NTNU
Innovation and Creativity

# Pipeline Liquid Control using Nonlinear MPC and OLGA

Optimal Utilization of Available Liquid Buffer Volume during Pipeline Transients

**Håvard Torpe**

Master of Science in Engineering Cybernetics

| | |
|---|---|
| Submission date: | June 2007 |
| Supervisor: | Ole Morten Aamo, ITK |
| Co-supervisor: | John-Morten Godhavn, Statoil |

Norwegian University of Science and Technology
Department of Engineering Cybernetics

## Problem Description

At the Hammerfest LNG plant, a huge slug catcher is installed for receiving gas, condensate and water from the Snøhvit field pipeline. The purpose of the slug catcher is to separate gas, condensate and water and to buffer varying gas and liquid rates from the pipeline. The gas flow from the slug catcher should be as constant and predictable as possible for optimal operation of the LNG plant. The sizing of the slug catcher is based on handling the liquid flow from a desired fast start-up of production after a set of selected scenarios with low or no production. The question raised in this exercise is whether the slug catcher size could have been reduced by applying nonlinear model predictive control (NMPC) using some combination of input. Possible manipulated variables include: the subsea choke (at the pipeline inlet), the topside choke (at the pipeline outlet) and the gas rate out of the separator. A simple feedback scheme for liquid flow control was developed, implemented and tested in a project. The results were promising, but a potential for more advanced control was identified.

1. Literature review on MPC with particular focus on NMPC.
2. Implement an NMPC algorithm in the in-house Statoil MPC tool, SEPTIC. Also test it on an example to demonstrate the implemented algorithm’s performance.
3. Develop an interface between SEPTIC and OLGA
4. Implement an NMPC, using OLGA for predictions and as true process (no model errors) and test it on selected cases.
5. Consider the strengths and weaknesses of different combinations of manipulated variables and controlled variables.

Assignment given: 08. January 2007
Supervisor: Ole Morten Aamo, ITK

# Summary

PIPELINES with multiphase flow will exhibit large and highly nonlinear liquid rates during transients caused by changes in production rate. This requires either a large separator (slug catcher) downstream, capable of handling all disturbances, or, some sort of control of the rate change in order to ensure that the downstream processing equipment can cope with the disturbances. In this report, it is proposed to use smart liquid control to optimally control the production during rate changes. The term smart liquid control was coined to describe the use of nonlinear model predictive control (NMPC) and a nonlinear pipeline model to control liquid levels.

The pipeline from the Snøhvit field to the Hammerfest LNG plant was used as a test case on which to implement smart liquid control. The technique was proved using a simplified OLGA pipeline model for both predictions and as process, i.e. perfect model. Good results were shown, especially when compared to manual control. Smart liquid control therefore presents a possibility to reduce separator size for new projects. Another highly interesting use of smart liquid control is to optimize the use of separator buffer capacity in order to maximize production.

The NMPC method used in this report, a single shooting multistep quasi-Newton method, was elaborated on in a literature review chapter. Also an implementation of the method was made and described in the context of NMPC. The implementation of the algorithm was made in the in-house STATOIL MPC-tool, SEPTIC. In order to assess the performance of the algorithm, it was tested on a small continuously stirred tank reactor system and the results reported and discussed.

# Acknowledgments

THIS report and the work presented in it, was done for the degree of Master of Science in Engineering Cybernetics, during the period from January to June 2007. The assignment was given in cooperation between the Norwegian University of Science and Technology (NTNU) and the STATOIL research center at Rotvoll.

During the course of this work, I have had help from many people. I would like to start by thanking my supervisor at NTNU, professor Ole Morten Aamo and my supervisors at STATOIL, Dr.Ing. John-Morten Godhavn and Dr.Ing Petter Tøndel for excellent guidance.

Apart from my supervisors, several people at STATOIL is due thanks. I would like to mention the programming help from Staff engineer Morten Fredriksen, the help on the Hammerfest LNG plant from Senior engineer Silja Eriksson Gylseth and especially the invaluable help on MPC in general and the STATOIL MPC tool, SEPTIC, in particular from MCP specialist Dr.Ing. Stig Terje Strand.

A thanks is also due to STATOIL which let me sit at their research center at Rotvoll during the course of this work which, without doubt, helped both the pace of the work as well as the quality of it. Extended access to the research center and programs was generously granted by the research manager for the process control section, Dr.Ing. Gunleiv Skofteland.

Finally, I would like to thank my fellow master students sharing an office at STATOIL, John Petter Jensen for many fruitful discussions and Patrick Meum for the collaboration on the NMPC algorithm and Chapters 2-3.

# Contents

# Chapter 1

# Introduction

THE problem addressed in this report is liquid handling during production rate transients in pipelines carrying multiphase flow. During the transient, liquid rates can become much larger than the nominal rates, and will in general exhibit nonlinear responses to the rate change. The cause of the variation in liquid rates is a phenomenon called entrainment. This is when one fluid is moved by another. In the pipeline investigated, water and oil are entrained by the gas. When the gas velocity is low during periods with low production, the ability of the gas to carry liquid along is reduced. Thus, liquid is accumulated in the pipeline until a new equilibrium is reached. The liquid handling challenge arises when the production rate increases, resulting in greater entrainment because of the increased gas velocity. This causes the accumulated surplus of liquid to be discharged from the pipeline, transiently increasing liquid rates.

The goal of this work is to design a control system that optimizes the use of available separator volume during such pipeline transients. The utilization of separator volume is interesting for two reasons. Firstly, because of the large separator volumes required if no control is used. Large separators require both large areas for construction as well as a large initial investment. From an economic point of view, large expenses before revenues are generated are unfortunate for the net present value. Reducing the size and cost of the separator thus presents a possible increase of the profitability of a development. Secondly, building a separator large enough to handle the liquid flows without control are often only an option onshore. Offshore, the price of building a large separator is much greater as space is a limited resource. Here, using the available separator volume effectively means a direct increase in production, thus also increasing profits.

Due to the nonlinear liquid rate responses, a nonlinear model predictive control scheme (NMPC) is used for liquid control. An OLGA model is used as pipeline model. OLGA is a dynamic multiphase flow simulator (Scandpower, 2006). The exact same model is used for both for predictions as well as process, i.e. zero plant-model mismatch. Although artificial, the setup with perfect model demonstrates the best performance that can be expected. Model error will in general reduce the performance achieved, and must be addressed in order to obtain good control.

*Smart liquid control* is introduced as a name for using NMPC and a nonlinear pipeline model for liquid control and tested using different controller setups. One controller setup to be tested introduces a choke upstream the separator as an extra input variable. The effect of the extra input will be discussed. The NMPC will be implemented in STATOIL's in-house MPC tool, SEPTIC. SEPTIC will be enhanced with an implementation of a single shooting multistep quasi-Newton method (SSMQN) algorithm based on the paper by Oliveira and Biegler (1995).

A model of the Snøhvit pipeline was provided by STATOIL and will be used for testing smart liquid control. The pipeline transports gas, oil and water from the Snøhvit field to the Hammerfest LNG plant. Flow assurance for this system and separator dimensioning is addressed an internal STATOIL report (Knutson, 2005) and in one engineering report by BMH Eagleton (Eagleton, 2001). Two cases are in these reports identified as the most difficult cases for liquid handling: pipeline rate ramp-up and pipeline start-up. These two cases will form the basis for the tests performed. Similar tests using PI-controllers for liquid control of the Snøhvit pipeline was performed in Torpe (2006). Here, NMPC was identified as a possibility for improving control.

In the following, a brief description of the different chapters for quick reference. Chapters 2 and 3 as well as the implementation of the SSMQN method are written in collaboration with Patrick Meum, who has used the same method to perform reservoir optimization on a reservoir model in ECLIPSE. The interested reader are referred to Meum (2007).

- *Chapter 2* gives a brief introduction to MPC and NMPC together with references to sources with additional information.
- *Chapter 3* narrows the perspective to the NMPC algorithm used in this report, a *single shooting multistep quasi-Newton* method. The chapter starts by giving a short historic introduction, before describing the steps in the algorithm. It concludes by discussing the implementation developed for SEPTIC and shows the performance of the method on a continuously stirred tank reactor system.
- *Chapter 4* is a description of Hammerfest LNG, which is used as a test system for implementation of smart liquid control.
- *Chapter 5* contains a short review of multiphase flow, intended to give the unfamiliar reader the background needed to assess the results given later in the report.
- *Chapter 6* describes the problem in greater detail, as well as the setup used for simulations. Some practical aspects of NMPC implementation is discussed in the context of the specific setups.
- *Chapter 7* give the results obtained using smart liquid control. The two scenarios simulated are described, and results given for different sizes of separator size. Also, the different controller setups (described in Chapter 6) are tested and compared. Finally, smart liquid control is compared to manual control.
- *Chapter 8* discusses the different control setups and the benefits of introducing a topside choke. Also, some conclusions on the subject of when to use smart liquid control are drawn based on results obtained.

- *Chapter 9* contains the conclusion and summarizes the results found in this report.
- *Chapter 10* proposes several issues to be addressed in future work.
- *Appendix A* contains a description of the library developed for communication between SEPTIC and OLGA

# Chapter 2

# Model Predictive Control: A Short Survey

Model predictive control (MPC) is one of today's most commonly used techniques within advanced control. Although MPC is the widely used term and by now the conventional name of this technique, one can also find the term receding horizon control (RHC) used on occasions in older literature. For simplicity this report will mostly refer to the former term, though RHC will be used when found appropriate. Mayne, Rawlings, Rao and Stockaert (2000) describes MPC as:

> a form of control in which the current control action is obtained by solving *on-line*, at *each* sampling instant, a finite horizon open-loop optimal control problem.

By repeatedly solving the open-loop control sequence at each time step, the controller has an inherently closed-loop effect. As the new control sequence is calculated from present state of the system[1], a stabilizing feedback control can be obtained. This is where this technique has its main difference from other pre-computed, optimal control laws. The control law is calculated for a given horizon, $T_c$, and the dynamic behaviour of the system is predicted for a horizon $T_p$, where $T_c \leq T_p$. As the controller moves forward in time, so does the horizon (hence the mentioned term receding horizon control). The basic idea of this is illustrated in Figure 2.1. A system is sought to be controlled at a set point, given by $r(t)$. The controller calculates an optimal input sequence, parameterized as a piecewise constant function of time, for the control horizon. As time progresses all horizons are moved ahead as well, so they slide along by one sampling interval at each step.

The MPC is now described in a more formal, mathematical formulation, following notation from Allgöwer, Findeisen and Nagy (2004). Consider a general class of continuous time systems described by some differential equation

$$\dot{x}(t) = f(x(t), u(t)), \quad x(t_0) = x_0 \tag{2.1}$$

[1]In case of the system state not being fully measured, but estimated in an observer, the best estimate available is used as the basis for the calculation

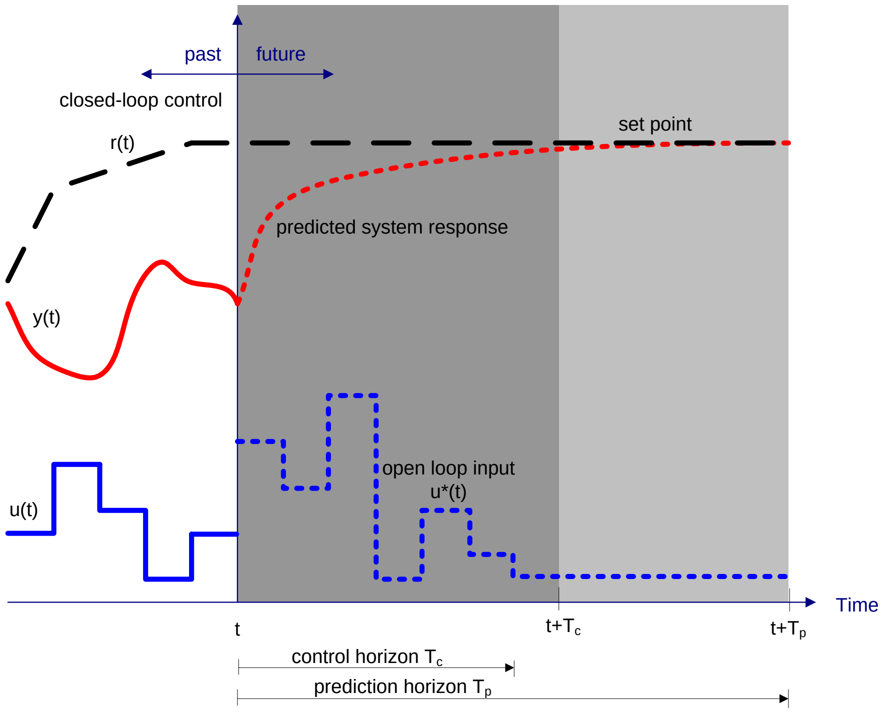

Figure 2.1: Principle of MPC

which is subject to input and state constraints of the form:

$$u(t) \in U, \, \forall \, t \geq t_0, \tag{2.2}$$

$$x(t) \in X, \, \forall \, t \geq t_0. \tag{2.3}$$

The inputs are given in the vector $u(t) \in \mathbf{R}^m$ and $x(t) \in \mathbf{R}^n$ denote the state vector, and the sets $U$ and $X$ are assumed to satisfy necessary topological properties. The optimal open-loop control is given by solving problem at every time instant:

$$\min_{\bar{u}(\cdot)} J(x(t), \bar{u}(\cdot)) = \int_t^{t+T_p} F(\bar{x}(\tau), \bar{u}(\tau)) \mathrm{d}\tau \tag{2.4a}$$

subject to

$$\dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \; \bar{x}(t) = x(t) \tag{2.4b}$$

$$\bar{u}(\tau) \in U, \, \forall \, \tau \in [t, t + T_c] \tag{2.4c}$$

$$\bar{u}(\tau) = \bar{u}(t + T_c), \, \forall \, \tau \in [t + T_c, t + T_p] \tag{2.4d}$$

$$\bar{x}(\tau) \in X, \, \forall \, \tau \in [t, t + T_p], \tag{2.4e}$$

where $T_p$ and $T_c$ refers to the control horizons already introduced above. The $\bar{u}$ denotes internal controller variables, and $\bar{x}$ refers to the system response to the input vector $\bar{u}$, i.e. the solutions to (2.4b). The cost functional $J$ is a sum of performance costs, $F(\cdot)$, at each time step. The cost function can in principle take any shape or form, but often arises from some economical consideration on the systems operational point $(x_{op}, u_{op})$ through a quadratic form:

$$F(x,u) = (x - x_{op})Q(x - x_{op}) + (u - u_{op})R(u - u_{op}). \tag{2.5}$$

Thus, the cost is given as a result of deviations from the operational set point, specified by positive definite *weighting matrices* $Q$ and $R$. Often one can also find additional terms in $F$ which penalize movements of the inputs where that is appropriate.

We define the solution to (2.4) as $\bar{u}^*(t; x(t))$, where the first set of inputs are applied to the system:

$$u(t) = \bar{u}^*(t_0; x_0) = \bar{u}_0^*. \tag{2.6}$$

The optimal cost yielded by $\bar{u}^*$ is then a function of the state $x(t)$ alone. This optimal cost is often referred to as the value function

$$V(x) = J(x(t), \bar{u}^*(t; x(t))). \tag{2.7}$$

This section gives a short review of selected portions of the existing theory available today on the topic of model predictive control. It does by no means set out to include every aspect of the subject, as that would be far out of scope for this report. This review relies heavily on the reviews made by Mayne et al. (2000), Qin and Badgwell (2003), Morari and Lee (1999), Allgöwer et al. (2004) and the book by Maciejowski (2002), and the reader should assume to find all that is mentioned here in these excellent papers. Citations are made in the largest extent possible, while simultaneously trying to maintain some degree of readability. The interested reader should in any case seek to investigate these surveys for a more comprehensive picture. Also recommended is the book by Allgöwer and Zheng (2000). Now, we will first focus on the underlying fundamentals which make up MPC by outlining the historical development from linear theory. Then we expand our horizon and move the attention the work made on nonlinear MPC (NMPC), arising new issues on i.e. solution time, algorithm design and feasibility considerations to name a few. The chapter ends with some discussion on the topics of stability and robustness for MPC in general and NMPC in particular.

## 2.1 The History and Evolution of MPC

*Predictive control* is the one general class of advanced control methodologies to have a significant impact on the practice in industrial control engineering, Maciejowski (2002) states. Much of this class is covered by varieties of MPC, though there other subclasses can be found which share common properties with MPC (e.g. general predictive control). We will here only consider MPC, without much loss of generality. MPC started out as an industrial success only, because the advantages of this control were first recognized by the industrial engineering community alone. It was within the industry that much of the early work was done on the topic, covering only the analysis necessary to perform satisfactory performance. Only in the past 20-25 years have MPC gotten the attention it deserves from academia, which has made important contributions to clarifying the properties of stability and robustness for predictive control.

In the following of this subsection we consider a discrete, linear time-invariant (LTI) model

$$x_{k+1} = Ax_k + Bu_k \tag{2.8a}$$
$$y_k = Cx_k. \tag{2.8b}$$

### 2.1.1 Origin of MPC

Though MPC in the strong sense evolved *within* communities of industrial engineering, it did in fact evolve *from* important academic work done in the 1960s on the topic of optimal control. Mayne et al. (2000) states that MPC links

> Hamilton-Jacobi-Bellman theory (Dynamic Programming), which provides sufficient conditions for optimality and a constructive procedure for determining an optimal *feedback* controller $u = \kappa(x)$, and the maximum principle, which provides necessary conditions of optimality and motivates computational algorithms for the determination of the optimal open-loop control $\bar{u}^*(\cdot; x)$ for a given initial state x. The link is
>
> $$\kappa(x) = \bar{u}_0^* \tag{2.9}$$

This represent the ideal case of optimal control. The feedback control is given by the solution to the open-loop control problem, as in (2.4), for every $x$. Kalman (1960a) made an important complementary observation to this when showing that optimal control does not imply stability in the general finite horizon case. He found that stability can be shown, with some assumptions on the system conditions, for an infinite horizon optimal controller, known as the Linear Quadratic Regulator (LQR) (Kalman, 1960a,b). The LQR generates the optimal control sequence from a static state feedback law where the feedback gain is found via the solution of an Algebraic Riccati Equation (ARE), because the Hamilton-Jacobi-Bellman equations simplifies to an ordinary differential equation for a LTI system, the *Riccati* equation. Stability can be guaranteed by ensuring that the problem objective function is positive definite through the choices of weighting matrices (Morari and Lee, 1999).

However, the stability properties alone were not enough to make the process industry embrace the LQR Qin and Badgwell (2003) reports, listing the following subjects that the theory failed to address:

- *constraints* - typically the most economically profitable operating point of a process lies in an intersection of several input or output constraints;
- *process nonlinearities* - LQR are based on LTI models;
- *model uncertainties* - LQR stability guarantees are made on assumptions of a perfect model;
- *performance criterion flexibility* - many process units may require to combine several objectives of different nature.

To include constraints in the infinite horizon controller, ones again needs to solve (2.9) as the solution based on ARE was no longer valid. However, computing the solution to the Hamilton-Jacobi-Bellman equations in the infinite case is a difficult task. From an on-line perspective this is usually not even close to practical to obtain. In fact, it often can not be found analytically at all, even in the simplest form, the unconstrained case (Allgöwer et al., 2004).

The solution to the infinite horizon problem was to redefine it as a receding horizon optimal control problem. Mayne et al. (2000) mentions the work of Kleinman (1970); Thomas (1975); Kwon and Pearsons (1977) and Kwon, Bruckstein and Kaliath (1983) as important here, as they all proposed different extensions to a stabilizing receding horizon alternative. With some variations the all shared a common approach, by introducing various types of terminal constraints. Although, as we will see later on in Section 2.2.3, the concept of adding a terminal constraint to the problem would show to have large impact on later research, the stability results of Kleinman, Thomas and Kwon et al. was limited to hold for unconstrained linear systems only. Hence, they also lacked many of the same properties as the LQR.

### 2.1.2 Industrial applications

The above listed weaknesses of the LQR were indeed reasons for the lack of its industrial support, but in addition, there have been claimed that the main reason was a cultural difference represented by LQR and the process industry. Control engineers either had no exposure to LQR concepts or regarded them as impractical (Qin and Badgwell, 2003). So the industry developed its own methodology, by including properties such as input/output constraints and explicit process models from which could be estimated from test data.

In the following the first industrial MPC applications are presented, as presented in Qin and Badgwell (2003). These represent pioneering work which has influenced later MPCs in one way or the other.

**IDCOM** Though Lee and Markus (1967) was the first publication which explained MPC in the broader sense, Richalet, Rault, Testud and Papon (1976) are the ones credited for describing the first MPC control application, with their IDCOM (Identification and Command). IDCOM was described by the authors as a model predictive *heuristic* control (MPHC), because a transfer function of the control law was not available. This is due to the fact that MPC is not a linear controller, since it behaves nonlinear in terms of dealing with constraints. Even so, the IDCOM is what today is known as a linear MPC, because of its linear model representation. This was an impulse response model, known as a *finite impulse response* (FIR) model. The model had inputs called manipulated variables (MVs), if adjustable by the controller, and disturbance variables (DVs), if not available for control. The outputs were termed controlled variables (CVs). The FIR was identified from plant test data using a parameter estimation algorithm to minimize plant and model outputs. To calculate the control problem IDCOM used the same algorithm, by noting that control is the mathematical dual of identification (Qin and Badgwell, 2003). Most importantly, the IDCOM included what the LQR lacked, an explicit formulation of input and output constraints. These were included in the control calculations by checking for feasibility in all algorithm iterations.

The contributions by Richalet et al. are important because they proposed an application that satisfied particular demands of process control. Also, they pointed out the importance of embedding dynamic control in a control hierarchy to be effective. The significant economical benefits lies not in the low level dynamic control by reducing output variations, but in the above level by dynamically allocation variable set point as close to the operational constraints as possible. This has since been one of the crucial arguments in favour of MPC to be chosen as the control application.

**DMC** Dynamic Matrix Control (DMC) was presented by Cutler and Ramaker (1979, 1980) and was the other algorithm to form the first generation of MPC, together with IDCOM. DMC computed the optimal inputs as a solution to a least-squares problem, and did not include constraint handling in the original version. This was taken care of by Prett and Gillette (1980) which contributed with a modification to the DMC in which constraint handling was included for absolute input constraints.

The DMC used a linear step response model where the output is described by a weighted sum of past input changes, i.e. an integral of the impulse response. For multiple outputs DMC imposed the superposition principle. By utilizing this step response model the predicted future output changes can be written as a linear combination of future input moves, given an initial state. This way one can parameterize the computation of the optimal inputs as relative movement from the initial input, a fact which is still appreciated in MPC today. Considering the system from (2.8), and the fact that every future output can be written as a sequence of changes from the current value

$$\begin{aligned}
u_k &= \Delta u_k + u_{k-1} \\
u_{k+1} &= \Delta u_{k+1} + u_k \\
&= \Delta u_{k+1} + \Delta u_k + u_{k-1} \\
u_{k+2} &= \Delta u_{k+2} + u_{k+1} \\
&= \Delta u_{k+2} + \Delta u_{k+1} + \Delta u_k + u_{k-1} \\
&\vdots \\
u_{k+H_u-1} &= \Delta u_{k+H_u-1} + u_{k+H_u-2} \\
&= \Delta u_{k+H_u-1} + \Delta u_{k+H_u-2} \\
&\quad \cdots + \Delta u_{k+1} + \Delta u_k + u_{k-1},
\end{aligned}$$

the future outputs can be written over the prediction horizon $H_p$ as

$$\begin{aligned}
x_{k+1} &= Ax_k + B(u_{k-1} + \Delta u_k) \\
x_{k+2} &= Ax_{k+1} + B(u_k + \Delta u_{k+1}) \\
&= A^2 x_k + (AB + B)u_{k-1} + (AB + B)\Delta u_k + B\Delta u_{k+1} \\
&\vdots \\
x_{k+H_p} &= A^{H_p} x_k + (A^{H_p-1}B + \ldots + AB + B)\Delta u_k \\
&\quad \ldots + (A^{H_p-1}B + \ldots + AB + B)\Delta u_{k+1} \\
&\quad \ldots + (A^{H_p-H_u}B + \ldots + AB + B)\Delta u_{k+H_u-1} \\
&\quad + (A^{H_p-1}B + \ldots + AB + B)u_{k+1}.
\end{aligned}$$

Now we can stack the states and input changes over the prediction horizon in two vectors $\mathcal{X}(k)$ and $\Delta\mathcal{U}(k)$ respectively, and write it more compactly as

$$\mathcal{X}(k) = \psi x_k + \upsilon u_{k-1} + \theta \Delta\mathcal{U}(k),$$

where

$$\psi = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{H_p} \end{bmatrix}, \; \upsilon = \begin{bmatrix} B \\ AB \\ \vdots \\ \sum_{i=0}^{H_p-1} A^i B \end{bmatrix},$$

$$\theta = \begin{bmatrix} B & \cdots & 0 \\ AB + B & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{H_p-1} A^i B & \cdots & \sum_{i=0}^{H_p-H_u} A^i B \end{bmatrix}.$$

The linear combination ties future inputs and outputs together through a so-called *Dynamic Matrix*. The dynamic matrix is a sensitivity matrix, $\mathcal{S}$, as will used later on in this report, which expresses the influence from every input to every output in the discrete parameterization over the prediction horizon. This matrix founds the basis for the least-squares problem definition, as the solution is given by (Maciejowski, 2002)

$$\Delta\mathcal{U} = \mathcal{S} \backslash \left[\mathcal{X}_{ref} - \mathcal{X}\right],$$

where $\mathcal{X}_{ref}$ depicts the control target.

Morari and Lee (1999) states that DMC had a tremendous impact on industry, estimating that probably every major oil company in world has a DMC inspired application installed in most new installations or revamps. But even if DMC, and IDCOM, did get the recognition LQR failed to receive in process industry, there still was some weaknesses in their constraint handling.

**QDMC** The breakthrough in constraint handling came when García and Morshedi (1986) showed how the DMC objective could be written as a standard quadratic program (QP), by introducing a quadratic cost function similar to (2.5). The QP provides efficient constraint handling in the control algorithm through the performance objective, both for inputs and outputs. As in standard DMC the dynamic matrix plays a important role, as the process constraints can be related directly to the input moves, re-written from $u_k$ to $\Delta u_k$.

The beauty behind QDMC lays in the fact that a QP is a simple optimization problem to solve in the convex case. Since the Hessian of the QP is positive definite for linear plants, QDMC could easily calculate optimal control inputs using standard commercial optimization codes. This allowed the QDMC scheme to grow in model size and complexity, since not much time was spent on calculations.

Though several other industrial MPC applications have been proposed after the QDMC, this report will end this part of the review here. The interested reader is referred to Qin and Badgwells excellent survey on the topic to learn more about the third and the fourth generation of MPC. The next subsection will take a look at MPC from a nonlinear perspective, as this is where the interest lays currently. The QDMC seems like the natural place to stop, as it has proven to be the basis of several attempts to extend the MPC technology to incorporate nonlinear process models.

## 2.2 NMPC

This chapter addresses various elements on the topic of NMPC, from a control theoretical point of view. First possible model representations are presented, then different system discretizations and solution methods. Stability and robustness will also be given some attention, before ending with some notes on feasibility.

### 2.2.1 Model

The model is the heart of MPC. Qin and Badgwell (2003) states that, in principle, the model can take any required mathematical form. Unsurprisingly, a wealth of model formulations are used, although some are more common than others. In Qin and Badgwell (2003) some of the model forms used in commercial products are listed; finite impulse response (FIR), velocity FIR, Laplace transfer function, linear state-space, auto-regressive with exogenous input, Box-Jenkins and multi-model, input-output, first-principle, nonlinear state-space, nonlinear naural net and static nonlinear polynomial. These labels are not mutually exclusive, nor do they represent an exhaustive list of models that can be used. This section will look at some different categories in which most models can be sorted. The classification is based on Meadows and Rawlings (1997) and Rawlings (2000).

Perhaps the most important classification is the divide between *linear* and *nonlinear* models. For linear models, the superposition principle holds. That is, any linear combination of solutions for the linear system, is in itself also a solution. Powerful tools for the analysis and control of such systems are available. Nonlinear models will in general have no special characteristics, and are perhaps only characterized by not being linear. An introduction to nonlinear models and nonlinear model identification, is presented in Pearson (1997).

Similarly, models can be classified as either *first principles* or *experimental*. First principles models are based on physical knowledge of the system and are also referred to as physical models. In general a first principles model will be given as differential equations, either as ordinary differential equations (ODE) or as differential algebraic equations (DAE). Experimental models, also called black-box models, are used to obtain models that fit the process data sets. An experimental model will only contain the characteristics that were exhibited by the system during the identification process. Hence, the data used for model identification should be carefully selected, as the predictive ability of an experimental model is small outside the range of data that was used to identify it. Common experimental model formulations include step response and auto regressive external input models. The use of neural networks as models in MPC is an active area of research. The interested reader can consult Su and McAvoy (1997) with references. As a first principles model are based on physical insight into the system, it can be expected to describe system dynamics more completely than empirical models (Pearson, 1997). It should be mentioned that there exist model representations that tries to combine both experimental and first principles, so-called *hybrid* models. These can be thought of as experimental models based on fundamental physical laws of the underlying process, which is tuned with respect to model parameters to be in accordance to plant data.

Other classifications are possible. *Continuous time* or *discrete time*, *distributed parameters* or *lumped parameters*, *deterministic* or *stochastic*, *input-output* or *state-space* and *frequency domain* or *time domain*. Although these classifications can be used to further categorize models, they are less important than the first two already discussed.

### 2.2.2 Solution methods

There are three main strategies for solving the NMPC optimization problem (Allgöwer et al., 2004; Strand, 1991; Tenny, Wright and Rawlings, 2004; Barclay, Gill and Rosen, 1997). These can be categorized as either *sequential* or *simultaneous* approaches (Biegler, 2000).

If the behaviour of a system is completely determined by its initial values it is called and initial value problem (IVP). In order to determine the states in a time interval, $[t_0, t_1]$ only one simulation is necessary (single shot). This approach is also described as single shooting, sequential approach or feasible path approach and are more thoroughly described in Oliveira and Biegler (1995); Silva and Oliveira (2002); Tenny et al. (2004). A fuller description of single shooting is given in Section 3.

*Multiple shooting* differs in that the time interval is divided into multiple intervals. In each of these subintervals an IVP is solved independently. These IVPs must be solved iteratively with updated initial values that converge to the end value in the preceding interval. Multiple shooting is therefore also coined as a sequential approach. Diehl, Bock, Schlöder, Findeisen, Nagy and Allgöwer (2002) lists several advantages with multiple shooting:

- As a simultaneous strategy, it allows to exploit solution information in controls, states and derivatives in subsequent optimization problems by suitable embedding techniques.
- Efficient state-of-the-art DAE solvers are employed to calculate the function values and derivatives quickly and accurately.
- Since the integrations are decoupled on different multiple shooting intervals, the method is well suited for parallel computation.
- The approach allows a natural treatment of control and path constraints as well as boundary conditions.

The third option is to solve the differential equations and the optimization problem simultaneously, thus the term simultaneous approach. The differential equations are discretized and enter the optimization problem as equality constraints. The manner in which the differential equations are discretized is important. Silva and Oliveira (2002) lists weighted residuals, orthogonal collocation and finite differences schemes as possible techniques for discretization/parameterization. Of these, *collocation* is the technique most often referenced. Barclay et al. (1997) explains collocation as *"a form of multiple shooting in which an appropriate implicit Runge-Kutta (IRK) formula is used to solve the initial-value problem"*.

### 2.2.3 Stability

Before theory on stability and robustness is presented, it is appropriate to define what the terms refers to. Skogestad and Postlethwaite (2005) mentions two types of stability — nominal stability (NS) and robust stability (RS). If the system is stable for the nominal plant, it is said to be nominally stable. Similarly, if the system remains stable for all plants in the uncertainty set, it is said to be robustly stable. In this report, stability is taken to mean nominal stability, while robustness is taken to mean robust stability.

Stability of NMPC is becoming a mature area of research, if not as mature as the stability of linear MPC. A good starting point to NMPC stability literature is provided in Mayne et al. (2000) and somewhat simpler stated in Allgöwer et al. (2004), the two of which this discussion is based. The literature is focused on the stabilization of a steady state.

Due to constraints, even linear model predictive control will result in a nonlinear control law, requiring nonlinear tools for the study of stability. For almost all stability analysis, Lyapunov theory using the value function (recalling (2.7)) is therefore used. Mayne et al. (2000) starts the stability analysis by listing several modifications to the MPC scheme such that stability can be guaranteed.

Since MPC solves a open loop problem over a horizon, the easiest method of stabilization to grasp is to extend the horizon infinitely. Due to Bellmanns principle of optimality, the optimal trajectory in the next sample will be the remaining trajectory of the previous sample. Since an infinite horizon input trajectory in general will be impossible to compute in finite time, several approximations exists that ensures closed loop stability with finite horizon. One can, without loss of generality, assume that the system is to be stabilized at the origin.

*Terminal equality constraint* ensures stability by requiring that all states should be set to zero at the end of the prediction horizon and the control inputs used to maintain the system at the origin should also be zero,

$$\bar{x}(t+T_p) = 0. \tag{2.10}$$

The obvious disadvantage of a terminal constraint is that it will force the system to a selected state in finite time, heavily decreasing the solution space of the problem. Also, there is in addition the extra computational burden of finding an exact satisfaction to the imposed equality, as this transforms the problem to a boundary-value problem (BVP). *Terminal cost function* uses no exact terminal constraint. Instead, a terminal cost is used to ensure the stability. Such a terminal cost can unfortunately only be obtained generally for linear unconstrained or linear constrained stable systems. The terminal costs can in these cases be computed by solving the Lyapunov equation, see for example Maciejowski (2002). A method related to terminal equality constraints is the *terminal region constraint* where the states are required to lie within a terminal set at the end of the prediction horizon,

$$\bar{x}(t+T_p) \in \Omega. \tag{2.11}$$

This terminal region, $\Omega$, are defined by calculating a controller that drives all states within the terminal set exponentially fast to the origin. Finally, it is possible to combine terminal set with terminal cost. A cost $E$ is added to (2.4a) for the terminal state, giving

$$J(x(t), \bar{u}(\cdot)) = \int_{t}^{t+T_p} F(\bar{x}(\tau), \bar{u}(\tau))\mathrm{d}\tau + E(\bar{x}(t+T_p)). \tag{2.12}$$

If the terminal cost equals the residual of weights for all $t > T_p$, the solution of (2.4) extended with (2.11) and (2.12) will, in effect, become the infinite horizon solution. The closer the terminal cost is to the infinite horizon weight, the more of the benefits gained by infinite horizon are realized. A deduction of an infinite horizon scheme is given in Chen and Allgöwer (1998a) and Chen and Allgöwer (1998b).

Allgöwer et al. (2004) gives the following theorem ensuring stability of the NMPC, after modifying the original problem (2.4) with (2.11) and (2.12):
**Theorem 2.1.** *Assume that:*

1. *$U \subset \mathbf{R}^m$ is compact, $X \subseteq \mathbf{R}^n$ is connected and the origin is contained in the interior of $U \times X$.*

2. *The vector field $f$: $\mathbf{R}^n \times \mathbf{R}^m \rightarrow \mathbf{R}^n$ is continuous in $u$ and locally Lipschitz in $x$ and satisfies $f(0,0) = 0$.*

3. *$F$: $\mathbf{R}^n \times U \rightarrow \mathbf{R}$ is continuous in all arguments with $F(0,0) = 0$ and $F(x,u) > 0 \forall (x,u) \in \mathbf{R}^n \times U \backslash 0, 0$.*

4. *The terminal penalty $E : \Omega \rightarrow \mathbf{R}$ is continuous with $E(0) = 0$ and that the terminal region $\Omega$ is given by $\Omega := x \in X | E(x) \leq e_1$ for some $e_1 > 0$ such that $\Omega \subset X$.*

5. *There exists a continuous local control law $u = k(x)$ such that $k(x) \in U$ for all $x \in \Omega$ and $\frac{\mathrm{d}E}{\mathrm{d}x} f(x, k(x)) + F(x, k(x)) \leq 0, \forall x \in \Omega$.*

6. *The NMPC open-loop optimal control problem has a feasible solution for* $t = 0$*. Then for any sampling time* $0 < \delta \leq T_p$ *the nominal closed-loop system given by the problem* (2.4) *extended with* (2.11) *and* (2.12)*, and the input* (2.6)*, is asymptotically stable and the region of attraction* $\mathbf{R}$ *is given by the set of states for which the open-loop optimal control problem has a feasible solution.*

Mayne et al. (2000) concludes the discussion about stability with the conclusion that a combination of terminal set and terminal cost seems to be the best way to ensure both stability and good performance.

### 2.2.4 Robustness

When NMPC is applied in practice, it is unrealistic to assume that the model used for prediction will accurately match the process and similarly that no un-modelled disturbances enters the process. Obviously a successful implementation of NMPC relies on being able to cope with these issues. The MPC formulation has an inherent robustness, due to its similarity with optimal control. (Allgöwer et al., 2004) According to Qin and Badgwell (2003) most commercial MPC products relies on this robustness and brute force evaluation of model mismatch to ensure robustness. Qin and Badgwell (2003) calls for MPC formulations that ensure robust stability. A survey of different schemes to ensure robust stability can be found in Allgöwer et al. (2004) and Mayne et al. (2000) which is also the basis for this discussion.

Three approaches to achieve robust stability are outlined. The first of these is to *solve an open-loop min-max problem*, which minimizes the maximum objective function value for a set of uncertainties. The input sequence must be feasible for every realization in the set of uncertainties, possibly giving feasibility problems or conservative solutions. *$H_\infty$ MPC* is another approach, using $\text{H}_\infty$ control to achieve robustness. Large computational cost and the need for a global optimum are drawbacks for this approach. The third approach is to use the parameters of a *feedback controller* as optimization variables instead of using inputs directly. Because a feedback controller is used, disturbances are rejected also between sampling times, reducing the need for conservative control.

Mayne et al. (2000) concludes the discussion of robustness by stating that current robustness schemes must be regarded as conceptual rather than practical.

### 2.2.5 Feasibility

Model based predictive control is based on finding a solution to the optimization problem (2.4). Feasibility problems arise when no solution to this problem can be found. Normally, feasibility problems are solved by relaxing process constraints. Process constraints are often output constraints that define safety constraints, process equipment limitation or product specifications. Such constrains are by nature different from input constraints that are physical constrains imposed by the actuator. Such constrains can be maximum power, maximum opening etc. Thus, input constraints can normally not be violated.

Different ways of imposing soft constraints are possible. Oliveira and Biegler (1994) advocates for the use of exact penalty functions. Hence, process constraints are only violated if no other feasible solution exists. It is also possible to penalize constraints relaxation using a normal quadratic penalty. A quadratic penalty implies that, if the optimal operation is on the constraint, a small violation of the constraint will be allowed. This violation will be reduced as the penalty is increased. However, as long as the penalty is finite, the violation will also be finite.

When discussing feasibility of MPC, there is one additional type of constraints that needs to be considered, namely model constraints. Naturally, these constraints cannot be relaxed. For a linear system, the model constraints can be solved exactly and simultaneously with the optimization problem. In general, this will not be possible for nonlinear models. A discussion of how nonlinear equalities enters the optimization can be found in Section 2.2.2. A short discussion of the different solution methods impact on the feasibility of the model follows.

Single shooting enjoys the advantage that the model equations are satisfied along the whole prediction horizon. Thus, each iterate will be a feasible albeit suboptimal solution of the optimization problem. Multiple shooting and collocation allows iterations, inconsistent with the system dynamics. (Tenny et al., 2004) Thus, the solution cannot be guaranteed feasible until convergence is achieved.

This concludes the chapter on MPC theory. From a focus on methodological overview and important properties such as stability and robustness, the following chapter will present a dedicated NMPC application, derived from a SQP principle.

# Chapter 3

# A multistep quasi-Newton method

This chapter will focus on the method developed in this report, a single shooting, multistep, quasi-Newton method (SSMQN). The first section contains the motivation for using NMPC along with some background on the chosen method. A detailed description of the single shooting multistep quasi-Newton method follows in the second section, while the implementation used for this report is described in the third section. Finally, an example used extensively in the development of the SSMQN — NMPC used on a CSTR system, is presented.

## 3.1 Background and motivation

When the model used for predictions in an MPC application is changed from a linear model to a nonlinear model, the optimization problem to be solved online changes from a QP (or LP if a linear objective function is used) to a NLP[1]. While QP's are solved fast and reliably with standard QP solvers, algorithms for solving NLP's are much more an area of research. In this report, the NLP will be solved using a single shooting multistep quasi-Newton method. The method was described by Li and Biegler (1989) extending the previous single-step method by Li, Biegler, Economou and Morari (1990). An algorithm very similar to the one used in this report can be found in Oliveira and Biegler (1995) where more general objective functions can be accommodated. A linearized input output model is obtained from a black box nonlinear model around the nominal input sequence found at the previous sample. A search direction is found by formulating a QP sub problem using the linearized model. In order to avoid algorithm divergence, a linesearch is performed along the calculated search direction, ensuring descent in the objective function. The solution found is checked against a convergence criterion, and, if not satisfactory, set as nominal input trajectory and the procedure repeated. All in all, the algorithm is a SQP algorithm especially adapted to the NLP's generated by a NMPC.

Single shooting has some known stability and robustness issues (Barclay et al., 1997; Biegler, 2000). If the system is locally unstable, the states can diverge, giving immedi-

[1]NLP is a commonly used acronym for *nonlinear programming*

ate failure. It can also be difficult to find input trajectories with bounded outputs except very close to the optimal trajectory making single shooting unsuitable as an optimization strategy. Despite these problems, single shooting has been reported to successfully control some nonlinear systems, e.g. batch processes in Silva and Oliveira (2002).

## 3.2 Algorithm description

### 3.2.1 Outline of algorithm

The following outline of the algorithm is based on the algorithm described in Oliveira and Biegler (1995)

1. Set QP sub problem iteration counter to zero.
2. Compute input sensitivities for the nominal input trajectory.
3. Solve the QP sub problem with the linearized model to find a search direction
4. Employ a line search algorithm to determine a suitable step size along the search direction.
5. If the solution found satisfies some convergence criteria, the first input of the computed input trajectory given to the controlled system, and the new input trajectory shifted one sample and set as the nominal input trajectory.
6. If the solution fails to satisfy some convergence criteria, the QP sub problem iteration counter is checked, and if less than the iteration limit, it is incremented and the algorithm jumps to Step 2. If the maximum iteration limit is reached, the algorithm is stopped and the best input trajectory found returned.

The following subsections will describe some of these steps in greater detail. It can be practical to return to this algorithm outline to keep the right perspective.

### 3.2.2 Input sensitivities

*Step 2* of the algorithm is to linearize the model by computing sensitivities. Depending on the level of model control, there are several different possibilities as to which sensitivities should be calculated. Silva and Oliveira (2002) shows figures of two possible methods of calculating sensitivities. The first method is to calculate the sensitivities from inputs to state, state to state and state to output. The second is to only calculate input to state and state to output sensitivities. The drawback of this approach is that the input sensitivity for one input must be calculated for all subsequent states as opposed to calculating only the sensitivity to the states in the next sample, and then to use state to state sensitivities. However, it can be seen as an advantage that it is not necessary to perturbate the states, as this is problematic for black box models.

The scheme for calculating sensitivities used in this report, can be seen as a variant of the second approach. Sensitivities are calculated from input to outputs directly, analogous to defining the outputs as the state vector $x$ and using a unity measurement sensitivity. Inspired from Silva and Oliveira (2002) the sensitivities are given in a matrix with elements given by

$$\mathcal{S}_{j+1,i} = \frac{\partial x_{j+1}}{\partial u_i}, \quad j \geq i, \tag{3.1}$$

$$\mathcal{S}_{j+1,i} = 0, \quad \text{else}, \tag{3.2}$$

$$j \in [k, T_p - 1],\ i \in [k, T_c]. \tag{3.3}$$

The sensitivities calculated are shown in Figure 3.1. As can be seen, the sensitivity from each input parameter to each subsequent coincidence point is calculated. In this report, sensitivities will be obtained numerically by perturbation. Notice that in Figure 3.1 only perturbations in the inputs are necessary.

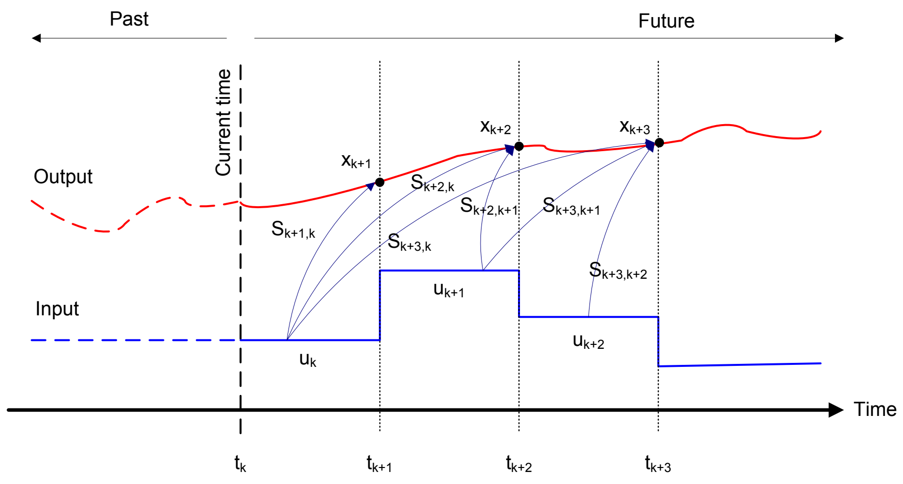

Figure 3.1: Calculation of sensitivities

The sensitivity matrix calculated gives a linearized input-output model of the form:

$$\begin{bmatrix} x_{k+1} \\ x_{k+2} \\ \vdots \\ x_{k+T_p} \end{bmatrix} = \begin{bmatrix} \mathcal{S}_{k+1,k} & 0 & \dots & 0 \\ \mathcal{S}_{k+2,k} & \mathcal{S}_{k+2,k+1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{S}_{k+T_p,k} & \mathcal{S}_{k+T_p,k+1} & \dots & \mathcal{S}_{k+T_p,k+T_c} \end{bmatrix} \begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \vdots \\ \Delta u_{k+T_c} \end{bmatrix} + \mathbf{b}, \tag{3.4}$$

where $x_{k+i}$ is the output at time $k + i$, $\Delta u_{k+i}$ is the change in input at time $k + i$ and $\mathbf{b}$ is a vector containing the constant terms of the linearizations.

The sensitivity matrix is obtained by *perturbating* the inputs by small, but finite, values. The size of the perturbations influences the accuracy of the sensitivity, and hence also the convergence rate of the SQP algorithm. Too small perturbations give sensitivities dominated by numerical noise, while too large perturbations will give inaccurate sensitivities and possibly problems with convergence. It is therefore desirable to select a perturbation size as small as possible without encountering numerical difficulties. The lower limit on perturbations is determined by the accuracy of the solver used to simulate the nonlinear model. An example showing the effect of too large perturbation is shown in Figure 3.2. The large perturbation will wrongly indicate an increase in the objective function value moving from right to left, where as the small perturbation correctly shows a descent direction towards the minima.

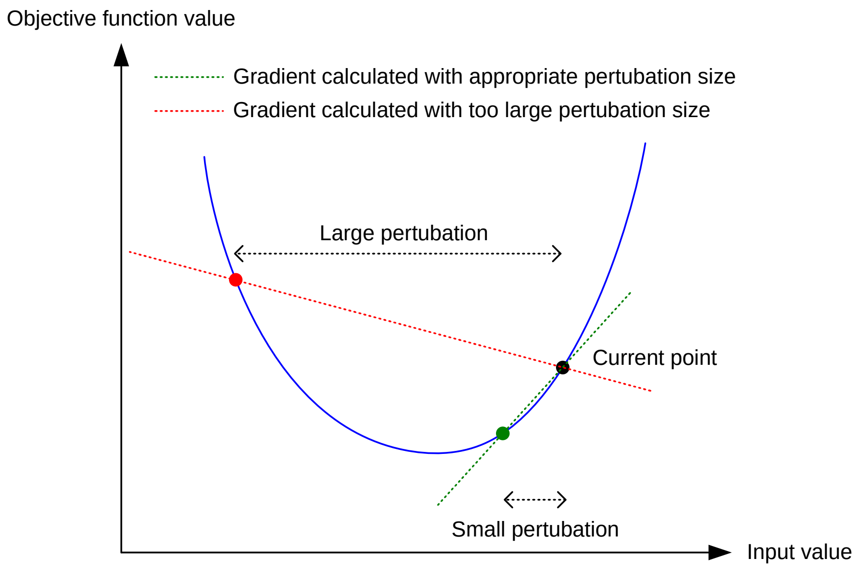

Figure 3.2: Effect of too large pertubation on numeric gradients

### 3.2.3 The QP sub problem

This section will discuss the present the QP sub problem generated by SQP algorithms. Solving the QP sub problem is done in *Step 4* in order to find a search direction. The introduction and the notation will follow Nocedal and Wright (1999).

Assuming a nonlinear problem of the form:

$$\min f(x) \tag{3.5a}$$
$$s.t.\ c_i(x) = 0, i \in E \tag{3.5b}$$
$$c_i(x) \geq 0, i \in I, \tag{3.5c}$$

where $f(x)$ is the function to be minimized, subject to state constraints $c_i(x),\ i \in E \cup I$. $E$ is the set which contains the indices of equality constraints, as $I$ is the set of indices of inequality constraints.

The QP subproblem (3.6) to be solved in order to find the search direction, $p$, are obtained by linearizing both equality and inequality constraints

$$\min \frac{1}{2} p^T W_k p + \nabla f(x_k)^T p \tag{3.6a}$$
$$s.t.\ \nabla c_i(x_k)^T p + c_i(x_k) = 0, i \in E \tag{3.6b}$$
$$\nabla c_i(x_k)^T p + c_i(x_k) \geq 0, i \in I \tag{3.6c}$$

Where $W_k = W(x_k, \lambda_k) = \nabla^2_{xx} L(x_k, \lambda_k)$ denotes the Hessian of the Lagrangian of (3.5), with the Lagrangian defined as $L(x, \lambda) = f(x) - \lambda^T c(x)$.

For general SQP algorithms it is important to maintain $W_k$ positive definite to avoid generating non-descending search directions, $p$. This is necessary in order to guarantee convergence on non-convex problems and from remote starting points. There are several possibilities as to how this can be accomplished. Nocedal and Wright mentions using full quasi-Newton approximations, such as the BFGS formula, using the Hessian of an augmented Lagrangian functions or using reduces-Hessian approximations. However, none of these techniques are required for NMPC if using a special choice of Hessian, as proven in the following section.

In MPC, the objective function $f(x) = \frac{1}{2}x^T Q x$ are quadratic in $x$. Thus, the Hessian becomes $\nabla_{xx} f(x) = Q$ and will be positive definite as long as $Q$ is chosen positive definite. The only possibility for $W_k$ not to become positive definite is if $-\nabla^2_{xx}\lambda^T c(x)$ is negative definite. By linearizing the constraints $c(x)$ before the QP sub problem is formulated, the contribution to $W_k$ from the Hessian of the constraints will always be zero. This is equivalent to the “Gauss-Newton” choice of Hessian where the effects of the nonlinear model equations on the Hessian is neglected. The effects of neglecting the contribution from the nonlinear model is described (among other choices for Hessians) in Tenny et al. (2004). Here it is stated that this choice in not asymptotically equivalent the true Hessian of the Lagrangian unless the model is linear or the Langrange multipliers for the model is zero. It is also pointed out that when $x$ in (3.5) is close to its optimal value, the state equations will be only weakly active, and their Lagrange multipliers close to zero. The Gauss-Newton choice of Hessian will therefore be close to the true Hessian. In fact, according to Biegler (1998), a NMPC has a structure that, for a solution interior to the constraints, will have $W_k \approx \nabla^2 L$ when using the “Gauss-Newton” choice of Hessian. The approximated Hessian of the Lagrange function will be asymptotically equivalent to the actual Hessian at the solution, resulting in a Q-quadratic convergence rate.

### 3.2.4 Linesearch

The QP sub problems are based on a quadratic approximation of the original NLP. If the approximation is poor, the solutions found may in fact cause the algorithm to diverge, if implemented unbounded. In order to ensure that the step taken is a descending step, a linesearch algorithm is employed in *Step 4*. This algorithm must decide the fraction of the step to be taken. Newton methods have a natural step length of one. The fraction, $\alpha$, of the step taken should therefore be in the interval $(0, 1]$. One can potentially solve an optimization problem, finding the step length, $\alpha \in (0, 1]$, minimizing the objective function. However, it will in general be a compromise between doing a thorough linesearch with fewer QP solutions, or a approximate linesearch with more QP solutions. Nocedal and Wright (1999, chap. 3) describes some approaches to linesearch based on the Wolfe conditions — the sufficient decrease condition and the curvature condition. The sufficient decrease condition ensures that the step is descending while the curvature condition makes certain that the step length is not unacceptably short steps.

### 3.2.5 Convergence criterion

The optimization termination criteria in *Step 5* can be specified as in generic SQP algorithms. However, since the single-shooting quasi-Newton methods searches along a feasible path, optimization can be terminated before exact convergence is achieved, still yielding feasible solutions although not optimal. Especially since NMPC generates a series of NLP's, where the solution of the previous NLP can be used as a starting point for the next NLP, and where the solution often converges over time, the convergence criterion can be relaxed. Since the accuracy of the sensitivities are bounded by the perturbation size (as discussed earlier in section on sensitivities (Section 3.2.2)), and thus also the solver accuracy, it can be convenient to use a relaxed convergence criterion.

## 3.3 Implemented algorithm

The previous section described some issues in the single shooting multistep quasi-Newton algorithm used in this report. In this section, the solutions used are described, as well as a pseudo code of the implementation.

### 3.3.1 Description of SEPTIC

The NMPC application was implemented in STATOIL's in-house MPC tool, SEPTIC (Statoil Estimation and Prediction Tool for Identification and Control). SEPTIC was first used in 1997 Strand and Sagli (2003), and has since then been implemented on 70 installations. Most of these applications have been implemented using linear and experimental models, as these have proved sufficiently accurate for the respective processes. There is however possible to use first principles nonlinear models, either as models programmed in SEPTIC, or through a interface against an external simulator. Either way, SEPTIC treats the model as a black-box model. None of the previous applications have used nonlinear simulator models, and as such SEPTIC did not include a robust algorithm for an iterative NMPC scheme. Hence, there was a need to develop the algorithm presented here.

### 3.3.2 Computation of sensitivity

Because of the simple model interface in SEPTIC, each column in the sensitivity matrix in (3.4) is calculated by simulating the system from $t_k$ to $t_{k+T_p}$, which might seem unnecessary as all columns except the one to the extreme left contains one or more zero elements in the top. Two facts, however, implies that this implementation is not as wasteful as it might seem at first sight. Firstly, input blocking are normally implemented with increasing block length. Consequently the number of zero elements in the sensitivity matrix will be much less than half the total number of elements. Secondly, if the sensitivity matrix is found from parallel processing where each input parameter sensitivity is simultaneously calculated on separate CPUs, all processes calculating sensitivities for inputs $u_i, i \in [k+1, k+T_c]$ must wait for the process calculating the sensitivity for $u_k$ to end. Simulating shorter horizons for $u_i, i \in [k+1, k+T_c]$ will therefore have no effect on the algorithm run time.

### 3.3.3 QP solver

The QP solver used in SEPTIC is a C compilation of a Fortran implementation developed by Schittkowski (2005) This implementation is based on the dual method developed by Goldfarb and Idnani (1983) The motivation for the development of this algorithm was the need for fast and robust solutions to the QP sub problems generated by a SQP algorithm. The dual QP solver implemented requires the QP to be positive definite. Earlier in this chapter, in section 3.2.3, it was shown that this will be true for the Gauss-Newton choice of Hessian approximation used in SEPTIC.

Compared to a primal algorithm, the advantage presented by a dual method is that there is no need for an expensive search for a feasible starting point (Goldfarb and Idnani, 1983). The unconstrained solution of the primal problem is a feasible solution for the dual problem. A feasible solution for the primal problem is obtained by solving series of sub problems where violated constraints are added until all constraints are satisfied. This procedure is in Goldfarb and Idnani (1983) considered superior to primal algorithms when no feasible starting point is immediately available.

In conclusion the QP sub problem solved in SEPTIC are stated in (3.7) where the Hessian of the objective function will be positive definite as long as the $Q$ used is positive definite.

$$\min \frac{1}{2} p^T Q p + \nabla f(x_k)^T p \tag{3.7a}$$

$$s.t.\ \nabla c_i(x_k)^T p + c_i(x_k) = 0, i \in E \tag{3.7b}$$

$$\nabla c_i(x_k)^T p + c_i(x_k) \geq 0, i \in I \tag{3.7c}$$

### 3.3.4 Linesearch

Backtracking is the implemented method of linesearch. The algorithm is a modified version of the backtracking linesearch found in Nocedal and Wright (1999, procedure 3.1, pages 41-42). First the full step is evaluated, then smaller and smaller fractions is evaluated until an acceptable step length is found. The difference between subsequent steps is a constant factor, $\xi$. For a step $\xi^i$ to be accepted, two criteria needs to be fulfilled. First, the objective value with step length $\xi^i$ must be less than the objective value with step length $\xi^{i+1}$. Second, the objective value with step length $\xi^i$ must be less than the objective value with step length zero. Figure 3.3 shows a case designed to illustrate the convergence criteria. Here, the algorithm will choose a step length $\alpha = \xi^3$, since this is assumed to be a local minima with greater objective function values on both sides.

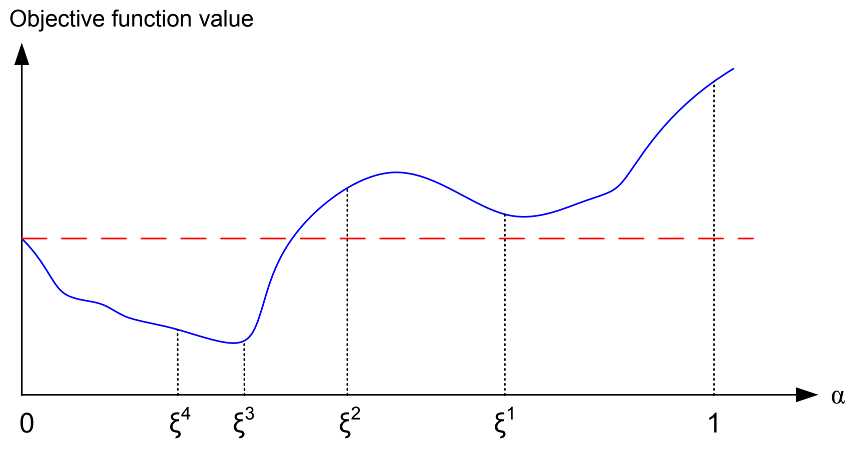

Figure 3.3: Backtracking linesearch

### 3.3.5 Convergence criterion

Convergence criteria considered in the algorithm developed in this report were linearization error, change in objective function and change in input parameters. The original SEPTIC algorithm used the linearization error criterion — after a step, the difference in the response of the controlled variables using the nonlinear and the linearized model is computed. If the linearization error is below a limit, the linear model is assumed to be an accurate description of the nonlinear model along the solution. However, consistent performance was difficult to obtain with this criterion.

Instead, the norm of the input change was used. Li and Biegler (1989) proposed this in their algorithm, as $\sum_k \left\|\bar{u}_k - \bar{u}_{k-1}\right\|^2 \leq \epsilon$, but noting "*that it is not really necessary to solve the QP problem (...) repeatedly (i.e. $\epsilon$ can be large)*". This was found not to be true using models of such a degree of nonlinearity as in this report. Thus, a small value should be chosen for $\epsilon$, so that a sufficient number of QP's are solved at each iteration.

$$\sum_k ||\Delta u_k||^2 < MV_{norm}. \tag{3.8}$$

The explicit criterion used is given in (3.8) where $MV_{norm}$ is the convergence limit. This criterion gave consistent performance while the resulting algorithm was significantly simpler. Alternatively, the change in the objective function could have been used. The obvious drawback is that if a step gives a reduction in the objective function value below the limit, the step may still be of significant length and the sensitivity around the new solution may be quite different from the previous. Using only the descent in objective function will then stop the optimization even if the new solution can result in a significant improvement in objective function value. Possibly, a criterion combining change in objective function with the norm of the input change would be better. However, the algorithm in its current form performs well.

### 3.3.6 Algorithm pseudo code

The SQP algorithm implemented in SEPTIC is given as pseudo code in Algorithm 1. The linesearch used is also given in pseudo code in Algorithm 2. Pseudo code is used in order to be able to state the algorithms more clearly, as the actual SEPTIC code makes extensive use of specialized methods and data structures which would have made the presentation of the algorithm unduly involved. Some previously presented and some new parameters will be used. $MVnorm$ and $\xi$ was described in Sections 3.3.5 and 3.3.4 respectively. In addition will $N_{QP,max}$ be used to represent the maximum number of QP iterations allowed in one NMPC step, $N_{ls,max}$ the maximum number of linesearch iterations and $U_{valid}$ is the feasible input values. In addition will some of the variables used in Algorithm 1 be used in Algorithm 2. The used variables are declared at the top of Algorithm 2.

**Algorithm 1** SEPTIC SQP algorithm

```
Require: $u_{nom} \in U_{valid}$, $MV_{norm}$ and $N_{QP,max}$
  $Y_{best} \leftarrow$ calcObjectiveValue($u_{nom}$)
  $S \leftarrow$ calcSensitivity($u_{nom}$)
  $\Delta u \leftarrow$ solveQP, $n_{QP} \leftarrow 1$
  $Y_{new} \leftarrow$ calcObjectiveValue($u_{nom} + \Delta u$)
  $runSQP \leftarrow$ true
  while $runSQP$ do
    $\alpha_{best} \leftarrow$ doLineSearch
    $u_{nom} \leftarrow u_{nom} + \alpha_{best} * \Delta u$
    $Y_{best} \leftarrow$ calcObjectiveValue($u_{nom}$)
    if $||\alpha_{best} * \Delta u|| < MV_{norm}$ then
      $runSQP \leftarrow$ false
    else if $n_{QP} == N_{QP,max}$ then
      $runSQP \leftarrow$ false
      $warning \leftarrow$ maximum number of iterations without convergence
    else
      $S \leftarrow$ calcSensitivity($u_{nom}$)
      $\Delta u \leftarrow$ solveQP($u_{nom}$), $n_{QP} \leftarrow n_{QP} + 1$
      $Y_{best} \leftarrow$ calcObjectiveValue($u_{nom}$)
    end if
  end while
```

## 3.4 Test system: a continuously stirred tank reactor

This section shows NMPC tested on a small continuously stirred tank reactor (CSTR) model. Even though the intension behind developing the SSMQN algorithm was to apply it to a computational expensive simulator model, developing it on such models hardly proved practical considering the time spent on retrieving results after algorithm modifications or tuning. To develop and verify the NMPC application there was a need for quick and

**Algorithm 2** doLineSearch

```
Require: $Y_{new}$, $Y_{best}$, $u_{nom}$, $\Delta u$, $\xi$ and $N_{ls,max}$
  $n_{ls} \leftarrow 0, \alpha \leftarrow 1$
  $runLS \leftarrow$ true
  while $runLS$ do
    $n_{ls} \leftarrow n_{ls} + 1$
    $\alpha_{new} \leftarrow \xi * \alpha$
    $Y_{old} \leftarrow Y_{new}$
    $Y_{new} \leftarrow$ calcObjectiveValue($u_{nom} + \alpha_{new} * \Delta u$)
    if $Y_{old} <= Y_{best}$ and $Y_{old} <= Y_{new}$ then
      $runLS \leftarrow$ false
      $\alpha_{best} \leftarrow \alpha$
    else if $n_{ls} == N_{ls,max}$ then
      $runLS \leftarrow$ false
      $warning \leftarrow$ no good step length found
      if $Y_{new} <= Y_{best}$ then
        $\alpha_{best} \leftarrow \alpha_{new}$
      else
        $\alpha_{best} \leftarrow 0$
      end if
    else
      $\alpha \leftarrow \alpha_{new}$
    end if
  end while
  return $\alpha_{best}$
```

simple model, which could simulate a number of runs in only a few seconds time, but still exhibit severe nonlinearities in order to challenge the algorithm robustness.

### 3.4.1 CSTR model description

The test model used is a CSTR with multiple steady states, previously investigated by both Oliveira and Biegler (1995) and Martinsen, Biegler and Foss (2004). The model equations are given in (3.9). The equations are similar to the system given in Martinsen et al. (2004), with the exception that the effect of inputs $u_1$ and $u_2$ on $\frac{\mathrm{d}x_1}{\mathrm{d}t}$ is quadratic. This modification was done in order to make the system more nonlinear, and does not reflect any physical interpretation.

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = u_1^2 + u_2^2 - k_1\sqrt{x_1} \tag{3.9a}$$

$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = (C_{B_1} - x_2)\frac{u_1}{x_1} + (C_{B_2} - x_2)\frac{u_2}{x_1} - \frac{k_2 x_2}{(1 + x_2)^2} \tag{3.9b}$$

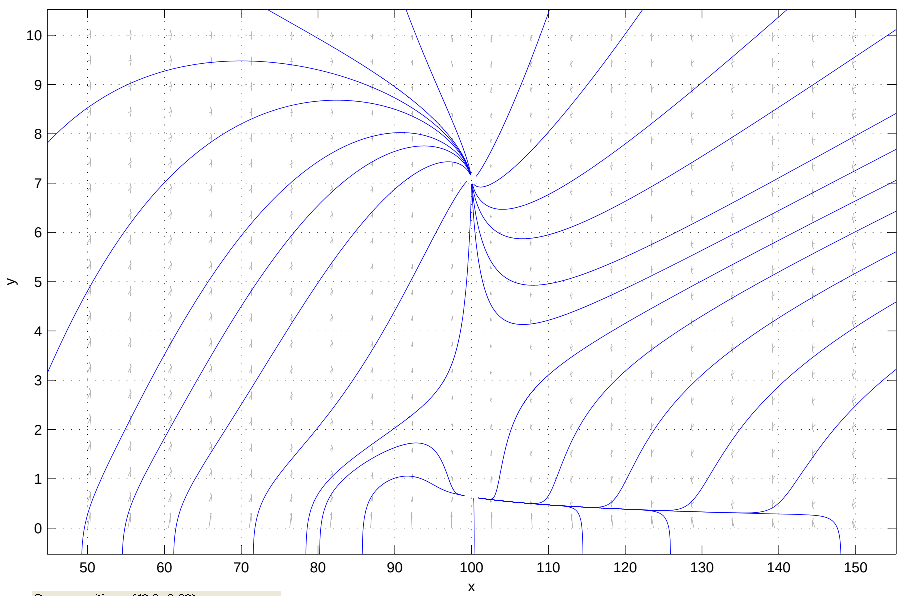

Figure 3.4: Solution trajectors for CSTR system

The initial parameters, following Martinsen et al. (2004), are set to: $k_1 = 0.2$, $k_2 = 1$, $C_{B_1} = 24.9$ and $C_{B_2} = 0.1$. With inputs $(u_1, u_2) = (1, 1)$ the system has three equilibrium points. These can be found at $x_1 = 100$ and $x_2 \in (0.633, 2.78, 7.07)$, the middle one being unstable. Figure 3.4 shows computed trajectories from different starting points with the initial parameter values. The plot was made using pplane7 and the Dormand-Prince solver, and shows the systems two stabile equilibrium points, and the unstable one in $(x_1, x_2) = (100, 2.78)$.

Martinsen et al. (2004) uses a nonlinear combination of variables as the controlled variable to be held at a constant reference value,

$$y = k_1 \sqrt{x_1} x_2, \tag{3.10}$$

giving the extended CV vector

$$x = \begin{bmatrix} x_1 & x_2 & y \end{bmatrix}^T. \tag{3.11}$$

The MVs are gathered in the vector

$$u = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^T. \tag{3.12}$$

To impose additional nonlinearities to the system, $y^{ref}$ is introduced as an arbitrary reference trajectory for $y$ to follow, transforming the problem into a trajectory tracking problem.

### 3.4.2 Algorithm Performance

This section presents the performance of the SSMQN algorithm applied to the CSTR example. The sampling time $T_s$ was set to one second, and simulations where carried out for 20

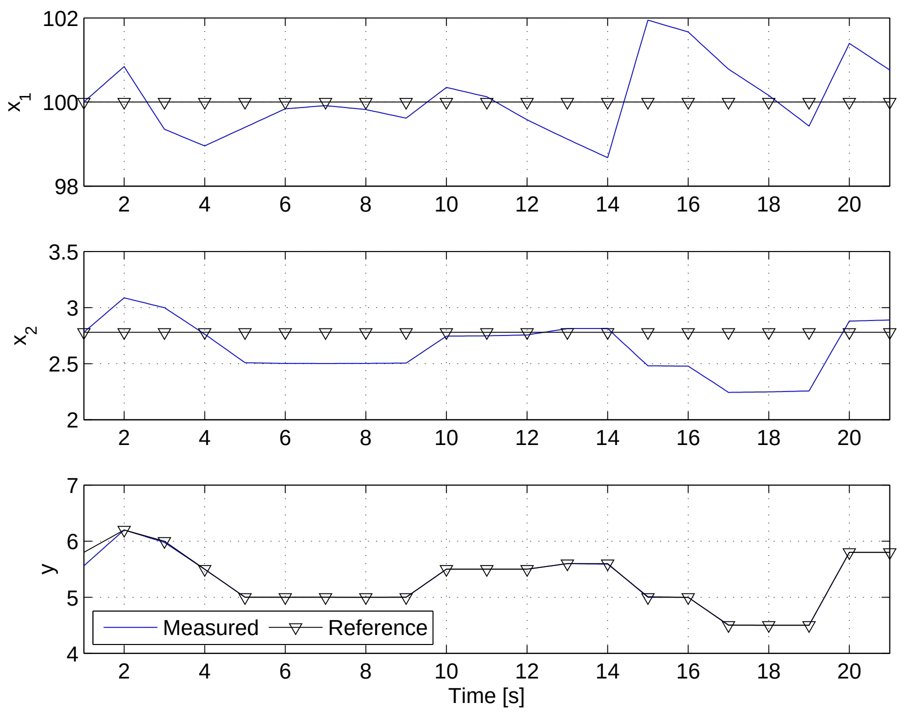

Figure 3.5: CSTR state plot for $y_1^{ref}$

seconds. $T_c$ and $T_p$ was chosen to 30 steps, parameterizing each input with 10 blocks, yielding a total of 20 optimization parameters considering both $u_1$ and $u_2$. Weighting matrices was chosen as

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 4e10^4 \end{bmatrix}, \quad R = P = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}, \tag{3.13}$$

where $P$ is a penalty term added to the cost functional on movements in $\Delta u$ giving

$$\begin{aligned} F(x,u) =& (x - x_{op})^T Q (x - x_{op}) + (u - u_{op})^T R (u - u_{op}) \\ &+ \Delta u^T P \Delta u. \end{aligned} \tag{3.14}$$

The system is attempted driven to the unstable equilibrium point $\begin{pmatrix} x_{1,op} & x_{2,op} \end{pmatrix} = \begin{pmatrix} 100 & 2.78 \end{pmatrix}$, $u_{op} = \begin{pmatrix} 1 & 1 \end{pmatrix}$ and $y = y^{ref}$ for two different reference trajectories.

Simulations for both reference trajectories were done with different values for the linesearch factor $\xi$ and the $MV_{norm}$ convergence criterion, as described in Section 3.3, to investigate the effects of these parameters on algorithm performance. Simulation results are shown in Figure 3.5 for $y_1^{ref}$ and in Figure 3.6 for $y_2^{ref}$. We only show one simulation for each reference vector, since the state plots differ only slightly from one parameter setting to another. As we can se from both figures, the algorithm controls the system in a satisfying manner. The reference trajectory is closely followed by $y$, and only some slight deviations for $x_1$ and $x_2$, as a result of the tuning in $Q$. An additional remark on the plot for $x_2$ must be made in Figure 3.6. The deviation from set point here is caused by the values in $y_2^{ref}$

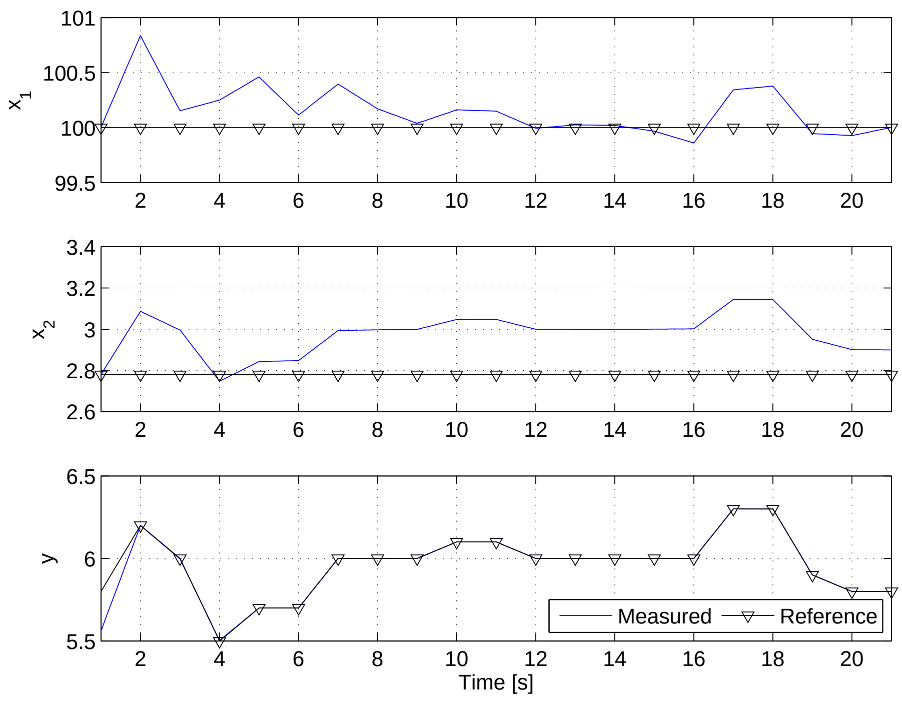

Figure 3.6: CSTR state plot for $y_2^{ref}$

which are found further away from the "nonlinear equilibrium"

$$y_{op} = k_1 \sqrt{x_{1,op}} x_{2,op} \tag{3.15}$$

$$= 0.2 \cdot \sqrt{100} \cdot 2.78 = 5.56. \tag{3.16}$$

Figure 3.7 and Figure 3.8 shows the difference in the algorithm performance for different values of $MV_{norm}$ for both reference trajectories. Both figures shows the performance for the different $MV_{norm}$ relative to $MV_{norm} = 0.01$, along with the total number of simulations done at each sample interval for each value of MV convergence criterion tested. As one might expect, the more relaxed $MV_{norm}$ values perform more poorly then for the more rigid (lower) convergence criteria, but only in a few isolated samples, e.g. sample 11-14 in Figure 3.7 and samples 2-3 and 17-19 in Figure 3.8. The cost of a rigid $MV_{norm}$ value is a high number of extra simulations at each timestep, and may in many cases not be justifiable. The figures show that performing an extensive number of extra simulations does in many cases not yield a significant objective value improvement. When using large commercial simulator models for prediction the simulation time overhead can be of considerable size if the algorithm performs additional simulations without producing objective improvement.

Figure 3.9 and Figure 3.10 shows the difference in the algorithm performance for different values of $\xi$ for both reference trajectories. Both figures shows the performance for different $\xi$ values relative to $\xi = 0.7$, along with the total number of simulations done at each sample interval for each value of backstepping factor tested. Here it is not so obvious what is the effect of the choice of $\xi$, as there are no clear trends shown like in the case of the varying

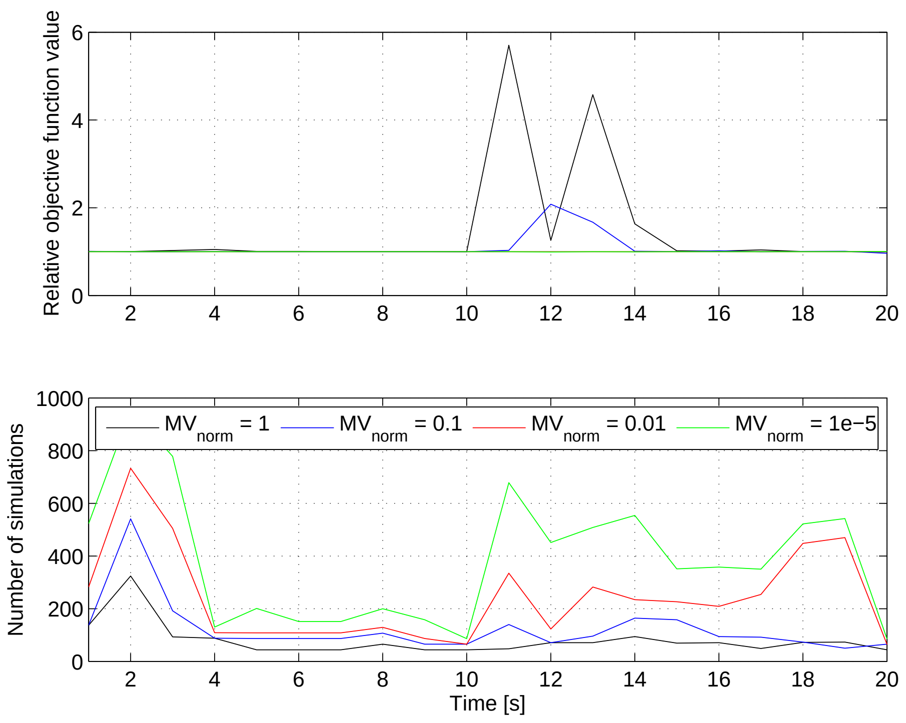

Figure 3.7: Algorithm performance for different $MV_{norm}$ using $y_1^{ref}$

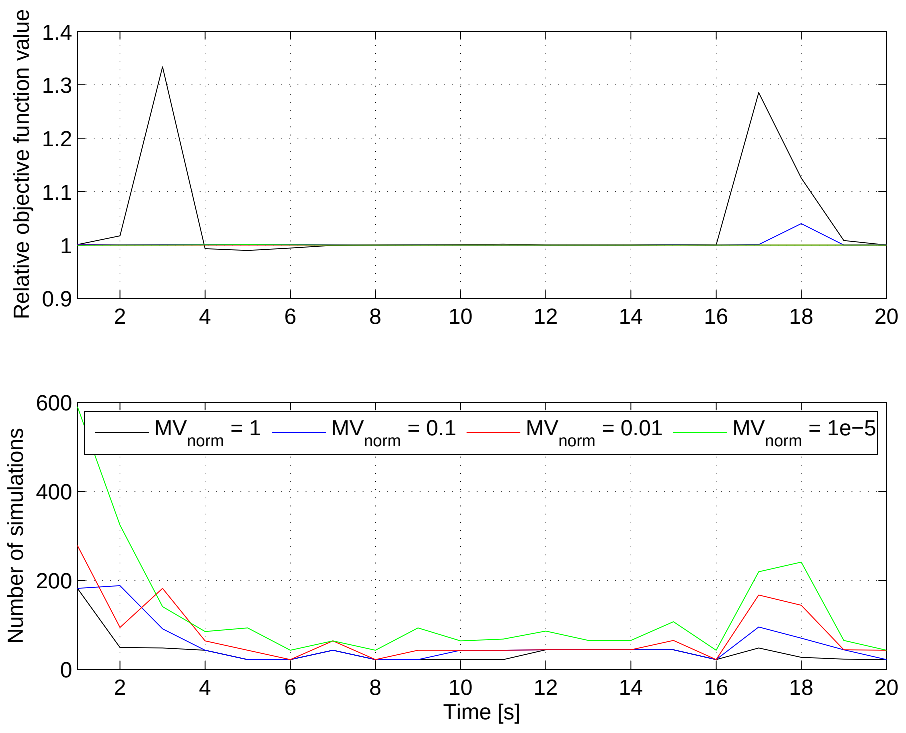

Figure 3.8: Algorithm performance for different $MV_{norm}$ using $y_2^{ref}$

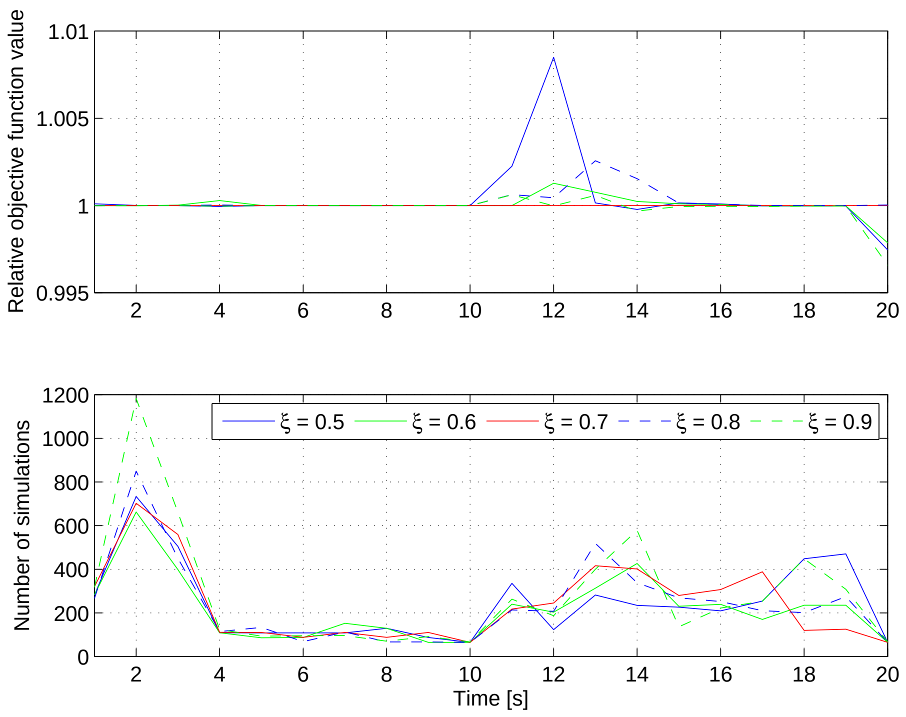

Figure 3.9: Algorithm performance for different $\xi$ using $y_1^{ref}$

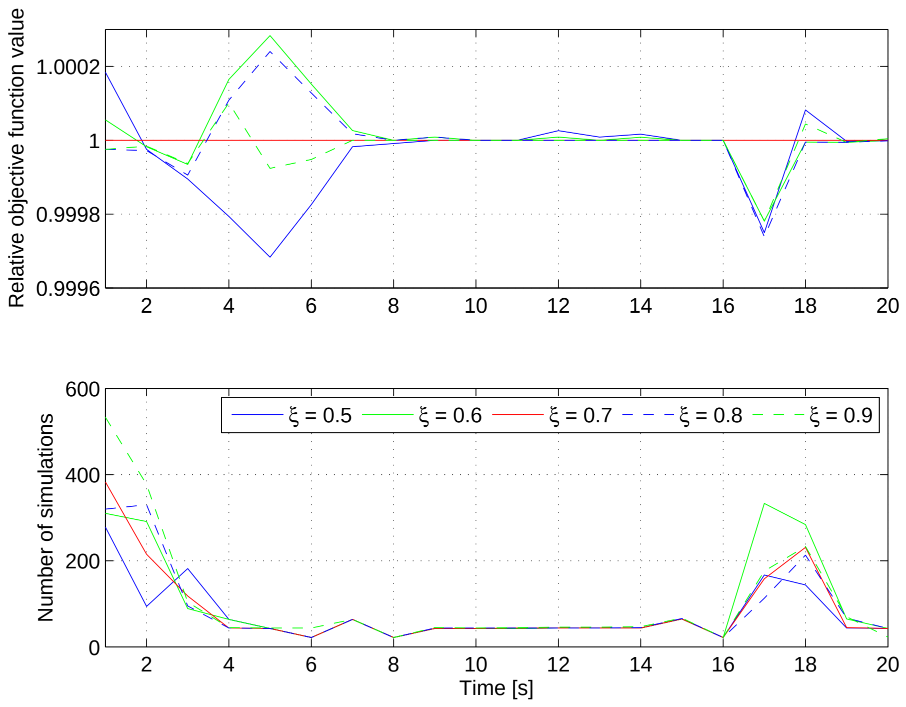

Figure 3.10: Algorithm performance for different $\xi$ using $y_2^{ref}$

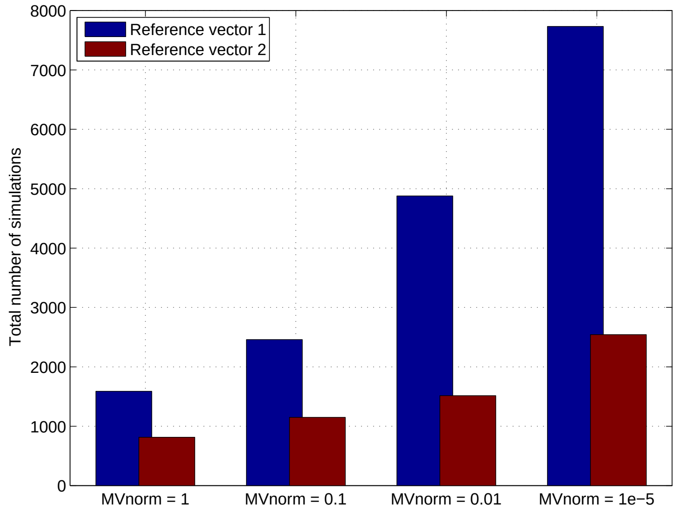

Figure 3.11: Total number of simulations for different values of $MV_{norm}$

$MV_{norm}$. The performance of the different values of $\xi$ gives no indication of any value being superior to the others, and ought to be chosen from knowledge of the underlying process to be controlled. It should be noted that in these simulations $N_{ls,max}$ were chosen large enough to avoid any linesearch from being terminated before finding a satisfying step length, $\alpha_{best}$. If the control application is facing tight real-time deadlines, leaving only a limited number of simulations available for the algorithm to converge and obtain a solution, the $N_{ls,max}$ must be chosen low. If this is the case, the algorithm performance will benefit from a lower $\xi$, as the linesearch will cover a greater part of the $\alpha$ interval $(0, 1]$ in fewer iterations.

Figure 3.11 and 3.12 shows bar plots of the total number of model simulations for each reference vector and for the different choices of $MV_{norm}$ and $\xi$ respectively. These figures further confirm the trends discussed earlier, as we can see a clear effect on the number of simulations from the value of $MV_{norm}$, the number of simulations increasing as the convergence criterion is tightened. The effect of the value of $\xi$ is, however, not as evident. The problem seems to need a certain number of simulations for convergence, regardless of the value of $\xi$.

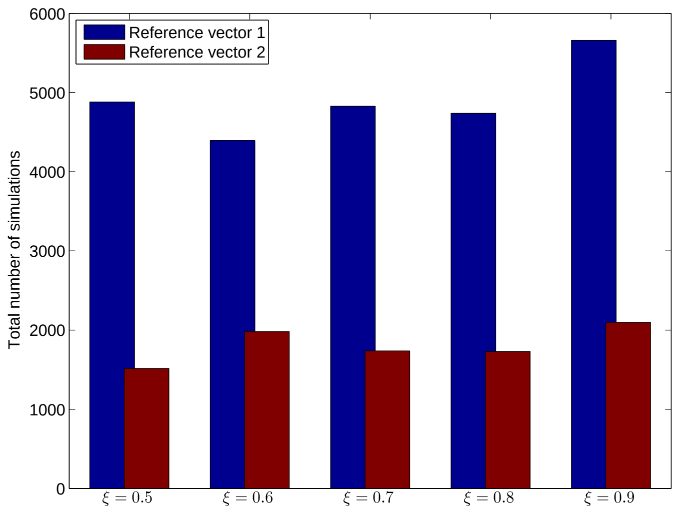

Figure 3.12: Total number of simulations for different values of $\xi$

# Chapter 4

# Hammerfest LNG — from reservoar to shipping

The system simulated in this report is a simplified subsystem of Hammerfest LNG[1] (often referred to as Snøhvit). This chapter attempts to give an overview over Hammerfest LNG, from reservoir to shipping. It is not necessary to read this chapter in order to be able to interpret results later; it can however be desirable to gain a wider perspective on the total process in order to appreciate the choice of boundaries for the simulated system, and the conditions imposed by the surrounding process. A simple and readable review of the Hammerfest LNG subsystems which is the basis for this review can be found at the Snøhvit web pages. (STATOIL, 2005)

The products from Hammerfest LNG is LNG, LPG[2] and condensate which is transported to customers by ship. A schematic diagram of the connection between Snøhvit subsystems is shown in Figure 4.1. Later chapters in this report will focus on the system between the pipeline end manifold (PLEM) and (including) the slug catcher. The slug catcher will in later chapters be referred to as the separator. (While the Hammerfest LNG plant contains several, the subsystem simulated later contains only this one)

From the onshore plant, $CO_2$ and mono ethylene glycol (MEG) is sent subsea. The $CO_2$ is injected into the reservoir for storage, while MEG is used as an anti-freeze and mixed with the well stream in order to avoid the formation of hydrates in the pipeline. The well stream from several templates, each with three wells, is gathered in the PLEM before it is sent into the pipeline. Pressure is controlled to a suitable level with a valve situated at the top of each well. In the simulations, however, the choke situated at the PLEM, called the subsea choke in Figure 4.1, is used to control the flow.

Between the subsea equipment and the onshore plant, a $143\,\mathrm{km}$ pipeline is used for the transportation. The pipeline flow is multiphase, composed of an aqueous phase, consisting of water and MEG, a condensate phase (oil) and a gaseous phase. The pipeline stream is cooled along the pipeline, from about $25\,^{\circ}\mathrm{C}$ at the PLEM to the sea temperature of about

[1]Liquid Natural Gas

[2]Liquid Petroleum Gases; a mix of butane and propane

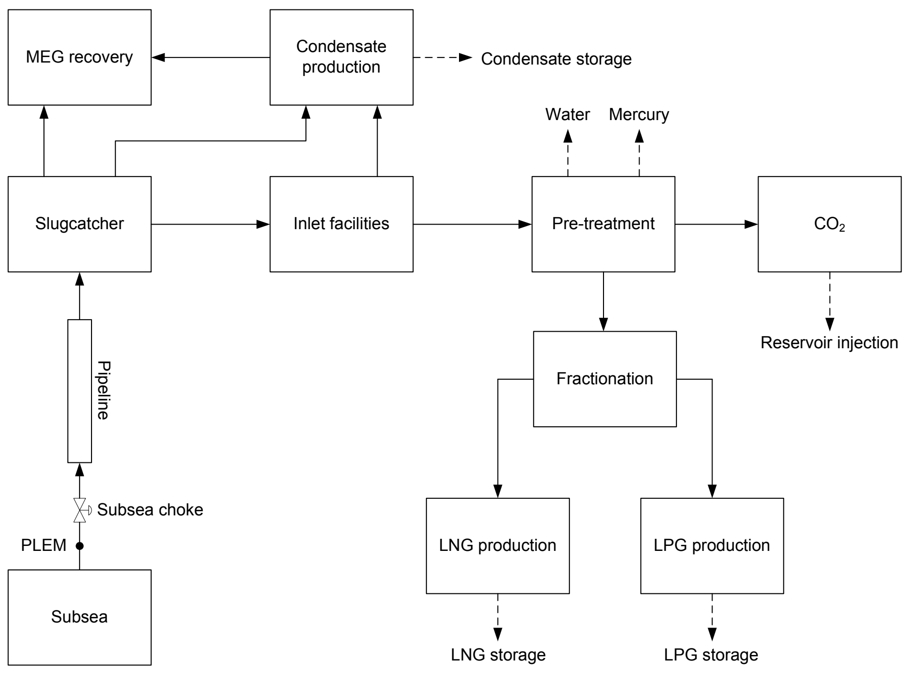

Figure 4.1: Schematic diagram of Hammerfest LNG

$4 - 5\,^{\circ}\mathrm{C}$ when arriving onshore. Because of the fall in temperature, gas will condense along the pipeline. Multiphase flow will be discussed in greater detail in Chapter 5.

Onshore, the pipeline flow is received in the slug catcher. The slug catcher is basically a volume used as a buffer between the field and the plant, smoothing variations in flow and pressure. In addition, the Hammerfest LNG slug catcher provides a first separation between water, gas and condensate.

From the slug catcher the gas is sent to the inlet facilities where it is stabilized at $70\,\mathrm{bar}$ and where a second separation of condensate takes place and particles removed. The condensate is sent to the condensate separator, while the gas is then sent to pre-treatment, where $CO_2$ is removed, the gas dehydrated and mercury removed, in that order. The $CO_2$ is sent offshore for injection. From pre-treatment, the gas is sent to fractionation, where heavier components such as propane and butane (C3 and C4), known as natural gas liquids (NGL), are separated and liquefied to LPG product. The remaining gas is then sent to liquefied natural gas (LNG) production, which basically consists of cooling in three steps. During this cooling, nitrogen is removed. Product LNG has a temperature of about $-163\,^{\circ}\mathrm{C}$.

The condensate production receives streams from different parts of the process. The condensate enters a separator, where gas and water/MEG is removed. Then, light components are removed from the condensate in a separator column and transferred to gas processing (not shown in Figure 4.1).

The aqueous phase from the separator (slug catcher) and the condensate separator (located in the condensate production box in the figure) is sent to MEG recycling. Here, particles, salt and most of the water is removed. The recycled MEG is then sent offshore, once again to be injected into the well stream.

As the name of this chapter suggest, the products produced at Hammerfest LNG are transported to the markets by ship. The LNG are transported on ships especially build for LNG transportation. Main markets for the gas are stated to be USA and countries in southern Europe.

# Chapter 5

# Multiphase flow

THE previous chapter described the Hammerfest LNG systems. However, before the control system developed is described and tested, it is appropriate to describe the pipeline, which is the system to be controlled, more closely. Some of the relevant properties is flow regimes in multiphase flow and how liquid is transported by gas (entrainment). Examples from simulations will be given.

## 5.1 Flow regimes

Multiphase flow has several different flow regimes, also called flow patterns. A short description is given in Fuchs (1997) which is the basis for this section. For horizontal flow, the flow regimes possible includes stratified, slug, annular and bubbly flow, or some intermediate state during transition from one flow regime to another. Only slug and stratified flow occurs in the simulations done in this report. Stratified is distinguished by a layer of liquid film in the bottom of the pipeline, and gas flowing above. Slug flow is characterized by alternating film thickness — slugs where the liquid film is so thick that it occupies the entire pipe cross section area are separated by sections where the liquid film thickness is significantly less.

The pipeline incline is important for when the transition between slug flow to stratified flow or vice versa occurs. If the incline is negative in the direction of the flow, gravity will contribute to liquid flow, reducing film thickness, extending the region of stratified flow. Conversely, if the incline is positive in the direction of the flow, gravity will impede liquid flow, extending the region of slug flow. Even gentle slopes influence the conditions where the transition between slug and stratified flow occurs.

## 5.2 Entrainment

The subject treated in this report is flow assurance in the face of large pipeline transients. Such transients can be pipeline ramp-up or pipeline start-up. The challenge posed by such transients is caused by a phenomenon called entrainment. Entrainment, one fluid moved by another, can be described by the drag equation (Fuchs, 1997):

$$Dr = \frac{A_i}{V} f \rho_G (u_G - u_L)^2 \tag{5.1}$$

where $Dr$ is drag [N/m$^3$], $\frac{A_i}{V}$ interphasearea [m$^2$/m$^3$], $f$ interaction coefficient, $\rho_G$ gas density and $u_G - u_L$ difference in velocity between liquid and gas. Because all parameters except the last are relatively constant, entrainment of liquid is largely decided by gas velocity — low pipeline production and low gas velocity will give a larger hold-up of liquid, while high pipeline production give higher gas velocity and a smaller liquid hold-up. A rate change from high production to low production will therefore start a build-up of liquid in the pipeline. The opposite case, a rate change from low production rate to high production rate, will thus entail the discharge of such a liquid build-up.

## 5.3 Simulations of Snøhvit pipeline — liquid transportation

The preceding section discussed some general mechanisms for liquid transportation. In this section, some multiphase flow effects will be demonstrated using the Snøhvit pipeline.

As earlier stated, it is ramp-up of the production rate that can cause trouble for the processing plant. The trouble is due to a transient period with elevated rates of liquid after the ramp-up. The case simulated is ramp-up of production from 60% (12.8 MSm$^3$/d) to 100% (20.8 MSm$^3$/d) over one hour. Transient effects can be observed over a period of about 25 hours after the production rate ramp-up. Figure 5.1 contains graphs of the liquid hold-up along the pipeline during this period, while Figure 5.3 contains the same graphs for the liquid mass flow. The graphs of liquid hold-up and liquid mass flow have been smoothed using weighted moving average in order to improve readability, since the true graphs are "noisy". A comparison of true and filtered liquid hold-up and liquid mass flow can be found in Figures B.1-B.2.

Starting with the figure of liquid hold-up, it is obvious that the amount of liquid at steady-state conditions depends on the gas velocity, and through gas velocity, also gas rate. The graph showing liquid hold-up at steady state 100% production is below the graph showing liquid hold-up at steady state 60% production through the entire length of the pipeline. It is interesting to note the correlation between low liquid hold-up at steady state and the pipeline profile shown in Figure 5.2. The decline in pipeline height at about 110 - 120 km corresponds to low liquid hold-up. Conversely, the upward slope between 50 and 100 km has high values of liquid hold-up.

While the liquid hold-up is closely related to the incline of the pipeline under steady state conditions, the correlation is weaker during the transient. The reason for this can be found in the plots of liquid mass flow in Figure 5.3. The ramp-up starts a wave of liquid that moves through the pipeline. When the liquid wave passes through some section of the pipeline, the liquid hold-up in that section will increase relative to the rest of the pipeline, thus obliterating the correlation between incline and hold-up.

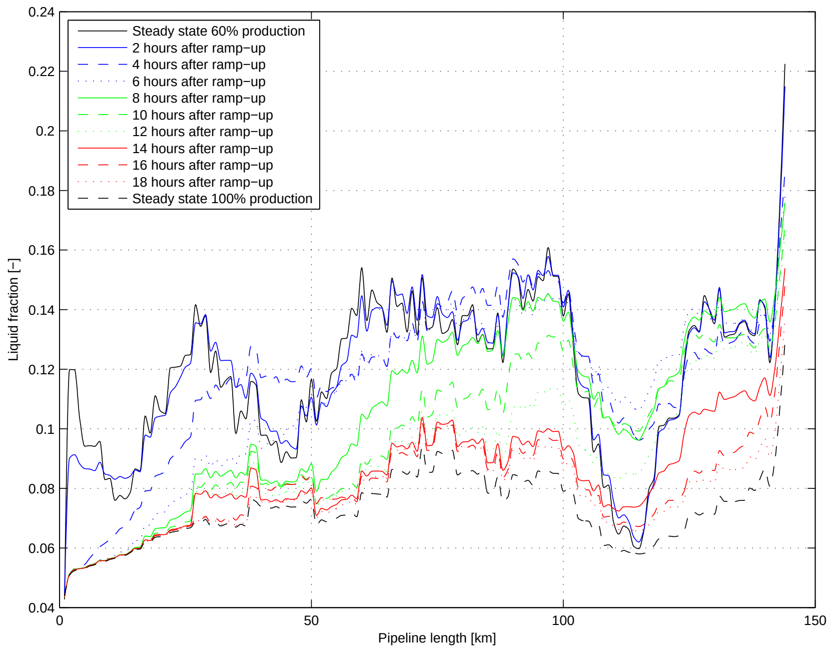

Figure 5.1: Liquid hold-up during step in production rate from 12.8 MSm$^3$/d to 20.8 MSm$^3$/d

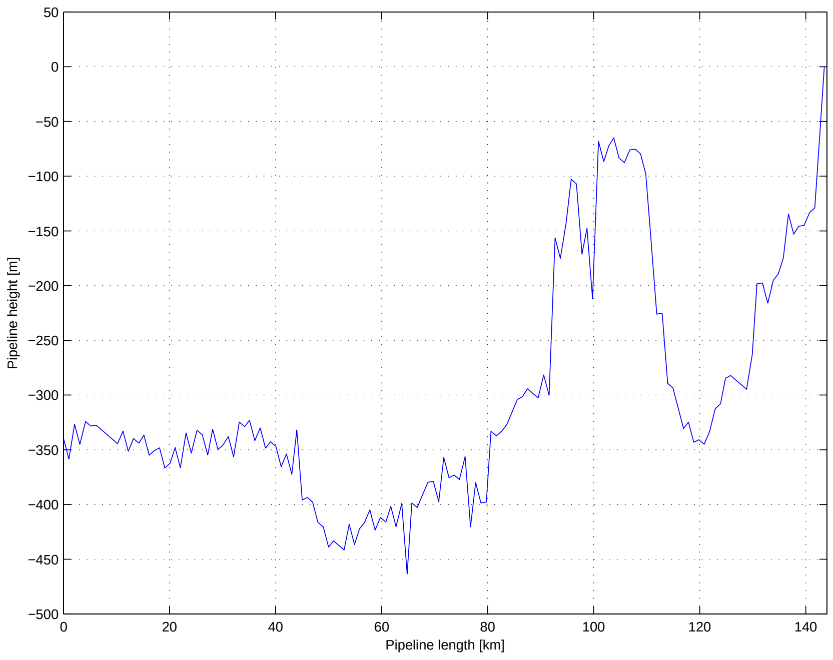

Figure 5.2: Pipeline profile

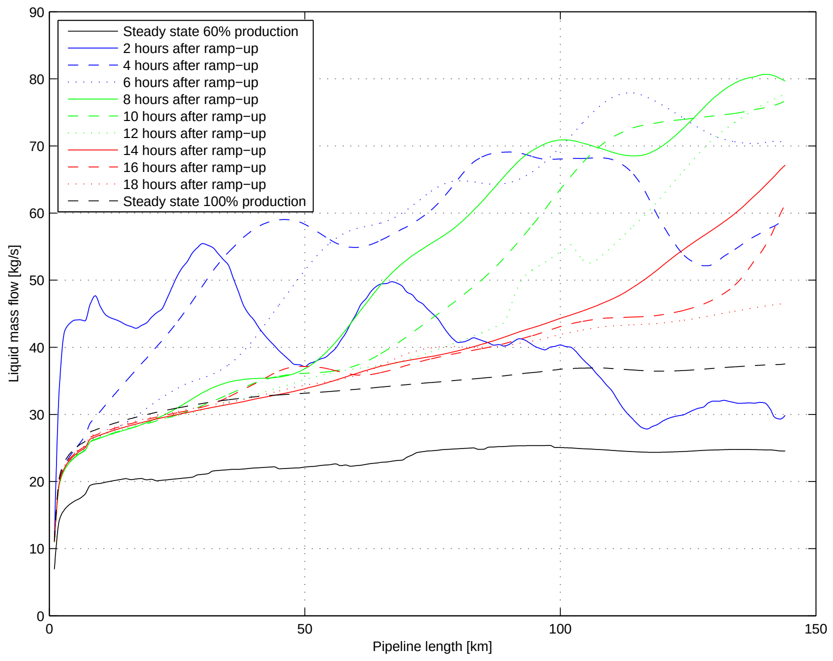

Figure 5.3: Liquid mass flow during step in production rate from 12.8 MSm$^3$/d to 20.8 MSm$^3$/d

During the ramp-up, the pipeline model estimates that the pipeline flow will be stratified everywhere, except at some sections with high hold-up relative to the rest of the pipeline, where slugging may occur. This form of slugging are referred to as *hydrodynamic slugging* by Storkaas (2005). It is described as caused by liquid waves growing on the gas-liquid interface eventually closing the cross-section. As the slugs disintegrate after the uphill pipeline section, this sort of slugging does not present any trouble for the downstream separation equipment.

# Chapter 6

# Problem description and simulation setup

THIS chapter starts with a presentation of the problems to be addressed. These will be further demonstrated through simulation results in Chapter 7, and discussed in Chapter 8. Also, this chapter contains a description of the setup used for the simulations with emphasis on models and controller structures used.

## 6.1 Problem to be addressed

In the previous chapter it was described how changing from low to high production of a pipeline will result in elevated liquid rates for a period. It is this discharge that poses the challenge which will be addressed in this report. For the purpose of maximizing production, it is ideal to use a separator large enough to handle all possible liquid rates. However, the cost of such a separator will often be very large (the Hammerfest LNG slug catcher costs 500 MNOK) and may not be possible due to space limitations. Downstream process facilities with limited liquid handling capacity, will thus requiring some sort of strategy in order to avoid overflowing the separator during ramp-ups. Such a strategy can be guidelines for ramp-up for the operators to follow, or it can be some sort of closed-loop control. The strategy developed and tested in this report falls into the latter category. The idea is to use model information about liquid inventory in order to predict onshore liquid arrival and to meet the constraints on liquid arrival imposed by the receiving facilities. The production rate should be maximized while respecting the constraints on liquid discharge. Optimizing ramp-up speed can be exploited in two different concepts. Firstly, new projects can consider reducing separator size, as better ramp-up speeds are possible. Secondly, existing separators can be used more efficiently, reducing production loss during pipeline ramp-ups.

## 6.2 Pipeline model

Although the pipeline model was provided by STATOIL, and a commercial product, OLGA version 5.0.2 (Scandpower, 2006), was used to simulate the pipeline, some modifications was necessary in order to generate a model suitable for control. Also, using OLGA as a prediction model required a new model interface. This section contains a short description of the work done on the pipeline model.

### 6.2.1 Using OLGA as model for prediction

OLGA is in this report used as the prediction model, which requires that the MPC is able to start simulations and wait until they are completed before moving on. Waiting on simulations are not necessary when using OLGA as process model — measurements from the process model are simply sent to the MPC when they become available, and the MPC sends the control inputs back when they are computed. Previously, STATOIL has used OLGA as the MPC process model, communicating over OPC (OLE for Process Control). It would probably be possible to use a similar solution using flags to signal simulation start and finish between the MPC and OLGA. However, such an implementation would probably require much more logic than the chosen solution which is to make a dynamic link library (DLL) from the OLGA-MATLAB toolbox (Scandpower, 2005b). A description of how such a DLL can be generated is given in Appendix A. It is not necessary to know how the link is implemented in order to understand the results given in this report. The description is, however, attached for completeness and because considerable time was spent developing this solution.

### 6.2.2 Simplification of OLGA model

The original OLGA model had a detail level that enabled the simulations to be performed at about 20 times real-time speed. While this speed is suitable for testing different scenarios, it would have made for very long computation times if used as predictive model for calculating sensitivity matrices. Consequently, the smallest pipeline section lengths was increased from $200\,\mathrm{m}$ to about $1000\,\mathrm{m}$ using OLGA's integrated profile tool, Grid Generator (Scandpower, 2005a). Grid Generator is able to do this simplification while the angle distribution is retained. The angle distribution is plays an important role in the calculation of liquid hold-up. Maintaining the angle distribution is done at the cost of changing the pipeline profile. The amount of change in pipeline profile is dependent on the amount of simplification. Original and simplified pipeline profiles is shown in Figure 6.1. The extra bulges seen on the simplified profile are due to keeping the angle distribution and increasing the section length.

Model behaviour changed due to the simplification. While the steady state rates of gas, oil and liquid are largely correct, the same can not be said for liquid hold-up. Since the liquid hold-up are changed, so are also the rates during pipeline production ramp-up when surplus liquid is discharged. A more accurate model with similar reduction in simulation

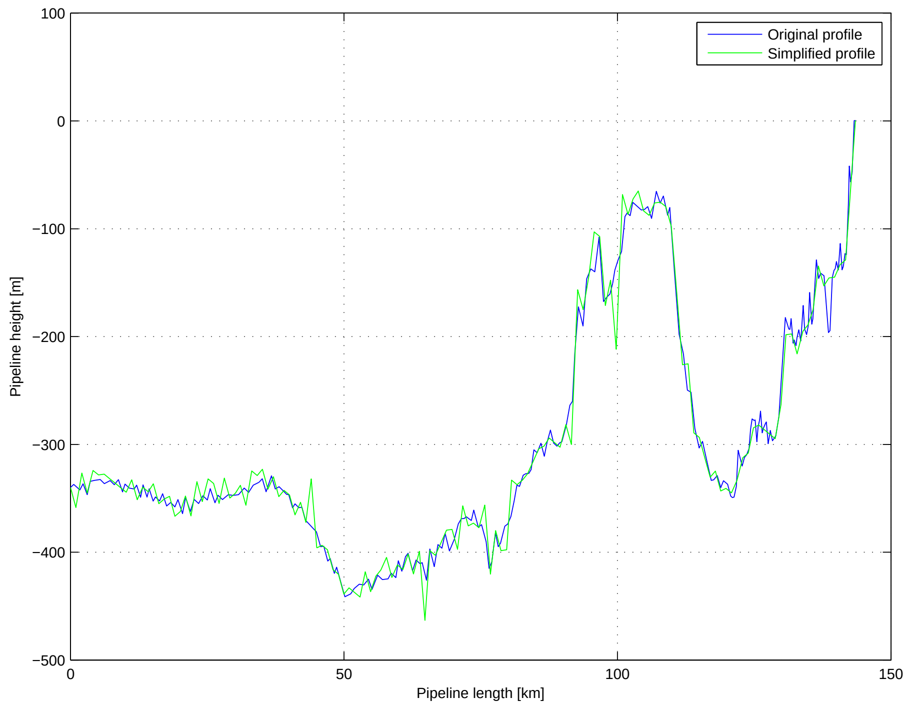

Figure 6.1: Original and simplified pipeline profiles

time could probably have been obtained. However, pipeline modeling was not the subject of this report, and the simplified pipeline model is assumed to be correct. The use of the simplified pipeline was made necessary by the fact that during the course of this report, a total of 3 to 4 years of pipeline flow was simulated. A fast model was therefore required. By changing the model, the relevance to the Snøhvit pipeline and the Hammerfest LNG plant is, however, somewhat reduced.

### 6.2.3 Pipeline source

Subsea, the pipeline end manifold (PLEM) connects the stream from all the wells before it is sent through the pipeline. The separate wells and the pipelines connecting them to the PLEM was not simulated. Instead, the PLEM was simulated as a boundary having constant temperature and constant pressure. A choke was then used in order to limit the flow through the boundary. Not modeling the different wells are of course a simplification, it is however a question of how much of the system that should be modeled, and the PLEM was considered to be a natural boundary.

## 6.3 Separator model

The separator (slug catcher) downstream from the pipeline needed to be modeled, as the separator liquid levels and the pressure was used as controlled variables. Flexibility considerations demanded that the separator should be modeled outside OLGA. If the separator had been modeled in OLGA, a separate initial state file would have been required for each size of separator used. Also, by modeling the separator outside of OLGA, full control over liquid controllers could be afforded. For practical reasons, the separator was modeled in SEPTIC.

The modeling accuracy reflects the fact that separator dynamics is not the subject of this report. The main quality of the model is modeling liquid levels. Several simplifications were used:

1. Water and oil (condensate) are incompressible
2. The pressure is the same in the entire separator and is a linear function of gas density.
3. The separator has a constant temperature of $5\,^{\circ}\mathrm{C}$
4. There are no phase transition between gas and oil (condensate)
5. Separation is assumed perfect

Modeling the separator outside OLGA raises the question of how to provide feedback from the separator model to the pipeline model. In the OLGA model, the pipeline ends in a pressure boundary. By updating the boundary pressure with the separator pressure a connection between the models are made. Figure 6.2 shows the pressure boundary. A minimum pressure boundary update frequency is required unless the separator pressure will become unstable. This is due to the fact that the boundary pressure is held constant over a sample. For example, let the current boundary pressure be $80\,\mathrm{bar}$. OLGA calculates a mass rate, say $40\,\mathrm{kg/s}$ which is sent to the separator model. Now, let the mass flow out of the separator be $30\,\mathrm{kg/s}$. If the sampling time, $\Delta t$ is 5 minutes, this will result in $3000\,\mathrm{kg}$ of additional mass in the separator. If little capacity is available, either because the separator is small or that it is nearly full, the extra mass will result in a significant increase in pressure. The increased pressure will cause OLGA to calculate much smaller mass rate in the next sample, which causes the pressure to drop. This effect may result in an oscillating system. Thus, in order to retain the self stabilizing quality of the true system, it is necessary to select the sampling time sufficiently small. The discussion is analogous to that of a digital controller implementation.

In Figure 6.2, let $Q$ denote volume rate, $V$ denote volume, $m$ denote mass rate,$M$ denote mass and $\Delta t$ denote the sampling time. Then the separator equations can be given as (6.1)-(6.3).

$$V_{water,k+1} = V_{water,k} + \Delta t(Q_{water,in} - Q_{water,out}) \tag{6.1}$$
$$V_{oil,k+1} = V_{oil,k} + \Delta t(Q_{oil,in} - Q_{oil,out}) \tag{6.2}$$
$$M_{gas,k+1} = M_{gas,k} + \Delta t(m_{gas,in} - m_{gas,out}) \tag{6.3}$$

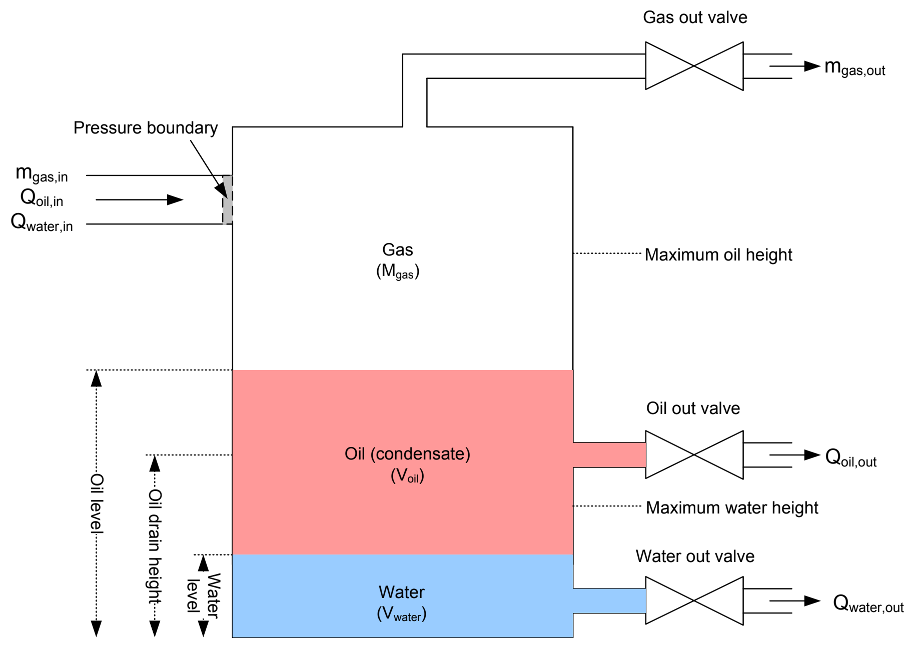

Figure 6.2: Separator model

To calculate pressure, first the density must be obtained. This is done simply by dividing the gas mass by the available volume:

$$\rho_{gas} = \frac{M_{gas}}{V_{total} - V_{water} - V_{oil}} \tag{6.4}$$

The relationship between density and pressure is obtained by linearizing data from the OLGA file containing the fluid properties. Equation (6.5) shows the relationship used. The factors $a$ and $b$ was found for an operating point of $80\,\mathrm{bar}$ and $5\,\mathrm{degC}$.

$$\begin{aligned} p_{separator} &= a * \rho_{gas} + b \\ a &= 7.2085 * 10^4\,\mathrm{m^2/s^2} \\ b &= 1.6540 * 10^6\,\mathrm{Pa} \end{aligned} \tag{6.5}$$

Figure 6.2 shows valves on the outlets for gas, oil and water. These valves are not modeled. Instead, the desired outlet rates are set directly. This is analogous to having perfect flow controllers. For oil and water, the outlet rates are set by simple P-controllers. The choice of controller is not important, as long as reasonable performance is achieved.

The parameters for the full-size separator are given in Table 6.1. The separator are conceptual and several aspects such as the specific shape are left undetermined. It is however assumed that the separator is cylindrical, with a fixed height and fixed maximum water and oil drainage rates. The full-size separator parameters in Table 6.1 will later be referenced as the 100% separator, while smaller separators will be referred to as $p\%$ separators where $p$ will be the new base area. Volume and rates are given in Knutson (2005) and Eagleton (2001).

| Parameter name | Value |
|---|---|
| Base area | $100\,\mathrm{m}^2$ |
| Total height | $32.45\,\mathrm{m}$ |
| Oil drain height | $9.15\,\mathrm{m}$ |
| Water drain height | $0\,\mathrm{m}$ |
| Maximum oil drainage | $295.3\,\mathrm{m}^3/\mathrm{h}$ |
| Maximum water drainage | $32.4\,\mathrm{m}^3/\mathrm{h}$ |

Table 6.1: Separator parameters

## 6.4 NMPC implementation

After having decided which models to use, the question of how to implement the NMPC remains. Implementation poses several issues — which CVs to control and which MVs to control them with, weight matrices, sampling time, prediction horizon and control horizon.

### 6.4.1 Prediction and control horizon

Since the purpose of the controller is to discharge liquid optimally during a pipeline transient, it is essential that the controller sees the entire course of the transient. The prediction horizon, $T_p$, therefore needs to be set larger than the transient behaviour. For the choice of control horizon, $T_c$, Strand (1991) states that robustness usually increases with decreasing $\frac{T_c}{T_p}$, while nominal performance increases when $\frac{T_c}{T_p}$ approaches unity. Since robustness is not an issue in this report, $T_c$ was chosen equal to $T_p$.

### 6.4.2 Control structure

Finding an optimal start-up strategy requires that all relevant degrees of freedom are available to the controller. Determining which inputs to use is therefore of great importance. Controller structures using different combinations of inputs were therefore tested, along with adding a topside choke as an extra MV. The original control structure at Hammerfest LNG is shown in Figure 6.3. The PLEM and the separator outlets were chosen as system boundaries.

Inside the system boundaries, multiple objectives can be identified. Not all objectives needs to be accounted for in each proposed control structure. The control objectives in descending order are:

1. Keep separator pressure and separator water and oil levels within given limits
2. Keeping the gas rate out of the separator smooth and non-decreasing during the pipeline ramp-up.
3. Maximize gas rate out of separator
4. (Avoid unnecessary topside choking)

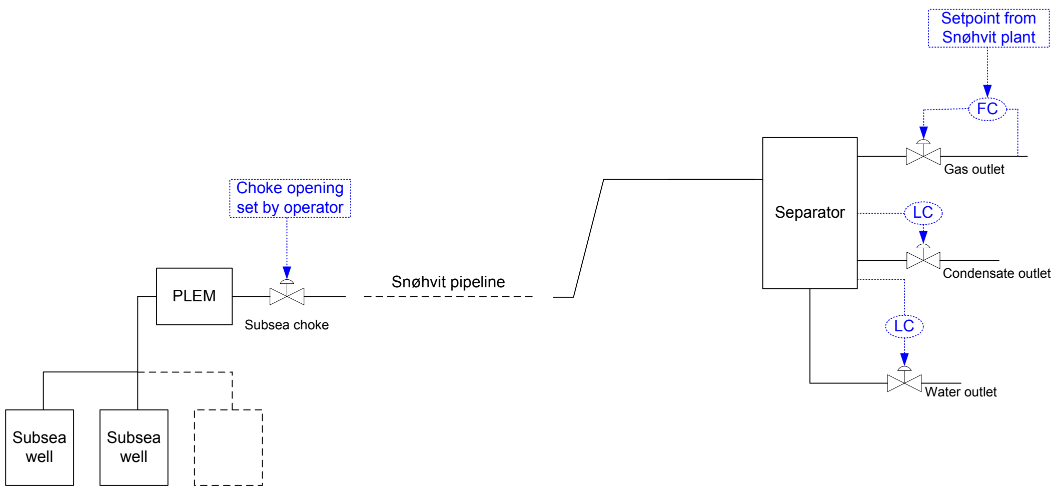

Figure 6.3: Original pipeline control structure at Hammerfest LNG

The last objective is given in parenthesis as this is only a relevant if a topside choke is present. Now, keeping these objectives in mind, different control structures are discussed.

### Original control structure

The solution that will be used at Hammerfest LNG is given in Figure 6.3. The pipeline rate is set by using the subsea choke, while the gas rate required by the plant is drawn by using the gas outlet choke. No restriction of pipeline flow is needed since the separator is designed to be large enough to accept the surge waves generated by ramp-ups and start-ups (See Knutson (2005) and Eagleton (2001)). Separator pressure floats on the pipeline pressure. Thus, as long as roughly the same gas rate flows into the pipeline as flows out of the separator, the separator pressure will be relatively slow varying and stable.

### Controlling the gas rate

As gas rate has great impact on liquid movement in the pipeline, any control structure with the purpose of controlling liquid discharge needs to in some way control the pipeline gas flow. The simplest control scheme is to only control the gas rate out of the pipeline. Figure 6.4 shows the structure used, except that the subsea choke will be controlled by some simple controller, or set by operators. In later simulations, a basic controller controlling the separator pressure will set the subsea choke in this setup. Thus, the objectives to be addressed in the NMPC are: liquid levels, smooth and non-decreasing gas rate and maximizing gas rate output.

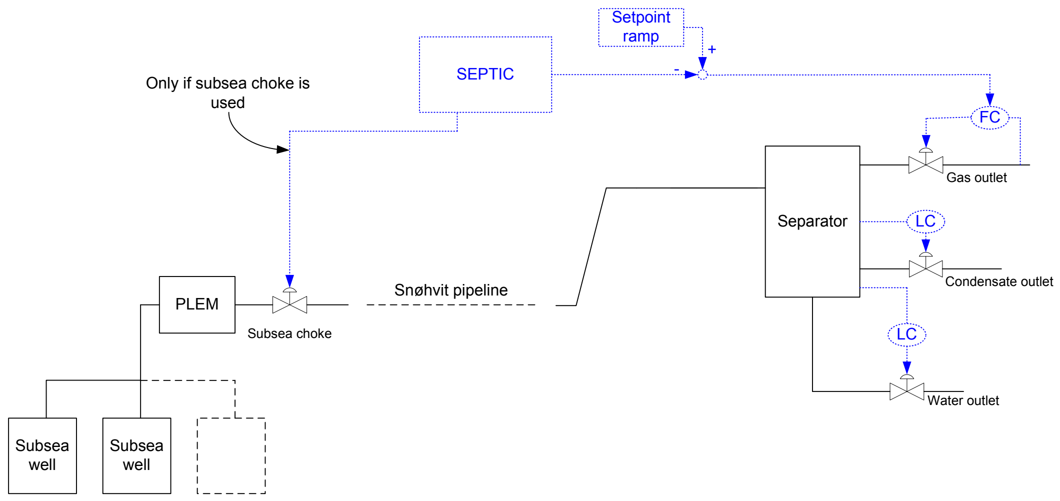

Figure 6.4: Control structure - controlling subsea choke and topside gas rate

### Controlling gas rate and subsea choke

Enhancing the controller setup with NMPC control of the subsea choke, the controller structure in Figure 6.4 is obtained. Because the subsea choke is included as a MV, it is possible for the NMPC to adjust the separator pressure. Hence, in addition to the objectives stated for the previous setup, the separator pressure must also be controlled.

### Controlling subsea and topside choke

By introducing a topside choke (see Figure 6.5), a dynamic degree of freedom is introduced. Using the topside choke, the pipeline downstream pressure can be changed more rapidly, compared to using the choke on the gas outlet from the separator. The topside choke is controlled by a PI flow controller whose set point is provided as a MV to the MPC. Since the pressure difference over the topside choke can vary significantly, gain scheduling are used to achieve consistent performance for various pressure differences. Gain scheduling for the topside choke is described in Torpe (2006).

While the benefit of using a topside choke will be discussed in later chapters, some other effects of introducing a topside choke needs to be discussed. When using a topside choke the separator pressure potentially becomes uncoupled from the pipeline pressure. The more the topside choke is closed, the more uncoupled from the pipeline pressure the separator pressure becomes. As the separator volume is much smaller than the pipeline volume, this makes the separator pressure more sensitive for differences in flow in and out of the separator. If the NMPC is to control the pressure with the topside choke, a maximum sample time is imposed if the pressure is to be kept within the pressure limits. The discussion is quite similar to that in Section 6.3. There, it is argued that the maximum sample time is dependent on the separator size. The smaller the separator size, the smaller the maximum sample time becomes. This limitation is unnecessary, as model predictive control is not needed for pressure control of the separator. A simple PI-controller is sufficient for this

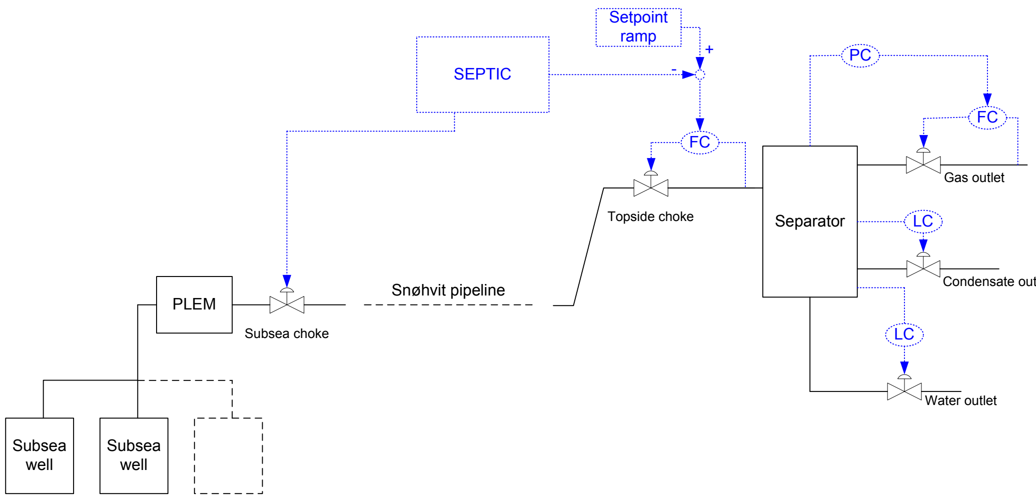

Figure 6.5: Control structure - using a choke upstream the separator

purpose. The pressure controller sets the set point for the flow controller controlling the gas rate out of the separator. A disadvantage with this solution is that the adjustments made in the flow out of the separator become a disturbance for the Melkøya plant, undermining the separators function as a buffer.

The NMPC will in this setup have no direct control of the gas flow out of the separator. The control will be indirect through keeping the separator pressure high enough to maintain the wanted gas flow. A potential enhancement of this setup would be to provide the separator pressure setpoint as an additional MV.

### Loss of performance from using distributed level control

Some low-level control is built around the separator to limit the size and complexity of the MPC. Two PI-controllers controls the water- and the oil level respectively. This removes the need for level set point and low level limit. However, because of the maximum liquid outlet rates (i.e. actuator limitations), low level control cannot ensure that liquid level high limits will be respected.

The computational cost of calculating the sensitivity matrix is proportional to the number of input parameters. Keeping the number of inputs small is therefore vital in order to keep computation times reasonable. Control of separator liquid levels were considered of less importance to the control performance. This section is an attempt to justify that assertion.

During normal operation, control of liquid levels is fairly straight forward. The levels should be held low and a smooth flow of condensate out of the separator maintained. However, during transients, when liquid are discharged from the pipeline, the liquid flow out of the separator should increase until the maximum separator liquid drain rates are reached. Both of these tasks can be performed adequately by correctly tuned P or PI controllers.

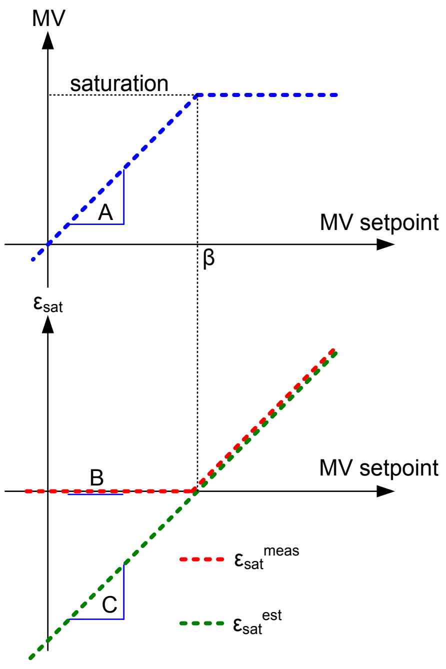

Figure 6.6: Saturation CV - using negative saturation

### 6.4.3 NMPC challenges

When implementing NMPC, some challenges were encountered. Two examples of such challenges and the solution chosen will follow.

#### Time varying actuator limitations

When actuator constraints are time varying, they cannot be included explicitly in the NMPC. Thus, the NPMC is not able by itself to determine if the inputs are saturated. The sensitivities obtained by the NMPC will therefore not necessarily be correct, as a perturbation in a saturated input will have no effect on the controlled variables. This section will describe a solution for this problem, and how it was implemented in the control structure shown in Figure 6.5.

For a NMPC to be effective, the controller needs to see how much the inputs can be changed until some CV reaches its limit. When an input is saturated, a change in a MV will have no effect on CVs, thus effectively removing the MV in question as a degree of freedom.

In Figure 6.6, a MV with a time varying high limit, $\beta$, is shown. If the MV set point exceeds $\beta$, the violation is penalized by introducing a CV that measures the saturation, $\epsilon_{sat}^{meas} = MV_{setpoint} - \beta_{measured}$, with a high limit of zero. Measured $\epsilon_{sat}$ can be seen as the red line in Figure 6.6. Point B illustrates the fact that as long as the MV is not saturated, $\epsilon_{sat}^{meas}$ (and its sensitivity to perturbations in the MV set point) will be zero. Hence, the MPC cannot judge how much the MV set point can be increased before $\beta$ is reached. The desired

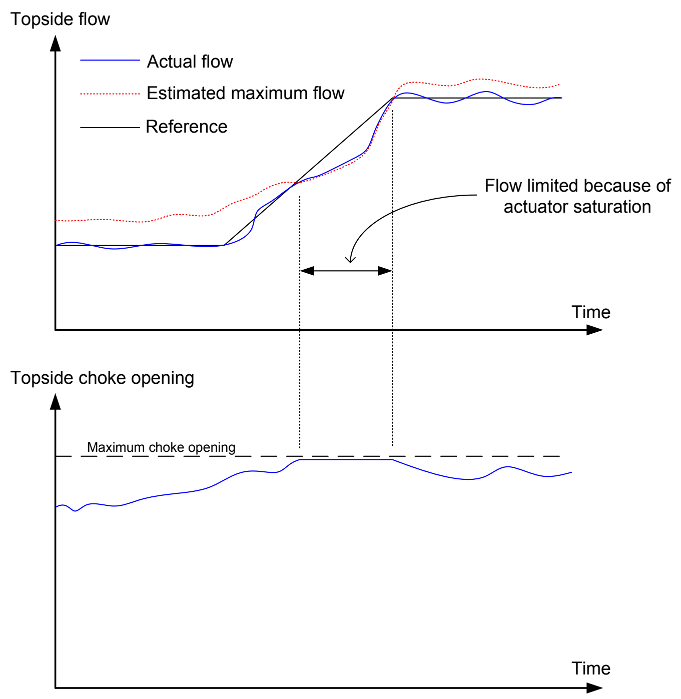

Figure 6.7: Saturation of flow controller

sensitivity is shown as point A. In order to estimate the sensitivity shown in point A, the point at which the MV saturates (here given as $\beta$) must be estimated when the MV is not saturated. $\epsilon_{sat}^{est}$ is then calculated as $MV_{setpoint} - \beta_{estimated}$ and is shown as the green line in Figure 6.6. The sensitivity obtained in point C is a good approximation of the sensitivity in point A as long as the estimate, $\beta_{estimated}$, is good.

In the controller setup where a topside choke is used, the set point to the topside flow controller is an example of an input with a time varying actuator limitation. The maximum flow is a function of the differential pressure over the topside choke, which will vary over time. This issue is managed using the above described technique. That is, when the topside choke is not fully opened, a simple model is used to estimate how much the gas rate can be increased before the topside choke controller is fully opened. Figure 6.7 shows the estimated maximum flow in red and the actual flow in blue. This estimation of negative flow limitation is necessary in order to obtain nonzero sensitivities as shown in Figure 6.6. However, keeping the topside flow controller within actuator limitations, that is, the controller is active, effects the sensitivities obtained from the subsea choke. The subsea choke controls the pipeline flow. However, the gas rate set point is set at the topside choke. If the topside gas rate controller is active, a perturbation in subsea choke opening will have little or no effect on the topside gas rate, effectively ruining the sensitivity accuracy. The solution is to allow the flow controller to be saturated by a small value. A perturbation in the subsea choke opening will then result in a change in the gas rate.

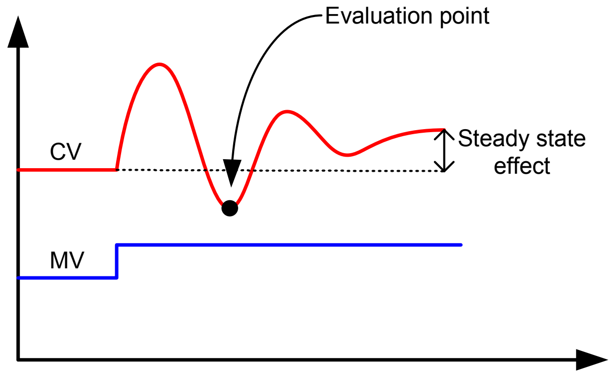

Figure 6.8: Inaccurate sensitivity due to dynamic CV response

### Sensitivity problems using CVs with very dynamic responses

Some controlled variables have naturally very dynamic responses. Obtaining correct sensitivities from such CVs are a challenge. Figure 6.8 shows a perturbation in a MV and the response of the perturbation in the CV. As can be seen, the change in CV is in the same direction as the perturbation. However, the CV value in the evaluation point shows a change in the opposite direction. Thus the sensitivity value will not only be wrong, but also have a wrong sign! Inaccurate sensitivities may cause the next search direction to be found to be non-descending, causing the SQP algorithm to abort. Generally, it would be preferable to model the CV such that large dynamic effects that need not be accounted for is ignored. It would also be possible to use a shorter the NMPC sampling time such that the dynamic response is visible to the NMPC. In cases where this is not possible, it is important to choose CVs and MVs such that the dynamic response of the CV is minimized.

# Chapter 7

# Results

THE simplified Snøhvit pipeline and the separator model discussed in the previous chapter has been used to test the proposed control system, *smart liquid control*. This chapter contains the most relevant and interesting results. First, a description of the two simulation scenarios used is given. Then, the results of the two scenarios are given respectively. A comparison between smart liquid control and manual control is made, before the last section gives some results used for demonstrating the effects of using a topside choke.

## 7.1 Scenarios simulated

Among the reports written on the Hammerfest LNG project, several adress flow assurance. Two reports, one engineering report by BMH Eagleton (Eagleton, 2001) and an internal STATOIL report (Knutson, 2005), discuss the slug catcher size and the scenarios that the slug catcher should be able to cope with. The information in this section is derived from these reports. Two cases have been identified as critical for liquid management of the Hammerfest LNG pipeline. Each case place emphasis on distinct capacities.

- *Case I*: Ramp-up of pipeline production from 60% to 100%
- *Case II*: Start-up of pipeline from shut-in conditions

The former case, *ramp-up of pipeline production from 60% to 100%*, places the greatest demands on water handling capacities. The pipeline should be produced at 60% until pipeline equilibrium is reached. Equilibrium in this context refers to the amount of liquid hold-up in the pipeline, that is, the volume of oil and the volume of water in the pipeline is constant. Some time after the equilibrium has occurred; the pipeline should be ramped up to full production as quickly as possible in order to maximize production. The latter case, *start-up of pipeline from shut-in conditions* will place the greatest demands on the oil handling. The pipeline is shut-in from full production, and remains shut-in until all states are stabilized, i.e. the temperature is converged to the ambient temperature. It is then assumed that the Hammerfest LNG plant has had a major stop, requiring the pipeline to be produced at 30% for 24 hours until all systems are fully operational. During the 24 hours where the pipeline is produced at 30%, no liquid will arrive onshore, causing a build-up of liquid in the pipeline. After the 24 hours, the plant is fully operational and the production should be ramped up as fast as possible. A closer investigation of the scenarios and the pipeline behaviour can be found in Torpe (2006).

## 7.2 Smart liquid control: ramp-up of production

This section will look at how separator size reduction affects the pipeline performance. That is, the total production of gas during the ramp-up. As discussed in the previous section, the ramp-up will be from 60% to 100% production and the scenario will challenge the water handling capacity. The control structure used was control of gas rate out of separator and basic control of the subsea choke. The total gas production during the 40 hours the system was simulated is given in Table 7.1. The reference is 40% separator size. This is

| Separator size | Total prod. [MSm$^3$] | Prod. loss [MSm$^3$] | Rel. prod. loss | Marginal prod. loss |
|---|---|---|---|---|
| 40% | 34.03 | | | |
| 33% | 33.67 | 0.36 | 2.057 | 2.057 |
| 20% | 33.03 | 1.00 | 2.000 | 1.625 |
| 5% | 32.65 | 1.38 | 1.577 | 0.507 |

Table 7.1: Total gas production during ramp-up with different separator sizes

the smallest separator that can follow the nominal ramp-up trajectory without exceeding the maximum water level, i.e. no control used. However, if no liquid control is used in an actual implementation, a more conservative ramp-up trajectory would have to be used to allow for disturbances and model errors. In Table 7.1, the third column shows the production loss compared to using the nominal (fast) ramp-up trajectory. Unsurprisingly, reducing separator volume also increases production loss. Less available water storage requires the rate to be increased more slowly in order to avoid the water level rising past the maximum water level.

The fourth column in Table 7.1 shows the reduction in gas production relative to the reduction in separator size compared to the 40% size separator. For example, the value for 33% separator are calculated by: $0.36/((40-33)/40) = 2.057$. It might be a surprise to learn that reducing the separator by a fixed volume has less impact for a small separator than a large.

It is interesting to note that reducing the separator size from 20% to 5% only causes a slight decrease in the total production of gas, even if the separator size is reduced by a fourth. To illustrate this, the fifth column in Table 7.1 shows the marginal gas production loss. That is, the reduction in gas production relative to the reduction in separator size between adjacent separator sizes. For example the value for the 5% separator size are given by: $(1.38-1.00)/((20-5)/20) = 0.507$. The effect of the water drainage rate are relatively larger for a small separator compared to a large separator, thus giving a smaller marginal gas production loss.

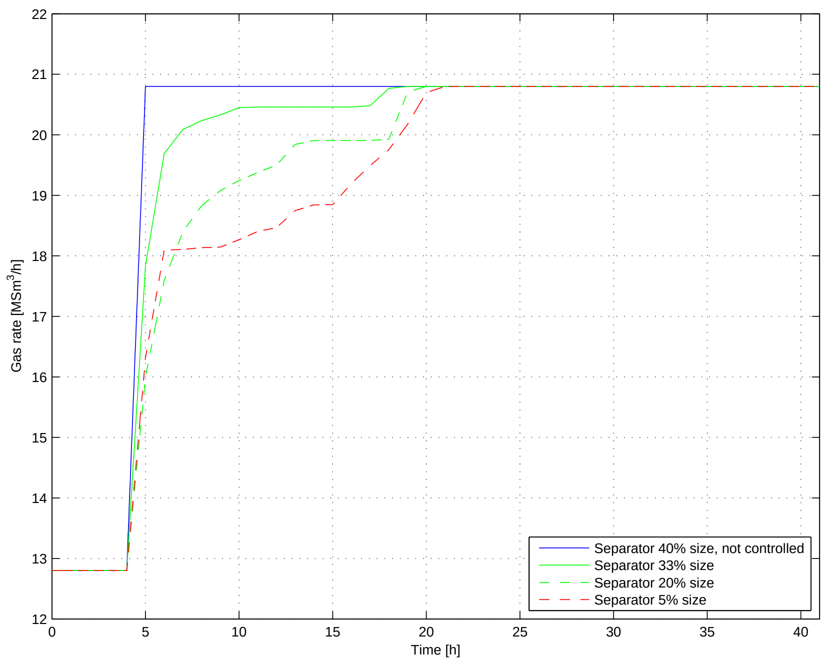

Figure 7.1: Gas rate during ramp-up of pipeline

Although the results that are economically important are given in Table 7.1, it is from a control perspective interesting to see the differences in the gas and water level trajectories resulting from the different separator sizes.

The gas rates are shown in Figure 7.1. Regardless of separator size, all gas rate trajectories show one common trait; a large increase in gas rate during the first couple of hours. There are at least two reasons for this. Not ramping up steeply at the beginning means that a huge amount of potential production is lost. Also, there must be an increase in gas velocity in order to start the discharge of liquid. After the initial increase in production two different behaviours are seen — the rates for separator sizes 33% and 20% shows a reduction in ramp-up speed until approximately 20 hours, while the rate for separator size 5% shows an increase in ramp-up speed during the same period of time. The reason for the different behaviours can be seen in Figure 7.2. The water level for the 33% and 20% separators comes up to the maximum water level only once during the simulation — at approximately 20 hours. The water level for the 5% separator, however, comes up to the maximum water level no less then three times during the simulation — between 10 and 20 hours. Thus, the gas rate for the 5% separator must be carefully controlled in this period of time as opposed to the gas rates for the 33% and 20% separators which need only to take into account the tangent of the maximum water level at 15 hours.

At first sight it might seem strange that the water levels return to normal at the same time for the different separator sizes. The reason is simply that the maximum water drainage rate and the amount of water to be discharged from the pipeline are the same independently of separator size.

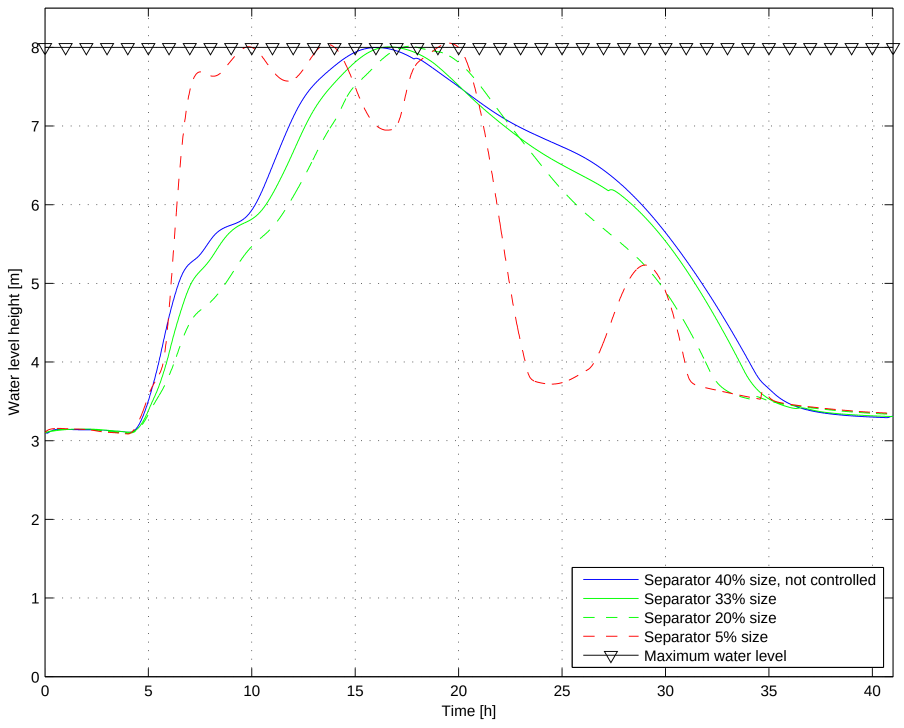

Figure 7.2: Water level during ramp-up of pipeline

## 7.3 Smart liquid control: pipeline start-up

The second case to be investigated is pipeline start-up (see Section 7.1). The control structure used was control of gas rate out of separator and basic control of the subsea choke. Table 7.2 shows the total production of gas during the 40 hours the case was simulated. (For a description of what the two right-hand columns contains and how they are calculated, see the previous section.) It is interesting that the reduction of the separator size seems to have

| Separator size | Total prod. [MSm$^3$] | Prod. loss [MSm$^3$] | Rel. prod. loss | Marginal prod. loss |
|---|---|---|---|---|
| 45% | 32.80 | | | |
| 33% | 31.91 | 0.89 | 3.338 | 3.338 |
| 20% | 29.92 | 2.88 | 5.184 | 5.052 |

Table 7.2: Total gas production during start-up with different separator sizes

a greater effect on production for a small separator than a large. This is opposite to what was found in the ramp-up scenario. An important cause of the difference is that the maximum oil drainage rate is only 140% of the oil rate at 100% steady-state production, while the maximum water drainage is 290% of the water rate at 100% steady-state production. Thus, the buffer volume is more important for oil handling than water since the drainage is relatively less for oil compared to water.

Control performance can be seen as gas rate in Figure 7.3 and oil level height in Figure 7.4. Clearly, the gas rate increase is adapted such that the oil level does not exceed the maximum level. The small overshoot of oil level is due to tuning and problem formulation. A quadratic penalty function is not exact; the limitation is violated by a small amount (soft constraint). The overshoot could have been reduced by increasing the penalty weight.

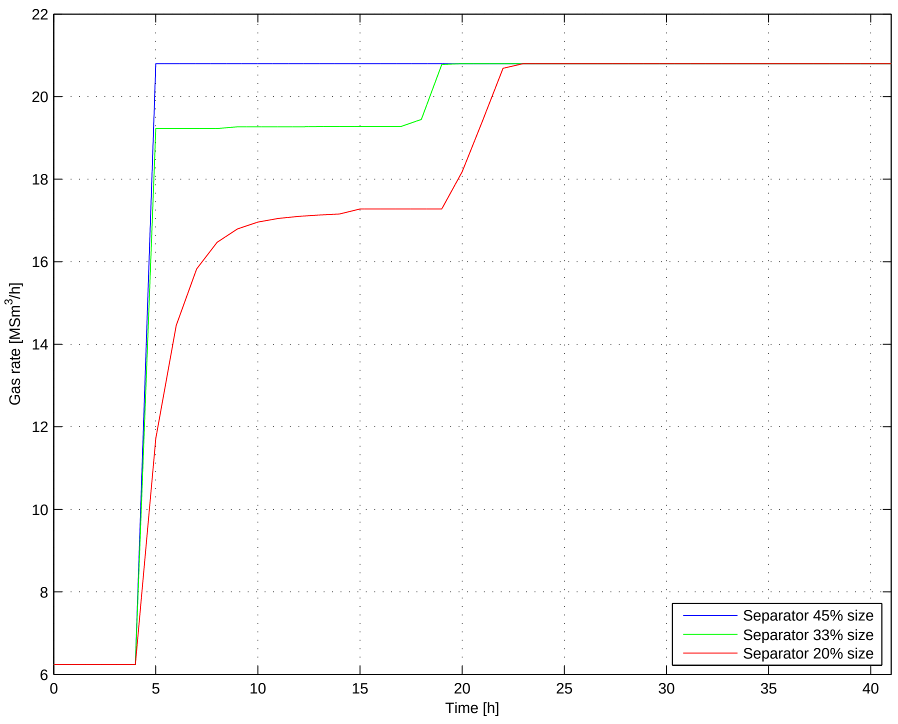

Figure 7.3: Gas rate during start-up of pipeline

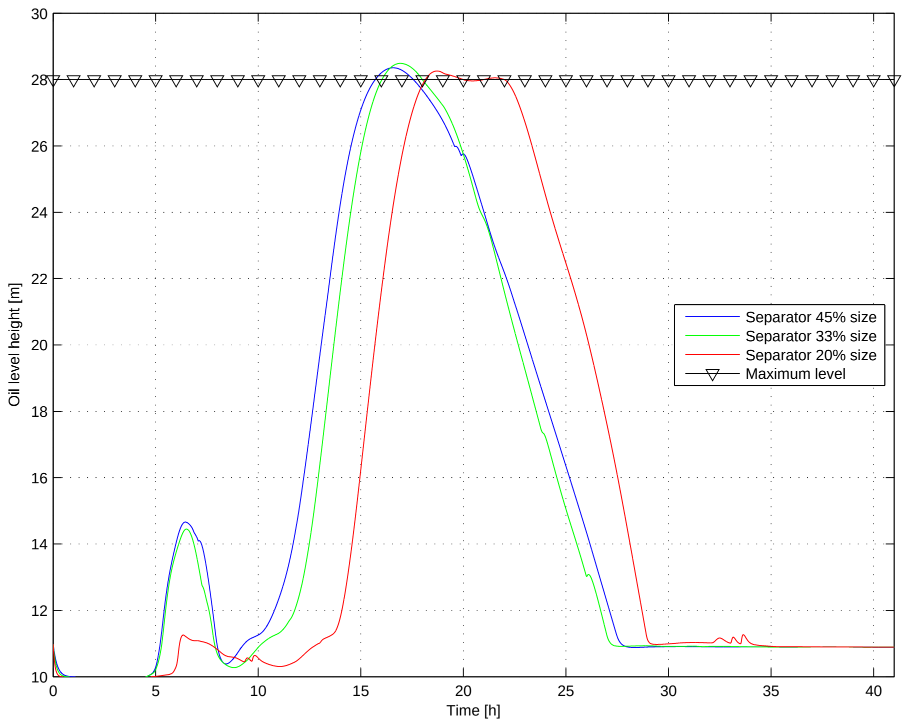

Figure 7.4: Oil level during start-up of pipeline

| | | |
|---|---|---|
| Ramp-up: | | |
| Smart liquid control | 33.03 | [MSm$^3$] |
| Manual control | 32.20 | [MSm$^3$] |
| Difference | + 0.84 | [MSm$^3$] |
| | | |
| Start-up: | | |
| Smart liquid control | 29.92 | [MSm$^3$] |
| Manual control | 27.95 | [MSm$^3$] |
| Difference | + 1.98 | [MSm$^3$] |

Table 7.3: Comparison of production using smart liquid control and manual control

## 7.4 A comparison between smart liquid control and manual control

As discussed in Section 6.1, there exists two possibilities for using smart liquid control. In this section some results showing the optimalization of liquid buffer volume for a fixed separator size will be shown. Smart liquid control was compared to a simple rate trajectory, such as may result from manual control. The cases tested was ramp-up (Figures 7.5 and 7.6) and start-up (Figures 7.7 and 7.8) of pipeline with a 20% size separator. The results are given in table form in Table 7.3. There are not much to comment on, except that the use of smart liquid control increases production significantly. The gain in profit is large, regardless of whether a price of $1\,\mathrm{NOK/Sm^3}$ or $2\,\mathrm{NOK/Sm^3}$ is used.

Figure 7.5 shows the difference in gas rates between smart and manual control during ramp-up. The manual control gas ramp-up was found by trial and error using the pipeline model to look ahead. Even if the rate change can be completed in less time using a linear gas trajectory, the total production during the ramp-up are much less than with smart liquid control. The water levels during ramp-up is shown in Figure 7.6. Both manual and smart liquid control complies with the maximum water level.

The improvement in production was even better in the start-up scenario than in the ramp-up scenario. Figure 7.7 shows the gas rates, while Figure 7.8 shows the oil level. The shape of the gas rates when using smart liquid control is distinctive. While the rates compared are almost equal at 7 hours, 19 hours and 32 hours, smart liquid control manages to utilize buffer volume to increase production in the intermediate time periods. As Figure 7.8 shows, both manual and smart liquid control respects the upper oil level limit.

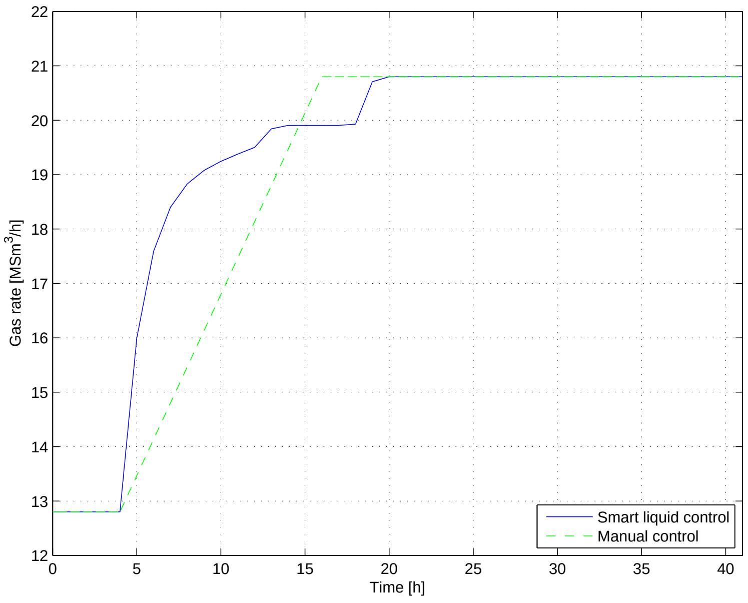

Figure 7.5: Gas rate during ramp-up

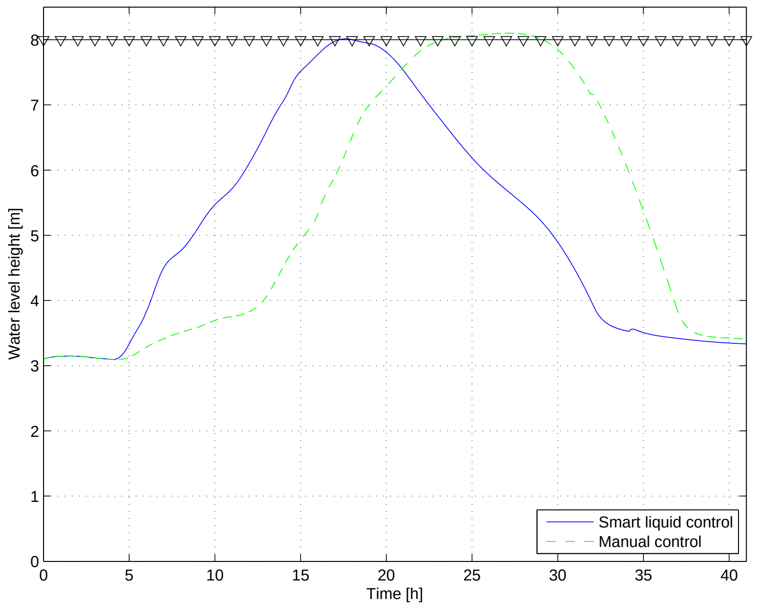

Figure 7.6: Water level during ramp-up

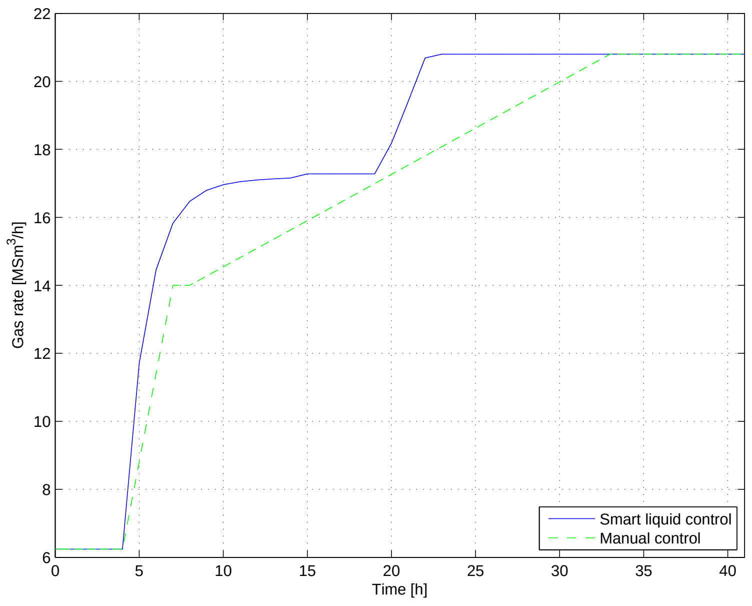

Figure 7.7: Gas rate during start-up

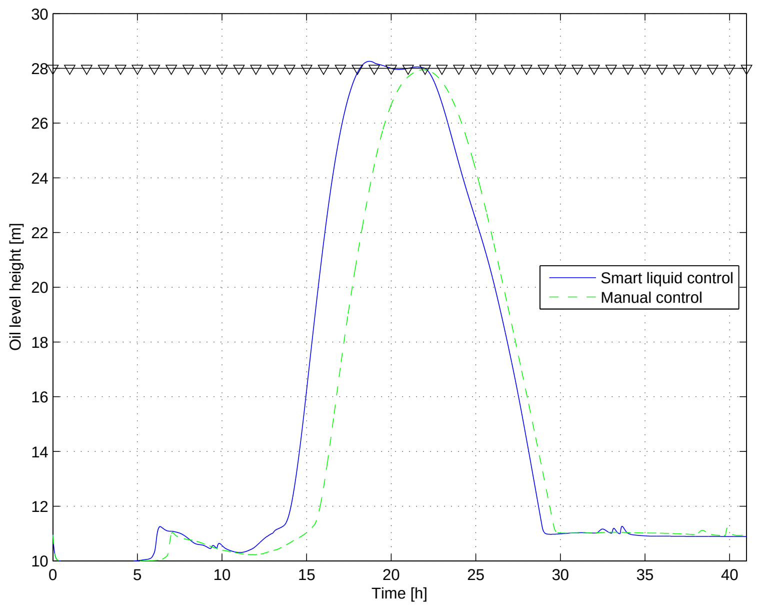

Figure 7.8: Oil level during start-up

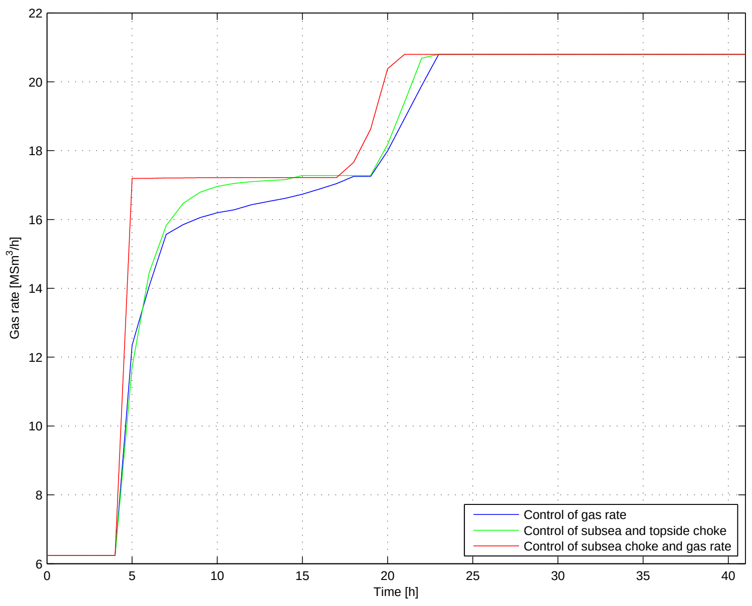

Figure 7.9: Gas rate during start-up using different controller structures

## 7.5 Use of topside choke

Three different controller setups was described in Section 6.4.2. One setup introduced a new manipulated variable, namely a topside choke. The effect of a topside choke is mainly dynamic, as in steady-state, the same amount of mass must enter and leave the separator. However, the topside choke can isolate the separator pressure from the pipeline pressure, something that cannot be done without it. In this section, the performance with the different controller setups will be compared. The case investigated is pipeline start-up with 20% size separator.

Figure 7.9 shows the gas rates for the different controller setups. As can be seen, the setup with control of the subsea choke and the gas rate out of the separator clearly outperforms the two others. This requires an explanation, as the setup using a topside choke should at least be as good. However, if one studies Figure 6.5 closely, it can be seen that the separator pressure is not controlled by SEPTIC. And indeed, Figure 7.10 shows that pressure, when using the setup with topside and subsea choke, behaves differently. Particularly interesting is the drop in pressure from 4 till 6 hours. When examining the oil rates in Figure 7.11, it can be seen that far more oil is discharged in this period with the subsea choke and gas rate controller than with the other controllers. This result implies that control of the separator pressure can change the discharge rates of liquid. A topside choke will give faster control of separator pressure, thus increasing the possibility of using the pressure to an advantage.

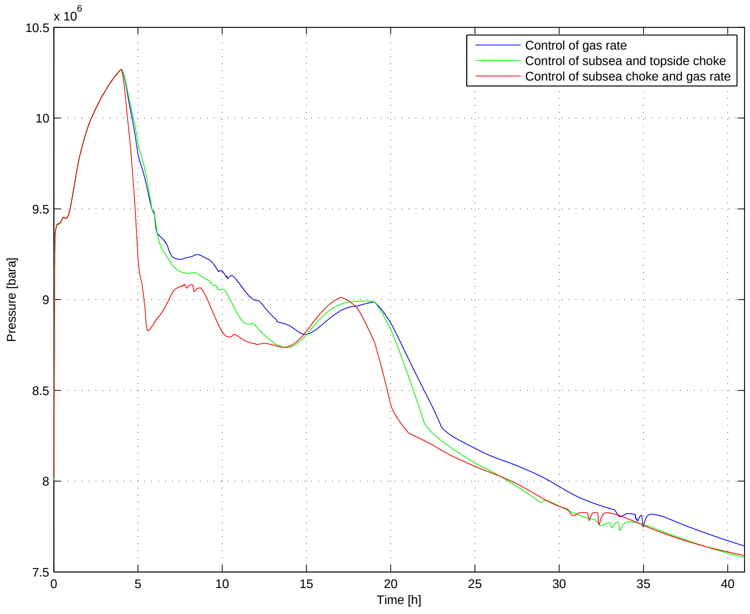

Figure 7.10: Separator pressure using different controller structures

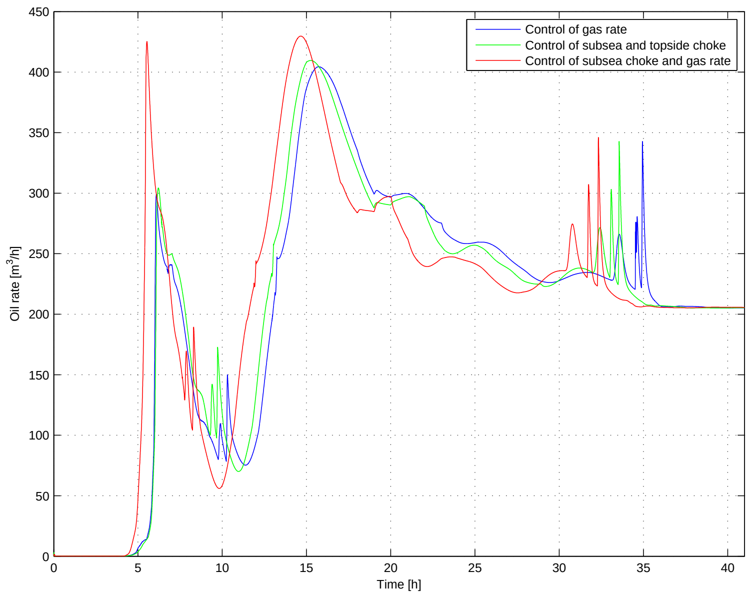

Figure 7.11: Oil rates during start-up using different controller structures

# Chapter 8

# Discussion of results

SEVERAL interesting results using smart liquid control were shown in the previous chapter. This chapter will further discuss the results, and tie any loose ends.

## 8.1 Should a topside choke be used?

While tests performed using a topside choke was insufficient to establish confidence in the benefit of using it as an extra MV, the test results in Section 7.5 nonetheless showed promise. A connection between separator pressure and liquid transportation was pointed out. Since, using a topside choke enables the separator pressure to be changed more quickly, an improved NMPC formulation may improve production during transients.

## 8.2 Controller structure

Due to computational effort, the number of MVs used should be kept to a minimum. However, for the Snøhvit case studied results indicate that at least the subsea choke as well the gas rate out of the separator should be used as inputs. Possibly, a topside choke should also be used as an extra input. A possible control structure to consider is given in Figure 8.1. The structure presents several advantages compared to the ones used in the report. Firstly, the separator pressure is controlled explicitly by setting the pressure controller set point. Secondly, the pressure control uses the topside choke to control the pressure. Hence, the adjustments made by the pressure controller will not become a disturbance for the downstream plant.

## 8.3 When to use smart liquid control?

Two different uses of smart liquid control have been outlined. The first use is in the design phase. Depending on the specific project, the optimal separator size may be smaller than

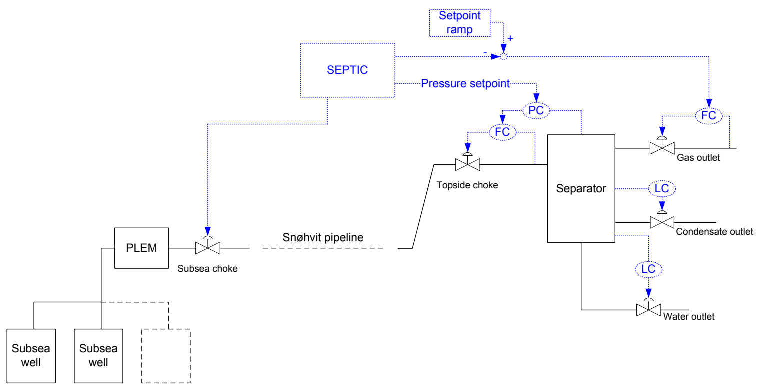

Figure 8.1: Suggestion to new controller structure to be investigated in future work

full-size separator if smart liquid control is used. The full-size separator refers to the size needed for handling liquid discharge without control. The reason for the possible change in optimal separator size is that smart liquid control reduces the production loss associated with choosing a smaller separator. Reducing the separator at Hammerfest LNG would be an example of such a use. Here, the chosen solution was to build a separator large enough to cope with surge waves without limiting ramp-up. Depending on several parameters, such as the frequency of large rate changes and the reduction in cost using a reduced size separator, an economic evaluation can be performed.

The second use is perhaps more relevant for retrofit projects. If the current separator is a bottleneck when ramping up production, losses can be reduced by using smart liquid control. Here, the gains from using smart liquid control only have to be assessed against the cost of implementing it. Another advantage when implementing smart liquid control retrofit is that it is possible, in advance, to judge the pipeline model accuracy.

# Chapter 9

# Conclusions

OVER the lifetime of a pipeline, production will be shut-down, started up, increased and decreased many times. Causes for this can be partial or full shut-downs of the processing plant due to troubles or revision stops, trouble with the reservoir or wells or changes in production due to market conditions. Large changes in pipeline production can potentially pose challenges for the liquid handling system. In this report it has been shown how using smart liquid control can improve production rates during pipeline rate changes due to better usage of available buffer volume. The term smart liquid control was coined to describe the use of NMPC to control liquid levels using a detailed pipeline model for prediction of liquid rates.

Although smart liquid control still is more conceptual than a practical solution, due to the perfect model assumption, two promising application has been identified. Firstly, as smart liquid control reduces production loss it may enable the use of smaller separators. In this scenario, smart liquid control must be considered in the design phase. Smart liquid control has shown potential to change how separator size is calculated by reducing the production loss associated with smaller buffer volume. Secondly, smart liquid control can optimize the use of separators that are already built, increasing production. This option allows smart liquid control to be applied retrofit, which might entail smaller financial risk compared to the first option. Both options was tested on two different cases, one with emphasis water handling, the other with emphasis on oil handling. Good simulation results were obtained for both cases using both options.

Several combinations of inputs were considered. Good dynamic control of the separator pressure was shown to be important for the performance achieved. With control of the subsea choke and the gas rate out of the separator smart liquid control was able to exploit the connection between separator pressure and liquid control to increase production. However, the use of a topside choke downstream the pipeline was identified as having even greater potential in this respect.

The smart liquid control was implemented in SEPTIC, a STATOIL in-house MPC tool, while OLGA was used as the pipeline model. SEPTIC functionality was enhanced with an implementation of a single-shooting algorithm (Li and Biegler, 1989; Li et al., 1990;

Oliveira and Biegler, 1994, 1995; Silva and Oliveira, 2002) and tested on a continuously stirred tank reactor example. A interface between OLGA and SEPTIC was generated using the OLGA-MATLAB toolbox as basis.

# Chapter 10

# Future work

While the results presented in this report are promising, there are several issues that need to be addressed before smart liquid control can be considered ready for implementation. The main obstacle is the subject of perfect model, or more precisely, that the prediction model will be imperfect, both in terms of structure and parameters. (Allgöwer et al., 2004) Thus, the NMPC application needs to address model updates and robustness. In addition, model validation will be important in order to obtain the best possible predictions.

## 10.1 Robustness

In chapter 2, the robustness of NMPC was discussed. It will be difficult to employ the academic approach used there on a large pipeline model. Robustness must therefore be shown through tests using different model errors such as structural errors and parameter errors. A rigorous scheme of tests should be used in order to show satisfactory robustness to model errors.

## 10.2 Feedback from measurements

Although smart liquid control can be used in open loop, feedback from measurements and model update should be used to realize as much as possible of the potential of the receding horizon formulation. Some state updates are simple, such as model update of separator pressure and liquid levels. Other, such as pipeline liquid hold-up requires more thought. If the measured pipeline liquid rates differ from the predicted rates, should the estimated total hold-up of liquid in the pipeline be changed, or should the calculated hold-up be trusted? This and similar questions needs to be answered before smart liquid control can be used in closed loop.

Although closed loop implementation of smart liquid control needs further work, it is the authors opinion that further research will be worthwhile. Especially open-loop implementation of smart liquid control for a pipeline where liquid handling are restrictive for production during transients look promising and can be tested without much further work.

## 10.3 Model validation

This report is based on the assumption that the model used for predictions are perfect, i.e. no model error. In a real application, this will not be the case. The previous section discussed future work to manage the plant-model mismatch. However, using measurement feedback cannot compensate for a poor model without sacrificing performance. Smart liquid control is based on using a model to predict the arrival of liquid. The resulting solution will be optimal only if the prediction is perfect. However, small differences from the predicted arrival of liquid can be accounted for by small input adjustments. Larger differences will require larger adjustments in input. The system is thus transferred away from the initial solution, reducing the optimality of the closed loop control. Also, in order to account for large model errors, more conservative control must be used, thus reducing performance. To avoid the problems described, emphasis should be placed on tuning the model to the pipeline responses

## 10.4 Meeting real-time constraints

The optimization algorithm described and used in this report is tailored to NMPC. Further algorithm refinements, reducing the required number of times the system needs to be simulated are possible. However, more reduction in calculation time could probably be gained from two other improvements.

The brute force approach for reducing the computation time is *parallel processing*. Given enough computers, the sensitivity matrix can be obtained by simulating the system through the prediction horizon only one time on each computer. The linesearch can be computed in parallel in a similar manner.

Running simulations in parallel can only reduce the number of simulations that need to be computed on the same computer – the simulation time over one prediction horizon will remain unchanged. In order to reduce the time needed to simulate the system over the prediction horizon once, the model must be simplified. Effort should be made in order to obtain the fastest model that still fulfils the demands made on accuracy.

## 10.5 Connection between changes in separator pressure and liquid rates

In Section 7.5, a connection between changes in separator pressure and the amount of liquid discharged from the pipeline was pointed out. This phenomenon should probably be investigated more closely in a multiphase flow setting in order to be able to use it to the full advantage. A proposal for a new control structure that might be investigated in order to exploit the effect of separator pressure on liquid rates was proposed by the author in Figure 8.1.

# Bibliography

Allgöwer, F. and Zheng, A. (2000), (Eds.). *Nonlinear model predictive control, progress in systems and control theory*, Vol. **26**. Birkhauser Verlag.

Allgöwer, F., Findeisen, R. and Nagy, Z. (2004). "Nonlinear model predictive control: From theory to application", *J. Chin. Inst. Chem. Engrs.*, Vol. **35**, no. 3, pp. 299–315.

Barclay, A., Gill, P. and Rosen, J. (1997), "SQP methods and their application to numerical optimal control". `citeseer.ist.psu.edu/barclay97sqp.html`. Last visited: 1. May 2007.

Biegler, L. T. (1998). "Advances in nonlinear programming concepts for process control", *Journal of Process Control*, Vol. **8**, no. 5,6, pp. 301–311.

Biegler, L. T. (2000). In *Nonlinear Model Predictive Control, F. Allgöwer and A. Zheng (Eds).*, Chapter Efficient Solution of Dynamic Optimization and NMPC Problems, pages 219–244. Birkhäuser. ISBN 3-7643-6297-9.

Chen, H. and Allgöwer, F. (1998a). "A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability", *Automatica*, Vol. **34**, no. 10, pp. 1205–1217.

Chen, H. and Allgöwer, F. (1998b). "A computationally attractive nonlinear predictive control scheme with guaranteed stability for stable systems", *Journal of Process Control*, Vol. **8**, no. 5-6, pp. 475–485.

Cutler, C. R. and Ramaker, B. L. (1979). "Dynamic matrix control - a computer control algorithm". In Proc. *AIChE 86th National Meeting*, Houston, TX.

Cutler, C. R. and Ramaker, B. L. (1980). "Dynamic matrix control - a computer control algorithm". In Proc. *The Joint Automatic Control Conference*, San Francisco, CA.

Diehl, M., Bock, H. G., Schlöder, J. P., Findeisen, R., Nagy, Z. and Allgöwer, F. (2002). "Real-time optimization and nonlinear model predictive control of processes governed by differntial-algebraic equations", *Journal of Process Control*, Vol. **12**, pp. 577–585.

Eagleton, B. (2001). "Snøhvit project: Hammerfest LNG project, slug catcher preliminary engineering report". *Engineering report.*

Fuchs, P. (1997). "Multiphase pipeline flow". Technical report, NTNU. Norwegian original title: Flerfase rørstrømning.

García, C. E. and Morshedi, A. M. (1986). “Quadratic programming solution of dynamic matrix control (QDMC)", *Chemical Engineering Communucations*, Vol. **46**, pp. 73–87.

Goldfarb, D. and Idnani, A. (1983). “A numerically stable dual method for solving strictly convex quadratic programs", *Mathematical Programming*, Vol. **27**, pp. 1–33.

Kalman, R. E. (1960a). “Contributions to the theory of optimal control", *Bulletin de la Societe Mathematique de Mexicana*, Vol. **5**, pp. 102–119.

Kalman, R. E. (1960b). “A new approach to linear filtering and prediction problems", *Transactions of ASME, Journal of Basic Engineering*, Vol. **87**, pp. 35–45.

Kleinman, B. L. (1970). “An easy way to stabilize a linear constant system", *IEEE Transactions on Automatic Control*, Vol. **15**(6), pp. 692.

Knutson, A. (2005). “Flow assurance operating strategy". *Internal Statoil report.*

Kwon, W. H. and Pearsons, A. E. (1977). “A modified quadratic cost problem and feedback stabilization of a linear system", *IEEE Transactions on Automatic Control*, Vol. **22**(5), pp. 838–842.

Kwon, W. H., Bruckstein, A. M. and Kaliath, T. (1983). “Stabilizing state-feedback design via the moving horizon method", *International Journal of Control*, Vol. **37**(3), pp. 631–643.

Lee, E. B. and Markus, L. (1967). *Foundations of optimal control theory*. Wiley.

Li, W. C. and Biegler, L. T. (1989). “Multistep, newton-type control strategies for constrained, nonlinear processes", *Chem. Eng. Res. Des.*, Vol. **67**, pp. 526 – 577.

Li, W. C., Biegler, L. T., Economou, C. G. and Morari, M. (1990). “A constrained pseudo-newton control strategy for nonlinear systems", *Computers and Chemical Engingeering*, Vol. **14**, no. 4/5, pp. 451 – 468.

Maciejowski, J. M. (2002). *Predictive Control with constraints*. Prentice Hall. ISBN 0-201-39823-0.

Martinsen, F., Biegler, L. T. and Foss, B. A. (2004). “A new optimization algorithm with application to nonlinear MPC", *Journal of Process Control*, Vol. **14**, no. 8, pp. 853 – 865.

Mathworks (2006), “The Mathworks, Inc. Software License Agreement - Deployment Addendum".

Mayne, D. Q., Rawlings, J. B., Rao, C. V. and Stockaert, P. O. M. (2000). “Constrained model predictive control: Stability and optimality", *Automatica*, Vol. **36**, pp. 789–814.

Meadows, E. S. and Rawlings, J. B. (1997). In *Nonlinear process control*, Chapter 5: Model predictive control. Prentice Hall PTR. ISBN 0-13-625179-X.

Meum, P. “Optimal Reservoir Control with Nonlinear MPC and ECLIPSE". Master’s thesis, Norwegian University of Science and Technology (NTNU).

Microsoft (1996), “DLLs for beginners". Microsoft Developer Support, Visual Studio 6.0.

Morari, M. and Lee, J. H. (1999). “Model predictive control: Past, present and future", *Computers and Chemical Engingeering*, Vol. **23**, pp. 667–682.

Nocedal, J. and Wright, S. J. (1999). *Numerical optimization*. Springer-Verlag New York. ISBN 0-387-98793-2.

Oliveira, N. M. C. and Biegler, L. T. (1995). “An Extention of Newton-type Algorithms for Nonlinear Process Control", *Automatica*, Vol. **31**, no. 2, pp. 281 – 286. ISSN 0005-1098.

Oliveira, N. M. C. and Biegler, L. T. (1994). “Constraint handling and stability properties of model-predictive control", *American Institute of Chemical Engineers Journal*, Vol. **40**, no. 7, pp. 1138–1155.

Pearson, R. K. (1997). In *Nonlinear process control*, Chapter 2: Nonlinear Process Identification. Prentice Hall PTR. ISBN 0-13-625179-X.

Prett, D. M. and Gillette, R. D. (1980). “Optimization and constrained multivariable control of a catalytic cracking unit". In Proc. *The Joint Automatic Control Conference*, San Francisco, CA.

Qin, S. J. and Badgwell, T. A. (2003). “A survey of industrial model predictive control technology", *Control Engineering Practice*, Vol. **11**, no. 7, pp. 733 – 764.

Rawlings, J. B. (2000). “Tutorial overview of model predictive control", *IEEE Control Systems Magazine*, Vol. **20**, no. 3, pp. 38 – 52.

Richalet, J., Rault, A., Testud, J. and Papon, J. (1976). “Algorithmic control of industrial processes". In Proc. *The 4th IFAC symposium on identification and system parameter estimation*, pages 1119–1167.

Scandpower (2005a), “OLGA 2000 Grid Generator". Scandpower Petroleum Technology.

Scandpower (2006), “User Manual: Transient Multiphase Flow Simulator, Version 5". Scandpower Petroleum Technology.

Scandpower (2005b), “User’s Manual: OLGA-Matlab toolbox ver 1.01". Scandpower Petroleum Technology.

Schittkowski, K. (2005). “QL: A Fortran code for convex quadratic programming - User’s guide, Version 2.11". Technical report, Department of Mathematics, University of Bayreuth.

Silva, D. C. M. and Oliveira, N. M. C. (2002). “Optimization and nonlinear model predictive control of batch polymerization systems", *Computers and Chemical Engineering*, Vol. **26**, pp. 649 – 658.

Skogestad, S. and Postlethwaite, I. (2005). *Multivariable Feedback Control: Analysis and Design*. John Wiley & Sons. ISBN 0470011688.

STATOIL (2005), “The long road to LNG". `http://www.statoil.com` - Snøhvit LNG project, Electronic report. Last visited: 5. June 2007.

Storkaas, E. (2005). *Stabilizing control and controllability: Control solutions to avoid slug flow in pipeline-riser systems*. PhD thesis, Norwegian University of Science and Technology.

Strand, S. and Sagli, J. R. (2003). “MPC in STATOIL - Advantages with in-house technology". In Proc. *ADCHEM 2003*, pages 97–103.

Strand, S. T. (1991). *Dynamic optimization in state-space predictive control schemes*. PhD thesis, The University of Trondheim, Norwegian Institute of Technology.

Su, H. T. and McAvoy, T. J. (1997). In *Nonlinear process control*, Chapter 7: Artificial Neural Networks for Nonlinear Process Identification and Control. Prentice Hall PTR. ISBN 0-13-625179-X.

Tenny, M. J., Wright, S. J. and Rawlings, J. B. (2004). “Nonlinear model predictive control via Feasibility-Pertubed Sequential Quadratic Programming", *Computational Optimization and Applications*, Vol. **28**, pp. 87–121.

Thomas, Y. A. (1975). “Linear quadratic optimal estimation and control with receding horizon", *Electronics Letters*, Vol. **11**(1), pp. 19–21.

Torpe, H. (2006). “Surge wave control: Can a topside choke and a controller replace a 500 million nok slug catcher?". *Project assignment report*, Department of technical cybernetics, Norwegian University of Science and Technology.

# Appendix A

# Using OLGA as model in MPC

THIS appendix will describe the implementation of the SEPTIC-OLGA interface aling with some of the general aspects of the solution. In Section 6.2.1 the need for extended interaction, requiring new capabilities, is explained.

## A.1 SEPTIC-OLGA interface

### A.1.1 Interface description

As OLGA is provided with a toolbox that can be used for online control of simulations from MATLAB (Scandpower, 2005b), this was thought to provide a good starting point for communication between OLGA and SEPTIC. The toolbox is in the form of MATLAB p-files, which means that the source code cannot be read. Hence, its functionality cannot be copied into a C++ library directly. The only way to utilize this code is therefore to call the p-files.

### A.1.2 MATLAB-generated DLL

MATLAB is equipped with a toolbox called MATLAB COMPILER. This toolbox can create C and C++ library files from MATLAB script files (m-files). Used in conjunction with MATLAB Component Runtime (MCR) this library can be used to access the functionality in m-files from a C or C++ application. The library, in the form of a DLL was generated using the command:

```
mcc -W cpplib:<name-of-interface> -T link:lib <list-of-m-files> -v
```

The argument "mcc" is used to invoke the Compiler. "-W cpplib:<name-of-interface>" specifies that a C++ interface should be built with interface and header files named "name-of-library". In addition an initialization and a termination function for including the library in applications are created. “-T link:lib” specifies that C/C++ wrapper files should be compiled to object form which is linked into a DLL. <list-of-m-files> is simply a list of files that should be included in the library. “-v” displays the compilation steps.

The compiler only accepts m-files when building libraries. The toolbox, however, are supplied as p-files. (MATLAB pseudo code files) Thus, MATLAB m-files calling the toolbox p-files had to be created.

Because the OLGA-MATLAB toolbox has been unchanged for the last couple of versions, the library generated can (at least) be used to interface OLGA versions between 4.13 and 5.1. As long as the toolbox is unchanged, no modification is needed. Also, if the toolbox is changed, it is fairly quick to generate a new library.

### A.1.3 Using the MATLAB-generated library from SEPTIC

Matlab Compiler generates <name-of-interface> files with the extensions cpp, h, lib, ctf, dll, exp and exports as well as a <name-of-interface>_mcc_component_data.c file. There are probably numerous possibilities as to how these files could be included in an application in Microsoft Visual Studio 6.0. However, the following recipe works and is the method chosen for this application.

- The file with the extension lib and the mclmcrrt.bil file residing in $MATLAB\extern\lib\win32 are added to the project. - The $MATLAB\extern\ include directory is added to the “Additional include directories” along with the directory containing the generated interface files.

After the correct files have been included, the project can be built. However, before any library functions can be called, the mclInitializeApplication and the <name-of-interface>Initialize functions needs to be called. In PipeModelModl this is accomplished by calling these functions the first time its model function is called.

Consider a MATLAB function of the following form:

```
[ ouput1, output2 ] = important_function ( input1 , input2 )
```

The function prototype for the C++ library function created from important_function will be:

```
void MW_CALL_CONV important_function ( int nargout
  , mwArray& output1
  , mwArray& output2
  , const mwArray& input1
  , const mwArray& input2 )
```

mwArray is the parameter type used for input to and output from MATLAB generated C++ libraries. For more information on this class, consult MATLAB help. The parameter nargout is the number of function output parameters.

Deployment of application requires the installation of MCR. MCR and the library created by MATLAB COMPILER can be deployed without the need for a MATLAB license, as can be seen in the MATLAB license agreement. Mathworks (2006)

The first time the application is started, some time will be spent extracting the Component Technology File (CTF) archive. It is also possible to do this without starting the application.

## A.2 Dynamic Link Libraries

Dynamic Link Libraries (DLLs) are shared libraries used for WIN applications. This discussion will only address DLLs for WIN32 applications. For additional information on DLLs, se for example Microsoft (1996). This section is based on information from this source.

Dynamic linking differs from static linking in that the application is linked to the library at run time. This is a significant difference compared to static libraries which is linked to the application at link time and incorporated in the applications executable file. This is an advantage if the code in the DLL needs to be rebuilt. Since applications are linked to the library only at run time, the application needs to be neither recompiled nor relinked.

Another difference is how the library is loaded into memory. With static libraries, the library code is incorporated in every application that uses it, thus occupying as many times the library size as there are applications using it. A dynamic library will often reside in its own memory and which are mapped into each application using it. Therefore, only one copy of the DLL will exist at run time. There is, however, possible for the DLL to contain separate sets of data for each of the applications that uses the DLL.

# Appendix B

# Additional simulation results

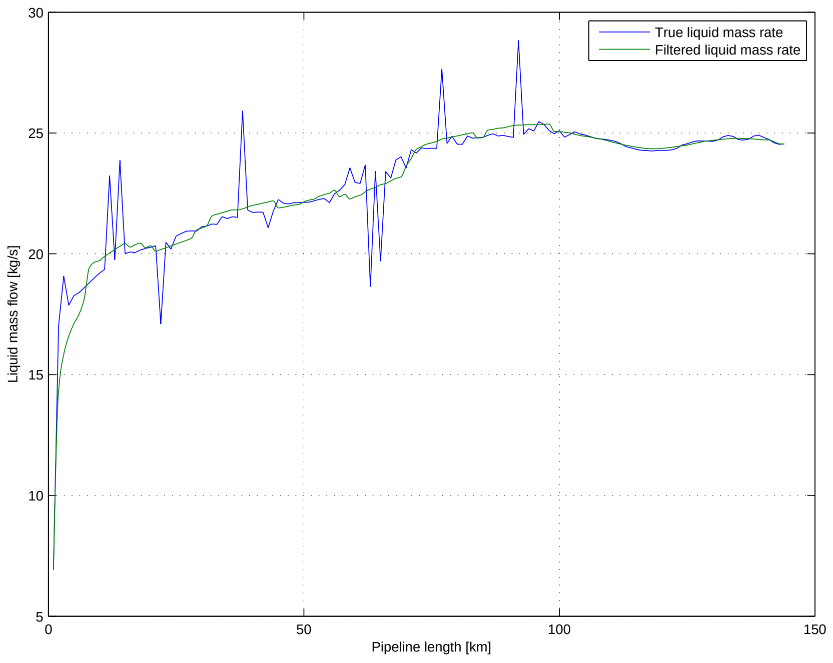

Figure B.1: Comparison of true liquid mass rate and filtered liquid mass rate during step in production rate from 12.8 MSm$^3$/d to 20.8 MSm$^3$/d

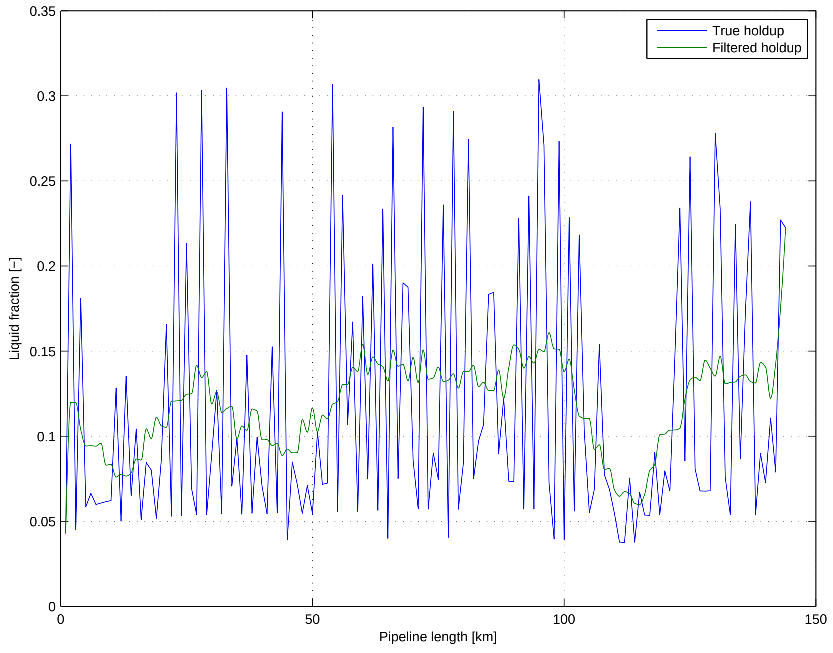

Figure B.2: Comparison of true liquid holdup and filtered liquid holdup during step in production rate from 12.8 MSm$^3$/d to 20.8 MSm$^3$/d

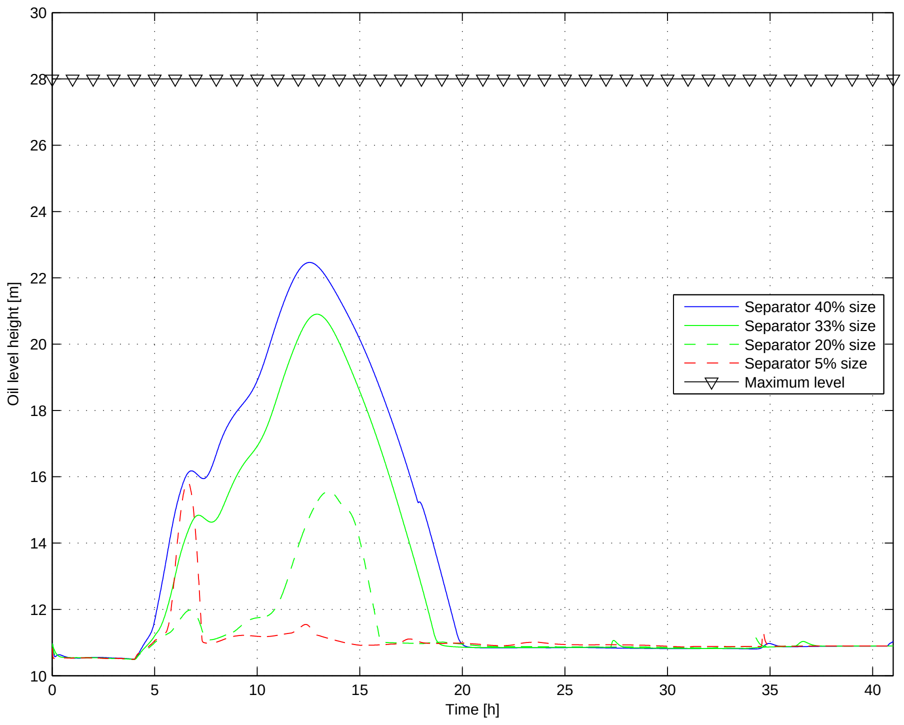

Figure B.3: Oil level during ramp-up of pipeline

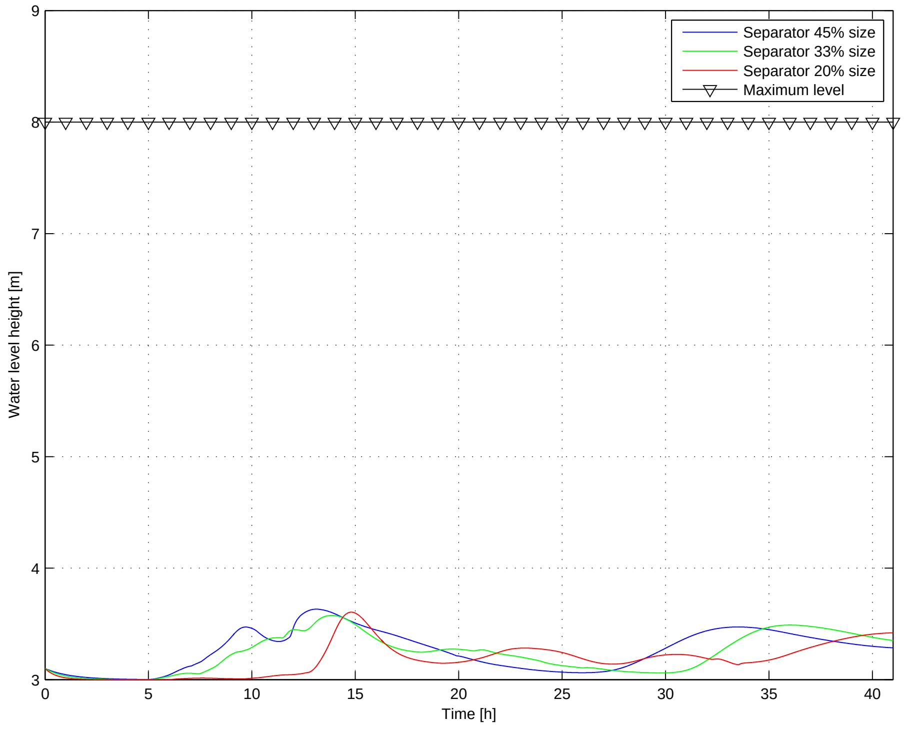

Figure B.4: Water level during start-up of pipeline