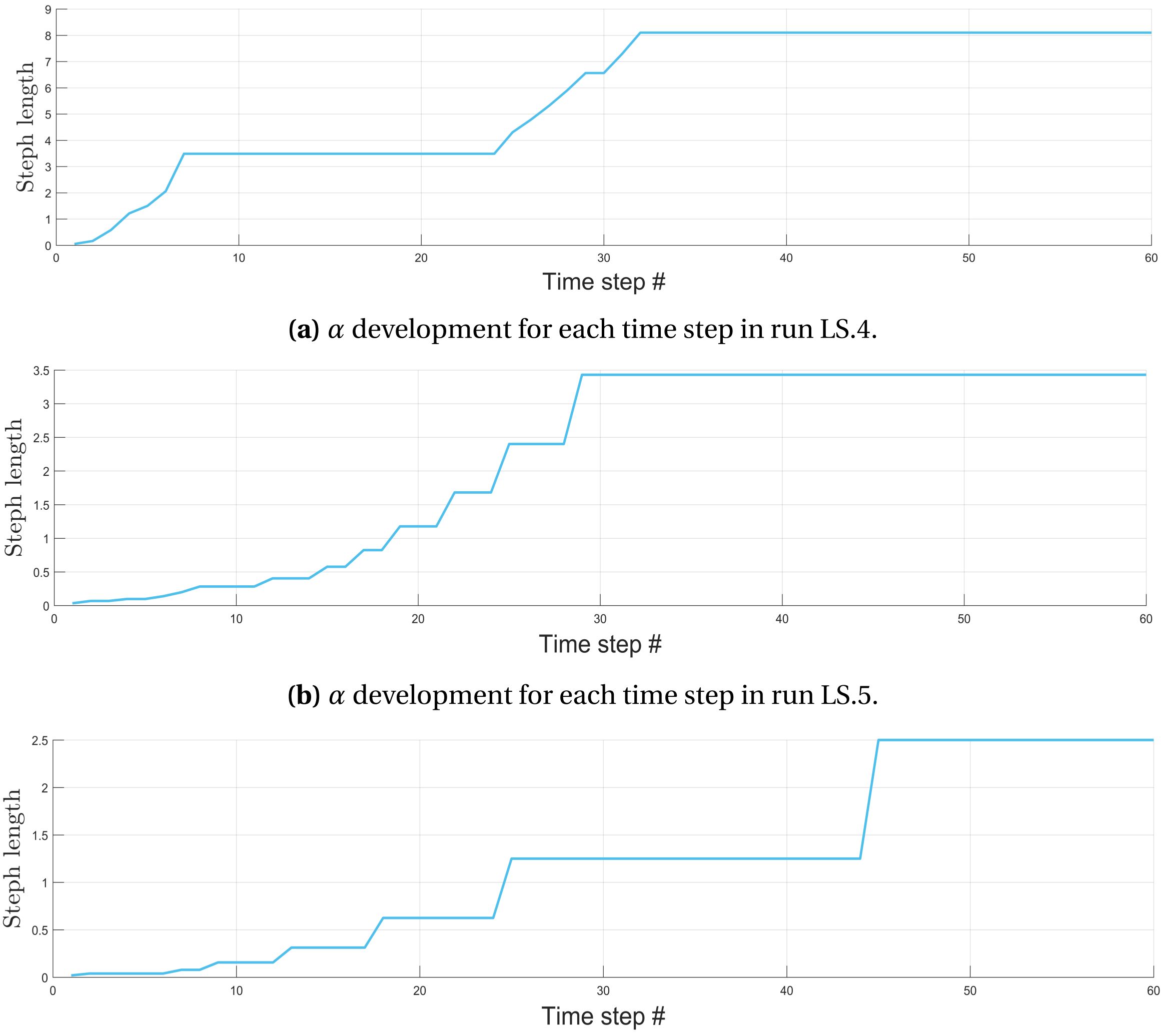
(b) The optimized prediction of lambda for the last time step, where c=0.1 and $\rho = 0.9$.

(c) The optimized prediction of lambda for the last time step, where c=0.5 and $\rho = 0.9$.

Figure D.4: The comparison of the lambda prediction horizon over sample instants N for the last time step. The results were from an extra run performed for the tuning of parameter c, with a different constant $\rho = 0.9$.

D.2 Tuning the Parameter ρ

The alpha development in the runs tuning ρ, LS.4,LS.5 and LS.6, are shown below in appendix D.2. The development of α is a mirroring of the iterations development, which as explained above is from the fact that α needs less updating close to the optimal solution.



(a) α development for each time step in run LS.4.

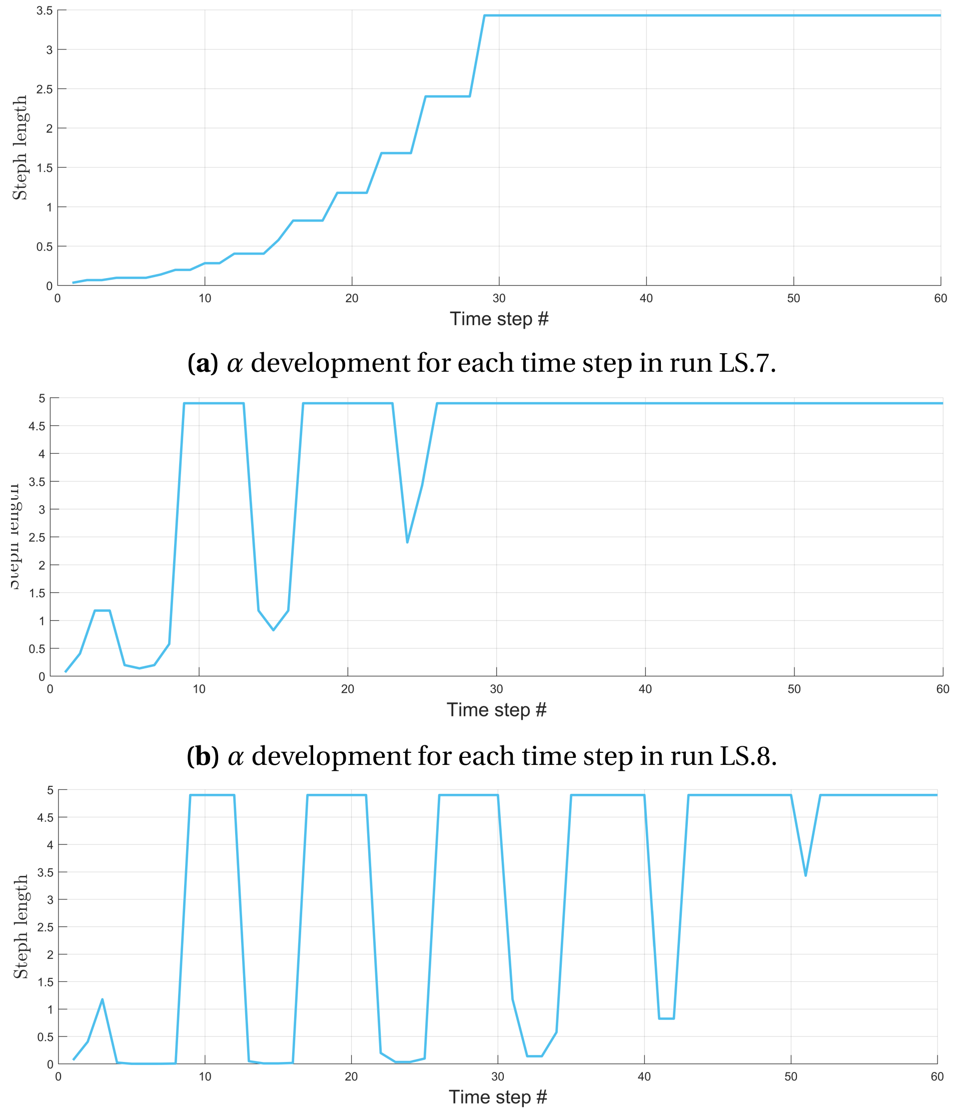
(b) α development for each time step in run LS.5.

(c) α development for each time step in run LS.6.

Figure D.5: The development of the α for each time step in the simulation for run LS.4, LS.5 and LS.6.

D.3 Line Search method Without Maximum Iterations

The α development for the runs performed without a cap on the iterations, LS.7,LS.8 and LS.9, are shown below in appendix D.2. The development of α is a mirroring of the iterations development, which as explained above is from the fact that α needs less updating close to the optimal solution. The dips in the α development that can be seen below, are the opposite of the peaks in the iteration development, due to the new method of measuring tolerance.



(a) α development for each time step in run LS.7.

(b) α development for each time step in run LS.8.

(c) α development for each time step in run LS.9.

Figure D.6: The development of the α for each time step in the simulation.

D.3.1 Run Aug.6 with the Measuring Tolerance

For run 6, which never reached the maximum iterations cap throughout the simulation, a run with the new measuring method for tolerance was performed. The run is shown below in fig. D.7. The new measuring method calculates the change in only the first λ on the prediction horizon, tol=$\lambda(1)_{k+1} - \lambda(1)_k$. This result is not included with the other runs, however, it had fewer iterations than any of the runs. This confirms that the method where tolerance is measured for only the first sample instant in the prediction horizon is the best for lowest number of iterations needed for simulation.

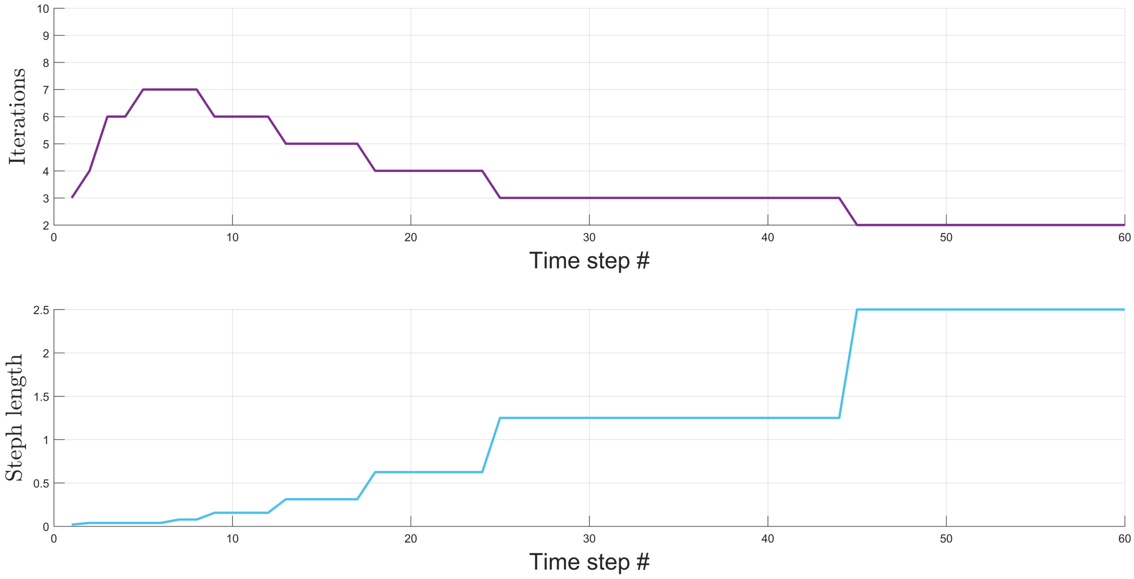


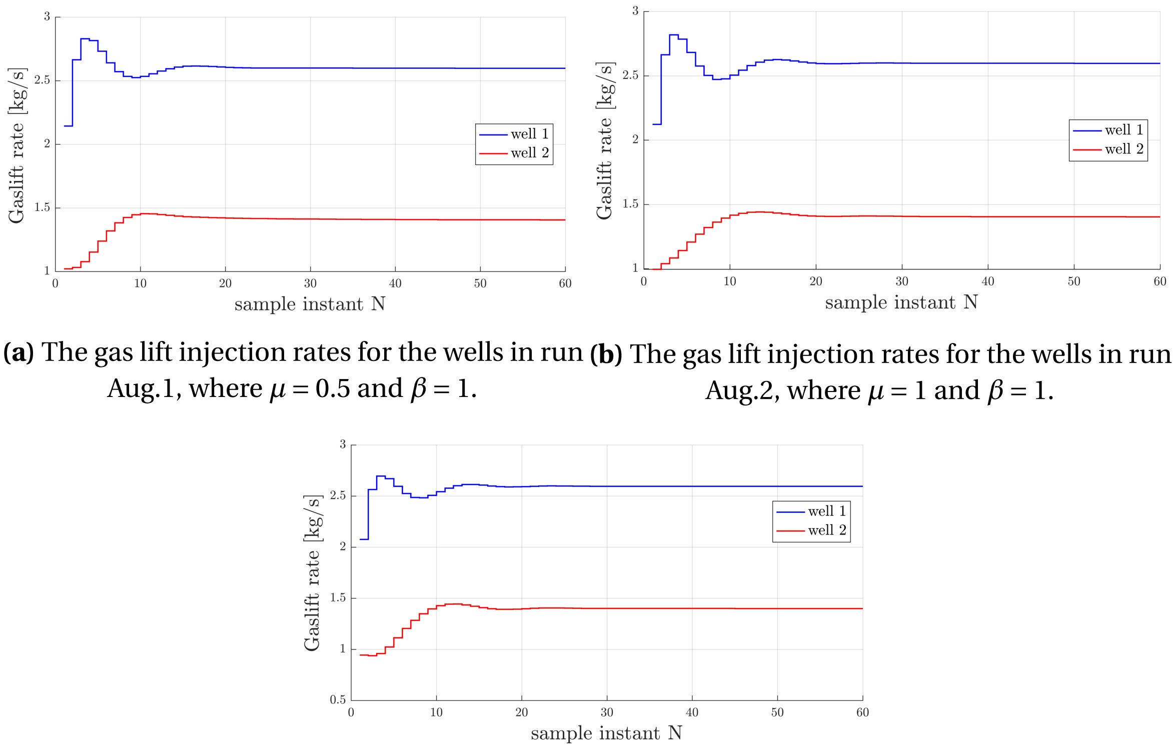
Figure D.7: The development of iterations and α for each time step throughout the simulation for the extra run, a modified run LS.6 with the new method of measuring tolerance.

Appendix E

Augmented Lagrangian Method Plots

E.1 Tuning the Parameter μ

The gas lift injection rate for run Aug.1, Aug.2 and Aug.3 are shown below in fig. E.1.



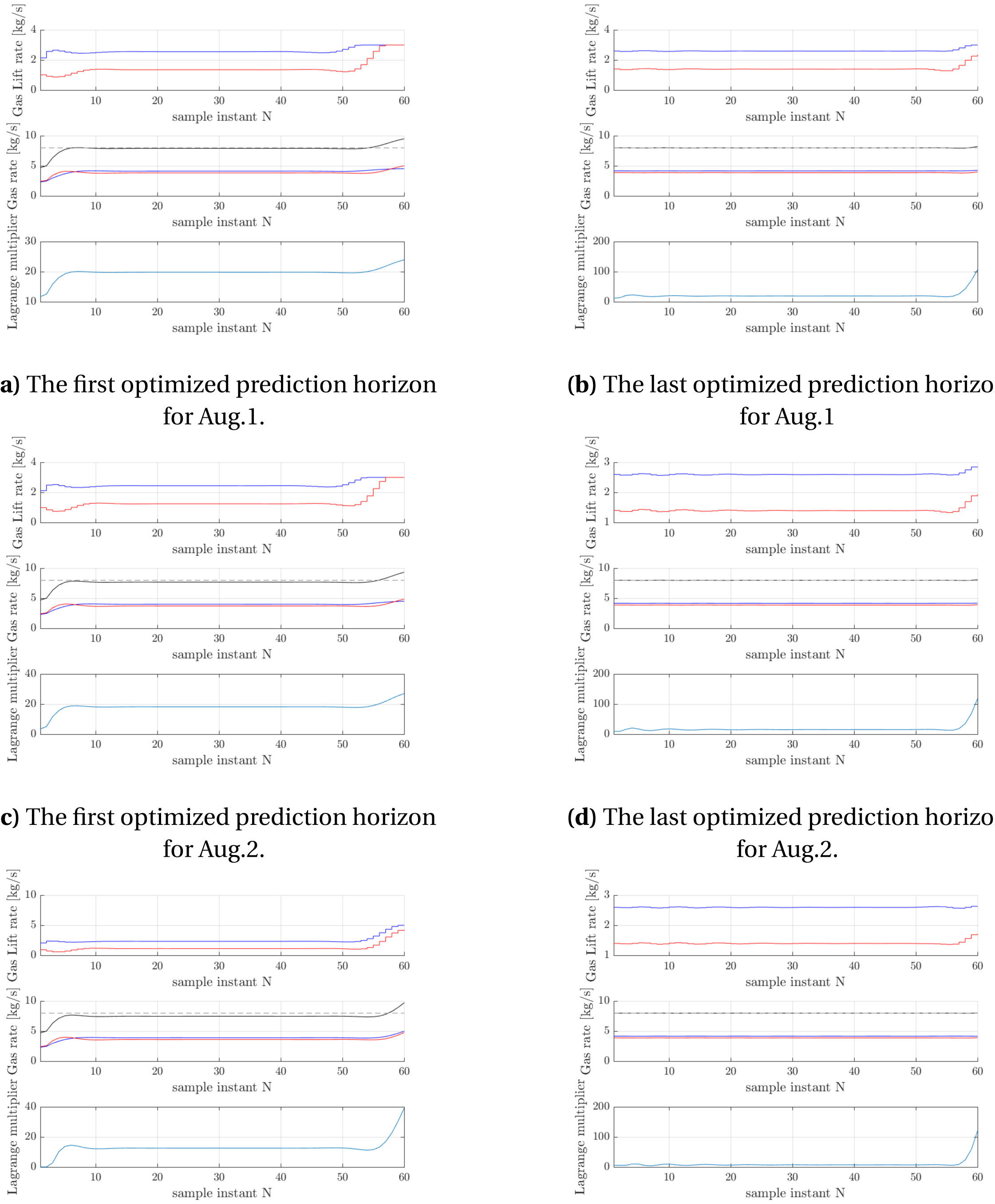
(a) The gas lift injection rates for the wells in run Aug.1, where $\mu = 0.5$ and $\beta = 1$.

(b) The gas lift injection rates for the wells in run Aug.2, where $\mu = 1$ and $\beta = 1$.

(c) The gas lift injection rates for the wells in run Aug.3, where $\mu = 2$ and $\beta = 1$.

Figure E.1: The gas lift rate for each well from the augmented lagrangian decomposition NMPC is shown up-close, for run Aug.1-Aug.3 to illustrate the trajectory of the simulation.

The prediction horizons for the first three runs for augmented lagrangian decomposition are shown in fig. E.2. For run Aug.1 the first and the last optimized prediction horizon are shown fig. E.2a and fig. E.2b, for run Aug.2 the first and the last optimized prediction horizon are shown in fig. E.2c and fig. E.2d and for run Aug.3 the first and the last optimized prediction horizon are shown in fig. E.2e and fig. E.2f.



(a) The first optimized prediction horizon for Aug.1.

(b) The last optimized prediction horizon for Aug.1

(c) The first optimized prediction horizon for Aug.2.

(d) The last optimized prediction horizon for Aug.2.

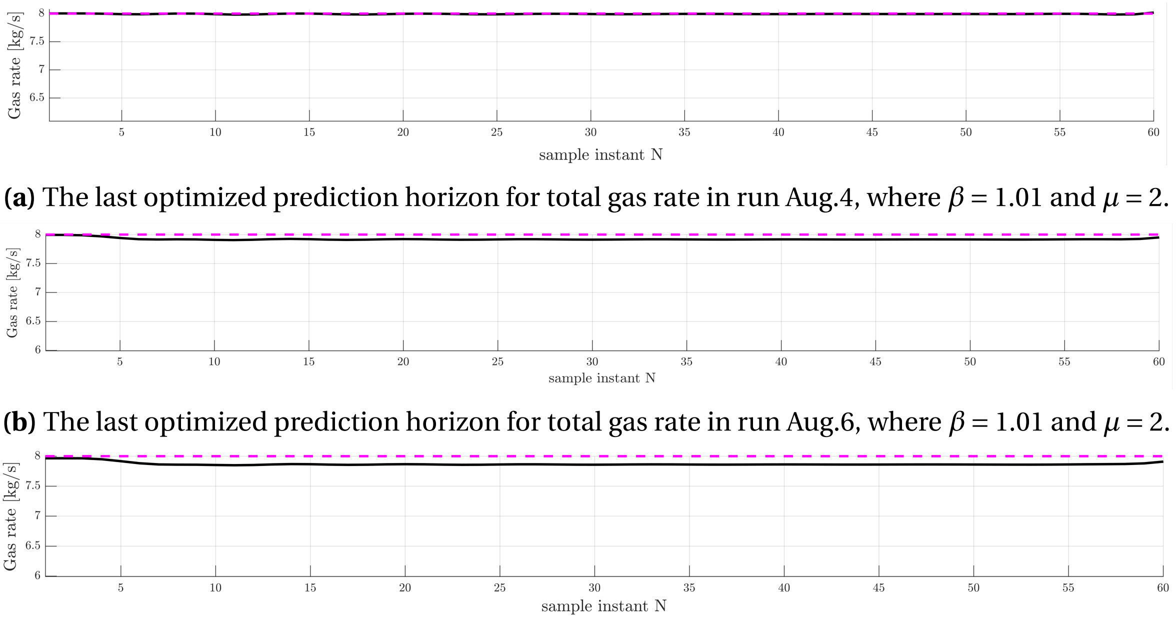
(e) The first optimized prediction horizon for Aug.3.

(f) The last optimized prediction horizon for Aug.3.

Figure E.2: The first and last optimized prediction horizon for run Aug.1-Aug.3.

E.2 Tuning the Parameter β

The last prediction horizon of the total gas rate from both wells are shown below in fig. E.3. They are included to show the deviation of the prediction from the optimal value. It can be seen that the last optimized prediction horizon does not reach the active constraint in run Aug.5 and Aug.6, and the runs are therefore not valid.



(a) The last optimized prediction horizon for total gas rate in run Aug.4, where $\beta = 1.01$ and $\mu = 2$.

(b) The last optimized prediction horizon for total gas rate in run Aug.6, where $\beta = 1.01$ and $\mu = 2$.

(c) The last optimized prediction horizon for total gas rate in run Aug.6, where $\beta = 1.01$ and $\mu = 2$.

Figure E.3: The last optimized prediction horizons for the total gas rate in run Aug.4-Aug.6, where β is tuned. The solid black line is the total gas rate over the sample instants and the dashed pink line marks the gas capacity constraint at 8 kg/s.