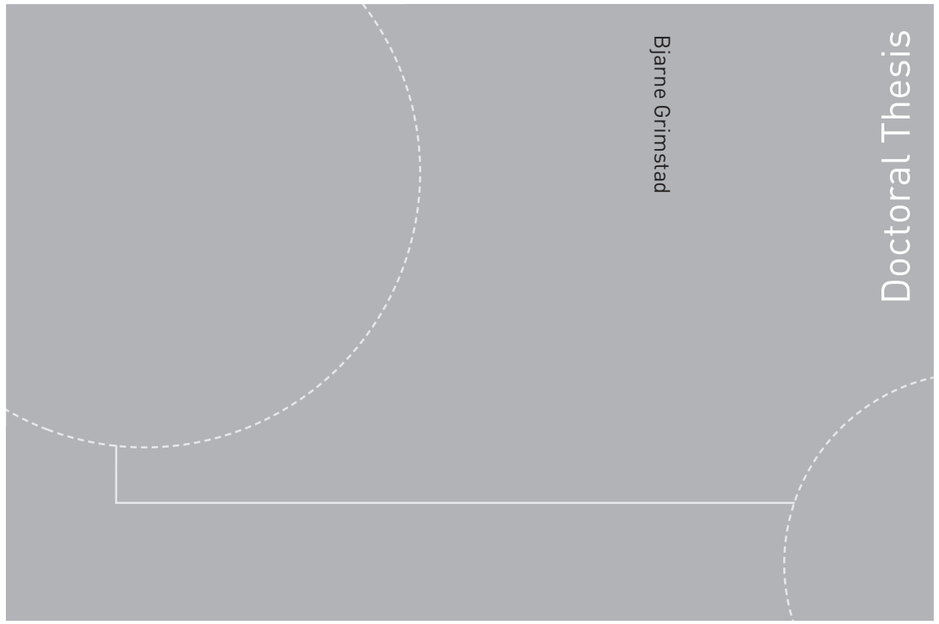
ISBN 978-82-326-1206-2 (printed version)
ISBN 978-82-326-1207-9 (electronic version)
ISSN 1503-8181

NTNU
Norwegian University of
Science and Technology



Doctoral theses at NTNU, 2015:275

Bjarne Grimstad

Daily Production Optimization for Subsea Production Systems

Methods based on mathematical programming and surrogate modelling

Doctoral theses at NTNU, 2015:275

NTNU

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Bjarne Grimstad

Daily Production Optimization for Subsea Production Systems

Methods based on mathematical programming and surrogate modelling

Thesis for the degree of Philosophiae Doctor

Trondheim, October 2015

Norwegian University of Science and Technology
Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Engineering Cybernetics

ISBN 978-82-326-1206-2 (printed version)
ISBN 978-82-326-1207-9 (electronic version)
ISSN 1503-8181
ITK Report 2015-6-W

Doctoral theses at NTNU, 2015:275

Printed by Skipnes Kommunikasjon as

To my parents

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of Philosophiae Doctor (PhD) at the Norwegian University of Science and Technology (NTNU). The research has been conducted at the Department of Engineering Cybernetics (ITK) from September 2011 to 2015. Funding for the research has been provided by the Center for Integrated Operations in the Petroleum Industry (IO center) and partly by the department, for which I am grateful. The financial support from an IBM PhD Fellowship Award is also highly appreciated.

First and foremost, I would like to express my sincere gratitude to my supervisor Professor Bjarne Foss, who gave me an opportunity to pursue a PhD on a topic of my interest. Throughout the PhD work he has facilitated a creative space for me to work in, filled with impulses from the academia and industry. Without his involvement this work would never have been written.

I would like to thank my co-supervisors Dr. Vidar Gunnerud and Dr. Dag Ljungquist. The innovative ideas and entrepreneurial spirit of Vidar Gunnerud has been an inspiration to me.

An important facet of this work has been the attempt to generate value from research results together with the industry partners in the IO center. These efforts would not have been as fruitful without the support of Dr. Malcolm Woodman and Mr. Richard Heddle at BP Exploration Operating Company Limited, in Sunbury, UK. Our numerous meetings and workshops have formed this work. I would also like to thank BP for hosting my stay in Sunbury during January-September in 2013.

I would like to thank the clever Master's students whom I have been so lucky as to supervise: Håvard Ausen, Shaurya Sharma, Anders Sandnes, Patrick M. Robertson, and Håkon Skibeli. Our collaborations have been rewarding and educational to me.

I would like to thank my friends and colleagues at the department: Brage Knudsen, Anne-Mai Ersdal, Anders Willersud, Joakim Haugen, Tor Aksel Heirung, Andres Codas, Kristian G. Hanssen, Morten Pedersen, Aleksander Veksler, Anders Wenhaug, and others. Our discussions and collaborations have been interesting.

Finally, I want to express gratitude for the unconditional support of my parents Wenche, Gary, Eva, and Arne. I am very lucky to have you.

Tuva: nå er jeg ferdig med tingene mine!

Bjarne Grimstad
Trondheim, May 2015.

Contents

List of Figures

List of Tables

Abbreviations

APC	Advanced Process Control
CPM/CPMS	Condition and Performance Monitoring System
EOR	Enhanced Oil Recovery
FA/FAS	Flow Assurance System
FEED	Front-End Engineering Design
FP	Feasibility Pump
FPSO	Floating Production Storage and Offloading
IP	Interior Point
LP	Linear Programming
MIP	Mixed-Integer Programming
MILP	Mixed-Integer Linear Programming
MINLP	Mixed-Integer Non-Linear Programming
MPC	Model Predictive Control
NCS	Norwegian Continental Shelf
NLP	Non-Linear Programming
OFP	Objective Feasibility Pump
PID	Proportional-Integral-Derivative
PO/POS	Production Optimization System
RTO	Real-Time Optimization
SCADA	Supervisory Control And Data Acquisition
SQP	Sequential Quadratic Programming
VFM/VFMS	Virtual Flow Metering System

Chapter 1

Introduction

This introductory chapter begins by giving the reader a notion of what subsea production systems *are*, why we have them, and how we operate them. Following this introduction, is a discussion on the information technologies that support the daily decision processes facing an operator of a subsea production system. This discussion is focused on technology for real-time optimization, which is the topic of this thesis. Next, a research objective is given, together with a motivation for spending resources towards reaching this objective. At the end of this chapter, the layout of the remainder of this thesis is given.

1.1. Subsea production systems

Norway has been in the forefront of the technological development of subsea production systems, together with other leading subsea nations like the USA and Brazil. In the following section, the history of subsea technology on the Norwegian continental shelf is briefly summarized. Then, the common components and structure of a modern subsea production system is presented. Following this is a short description of how a subsea production system is controlled. This introduction does not attempt to give a complete picture of subsea production systems, but will provide the abstractions needed to understand the rest of this thesis.

1.1.1. History of subsea technology on the Norwegian continental shelf

In June 1971, the Norwegian oil adventure began as the first barrels of oil were produced from the Ekofisk field. The following year, the Norwegian oil company Statoil was created to ensure Norway's participation in the production licenses on the Norwegian continental shelf (NCS). During the 1960-70s, several major oil fields were discovered on the NCS and investments to develop new fields accelerated quickly. Statoil became the first Norwegian oil company to be given operator responsibility in 1981 for the Gullfaks field in the North Sea (Statoil, 2015). During the 1980s and 1990s many of the major and easily accessible oil fields were developed, including the Oseberg, Statfjord, and the Troll natural gas and oil field. These giants contributed significantly to positioning Norway as Europe's largest producer and exporter of oil with a peak daily production of more than 3.4 million barrels in

2001 (Nedregaard, 2003; Norwegian Petroleum Directorate, 2015).

At shallow water depths up to 300 meters, the first major oil fields on the NCS were developed using concrete production platforms standing on the sea bed. These large and rigid structures did not, however, allow for an economically sensible development of smaller fields, satellite wells, and fields at deeper waters. Thus to increase the production on the NCS new technologies were needed that could address the challenges related to deeper waters. This came with the development of subsea technology – structures that could be placed and operated on the seabed. The idea dated back to the 1960s, and the first subsea well on the NCS was actually tested in the 1970s. Statoil's first subsea well started producing from Gullfaks in 1986 (Statoil, 2015). One year later, Tommeliten, the first field to use a subsea template structure, started to produce through a tie-in to Ekofisk (Norsk Oljemuseum, 2015; Solheim et al., 1989). The subsea technology was attractive because it allowed for flexibility in the layout of the production system, with satellite wells, tie-ins to existing fields, and production to floating production, storage and offloading vessels (FPSOs). It could also alleviate operational constraints, such as water and gas handling capacities on platforms with many wells.

The second generation of subsea technology, dated to the early 1990s, focused on cost-effective solutions to allow expansions of current fields – in particular Statfjord – by drilling wells previously out of reach for platform drilling. The next, third generation, dated to the late 1990s, moved subsea production systems to deeper waters and was an enabler for using dynamically positioned FPSOs. One development using third generation subsea technology was that of the Norne oil field, located at a water depth of 380 meters. Production from Norne to the Norne FPSO began in 1997.

During the 2000s, the fourth generation of subsea technology addressed difficult development projects with high pressure and high temperature gas reservoirs and long tie-ins to shore. Two especially challenging developments were those of the gas fields Ormen Lange and Snøhvit; the former residing on water depths ranging from 850 to 1,100 meters (Bernt, 2004), and the latter with a 160 km tie-back to processing facilities on shore (Witting, 2006). This generation also introduced the first subsea processing systems. The Tordis subsea separation, boosting, and injection system was developed by Statoil to pursue the ambition of a subsea factory (Statoil, 2015). The increasingly difficult operating conditions and remote wells accelerated what is often referred to as the *digitization* of the offshore petroleum industry. Digitization meant better control and sensing in so-called *smart wells*, fiber optic cables, and remote control technologies, to mention a few innovations. It lay the foundation for increased operational awareness – despite a higher system complexity.

Since 2007-2008, the fifth and current generation of subsea technology has focused on improving cost-efficiency through standardization and modularization. This has made fast-track subsea tie-back projects possible. Subsea processing has also become increasingly common with projects like the Tyrihans development, where raw seawater is injected into the reservoir by using pumps on the seabed (Grynning

et al., 2009). The current generation has also started to embrace the opportunities catered for by the digitization, with an extensive use of remote operations and software monitoring systems.

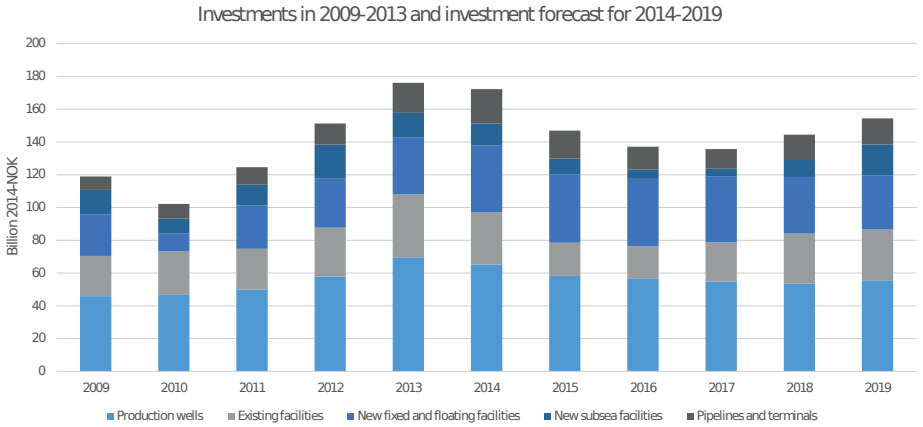


Figure 1.1.: Investements in oil and gas on the Norwegian continental shelf. Numbers from the Norwegian Petroleum Directorate (2015).

The future subsea technologies will certainly make use of instrumentation and software to allow operation of more advanced subsea factories. The subsea factory may enable developments at deeper waters and longer step-out distances. The next generation is expected to prepare the petroleum industry for the harsh environment of the arctic – the arctic challenge (Allen, 2011). Before this happens, however, the oil industry has to recover from the current economic slump caused by high supply and low oil price. Today, 14 years after Norway had its peak production in 2001, investments in new developments on the NCS are predicted to decline (DNB Markets, 2015; Norwegian Petroleum Directorate, 2015). As Figure 1.1 shows, increased investments in subsea facilities are predicted for the years that follow the current slump.

1.1.2. Subsea production system architecture

The purpose of a subsea production system is to produce hydrocarbons *from* and/or inject water or gas *to* a reservoir in a controllable, cost-efficient and safe manner. A subsea production system consists of several components which are configured and combined to serve this purpose. The system architecture must be designed to strict regulations and specifications, with future production and extensions in mind. On the NCS, compliance to regulations is commonly ensured by following the ISO and NORSOK standards (Standards Norway, 2015). Next, the most common components of a modern subsea production system are introduced.

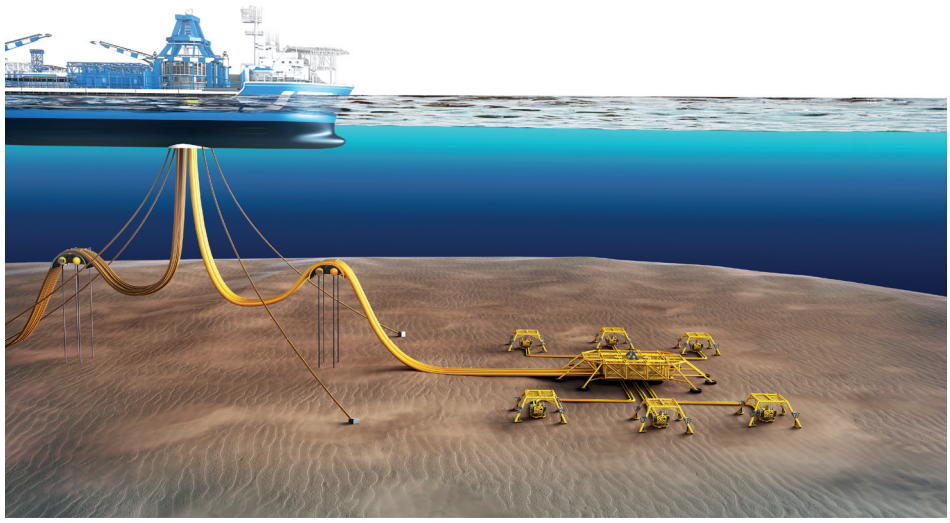


Figure 1.2.: Illustration of a subsea production system with six wells producing to a floating production storage and offloading vessel.

Wells. A well may have a single or multiple branches drilled vertically or directionally to reach areas of the reservoir most likely to yield high hydrocarbon production (the productive zone). After a well is drilled, it is completed with production tubing to isolate the produced fluids as they travel from the reservoir to the topside facilities. Several valves are installed in the well to control the flow from the reservoir. The valves ensure a safe operation during production and makes it possible to close the well for maintenance operations. In particular, a surface controlled subsurface safety valve (SCSSV) is installed down in the well, with a safety mechanism that automatically shuts in the well in case of accidents. The *Christmas tree*, located in the wellhead, houses multiple valves as illustrated in Figure 1.3. The well has at least one master valve for shutting in the well in case of emergencies or if the well is to be permanently shut. The Wing valve is a fail-safe valve for temporarily shutting in the well, for example during maintenance. The oppositely positioned Kill Wing valve – also known as the Non-Active Side Arm valve – is used for injection of fluids such as corrosion inhibitors and methanol. The top valve in the Christmas tree is called the Swab valve and is opened to allow well interventions like wireline operations. The *choke* or choke valve is located downstream the Christmas tree. It is a robust and adjustable valve used to control the flow from the well.

All the above-mentioned valves are controlled mechanically by a remotely operated vehicle or hydraulically by the control module. The control module, also residing in the wellhead, is a computer that collects the sensor readings from all the sensors in the well and communicates them to the control system topside. It

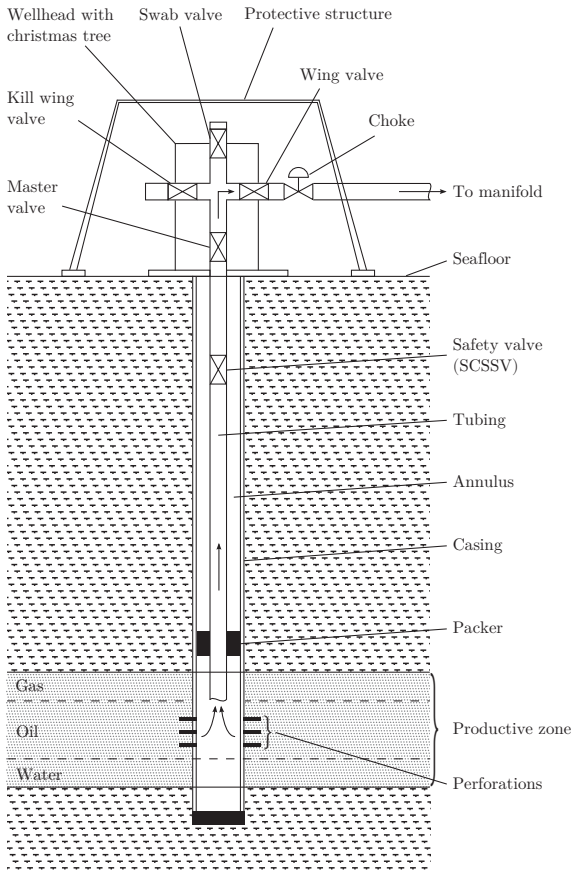


Figure 1.3.: Illustration of a subsea well and Christmas tree. Inspired by Nedregaard (2003).

may also relay control commands from the topside control system to the valves. Installed sensors typically include pressure and temperature sensors located before and after the choke, in the annulus of the well, and sometimes down-hole the well. In multi-branched wells, control valves may also be installed downhole to control the flow from each branch.

In addition to downhole sensors, some modern "smart wells" have installed artificial lift – such as an electrical submersible pump (ESP) or lift gas injection – that may increase production.

Manifolds. Often, it is economically sensible to commingle multiple well flows in fewer pipelines leading the fluid to the topside processing facilities – this is especially the case for subsea systems in deep waters or with long tie-ins. This commingling and routing of flow is done in the manifold, which essentially is a collection of pipes and on/off valves serving this purpose. Sometimes the manifold resides in a structure called a subsea template, which also may house several wellheads. Subsea

templates can have as much as 6 or 8 well slots. Often, some of the slots are left empty to accommodate future wells. Wells that do not sit in a template are referred to as satellite wells.

Pipelines and Risers. The fluid is transported in pipelines/flowlines between fields (tie-ins) or to shore (tie-backs). Pipelines leading the fluid topside are referred to as risers. The transported fluid is subject to hydrostatic and frictional pressure loss, as well as temperature loss. Some pipelines induce an unstable flow regime called slug flow, in which the gas phase exists as large bubbles separated by liquids. Risers, especially, are prone to severe/terrain slugging which may damage the production system if not handled properly. Hydrate forming is another issue related to hydrocarbon transportation in long pipelines; it can be prevented by injecting monoethylene glycol (MEG) in the production stream. Other issues related to flow assurance include the forming of solids such as wax and scale, and erosion due to sand production.

Subsea processing equipment. The catalogue of subsea processing technology has grown over the last years. Today, a wide range of passive and active subsea equipment is available for cost-efficient development of new fields.

Separation is an important function in the subsea factory. Subsea separation may be performed using the conventional gravitational principle, where a tank with sufficient volume is used for separation. It may also be performed by smaller, inline separators that establish a hydrocyclone in line with the flow, temporarily separating the fluid phases. Inline separation is based on exploiting the centripetal force and different fluid resistances of liquids and gas to separate the phases. Inline separators require less space, but generally offers a lower separation degree than gravitational tanks (Orlowski et al., 2012). For this reason, multiple inline separators are sometimes used in addition to a gravitational tank to obtain a high separation degree.

Active components such as subsea pumps and compressors (Lima et al., 2011) are also available in the subsea equipment catalogue; subsea pumps were for example installed in the Tordis IOR project (Gruehagen and Lim, 2009). Subsea pumping and gas compression enables production from low-pressure reservoirs, transportation of fluids over long pipelines, and injection of fluids to reservoirs. In so-called boosting stations, separation is combined with pumps and/or compressors to enable production from fields in deeper waters and more remote areas. By separating the phases, single-phase pumps with a high hydraulic efficiency may be utilized (Håheim and Gaillard, 2009; Gruehagen and Lim, 2009). Single-phase pumps and compressors require sufficient de-gassing and de-liquidizing, respectively, to operate efficiently. To maintain a sufficient separation degree is one of the biggest challenges with boosting stations.

The subsea factory introduces many challenges that have to be addressed by auxiliary systems. For example, sand production may degrade separator performance and quickly wear out pumps. This issue is addressed by installing a sand handling system (Vu et al., 2009). Another example is slugging, which may greatly affect the control of a subsea system. Slugging may be dampened and the flow stabilized

by installing a buffer tank at the inlet of a subsea factory.

Another, less used technology is subsea storage (Eie, 2015). The technology may become important as the arctic areas are developed.

Topside facilities. Located topside on a platform, FPSO, or on shore, are the facilities for treating the produced fluids. The topside facilities have equipment for separation, gas scrubbing, water filtration and cleaning, and many auxiliary functions such as safety systems, pneumatic and hydraulic system for operating valves. After treatment, the oil and gas is exported, while water is released or injected to the reservoir.

The subsea system is operated mainly by adjusting mechanically, hydraulically, or electrically operated valves. Many, but not all of the valves are located in the wells. Operations that are performed by adjusting valves include: shutting in wells (wing valve), adjusting the choke, injecting lift gas (kill wing valve), routing well flows (manifold valves), and injecting MEG (dedicated valve).

To measure key operational variables, sensors are placed throughout the production system. Subsea sensors that measure pressure and temperature are commonly installed. More advanced sensors for rate measurement, sand detection, etc., are less frequently installed due to high cost. Real-time data from the sensors is usually sent to a subsea control module, which forwards the information to the topside control system. By fetching real-time and historical data from the control system, the operator may monitor the production system and plan operation accordingly.

Next, we discuss how people and computer systems make use of the real-time data to control a subsea system.

1.1.3. Operating a subsea production system

Figure 1.4 illustrates the control loop for daily production optimization. Real-time data is collected and stored by the supervisory control and data acquisition (SCADA) system. The data is made available to operators and productions engineers who monitor the system. Control actions are given by the operators to the SCADA system, usually after having consulted with the production engineers. The control actions are often valve opening set-points that the control system delegates to the subsea control module responsible for the valve to be adjusted.

The production engineers make use of real-time optimization (RTO) technology, which may advice them on how to control the asset. Notice the "feedback" loop back to the RTO system, which symbolises that *engineering knowledge* is guiding the optimization advice. This is an effort towards increasing the probability of a good advice/prediction, and ultimately to providing better services to the operators of the field. Engineering knowledge may in this context be operational conditions which are not captured by the real-time data or RTO system, such as planned maintenance events that require well intervention. This information may invalidate advice from the RTO; hence, the production engineers must guide or prevent the RTO from making such advice, for instance by altering or adding operational constraints to the optimization problem being solved. Furthermore, at certain times

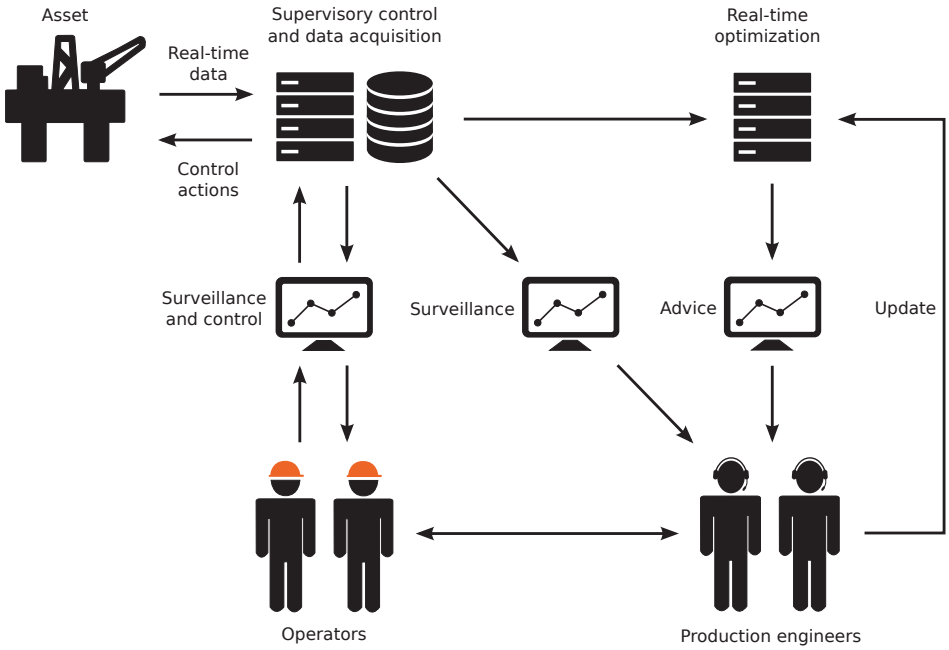


Figure 1.4.: Illustration of the control loop for daily production optimization of a subsea production system.

the production engineers must calibrate the models used by the RTO. Model calibration may be required when the operating point has moved significantly spatially (e.g. if valve settings have changed significantly) or temporally (e.g. if reservoir conditions have changed significantly with time).

JPT (2008) provides the following list of work tasks related to daily production management, typically performed by the production engineers on an asset.

- Monitor well performance
- Manage problem wells (control injection, stimulation, etc.)
- Execute and validate well tests
- Monitor and manage surface facility performance
- Manage daily operations
- Manage real-time lift, pressure, temperature, and flow control
- Update daily/monthly plan and forecast and take actions

These tasks are supported by a host of different information technologies for visualizing and analysing real-time data, predicting system behaviour during planning,

supporting communication and information sharing between onshore and offshore teams, alerting operators of safety critical events and trends, and so on. Some of the tools involved in real-time monitoring and optimization are discussed next.

1.2. Simulation and optimization technologies

The recent technological innovations in subsea sensing, communication (e.g. fiber optics), and electrification, often collectively referred to as the *digital oil field*, have made RTO solutions more viable than ever before.[1] Successful applications of simulation and optimization technologies have been documented by all the large E&P companies in numerous publications. The trend towards a higher quantity and quality of real-time measurements in digital oil fields may only have a positive effect on technologies reliant on extensive sensing, like RTO. Furthermore, the retrofitting and deployment of new technology on older fields opens for a wider use of decision support tools in the future.

The use of decision support tools such as RTO is motivated by field cases that document a positive correlation between the sophistication of the decision analysis in an oil company and its financial performance. Furthermore, any underestimation of the impact on reservoir uncertainties on future performance indicators can result in sub-optimal decision making and financial under-performance (Nicotra et al., 2005).

Below, some of the technologies related to control, simulation, and optimization of offshore production systems are grouped.

Asset management. These are technologies that facilitate analyses of strategic and economic long-term goals during development of the asset. Asset management is often performed using a coarse reservoir model to analyse: well and equipment investments (Tarhan et al., 2009), well placement (Güyagüler and Horne, 2004; Ozdogan and Horne, 2006; Bellout et al., 2012), field development and export (Akeze et al., 2009), and other life-of-field investments.

Reservoir management. These are technologies that aim to control operations to obtain the maximum possible economic recovery from a reservoir (Thakur, 1996). Reservoir management considers drainage, injection (e.g. water-flooding), and other EOR strategies to increase the total production over a horizon of several years. It is usually performed using several realisations of a high-fidelity reservoir model. Consequently, reservoir management is a time-consuming and computationally intensive technology often performed in batches. However, since it considers a horizon of months to years it can be run in real-time; it is then referred to as closed-loop reservoir management (Jansen et al., 2009).

Real-time Optimization (RTO). This is a broad class of technologies for estimation and prediction of states to improve upon economic goals such as increased production or reduced down-time. RTO is based on optimizing some objective

[1] For some reason, the E&P companies use different names for the digital oil field, including Intelligent Field, Smart Field, e-Field, i-Field, and Field of the Future.

subject to a representative model with predictive capabilities. The model may include the whole or parts of the production system, but must in any case represent the objective with sufficient accuracy. The most widely employed RTO technologies are virtual flow metering (VFM) systems, flow assurance systems (FAS), and more recently condition and performance monitoring (CPM) systems. These are real-time monitoring systems that have in common that they attempt to increase the operational awareness of the operator. RTO can also be used more actively to suggest operational moves that may increase production and/or lower operational cost. RTO with this capability is often referred to as production optimization (PO) systems.

Advanced Process Control (APC). Advanced control is used to maximize production, minimize energy consumption and other operational costs, and to minimize process variability. APC is usually implemented in the control system, in addition to regulatory control. APC technologies include model predictive control (MPC) and advanced regulatory control such as adaptive control. APC may require state and parameter estimation using, for example, Kalman filtering or moving-horizon estimation.

Regulatory control. Set-point (PID) control for simple control loops, usually implemented in a programmable logic controller (PLC). Commonly controlled variables are valve positions, pressures, liquid levels, pump and compressor speeds. The purpose of regulatory control is to ensure a safe and robust operation by automatizing the control of safety-critical equipment.

The scopes of the above technologies are illustrated in Figure 1.5 – the figure reflects the scopes of commercially available products. Note that there are some exceptions to the figure; for example, anti-slug control operates on wells with a sampling time of seconds (Aamo et al., 2005). Below, we attempt to outline the interrelations and a few of the major differences between the technologies.

Both APC and regulatory control are implemented as closed control loops. As already mentioned in Section 1.1.3, RTO (and reservoir and asset management) is performed with humans in the loop. RTO technologies are often classified as *model-based decision support tools* since they act as advisory systems in the decision processes of the operator. This makes RTO fundamentally different from APC, where the loop is closed. In the downstream industry, RTO is usually performed in a layer "on top" of APC. This may in many cases justify simplifications in the RTO model due to the linearizing effect of the lower level APC and regulatory control (Skogestad and Postlethwaite, 2005). Unfortunately, APC is infrequently utilized in offshore systems due to difficult operating conditions, namely: highly transient behaviour during shut-ins and start-ups which is difficult to model, many disruptive operations such as well interventions, significant disturbances (slugs), high degree of uncertainty (lack of measurements and low signal-to-noise ratios), and few or none advanced control specialists available offshore to ensure that the APC is working properly (Campos et al., 2013). The lack of APC in upstream production systems may have increased the scope, responsibilities, and expectations of current RTO technologies. Of course, the same set of difficulties apply to RTO, but perhaps to

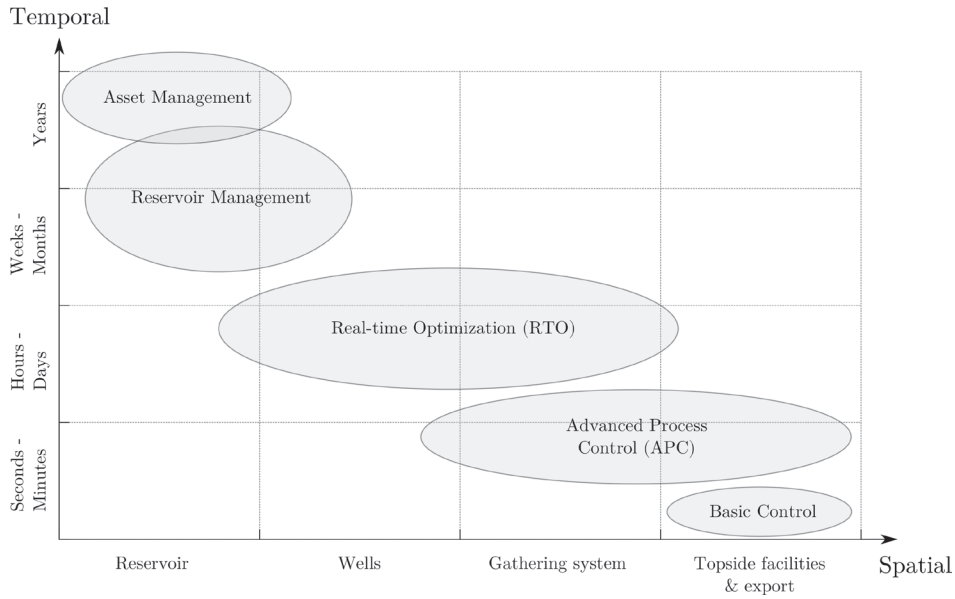


Figure 1.5.: Illustration of the scopes of different control and optimization technologies related to production. Note that the size of the ellipses do not reflect the extent to which the technology is used since it is highly variable between fields/systems.

a lesser extent due to an offloading of responsibility to the humans in the loop – the operators.

Except for regulatory control, the above technologies make extensive use of advanced models. In particular, asset and reservoir management make use of reservoir models; RTO make use of highly nonlinear multiphase flow models; and APC depends on dynamic models describing the topside process facilities. Next, we expand on some of the RTO technologies that are available today and their use of models.

Simulation – with data-driven and/or first-principle models – is combined with real-time measurements in VFM systems that estimate unmeasured states in the production system. VFM has been successfully and widely applied by the upstream industry in recent years – the application on the greater Ekofisk area is one of the success stories (Denney, 2012). Today, most offshore production systems have some form of VFM system installed. The high costs of subsea wells and the facilities that support them necessitates a high level of well-performance monitoring to protect the investments (Oberwinkler et al., 2006). In a primitive VFM system, each well is modelled by curve-fitting polynomials to well tests.[2] More advanced VFM systems model the whole production system with a combination of empirical and first-principle models; this makes it possible to include measurements from different

[2]Linear and quadratic inflow equations, sometimes combined with lift-curves, are commonly used to represent the relation between pressure and flow rate in wells.

parts of the production system in a large data reconciliation problem. CPM systems are similar to VFM systems, but include models for predicting equipment wear and tear. The positive impact of using CPM in daily operations on the Gjøa field is discussed by Roald et al. (2013). FAS is another specialization of VFM systems that focuses on predicting fluid-related flow issues such as wax and hydrate formation. To obtain trust-worthy predictions of flow assurance issues, accurate fluid and temperature models are required. A FAS has been used with success on the Ormen Lange subsea production and pipeline system to prevent hydrate formation and detect possible leakage and blockage (Holmås et al., 2013).

In addition to being used for estimation and assurance of flow during production, an important avenue for simulation is case studies: for example in *FEED* studies to unveil problems with subsea infrastructure designs, or in *what-if* studies to plan future operation by simulating different operational scenarios. Next, we discuss its use in what-if studies, which will lead us to technologies for production optimization.

Since the number of scenarios may be large, even infinite when continuous parameters are considered, a what-if study is limited to explore only a small subset of possible scenarios. Thus, a what-if study is likely to produce a suboptimal solution. By automatizing the what-if study, a much larger set of scenarios may be explored under structured constraints on which scenarios to consider. This automation is known as production optimization: it is the automated simulation and evaluation of different scenarios in search for a scenario that in some sense is best. Compared to flow estimation, which simulates one operating point at the time, production optimization explores many possible operating points. Hence, PO systems generally require better predictive capabilities of the model than VFM systems. This is perhaps the reason that real-time PO systems are used to a lesser extent than VFM systems. It is not trivial, and sometimes impossible, to obtain a model with sufficiently good predictive capabilities for production systems with limited sensing. However, PO systems are highly motivated from a financial point of view. Use of PO on subsea production systems is reported to achieve production increases in the range of 1-4% (McKie et al., 2001; Denney, 2008; Stenhouse et al., 2010; Teixeira et al., 2013). Similar gains are reported for real-time monitoring (Richardson et al., 2004). The opportunities and challenges related to RTO in the upstream industry are discussed by Foss (2012). The challenges are further elaborated on by Grimstad et al. (2014).

1.2.1. Modelling and simulation

Norway has in recent decades been leading in the development of technologies for multiphase flow simulation. Huge investments have been made to research and develop simulation models that may accurately predict multiphase flow in pipes. Several multiphase flow laboratories have been built to gain experience and empirical data, and to test different models. Emerging from this research are leading simulation technologies like the OLGA (Bendiksen et al., 1991) and

LedaFlow (Kongsberg Oil & Gas Technologies, 2015) multiphase flow simulators.

The multiphase flow simulator is a crucial component in the surveillance of subsea production systems; since only a fraction of the system states are measured, many states must be estimated from a model. Unfortunately, obtaining and maintaining an accurate model for multiphase flow simulation is a time-consuming task that requires some insight in the topic of multiphase flow.

A complex process hides behind every parameter in a production system model; the parameters in a slip model, for example, attempts to describe the relative velocities of gas and liquids as they travel together in the turmoil of slug flow. The above observation is testified by the significant investments in multiphase flow research. For a medium-sized production system, there are typically tens to hundreds of model parameters that must be tuned to obtain a useful model response. To calibrate the model parameters, experiments must be performed on the system to obtain information about the unknown states. The most common experiment is well testing, in which individual wells are routed to a test separator or multiphase flow meter that measures the flow rate at certain flow conditions. Since well testing is an expensive and disruptive operation, it is performed relatively seldom (Zenith et al., 2015). In general, the amount of useful information available for model calibration is very low.

To address the issues related to model maintenance, several oil companies have attempted to develop and use data-driven models (Goh et al., 2007). These are models that do not require insight into multiphase flow modelling, and that may be simpler to maintain in that regard. The use of data-driven models in the upstream business is still new, compared to physics-based models. It will be interesting to see how they compete with physics-based models in the future, as the number of subsea sensors and amount of information increase.

An advantage with physics-based models is that they may have valuable predictive capabilities, even without calibration – their predictive capability then stems from physical laws and flow correlations. Uncalibrated models are, for example, often used in the design phase to screen and make a preliminary sorting of different designs. Similarly, optimization may be performed with poorly calibrated models to provide a list of promising operating points, which the user may evaluate. Clearly, perfect model calibration is not a prerequisite for value. However, in some cases poor calibration may render an application worthless.

Next, we present some of the existing software products for RTO technologies such as VFM, FA, CPM, and PO.

1.2.2. Existing software products

Several software products exist for RTO of subsea production systems; be it real-time flow metering, condition and performance monitoring, flow assurance or production optimization. Some of the products are developed by software vendors and are commercially available, while others are in-house products used only by the owning oil company. A partial list of available products is provided in Table 1.1

(please note that this is an incomplete account of existing products).

Table 1.1.: Some commercial and in-house software products for simulation and optimization.

Company	**Software products**
ABB	OptimizeIT Enhanced Oil Production Suite (ABB, 2015): real-time monitoring, flow assurance and production optimization.
Baker Hughes	Neuraflow VFM (Baker Hughes, 2015): real-time flow measurement using neural networks.
Belsim	VALI (Belsim, 2015): flow metering and sensors monitoring solution.
BP	ISIS – Integrated Surveillance Information System (BP, 2015a; Foot et al., 2006). MBOS – Model-Based Operational Support (BP, 2015b).
Emerson	Roxar FieldWatch (Emerson, 2015): simulation and field monitoring system for flow assurance and production optimization, including the thermo-hydraulic calculator METTE.
FMC Technologies	FlowManager (FMC Technologies, 2015): steady-state and dynamic simulation for use in real-time multiphase metering, flow analysis, and oil and gas production optimization.
KBC	Petro-SIM for process simulation, Infochem Multiflash for flow assurance, and FEESA Maximus for integration and optimization of subsurface and surface models (KBC, 2015).
Kongsberg Oil & Gas Technologies	K-Spice and LedaFlow (Kongsberg Oil & Gas Technologies, 2015): dynamic multiphase flow simulation.
Petroleum Experts and Landmark, Halliburton	IPM (Petroleum Experts Ltd., 2014): including GAP for steady-state multiphase flow simulation, PROSPER for well performance modelling, and RESOLVE for model integration and production optimization.
Schlumberger	ECLIPSE (Schlumberger, 2014a): industry-reference reservoir simulator. OLGA (Schlumberger, 2014b): dynamic multiphase flow simulator. PIPESIM (Schlumberger, 2014c): steady-state multiphase flow simulation.
Shell	FieldWare Production Universe (Shell, 2015): real-time production surveillance and optimization.
Weatherford	Weatherford Field Office (Weatherford, 2015): a platform for real-time well and field optimization and operations.
Wood Group	Virtuoso (Wood Group, 2015): real-time monitoring and model-based virtual metering, advanced control, asset optimization, and advisory system.

In general, these products combine real-time measurements with simulation to estimate the current system state and to optimize production by predicting future states. Different technologies are taking advantage of this to obtain the best possible RTO solutions. Shell's FieldWare Production Universe, for instance, use model identification techniques to generate data-driven models for flow allocation and production optimization (Poulisse et al., 2006; Goh et al., 2007). Neuraflow from Baker Hughes also use a data-driven approach by employing artificial neural networks to model the complex behaviours seen in upstream production systems (Baker Hughes, 2015).

A more traditional approach, used by the majority of software in Table 1.1, is to combine first-principle and empirical models with real-time data. FlowManager, for example, models the whole production system by combining empirical and physics-based steady-state and dynamic models. This model is used to perform both real-time monitoring and optimization.

There are many flavours to the software in Table 1.1; some focus on very accurate modelling using dynamic multiphase flow simulation with advanced fluid and thermodynamic descriptions, while others use simplified models which are easier to optimize. Schlumberger offers a OLGA for dynamic multiphase flow simulation and PIPESIM for steady-state multiphase flow simulation. The latter is able to optimize flow for vast production networks over the complete lifecycle. Many of the products in Table 1.1 work by the principle: calibrate, then estimate. One exception is ABB's Well Monitoring System, which takes advantage of historical data when allocating rates by solving a combined allocation and calibration problem (Melbø et al., 2003). Regardless of modelling approach, products for real-time estimation and production optimization must serve several purposes, including: handling and storing real-time data, modelling and simulating (which requires a library of first-principal or data-driven models), calibrating models, optimizing production, and visualizing results, to mention a few.

Before we focus the discussion and review methods for daily production optimization, we would like to point out that multiphase flow simulation is an enabler for modern RTO solutions. Furthermore, model maintenance activities are crucial to achieving value from decision support systems like RTO. As the list in Section 1.1.3 portrays, the production engineers have limited time to perform all of the tasks related to RTO. This may be why Weatherford's sales pitch for the Field Office software (Weatherford, 2015) emphasizes on the automation of tasks: "Many of the repetitive and time-consuming processes of daily field operations can be distilled to automated functions. Our technology gives you a daily full-field analysis to predict potential issues and address existing optimization opportunities such as production problems, inefficient designs, and aging field equipment."

1.3. Methods for daily production optimization

In this section we discuss methods for daily production optimization using the framework of mathematical programming. Mathematical programming, or mathematical optimization, is a class of problem-solving techniques that is used in almost all scientific disciplines today. Its roots can be followed back to the beginning of *operations research*, which deals with the application of advanced analytical methods to help make better decisions. Mathematical programming formalizes the solution process of difficult maximization (or minimization) problems and decision problems such as the famous, NP-hard travelling salesman problem. The field of mathematical programming has been actively researched since Dantzig in 1947 developed the simplex algorithm for solving linear programming problems (Dantzig, 1987).

Operations-research concepts were adopted by the petroleum industry in the early 1950s (Carroll and Horne, 1992). Lee and Aronofsky (1958), for example, used linear programming to schedule crude oil production. From the 1950s to the 1980s, linear programming techniques were applied to macro level reservoir management and production system optimization. During this time, algorithms for integer programming emerged, allowing the inclusion of discrete decision variables in the problem formulation. In his early work, Bohannon (1970) used a MILP model to find an optimal way to develop a multi-reservoir pipeline system. Aronofsky (1983) provides an overview of optimization methods used in oil and gas development. The riveting development of mathematical programming methods in recent decades, especially in the subtopics of nonlinear programming and integer programming, has spawned a wide body of works on optimization applied to various production optimization problems; e.g. decisions problems related to all stages of a field's life, from development through tail production.

In the following, we discuss some of the methods used for (daily) production optimization in the lingo of mathematical programming. We will try to outline the evolution in the application of various methods as the needs, requirements, and expectations for accuracy and speed has increased with time.

First, we consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x), \tag{1.1}$$

where the *objective function* $f : \mathbb{R}^n \to \mathbb{R}$ maps a point $x \in \mathbb{R}^n$ to a real number. f is used to measure to which degree the objective is reached at x. In a production optimization setting, f would typically be specified so that profit (production of hydrocarbons) is maximized and cost (e.g. use of power and equipment wear-and-tear) is minimized. In this setting, x would be the operating point of the production system; that is, the valve settings and the corresponding states: pressures, temperatures, rates, and so on; and f would for instance be the daily oil production from the field. In practice, the objective function cannot be evaluated for different x without changing the actual valve settings and moving the operating point. This could result in a costly loss of production as many suboptimal operating points must be explored in order to identify a near-optimal solution. This process can be thought of as a mapping of the operational landscape by evaluating different operating points. To avoid this costly mapping process, and to account for the time-variance of the process/reservoir, engineers use process and reservoir simulators to evaluate different operating points.

To illustrate the use of simulation, let us consider two functions: f_1 and f_2. Let $f_1(x) = y$ denote a simulation evaluated at x with outputs y. Let $f_2(y) = z$ denote an objective measure where a cost z is computed from the simulation outputs y. Then, the objective f in (1.1) may be given by the function composition $f(x) = (f_2 \circ f_1)(x)$. There may be a considerable time cost related to an evaluation of f_1, that is, to one simulation. The expense of simulation is usually attributed to the numerical convergence of the simulation model. However, simulation calls may also

be costly due to software initialization, data communication, and other auxiliary functions.

Several optimization methods exist for the unconstrained problem in (1.1). The methods are often categorized as being *derivative-free* or *derivative-based*. When the function f is a known, analytical function with (first- and second-order) derivatives readily available, derivative-based methods are usually appropriate for solving (1.1). The most famous derivative-based method for unconstrained optimization is Newton's method (Nocedal and Wright, 2006). A Newton method handling multiple variables simultaneously was applied by Carroll and Horne (1992) to the optimization of production systems.

When using a derivative-based method, derivatives must be supplied by the user or by software such as an automatic differentiation scheme. Derivatives can also be approximated; for example by using a finite-difference scheme, or the BFGS method for approximation of the Hessian (Liu and Nocedal, 1989). However, when f is unknown, expensive to evaluate, or noisy, derivative-based methods may become impractical. For instance, f can be considered unknown it is implemented in legacy or proprietary software; the latter often being the case when optimizing a multiphase flow simulator. The various difficulties with simulation-based optimization, such as hidden constraints and simulation failure, is discussed by Digabel and Wild (2015). In these cases derivative-free methods may be utilized to solve (1.1).

Some derivative-free methods are similar to derivative-based methods with approximated derivatives. Examples include the trust-region methods, which iteratively solve analytical subproblems that are fitted locally to f, for example using regression or interpolation of polynomials (Conn et al., 2009) or radial-basis functions (Wild and Shoemaker, 2011). There is also a rich flora of derivative-free methods that do not compute nor explicitly approximate derivatives of f. Examples include direct-search methods and genetic algorithms. Two subclasses of direct-search methods are the directional direct-search methods, including the Mesh-Adaptive Direct Search (MADS) methods (Kolda et al., 2003), and the simplicial direct-search methods, including the simplex (Nelder-Mead) method by Nelder and Mead (1965). Fujii and Horne (1995) applied the Nelder-Mead method and a genetic algorithm for black-box optimization of networked production systems. Alternative methods referred to as *data-driven* methods, models f as an input-output relation by fitting a (nonlinear) function to available input-output pairs. These methods are attractive due to their simplicity, but require a sufficiently large amount of data to be useful. Oberwinkler and Stundner (2005) discusses a case where a neural network is used to model and optimize the oil production from a field based on historical data.

In their daily practice, the production engineers and operators perform a similar type of optimization as (1.1); that is, they change the operating point x to achieve their production targets. However, they do this while honouring operating constraints, such as the gas handling capacity of the processing facilities. As discussed in Sec. 1.1.3, the operational constraints ensure that the production system is operated in a safe manner. Thus, for an optimization method to produce mean-

ingful advice to the production engineer, all significant operating constraints must be included in the problem formulation. This leads to a *constrained* optimization problem, which can be written as

$$\begin{aligned} \text{min.} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0, \quad \text{for } i = 1, \ldots, m \\ & x \in \mathbb{R}^n, \end{aligned} \tag{1.2}$$

where m operating constraints $g_i(x) \leq 0$ are included in the problem formulation. Assuming f and g are known, analytical functions, different solvers can be used under conditions on the linearity and convexity of (1.2). In the simplest case, f and g are linear functions, and (1.2) is an linear programming (LP) problem. Large-scale LP problems can be efficiently solved by a simplex or interior-point (IP) algorithm. When f and g are nonlinear, (1.2) is a nonlinear programming (NLP) problem. Nonlinear problems can be solved by a sequential quadratic programming (SQP) or IP method. These methods are only guaranteed to converge to a global optimum if f and g are *convex* functions; if not, only local convergence can be guaranteed. Convexity is an important property also for speed and scalability since polynomial-time solution algorithms exist for convex problems (Nesterov and Nemirovskii, 1994). Nonconvex problems, however, are NP-hard and one should not expect low solution times for the general case. Solvers for global optimization can solve nonconvex problems to global optimality, as long as convex relaxations of f and g are available to the solver. Naturally, under the same conditions on f, a global solvers can solve the unconstrained problem in (1.1) to global optimality. Today, all publicly available global solvers implement what is known as *spatial branching*, in which continuous variables are branched upon until all local minima have been discovered or discarded. This technique is similar to the common branch-and-bound used to solve integer programming problems.

Wang et al. (2002) used a automatic differentiation to compute derivatives for an SQP method that optimized the production operations in petroleum fields. This approach is not possible when Problem (1.1) contains black-box functions without derivatives – as often is the case when some of the functions are implemented in a process simulator. Derivatives must then be approximated or a derivative-free method must be used. When derivatives are approximated, an inexact trust-region SQP algorithm may, for instance, be applied (Heinkenschloss and Vicente, 2002). There are several recent contributions to the topic of derivative-free methods for constrained optimization, including an extension of the MADS method, proposed by Audet and Dennis (2006). However, derivative-free methods have not been widely used to solve constrained optimization problems, and may in that regard be considered less suitable for (1.2) in all but some specific cases, including problems riddled with noise or where simulations are exceptionally expensive (Rios and Nikolaos V. Sahinidis, 2013). In the latter case, a radial basis function method has been successfully applied by Regis and Shoemaker (2005).

Now, let us bring the discussion back to the production optimization setting

and consider the production engineers' daily practice again. Let us imagine the following scenario: A set of wells produce to two pipelines in a fixed, but adjustable configuration until one of the pipelines must be closed for maintenance. Which wells should the production engineers route to the open pipeline and which wells should they shut in? These are exclusive *or* decisions: a well must produce to the open pipeline *or* be shut-in. As it turns out, it is likely impossible to pose the problem in the form of (1.2) due to these disjunctions. The above well scheduling problem is a discrete decision problem – a class of problems commonly encountered in operations research. Discrete decisions problems are usually posed as mixed-integer problems, which allow integer variables to be included in the problem formulation. In the above scenario, one could assign a binary (0-1) variable to each well and state that the well is shut in if the variable is zero, or producing if it is 1. Allowing integer variables in the problem formulation makes it possible to model all types of discrete decisions. Well routing and shutting in of wells are perhaps the two most common discrete decisions in daily production optimization. But, integer variables have been used to model the well placement problem (Lizon et al., 2014), well scheduling in shale gas systems (Knudsen et al., 2014), planning of future field developments (Sullivan, 1982), and many other decision problems.

With discrete decision added to the problem we obtain the following mixed-integer problem:

$$\begin{aligned} \text{min.} \quad & f(x,y) \\ \text{s.t.} \quad & g_i(x,y) \leq 0, \quad \text{for } i = 1,\ldots,m \\ & x \in \mathbb{R}^n, \quad y \in \mathbb{Z}^q, \end{aligned} \tag{1.3}$$

where y are integer variables. Problem (1.3) is generally in the NP-hard class of nonconvex MINLP problems; the intersection of integer problems and nonconvex problems, which both are NP-hard. Problems in this class have great modelling versatility, but do not scale very well due to the combinatorial nature of integer programming. For example, a problem with 40 binary variables have $2^{40} > 10^{12}$ possible enumerations. The situation is not improved when the objective and constraint functions are represented by a process simulator that may be time-consuming to evaluate. To achieve lower solution times, global optimality is often abandoned and a primal heuristic applied to obtain a satisfactory local solution. One example of a primal heuristic is the objective feasibility pump of Sharma et al. (2015), which was successfully applied to a well scheduling case. A common approach is to solve (1.3) using a two-level programming approach, where integer variables are handled by the upper level and the continuous variables are optimized in the lower level. Wang (2003) used the following two-level approach: the upper level, handling well connections (integer variables), was solved by the partial enumeration heuristic; the lower level, handling flow allocations (continuous variables), was solved by an SQP method.

The well-known Branch-and-Bound (BB) framework for solving mixed-integer programming problems is similar to a two-level approach. While integer variables are fixed in the two-level approach, they are relaxed by removing the integrality

condition and included as continuous variables in the "lower-level" subproblems of the BB algorithm. When f and g_i are convex functions, the BB algorithm may correctly discard branches that contain suboptimal solutions, and hence reduce the solution time considerable compared to a full enumeration. When f and g_i are nonconvex, the BB algorithm may be extended with spatial branching, to give a spatial BB (sBB) algorithm for global optimization. Several variations of the sBB algorithm exist in the literature and have been implemented in commercial solvers: for example, the Branch-and-Reduce algorithm by Ryoo and Sahinidis (1996) is implemented in the global solver BARON.

In addition to the problem with scalability due to the exponential growth in the number of enumerations in integer programming, there is a less obvious obstacle to solving (1.3). It is the assumption that f and g_i can be evaluated with integer variables as parameters. In particular, that we can evaluate, say, a pressure constraint g_i for two different routing combinations. Or in case BB is utilized, that we can evaluate g_i for a relaxed routing variable, which physically would mean a splitting of the flow. Unfortunately, this is seldom the case when the functions are implemented in a process simulator. Although there are some process simulators, like FlowManager from FMC Technologies (FMC Technologies, 2015), that do support this.

The problem with integer variable parameters, as discussed above, can be solved by *disaggregating* Problem (1.3). To disaggregate a problem here means to divide the (black-box) simulation into a set of smaller (black-box) simulation units, which are connected so that they represent the original problem. A common example would be the disaggregation of a production system into a network of pipes and equipment, where each individual pipe or equipment is represented by a simulation unit. By utilizing the structural information to disaggregate the network, connectivity logic, such as routing, can be kept separate from the simulation units. There are several advantages to disaggregation that we will address, but first consider the following disaggregated problem.

$$\begin{aligned} \text{min.} \quad & f(x) \\ \text{s.t.} \quad & \bar{g}_i(x) \leq 0, \quad \text{for } i = 1, \ldots, \bar{m} \\ & Ax + By \leq c, \\ & x \in \mathbb{R}^n, \quad y \in \mathbb{Z}^q. \end{aligned} \tag{1.4}$$

In (1.4), connectivity information and operational constraints are represented by the linear inequalities $Ax + By \leq c$. The nonlinear constraints $\bar{g}_i(x) \leq 0$ represent $\bar{m}$ (black-box) simulation units resulting from disaggregation. The number of simulation units $\bar{m}$ is not directly related to the number of operating constraints m. However, a disaggregation necessarily introduces new variables to represent the connectivity of the simulation units, causing the dimension of x in (1.4) to be higher than in (1.3). The connectivity variables can be thought of as the input and output variables of the simulation units. In most cases, the introduction of connectivity variables allows the user to specify operational constraints with linear constraints

and variable bounds. Notice that the integer variables y participate only in linear constraints and not as parameters in the (black-box) simulation units.

The disaggregated problem in (1.4) removes the issue with integer variable parameters, but introduces a new issue related to the evaluation of the process simulator. In (1.3), the process simulator, representing the whole production system, has to be evaluated; typically requiring several internal iterations to converge. In (1.4), each simulation unit $\bar{g}_i$ has to be evaluated separately. Assuming that all simulation units reside in one process simulator, this requires an extended machine-to-machine interface between the optimization algorithm and process simulator. Process simulators that lack this flexibility in their interface, may prevent disaggregation or complicate it significantly. A possible upside to disaggregation, however, is that each simulation unit is smaller and more robust to simulate than the complete production network. This brings us to another common issue with simulation-based optimization. Namely, that reliability of simulation is a requirement for dependable optimization results. In practice, as simulators become more complex, the reliability of the simulations degrade – that is, the simulator may experience convergence issues. This will cause problems for the optimization algorithm, unless specialized recovery measures are taken. Even with recovery measures in place, some regions of the operational space may be out of reach for optimization due to poor reliability, ultimately resulting in suboptimal solutions.

To address the above issues regarding reliability and speed of simulation, and inflexibility in the process simulator interface, we will in the next sections introduce the concept of surrogate modelling. Surrogate models will replace the simulation units, decoupling the optimization from the simulation during the solution process. As will become clear, this approach has several advantages in addition to accelerating solution times.

Before moving on, we emphasize that several difficult challenges related to simulation-based optimization were identified above. We are fortunate to have such a great toolbox as mathematical programming in hand for solving these problems; we will likely need more than one tool to solve the decision problems that the production engineers face in their daily operation. The hurdles are set high and the finish line seem to be receding – there is always an opportunity to improve the quality of advised decisions by increasing complexity: either by using more accurate and nonlinear models, or by adding more realistic operational constraints, or by considering more discrete decisions, or by integrating models. The preceding discussion on the different tools available in the framework of mathematical programming is summarized by Table 1.2.

1.3.1. Surrogate models for production optimization

In this section, the concept of surrogate modelling is introduced and its use in production optimization is motivated. A list of methods for surrogate modelling and their resulting model types, is provided. A few of the methods are elaborated on, but a thorough comparison is left for future research.

Table 1.2.: Comparison of methods for daily production optimization. Favourable characteristics are coloured gray.

Characteristics	**DFO**	**LP**	**NLP** (Nonconvex)	**MILP**	**MINLP**
Objective	Any	Linear	Nonlinear	(Piecewise) Linear	(Piecewise) Nonlinear
Constraints	Possible for some DFO methods	Linear	Nonlinear	(Piecewise) Linear	(Piecewise) Nonlinear
Achievable accuracy	High	Low	Intermediate–High	Intermediate–High	High
Discrete decisions	Possible for some DFO methods***	No	No	Yes	Yes
Derivatives required	No	Yes	Yes	Yes	Yes
Complexity class	–	P	P (NP-hard)	NP-hard	NP-hard
Expected speed	Fast	Fast	Fast (Slow)	Slow	Slow
Convergence	Local*	Global	Global**	Global	Global**

* Not guaranteed for all DFO methods; for example Nelder-Mead.
** Global solver required for nonconvex problems.
*** The literature on DFO methods for MINLP problems is limited (Liuzzi et al., 2015).

Surrogate models – also known in various contexts as proxy models, metamodels, response surface models, emulators, and reduced-order models – have been extensively studied and applied to overcome the challenges of simulation-based optimization (Jones, 2001). A surrogate model is an approximative model that mimics the input-output characteristic of a complex or unknown system. In this definition, a system may refer to a real system or a simulation – we are concerned with the latter in this thesis. A surrogate model is constructed from input-output data using a data-driven method, such as linear regression. Sometimes, the model is augmented with known first principles or relational information to improve the approximation. A surrogate model may, however, represent the input-output characteristic with high accuracy without an explicit statement of the underlying physics. For this reason, surrogate modelling is often referred to as black-box modelling. Furthermore, surrogate modelling can be used to obtain a computationally cheaper alternative to an expensive simulation. Reduced-order modelling, for example, aims at identifying a low-dimensional model with similar response characteristic as the high-fidelity model (Overschee and Moor, 1996). In general, surrogate modelling is concerned with the trade-off between model accuracy and computation/optimization tractabil-

ity.

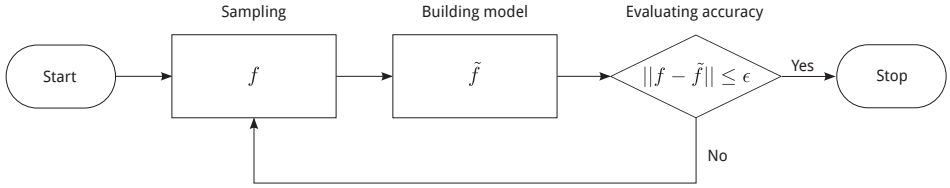


Figure 1.6.: Flowchart for building of a surrogate model $\tilde{f}$ that approximates f.

When simulations are expensive, a goal with surrogate modelling is to generate a sufficiently accurate model using as few samples as possible. The process of generating surrogate models can be separated into the following three steps, possibly carried out iteratively: sampling, model construction, and appraisal of model accuracy. The iterative process is illustrated by Figure 1.6. An initial design of experiments is generated, for instance using Latin hypercube sampling (Cozad et al., 2014). From the resulting samples a surrogate model is built. The surrogate model accuracy is then appraised by cross-validation or by performing additional sampling. If the estimated model error is sufficiently low, the process terminates; otherwise, the process is repeated.

High-fidelity reservoir simulations are particularly computationally expensive and may prevent timely optimization. The following quote from (Lerlertpakdee et al., 2014) motivates the use of surrogate models in optimization of multiphase-flow simulations:

> Automating model calibration and production optimization is computationally demanding because of the intensive multiphase-flow-simulation runs that are needed to predict the response of real reservoirs under proposed changes in model inputs.

Examples from the literature on the use of surrogate modelling for optimization include: Kriging response surfaces (Ahmed et al., 2013), neural networks and fuzzy logic (Sengul and Bekkousha, 2002), and piecewise linear models (Silva and Camponogara, 2014), to mention but a few.

In our mathematical programming framework, we replace the simulation units $\bar{g}_i$ with surrogate models $\tilde{g}_i$ to obtain:

$$\begin{aligned} \min. \quad & \tilde{f}(x) \\ \text{s.t.} \quad & \tilde{g}_i(x) \leq 0, \quad \text{for } i = 1, \ldots, \tilde{m} \\ & Ax + By \leq 0 \\ & x \in \mathbb{R}^n, \quad y \in \mathbb{Z}^q \end{aligned} \tag{1.5}$$

To simplify the following discussion we assume that each simulation unit is replaced by one surrogate model, i.e. $\tilde{g}_i \approx \bar{g}_i$ and $\tilde{m} = \bar{m}$. Note that the objective function in (1.5) is modelled with a surrogate model $\tilde{f}$.

A general requirement is that the error $\tilde{g}_i - \bar{g}_i$ introduced by using a surrogate model, must be much smaller than the error by which $\bar{g}_i$ models the real system. This is however only relevant within the operating envelope of the system. Many methods exist for constructing accurate approximations that can be used as surrogate models. Table 1.3 lists some of these methods, which stem from disciplines including statistics, machine learning, approximation theory, signal processing, and system identification. Notice that several of these methods are related to each other. For example, several of the methods, including radial basis function approximation and ordinary least squares, can be viewed as approximation with a single-layered artificial neural network.

In addition to accuracy, there are several other properties that are desirable in surrogate models meant for optimization. We identify the most important properties as:

1. approximation accuracy,
2. computational cost of construction and evaluation,
3. smoothness,
4. analytical derivatives,
5. convexity or the availability of a convex hull.

Properties 2 to 5 determine if the surrogate model is tractable for (global) optimization. Analytical derivatives, preferably first- and second-order derivatives, may accelerate the optimization for several reasons. Firstly, the computation time required by analytical derivatives is generally significantly lower than that required to approximate derivatives. In optimization involving reservoir simulations, analytical derivatives are efficiently obtained by using the adjoint equation (Jansen, 2011). Secondly, the required number of iterations of the optimization algorithm may reduced due to the accuracy of analytical derivatives[3] – resulting in better search directions than those obtained with approximated derivatives. Property 5, convexity or availability of a convex hull, is related to global optimization and will be discussed later in this thesis.

Next, we consider a motivating example where disaggregation and surrogate modelling is used to lower the Jacobian computation effort.

Example 1 (Structural information and Jacobian computation)**.** *Consider the system*

$$\begin{aligned} z_1 &= f_1(x_1), \\ z_2 &= f_1(x_1) + f_2(x_2), \end{aligned} \tag{1.6}$$

[3]As with any computation on a digital computer, the computation of analytical derivatives will have some round-off error.

Table 1.3.: Some surrogate models characterized by approximation method.

Type	**Short description**
Ordinary least squares	Also known as linear least squares. It can be used with linear basis functions, as in (generalized) linear regression, or with nonlinear basis functions. Common nonlinear basis functions include polynomials, exponentials, and logarithms; mixed bases can also be used. Evaluation cost is $\mathcal{O}(N)$. Construction involves solving a linear system of equations at cost $\mathcal{O}(N^3)$.
Radial basis function approximation	Approximation using ordinary least squares fitting with special basis functions. Several radial basis function forms exist, including Gaussian, multiquadric, inverse multiquadric, and thin plate spline. Radial basis function approximation can be viewed as a simple, single-layer artificial neural network. Evaluation cost is $\mathcal{O}(N)$. Construction involves solving a linear system at cost $\mathcal{O}(N^3)$ – however, the linear system generally requires preconditioning to avoid ill-conditioning.
Kriging	Also known as Gaussian process regression. A family of surrogate models with an embedded uncertainty model. RBF with Gaussian basis functions is a special case of Kriging. Kriging models are usually constructed by solving a nonconvex programming problem, which is expensive. Evaluation cost is $\mathcal{O}(N)$.
Artificial neural networks	Can be used to approximate unknown functions with a large number of inputs. Generally, it requires a large amount of data and computational power for training. Uses a composition of basis functions in the form of nonlinear weighted sums. Construction cost can be very high.
Piecewise linear interpolation	Interpolates points locally with a piecewise linear function. It is more flexible than linear regression, but may require a large number of samples to achieve high accuracy. Supports triangulation of scattered points; but is not practical for dimensions n higher than 3 or 4 due to a factorial increase in number of simplices. Like piecewise regression it results in a piecewise linear model.
Spline interpolation or smoothing	Built from piecewise polynomial basis functions and offers a high degree of smoothness. Like piecewise linear models, splines have compact support. The evaluation cost is $\mathcal{O}(p^2)$, where p is the spline degree ($p < N$ and usually $p \leq 3$). Construction involves solving a linear system of equations at $\mathcal{O}(N^3)$.
Wavelets and sinusoidal models	Built from sinusoidal basis functions and are mostly used to approximate time-series data. A disadvantage with sinusoidal models regarding optimization is that they are naturally oscillating (Pina and Jacob, 2013; Pereira et al., 2015). Evaluation cost is $\mathcal{O}(N)$. Construction cost is $\mathcal{O}(N \log(N))$ when the coefficients are calculated using the Fast Fourier transform.

* In this table, N refers to the number of basis functions used in the surrogate model.

where $x_1, x_2 \in \mathbb{R}$ and $z_1, z_2 \in \mathbb{R}$ may be considered inputs and outputs, respectively. When optimizing this system using a gradient-based solver, the following Jacobian

matrix must be evaluated at each iteration

$$J = \frac{dz}{dx} = \begin{bmatrix} \partial z_1/\partial x_1 & \partial z_1/\partial x_2 \\ \partial z_2/\partial x_1 & \partial z_2/\partial x_2 \end{bmatrix}. \tag{1.7}$$

If the system is considered to be black-box – i.e. the map $f : x \to z$ is considered unknown – the Jacobian must be approximated. Using central differences the Jacobian can be approximated at a point x by performing a total of $n_z \times (2n_x + 1)$ evaluations, where $n_x = 2$ and $n_z = 2$ are the number of inputs and outputs respectively. By utilizing the fact that z_1 is independent of x_2 we have that $\partial z_1/\partial x_2 = 0$, which saves us 2 evaluations each time J is approximated. That is, the single piece of structural information makes the Jacobian computation 20% faster.

Now, assume that f_1 and f_2 are surrogate models that have analytical derivatives. This would speed up the optimization considerable since the evaluation of the Jacobian would require $n_z \times n_x = 4$ evaluations. In this case, the structural information saves one iteration. Thus, only three evaluations are required for the Jacobian. A similar reduction in computational effort can be obtained for the Hessian matrix.

The choice of surrogate model determines the problem class of (1.5). Except for ordinary least squares with linear basis functions, the methods in Table 1.3 produce nonlinear surrogate models. In most cases, nonlinearity is necessary to achieve high approximation accuracy. With nonlinear surrogate models, (1.5) is an NLP, or a MINLP if $q > 0$.

Some of the simplest nonlinear surrogate models are obtained by polynomial interpolation. By increasing the degree of the polynomial, highly nonlinear functions can be approximated. However, polynomial interpolation is prone to Runge's phenomenon, where oscillations between the interpolation points occur. If many sample points are available, the situation can be improved by using polynomial fitting using ordinary least squares. However, the degree of polynomials is limited by round-off errors in the computer.

With radial basis function approximation, Kriging, and artificial neural networks, a flexible function form is obtained by including many basis functions. Radial basis functions, for example, is usually a sum of spatially dispersed functions that have the same function form. These surrogate models generally have a large number of parameters and may be prone to overfitting (Hawkins, 2004). The problem with overfitting can be handled easily by reducing the number of basis functions – the practitioner must however be aware of this concern.

Another way to obtain flexibility is to allow for piecewise-defined basis functions to achieve a property called *local support*. Splines, piecewise linear functions, and wavelets enjoy the flexibility derived from local support. Piecewise functions are used to model complex behaviour such as inflow performance relationship (IPR) curves (Guo et al., 2007) – see Figure 1.7 for an example. A piecewise linear function may in fact approximate any nonlinear function accurately, provided it consists of enough pieces. Piecewise linear surrogate models have been widely used for production optimization (Silva and Camponogara, 2014). A drawback with

piecewise functions is that they require logic for "selecting" the active piece – that is, the optimization solver must handle disjunctive constraints. With piecewise linear surrogate models this disadvantage is lessened since the resulting problem (1.5) can be formulated as a MILP, which can be efficiently solved by a MILP solver.

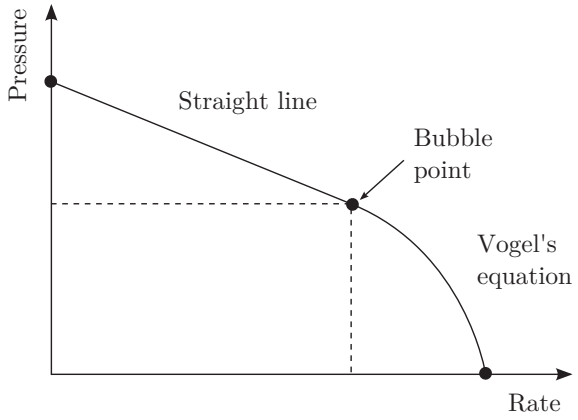


Figure 1.7.: A composite IPR curve consisting of a straight line and Vogel's quadratic equation.

In the novel work by Cozad et al. (2014), flexible surrogate models are obtained by solving a best subset problem; in which an optimal subset of a set of various basis functions is sought using integer programming.

In this thesis, a different approach is taken. Instead of addressing the issue with accuracy and flexibility as a best subset problem, we will resort to piecewise polynomial basis functions – also known as *splines*. We argue for this choice of surrogate model in the next section.

1.3.2. Spline surrogate models

This section provides a glance at splines and some of their remarkable properties. Splines, and in particular the B-spline, will be properly introduced in the next chapter of this thesis.

A spline is a function that is piecewise-defined by polynomial functions. The polynomial pieces are joined at points known as *knots* – which are similar to the breakpoints of piecewise linear functions. Put simply, the *B-spline* is a spline constructed to obtain a maximum degree of smoothness at the knots. Mathematically, a B-spline with N pieces can be expressed as

$$f(x) = \sum_{j=0}^{N-1} c_j B_j(x),$$

where $c_j \in \mathbb{R}$ are coefficients and $B_{j,p}(x)$ are B-spline basis functions of degree p. The B-spline basis functions are piecewise polynomials of degree p, defined by a recursive convex combination of polynomial pieces.

The B-spline has local support, meaning that at most $p+1$ basis functions are nonzero at any given point. This feature is exploited to obtain fast algorithms for evaluating the B-spline and its derivatives. Furthermore, since the basis functions are defined by convex combinations, these algorithms are numerically stable.

The coefficients $\{c_j\}_{j=0}^{N-1}$ of the B-spline are usually computed by solving a linear system of equations; most popular is the method of cubic B-spline interpolation. B-spline interpolation is not prone to Runge's phenomenon due to the flexibility from being piecewise-defined. Furthermore, round-off errors are minimal due to the numerical stability of the B-spline. Overfitting may occur, but can easily be handled by using a penalized B-spline (P-spline) that smooths the sample points (Eilers and Marx, 1996). The power of B-spline interpolation is most easily demonstrated with an example. In the following motivating example cubic B-spline interpolation is used to approximate the Rosenbrock function.

Example 2 (Approximation of the Rosenbrock function)**.** *To illustrate the accuracy of B-spline approximation, we consider the Rosenbrock function:*

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2,$$

on the domain $(x, y) \in X = [-2, 2] \times [-1, 3]$*. In this example,* f *is queried at 25 points on a regular 5-by-5 grid, as plotted in Figure 1.8. From these sample points a bilinear and bicubic interpolating B-spline is built using the* SPLINTER *function approximation library (Grimstad et al., 2015b). The resulting splines are shown in Figure 1.9. The absolute and relative approximation errors on the domain* X *are measured as:* $e_{X,abs} = ||f - \tilde{f}||_{X,\infty}$ *and* $e_{X,rel} = e_{X,abs}/(\sup_X f - \inf_X f)$*. The approximation errors for the two B-splines are given in Table 1.4. From 25 samples the bicubic B-spline approximates the nonlinear Rosenbrock function with a maximum error of 2.5% on the domain. As will become clear later in this thesis, a biquartic B-spline would have approximated the Rosenbrock function with zero error.*

Table 1.4.: B-spline approximation errors.

	$e_{X,\text{abs}}$	$e_{X,\text{rel}}$
Bilinear B-spline	403.84	0.161
Bicubic B-spline	61.96	0.025

In addition to being accurate, cheap to evaluate, smooth, and having analytical derivatives, the B-spline offers a convex hull without any significant computational effort – the vertices of the convex hull is basically given by the coefficients $\{c_j\}_{j=0}^{N-1}$

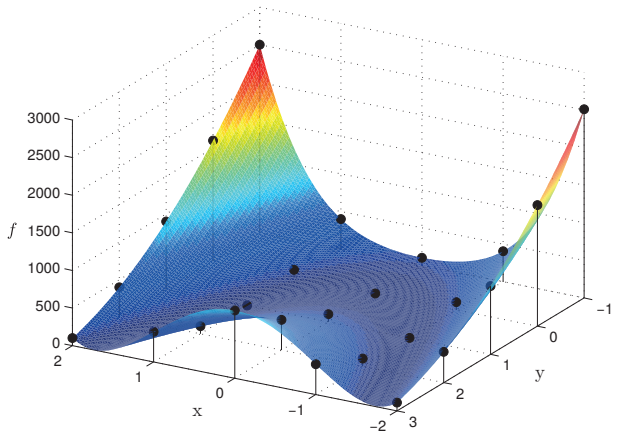


Figure 1.8.: The Rosenbrock function sampled at 25 points.

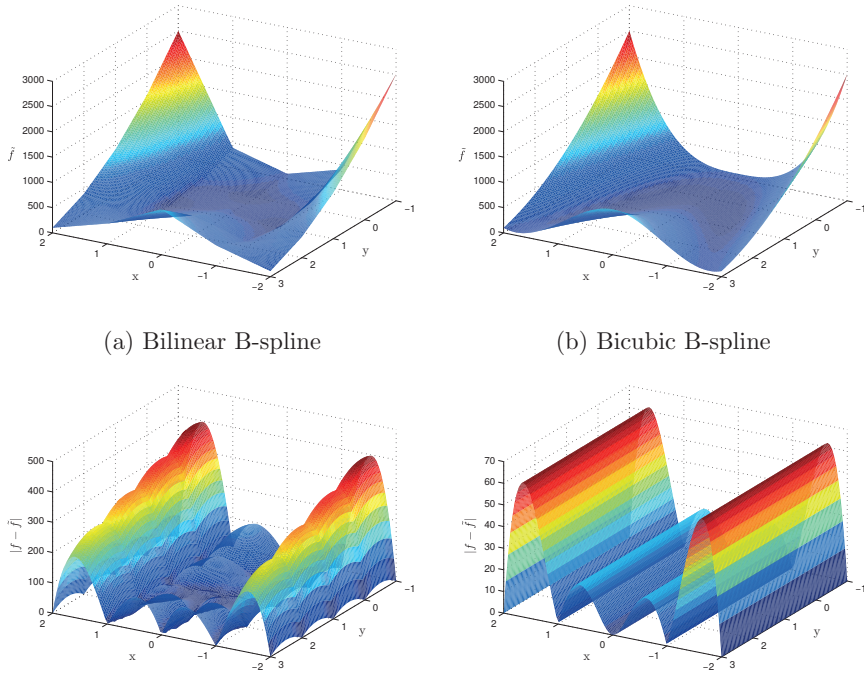
due to the convex combination construction of the B-spline. This property can be used to devise a global optimization algorithm for splines.

The B-spline may sound like the perfect surrogate model based on the one-sided discussion above. There are, of course, some drawbacks with using B-splines as surrogate models. The biggest challenge is the fact that B-splines are nontrivial to input in algebraic form to an optimization solver. As will be discussed, this motivated the development and implementation of a theoretical framework for spline optimization, to be presented in this thesis. Another drawback is that the B-spline does not inherently handle uncertainty like Kriging surrogate models.

1.4. Research objective and scope

Consider the following quote from an article in the Journal of Petroleum Technology titled *Holistic production optimization achieved one workflow at a time* (JPT, 2008).

> Sustained production optimization, a longstanding objective within the exploration and production sector, has witnessed only incremental and sporadic advances. Obstacles to adoption of a holistic approach seem intransigent – an apparently daunting challenge to individual stakeholder groups. Yet, within the industry, there is a renewed sense of drive toward consistency, and growing consensus, on key components.
>
> Effective production-data management, model management, and real-time production-optimization systems are necessary but not sufficient. Integration of these components by means of orchestrating critical workflows – to provide a human/system interface for efficient organizational,



(a) Bilinear B-spline

(b) Bicubic B-spline

(c) Approximation error of bilinear B-spline

(d) Approximation error of bicubic B-spline

Figure 1.9.: A bilinear (a) and bicubic (b) interpolating B-spline, and their corresponding absolute approximation errors in (c) and (d).

> operational, and technological collaboration—is required. To overcome inertia requires a reductionist approach to the problem. As the adage goes, we eat the elephant one bite at a time.

This quote aligns nicely with the industrial experiences of the supervisors and author of this thesis, and motivates the following research objective.

Research objective

To develop fast and reliable formulations and algorithms for daily production optimization, with the purpose of enhancing current model-based decision support tools.

To increase the chance for significant contributions, the scope of this thesis is limited to consider subsea production systems: from bottom hole of the wells to the separator topside. Thus, the reservoir and topside facilities are included in

the modelling only as boundary constraints. From a modelling point of view, this limitation is justified by the time scale argument given in (Grimstad et al., 2016). Furthermore, only steady-state models will be considered. The majority of daily production optimization problems fall within this scope. However, systems with significant short-term transients – such as production systems experiencing severe slug flow – cannot be modelled under these assumptions. These systems demand the application of a dynamic control scheme like nonlinear model predictive control (Willersrud et al., 2013).

Within the confinement of the scope there is a host of interesting and challenging problems related to production optimization. On the premise that existing multiphase simulation technology must be used, focus will be on building surrogate models that preserve sufficient simulation fidelity while attaining properties beneficial for optimization. The developed methods can then be used for real-time monitoring and optimization to improve current decision support tools.

Referring to the quote above, this will be the bite of the elephant that the author hopes to digest.

1.5. Outline and contributions of thesis

Outline and main contributions.

- **Chapter 2** contains the main result of a long trial-and-error process where the author, together with several supervised MSc students, attempted to use various surrogate models for production optimization. Some preliminary results from this work are documented in a conference paper by Grimstad (2012b) and the Master's theses of Ausen (2012); Sandnes (2013); Robertson (2014). Interpolation using linear, polynomial, piecewise polynomial, and finally spline surrogate models were tested. From this experimentation a novel branch-and-bound method for optimization with spline constraints emerged. The method, and its implementation in the code named CENSO, is presented in the paper by Grimstad and Sandnes (2015), which this chapter is solely based on.

- **Chapter 3** presents the first application of CENSO on real production optimization problems. The challenges related to solving real cases drove the initial two-year long development of CENSO. In cooperation with the oil company BP, in particular Dr. Malcolm Woodman and Mr. Richard Heddle, two production optimization cases were defined and attempted solved. Initial results showed that further development was necessary to bring down solution times for global optimization problems. This led to the implementation of bounds tightening techniques, node selection rules, branching rules, and various convex relaxations. Furthermore, the model had to be extended to include temperatures; a challenge that had not been addressed thoroughly by previous literature on oil and gas production optimization. The extension of

the model with temperatures allowed for modelling of temperature-dependant operational constraints. In particular, it allowed the inclusion of constraints on mixed in-situ velocity in the risers, to limit erosion of inner tube coating (corrosion inhibition layer). The in-situ velocity is highly sensitive to temperature; a change of only 1 degree Celsius could correspond to several hundreds standard barrels (STB) of oil per day. Thus, it was crucial that the temperatures were modelled accurately. The modelling efforts led to the modelling framework and ultimately the results presented by Grimstad et al. (2016), which is reproduced in this chapter.

- **Chapter 4** discusses the application of the method developed in (Grimstad and Sandnes, 2015; Grimstad et al., 2016) for real-time virtual flow metering. Specifically, the estimation problem is formulated as an optimization problem with B-spline surrogate models representing the pressure drop simulation units. This chapter is based entirely on the work by Grimstad et al. (2015c).[4]

- **Chapter 5** gathers and discusses the difficulties related to implementation of software for production optimization in the upstream oil and gas industry. The discussion is based on experience and several interviews with people working with production optimization in the upstream industry. The chapter is based entirely on the IO center report by Grimstad et al. (2014).

Note that each chapter listed above is *self-contained* and based on a single work.

Two appendices are attached to this thesis. Appendix A contains some notes by the author on optimization with B-spline constraints that has not been published. A description of the software developed by the author during the thesis work can be found in Appendix B.

1.5.1. Selected publications

The following publications were selected to form the basis of this thesis.

- Grimstad, B. and Sandnes, A. (2015). Global optimization with spline constraints: a new branch-and-bound method based on B-splines. *Journal of Global Optimization*.

- Grimstad, B., Foss, B., Heddle, R., and Woodman, M. (2016). Global optimization of multiphase flow networks using spline surrogate models. *Computers & Chemical Engineering*, 84:237 – 254.

[4]This work has sparked an ongoing effort by the author to investigate how state and parameter estimation for oil and gas production systems can be combined with statistical methods – such as the Generalized Likehood Ratio (GLR) method for hypothesis testing – to detect badly calibrated or uncertain models. In this regard, the author recently initiated and supervised the Master's thesis by Skibeli (2015) on this topic.

- Grimstad, B., Robertson, P., and Foss, B. (2015c). Virtual flow metering using B-spline surrogate models. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Florianópolis, Brazil.

The author has been the main contributor to these works.

1.5.2. Additional publications

The following peer-reviewed and relevant contributions by the author are not included in this thesis.

- Grimstad, B., Ausen, H., Lervik, V., Gunnerud, V., and Ljungquist, D. (2012). Optimization of a simulated well cluster using surrogate models. In *1st IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Trondheim, Norway.

- Grimstad, B. and Foss, B. (2014). A nonlinear, adaptive observer for gas-lift wells operating under slowly varying reservoir pressure. In *World Congress*, volume 19, pages 2824–2829, Cape Town, South Africa.

- Sharma, S., Knudsen, B., and Grimstad, B. (2015). Towards an objective feasibility pump for convex MINLPs. *Computational Optimization and Applications*.

- Foss, B., Grimstad, B., and Gunnerud, V. (2015). Production optimization – facilitated by divide and conquer strategies. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Florianópolis, Brazil.

- Jahanshahi, E., Grimstad, B., and Foss, B. (2015). Spline fluid models for optimization. Submitted for publication.

The author's contribution to the work by Sharma et al. (2015) was to help with the theoretic and algorithmic development of the objective feasibility pump (OFP).

The author's contribution to the work by Foss et al. (2015) was to develop the daily production optimization problem description and to construct an example problem on which different optimization approaches were compared.

1.5.3. Additional works

The author has also initiated and contributed to the following reports published within the intellectual property regime of the IO Center (2014). Note that these reports have not been peer-reviewed.

- Grimstad, B., Almklov, P., Foss, B., and Gunnerud, V. (2014). On why model-based production optimization is difficult in the upstream industry. Published as a report in the IO center.

- Jahanshahi, E. and Grimstad, B. (2014). Energy balance of three-phase mixing processes with standard conditions information. Technical report, IO center and Norwegian University of Science and Technology.

The report by Grimstad et al. (2014) is included to form a chapter of this thesis.

1.5.4. Dissemination

Throughout his PhD work, the author has shared his thoughts in many meetings and events, including the following.

- Grimstad, B. (2012e). Trustworthy production optimization. Stand. IO conference, Trondheim, Norway.
- Grimstad, B. (2012b). Optimization of a simulated well cluster using surrogate models. Presentation. The 8th International Conference on Integrated Operations in the Petroleum Industry, Trondheim, Norway.
- Grimstad, B. (2012c). Production optimization. Presentation. IBM, Oslo, Norway.
- Several workshops at FMC Technologies in Asker, Norway (Grimstad, 2014e,c).
- Several workshops at BP Exploration Operating Company Limited in Sunbury, UK (Grimstad, 2012d,a, 2013a,d,c,b,e, 2014d, 2015c). The author also stayed in London for 8 months during January–September, 2013.
- Presentations at the IO Center Technical Committee Meetings (Grimstad, 2014f,b).
- Grimstad, B. (2014a). A nonlinear, adaptive observer for gas-lift wells operating under slowly varying reservoir pressure. Presentation. The 19th IFAC World Congress, Cape Town, South Africa.
- Grimstad, B. (2014g). Short-term simulation-based production optimization. Presentation. BP, Imperial College, NTNU joint workshop at Imperial College, London, UK.
- Grimstad, B. (2015b). Daily production optimization. Presentation. IOC and BP workshop on daily production optimization, Rio de Janeiro, Brazil.
- Grimstad, B. (2015a). Black-box optimization with spline surrogate models. Seminar and lecture. Santa Catarina Federal University, Florianópolis, Brazil.

Together with Prof. Bjarne Foss, the author has defined and supervised the following MSc theses throughout 2011-2015: Ausen (2012); Sharma (2013); Sandnes (2013); Robertson (2014); Skibeli (2015).

Chapter 2

Global optimization with spline constraints

Grimstad, B. and Sandnes, A. (2015). Global optimization with spline constraints: a new branch-and-bound method based on B-splines. *Journal of Global Optimization*.

Summary

This paper discusses the use of splines as constraints in mathematical programming. By combining the mature theory of the B-spline and the widely used branch-and-bound framework a novel spatial branch-and-bound (sBB) method is obtained. The method solves nonconvex mixed-integer nonlinear programming (MINLP) problems with spline constraints to global optimality. A broad applicability follows from the fact that a spline may represent any (piecewise) polynomial and accurately approximate other nonlinear functions. The method relies on a reformulation-convexification technique which results in lifted polyhedral relaxations that are efficiently solved by an LP solver. The method has been implemented in the sBB solver CENSO *(Convex ENvelopes for Spline Optimization). In this paper* CENSO *is compared to several state-of-the-art MINLP solvers on a set of polynomially constrained NLP problems. To further display the versatility of the method a realistic pump synthesis problem of class MINLP is solved with exact and approximated pump characteristics.*

2.1. Introduction

Consider the mixed-integer nonlinear programming (MINLP) problem:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}, \\ & \mathbf{x} \in X' \cap (\mathbb{Z}^{n_d} \times \mathbb{R}^{n-n_d}) \end{aligned} \tag{$\mathbf{P}'$}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ are multivariate functions, and X' is a convex polyhedron bounding the n_d discrete variables and $n - n_d$ continuous variables. Solving $\mathbf{P}'$ to global optimality is generally an $\mathcal{NP}$-hard problem. Due to the hardness and generality of the problem, the majority of existing solution methods have

been developed to solve more tractable subclasses of $\mathbf{P}'$. For example, the successful Branch-and-Cut method solves mixed-integer linear programming (MILP) problems – for which f and $\mathbf{g}$ are affine functions (Nemhauser and Wolsey, 1999; Pochet and Wolsey, 2006; Padberg and Rinaldi, 1991). Other examples are the families of Branch-and-Bound, Branch-and-Cut, and Outer Approximation methods that solve *convex* MINLP problems – where f and $\mathbf{g}$ are convex (possibly nonlinear) functions (Bonami et al., 2008). Although MILPs and convex MINLPs are $\mathcal{NP}$-hard, they have the beneficial property of reducing to convex problems when the integrality constraints on the variables are removed. In particular, a MILP is relaxed to an LP problem, and a convex MINLP is relaxed to a convex NLP problem. This greatly simplifies the computation of a lower bound on the optimal value. There is an added complexity to *nonconvex* MINLP problems – where some of f and $\mathbf{g}$ are nonconvex. For nonconvex problems it is not enough to relax only the integrality constraints to obtain a convex lower bounding problem. In addition, any nonconvex function must be identified and replaced by a valid *convex relaxation*.

It is safe to say that a main concern in global optimization of nonconvex problems is to find convex relaxations that are tight, fast to compute, and that have favorable convergence properties under bisection/branching. Adding to this concern is the (probably contradictory) desire to obtain general convex relaxations, i.e., that are valid for a large class of functions. This has arguably been the motive of many works in global optimization. For instance, McCormick (1976) considered symbolic reformulations to generate convex relaxations of factorable nonconvex functions in his pioneering work. His idea has later been reused extensively, often in combination with spatial branching (branching on continuous variables). Today, derived approaches can be found in all commercially available spatial Branch-and-Bound (sBB) solvers for global optimization. A short review of the developments in nonconvex programming over the last two decades illustrates the widespread use of these two techniques.

The first method that was able to deal with the generic MINLP in $\mathbf{P}'$ came in 1996 with the Branch-and-Reduce of Ryoo and Sahinidis (1996). The method is characterized by its optimality- and feasibility-based domain reduction techniques. Although not a requirement, the factorable programming technique of McCormick was preferred to compute lower bounding problems. Shortly after, the αBB method for solving a subclass of nonconvex NLP (and later MINLP) problems was published (Adjiman et al., 1997, 1998b,a, 2000). The method introduced a new approach to generate convex relaxations of any function in $\mathcal{C}^2$ – the large class of twice-differentiable functions. The αBB-relaxations were combined with the tighter relaxations of McCormick to improve the lower bounding problems. Since then a range of different sBB methods for solving $\mathbf{P}'$ have been published, including: the Branch-and-Contract method of Zamora and Grossmann (1999) (for NLPs with univariate, bilinear, and fractional functions), the Symbolic Reformulation sBB approach of Smith and Pantelides (1997, 1999), the Generalized Branch-and-Cut of Kesavan and Barton (2000), and various interval-analysis-based sBB methods

(Hansen, 2003; Vaidyanathan and El-Halwagi, 1996).

There is a wide literature on convex relaxations past those of simple nonlinear terms (e.g., bilinear, fractional, and concave quadratic terms) and functions (e.g., exponential, logarithmic, and sine functions). A couple of examples include the convex relaxation of functions on polytopes (Locatelli and Schoen, 2014) and edge-concave functions (Meyer and Floudas, 2005a). A notable observation is that there seems to be a trade-off between the generality and tightness of a convex relaxation. This can be illustrated with the αBB relaxation which is general, but in many cases poor (see for instance Gatzke et al. 2002 where a hybrid reformulation is proposed). This may explain the popularity of the combination of symbolic reformulation and relatively simple convex relaxations. Realizing that the tightness of a convex relaxation is largely influenced by the variable bounds, the majority of the above-mentioned works have focused on domain reduction techniques. Indeed, this focus has yielded implementations, such as BARON (Tawarmalani and Sahinidis, 2002, 2004), LINDO (Lin and Schrage, 2009), and COUENNE (Belotti et al., 2009), which have documented an impressive performance.

In their recent survey, Burer and Letchford (2012) advocate to pursue development of algorithms for special cases of nonconvex MINLPs. They believe this may generate new techniques that later progress to general nonconvex programming techniques. In this spirit, the authors have considered the special case of nonconvex MINLPs with *spline* constraints – that is, where the functions f and $\mathbf{g}$ are spline functions. Splines are an important class of functions with wide applicability. They illusively appear as polynomials in most engineering optimization problems, typically as bilinear and quadratic functions. The class also covers piecewise linear functions, which traditionally have been handled with binary variables, special order sets, or special branching rules (Vielma et al., 2010; Keha et al., 2006). The ability of piecewise linear functions to approximate nonlinear functions has led to many applications, including: operations planning for oil and gas networks (Kosmidis et al., 2005; Martin et al., 2006; Gunnerud and Foss, 2010), process network synthesis (Bergamini et al., 2005), pooling problems (Misener et al., 2011), and merge-in-transit (Croxton et al., 2003), to mention a few. The same areas of application apply to splines.

To consider optimization with spline constraints the authors have chosen to use the mature *B-spline* framework – which is reviewed in the next section of this paper. The B-spline has many properties interesting for global optimization and have inspired a novel reformulation-convexification approach which covers polynomial, piecewise polynomial, and spline functions. The convexification results in a lifted polyhedral relaxation that can be solved by an LP solver. The new approach has been implemented in the sBB solver CENSO and compared with the three above-mentioned state-of-the-art solvers on a set of polynomially constrained problems.

The use of splines in optimization is not new. Meyer and Floudas (2005b) applied piecewise quadratic perturbation functions to improve the αBB convex underestimators. Dias et al. (2010) used the B-spline and nonlinear optimization for stochastic path planning. Functions similar to splines have been used in various

forms in stochastic and approximative (black-box) global optimization. For example, global (unconstrained) optimization methods based on response surfaces use linear, cubic, thin plate, multiquadric, or Kriging basis functions; cf. the works by Jones et al. (1998); Jones (2001) and the references therein. Response surface based methods have been applied also to black-box problems with constraints, but to a lesser extent (Sasena, 2002; Sasena et al., 2002; Regis, 2014). Examples of other works related to optimization with splines are: the global optimization method of McDonald et al. (2007) which uses radial basis function response surface models, and the blending function method of Meyer et al. (2002) for approximative optimization of nonfactorable functions.

In recent years, specialized BB methods based on the Bernstein coefficients have emerged and proved efficient in globally solving optimization problems with multivariate polynomial constraints (Garloff et al., 2003; Garloff and Smith, 2001; Nataraj and Arounassalame, 2007, 2011; Smith, 2009). The approach used by these methods is to first reformulate the problem by writing all multivariate polynomials in Bernstein form. Next, they perform a branch-and-bound search where bounding is done purely based on the Bernstein coefficients, thus avoiding any function evaluation. The method presented in this paper resembles the Bernstein methods in many ways – e.g. in that a reformulation is done before a BB search solves convex lower bounding problems based on the B-spline coefficients. However, the Bernstein form is a special case of the B-spline, which may represent any *piecewise* polynomial. Except from the related Bernstein approach for polynomial optimization, and a recent attempt on unconstrained global optimization using the multivariate B-spline (Park, 2012), the literature is scarce on optimization with B-splines.

The remainder of this paper is organized as follows. First, the B-spline and its relevant properties are presented in Sec. 2.2. This section also treats some important concepts and procedures for manipulating a B-spline. Next, Sec. 2.3 brings the B-spline into an optimization setting by showing how B-spline constraints can be handled. Sec. 2.4 presents a novel sBB algorithm for optimization of MINLP problems with spline constraints. A computational study of the algorithm follows in Sec. 2.5. The paper is concluded in Sec. 2.6, where some suggestions for further research are given.

2.2. Background on B-splines

In 1946 Schöenberg mathematically described the piecewise polynomial functions known as splines (Schönberg, 1946). Since then the theory of splines has been developed and explored thoroughly in many directions. It has seen numerous industrial applications; ranging from modelling of automobile bodies in the 1960s, to its use in modern computer-aided design (CAD) tools. The popularity of the spline is usually attributed to its versatility to model and approximate complex shapes and functions, and its excellent numerical properties. As the literature reveals, these

characteristics are aggregated from the mathematical construction of the spline. The well-established B-spline is one such construction – and the preferred spline representation in this paper. In the forthcoming sections, the B-spline and some of its properties will be highlighted from a mathematical programming point of view. It will then become clear to the reader that the B-spline has properties that make it suitable for global optimization. For example, its *convex hull* property allows for a straightforward construction of a relatively tight (convex) polyhedral relaxation.

To avoid referring the reader to other works most of the basic and relevant properties are covered here. These properties, and many more, can be found in any textbook on B-splines, e.g. Schumaker (2007); Piegl and Tiller (1997). The univariate B-spline is introduced before the multivariate B-spline, which will be used in subsequent sections. A remark is that the multivariate B-spline is presented using the Kronecker product and vectorized coefficients – a format suitable for implementation in computer code. After presenting the properties of the B-spline relevant for optimization, the very useful knot insertion algorithm is given. This algorithm is a keystone in the B-spline subdivision procedure given later. A supplementary discussion on how to obtain a B-spline form of a polynomial is given in Sec. 2.2.5.

2.2.1. Univariate B-splines

A univariate, degree p B-spline $f : \mathbb{R} \to \mathbb{R}$ is constructed from n *B-spline coefficients* $\mathbf{c} = [c_j]_{j=0}^{n-1}$ and $n + p + 1$ *knots* $\mathbf{t} = [t_j]_{j=0}^{n+p}$ as

$$f(x; \mathbf{c}, p, \mathbf{t}) = \sum_{j=0}^{n-1} c_j B_{j,p,\mathbf{t}}(x) = \mathbf{c}^{\mathsf{T}} \mathbf{B}_{p,\mathbf{t}}(x). \tag{2.1}$$

When the parameters $\mathbf{c}$, p, and $\mathbf{t}$ are given by the context $f(x; \mathbf{c}, p, \mathbf{t})$ is simply denoted $f(x)$. In (2.1), $\mathbf{B}_{p,\mathbf{t}}(x) = [B_{j,p,\mathbf{t}}(x)]_{j=0}^{n-1}$ is a column vector of pth-degree B-spline *basis functions*, defined by the recurrence relation

$$\begin{aligned} B_{j,p,\mathbf{t}}(x) &= \frac{x - t_j}{t_{j+p} - t_j} B_{j,p-1,\mathbf{t}}(x) + \frac{t_{j+1+p} - x}{t_{j+1+p} - t_{j+1}} B_{j+1,p-1,\mathbf{t}}(x), \text{[1]} \\ B_{j,0,\mathbf{t}}(x) &= \begin{cases} 1, & t_j \le x < t_{j+1}, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{2.2}$$

The B-spline basis functions are (overlapping) piecewise polynomials defined on the entire real line. However, the domain of the B-spline in (2.1) is considered to be $X = [t_0, t_{n+p}]$ since all basis functions are identically zero outside X. For the definition of the basis functions to make sense it is required that the *knot vector* $\mathbf{t}$ contains a nondecreasing sequence of real numbers (knots). Throughout this paper the knot vector is assumed to be *regular* in the following sense.

[1] Division by zero is handled by a '$0/0 = 0$' convention.

Definition 2.1 (Regular knot vector)**.** *A knot vector* $\mathbf{t}$ *is said to be* regular *if* $t_0 = t_p$*,* $t_n = t_{n+p}$*, and* $t_i \leq t_{i+1} < t_{i+p+1}$ *for* $i = 0, \ldots, n-2$.[2]

A regular knot vector ensures many important properties of the basis functions. For that reason a key aspect of procedures that manipulate the knot vector is to maintain regularity. Since some readers may be unfamiliar with the B-spline basis functions a few important properties are summarized below.

Property 2.1 (Nonnegativity)**.** $B_{j,p,\mathbf{t}}(x) \geq 0$ *for all* j*,* p*, and* x*.*

Property 2.2 (Local support)**.** $B_{j,p,\mathbf{t}}(x) = 0$ *for all* $x \notin [t_j, t_{j+p+1})$*.*

Property 2.3 (Partition of unity)**.** $\sum_{j=i-p}^{i} B_{j,p,\mathbf{t}}(x) = 1$ *for all* $x \in [t_i, t_{i+1})$*.*

As seen from the properties of the basis functions, the B-spline in (2.1) is a convex combination of the coefficients $\mathbf{c}$. It is also worth noting that it has local support, meaning that at most $p+1$ basis functions are nonzero at a point x. These properties allow fast and numerically stable schemes for evaluating the B-spline. The most popular among these is the recursive DeBoor-Cox algorithm (De Boor, 1972; Cox, 1972).

All derivatives of $f(x)$ exist in the interior of a knot span (where it is a polynomial). At a knot with multiplicity r, $f(x)$ is $p-r$ times continuously differentiable. With distinct knots – all knots having a multiplicity of one – $f(x)$ belongs to $\mathcal{C}^{p-1}$ on its domain X. In fact, its kth derivative is another B-spline of degree $p-k$ (Schumaker, 2007).

The linear vector space (or spline space) spanned by the basis functions is denoted $\mathbb{S}_{p,\mathbf{t}}$. Put informally, the Curry-Schoenberg theorem (Curry and Schoenberg, 1966) states that: any space of piecewise polynomials of degree $\leq p$ is a subset of $\mathbb{S}_{p,\mathbf{t}}$ on X, given an appropriate knot vector. An obvious consequence of this is that $\mathbb{P}_p \subseteq \mathbb{S}_{p,\mathbf{t}}$ on X, where $\mathbb{P}_p$ is the space of polynomials of degree $\leq p$ and $\mathbf{t}$ is an appropriate knot vector. To illustrate this consider the single segment $[0, 1]$ with no internal knots. On this segment a degree p B-spline has $p+1$ basis functions that correspond to the Bernstein polynomials. Hence it is a basis for all degree p polynomials on this segment. The relation between the B-spline basis functions and Bernstein polynomials is given below.

Property 2.4 (Bernstein polynomials)**.** *The regular knot vector*

$$\mathbf{t} = \{\underbrace{a, \ldots, a}_{p+1}, \underbrace{b, \ldots, b}_{p+1}\} \tag{2.3}$$

yields the basis functions

$$B_{j,p,\mathbf{t}}(x) = \binom{p}{j} \left(\frac{x-a}{b-a}\right)^j \left(\frac{b-x}{b-a}\right)^{p-j}, \tag{2.4}$$

[2] A knot vector satisfying the conditions in Definition 2.1 is also said to be $(p+1)$-regular or clamped.

for $j = 0, \ldots, p$ *and* $x \in [a, b]$. *When* $a = 0$ *and* $b = 1$, (2.4) *gives the Bernstein polynomials used to represent the famous Bézier curve.*

2.2.2. Multivariate B-splines

In the multivariate case a B-spline may be expressed as the tensor product of univariate basis functions. For this reason a multivariate B-spline is often called a tensor product B-spline. Let $\mathbf{x} \in \mathbb{R}^d$ and $\{\mathbf{B}_{p_1,\mathbf{t}_1}(x_1), \ldots, \mathbf{B}_{p_d,\mathbf{t}_d}(x_d)\}$ be the univariate basis function (column) vectors of sizes $\{n_1, \ldots, n_d\}$ with corresponding degrees $\mathbf{p} = \{p_1, \ldots, p_d\}$ and knot vectors $\mathbf{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_d\}$. Then the multivariate B-spline basis can be written as

$$\mathbf{B}_{\mathbf{p},\mathbf{T}}(\mathbf{x}) = \mathbf{B}_{p_1,\mathbf{t}_1}(x_1) \otimes \ldots \otimes \mathbf{B}_{p_d,\mathbf{t}_d}(x_d) = \bigotimes_{i=1}^{d} \mathbf{B}_{p_i,\mathbf{t}_i}(x_i), \tag{2.5}$$

where $\otimes$ denotes the Kronecker product. The size of $\mathbf{B}_{\mathbf{p},\mathbf{T}}$, that is, the number of basis functions, is

$$N = \prod_{i=1}^{d} n_i. \tag{2.6}$$

With this basis a multivariate B-spline may be compactly written as

$$f(\mathbf{x}; \mathbf{c}, \mathbf{p}, \mathbf{T}) = \mathbf{c}^{\mathsf{T}} \mathbf{B}_{\mathbf{p},\mathbf{T}}(\mathbf{x}), \tag{2.7}$$

where the B-spline coefficients are collected in the column vector $\mathbf{c} \in \mathbb{R}^N$. The domain of $f(\mathbf{x})$ is considered to be $X = [t_{0,1}, t_{n_1+p_1,1}] \times \cdots \times [t_{0,d}, t_{n_d+p_d,d}]$, where $\{t_{j,i}\}_{j=0}^{n_i+p_i}$ are the knots in $\mathbf{t}_i$. Furthermore, $f(\mathbf{x}) \in \mathbb{S}_{\mathbf{p},\mathbf{T}}$, where $\mathbb{S}_{\mathbf{p},\mathbf{T}}$ denotes the space of tensor product splines with degree $\leq \mathbf{p}$ and knot vectors $\mathbf{T}$.[3] Given appropriate knot vectors, it follows from the univariate case that $\mathbb{P}_{\mathbf{p}} \subseteq \mathbb{S}_{\mathbf{p},\mathbf{T}}$, where $\mathbb{P}_{\mathbf{p}}$ is the space of multivariate polynomials with degree $\leq \mathbf{p}$.

Since the multivariate B-spline consists of products of univariate basis functions it inherits many of the nice properties of the univariate B-spline. The algorithms for evaluating the univariate B-spline and its derivatives can be reused in the multivariate case.

A useful property of (univariate and multivariate) B-splines is that they possess affine invariance, meaning that affine transformations can be applied to the coefficients alone. In the multivariate case, affine transformations must operate on the vectorized coefficients $\mathbf{c}$. For example, let $\mathsf{A}_i \in \mathbb{R}^{n_i \times n_i}$ be a linear transformation of the univariate basis in variable x_i. Then, a linear transformation of the multivariate basis can be written as

$$\hat{\mathbf{c}} = \left(\bigotimes_{i=1}^{d} \mathsf{A}_i \right) \mathbf{c}, \tag{2.8}$$

[3] $\leq \mathbf{p}$ is here meant as element-wise inequality. Note that a multivariate spline or polynomial of degree $\mathbf{p}$ may have terms like $x_1^{p_1} \cdots x_d^{p_d}$, so in the conventional sense its degree is $\sum_j p_j$.

where $\hat{\mathbf{c}}$ is the new vector of B-spline coefficients. For a transformation in only one variable, say x_k, $\mathsf{A}_i = \mathsf{I}_i$, $\forall i \neq k$, where $\mathsf{I}_i \in \mathbb{R}^{n_i \times n_i}$ is an identity matrix. This property is extensively used in the spline procedures presented later in this work. Note that in the univariate case, (2.8) simplifies to $\hat{\mathbf{c}} = \mathsf{A}\mathbf{c}$.

2.2.3. The control points and control structure of a B-spline

The *control points* are a key feature of B-splines. They are best described by writing the B-spline in *parametric form*. Let $y = \mathbf{c}^{\mathsf{T}}\mathbf{B}_{\mathbf{p},\mathbf{T}}(\mathbf{x})$. Then, the parametric form is obtained by introducing independent parameters η and representing $\mathbf{x}$ and y as explicit functions of these parameters, i.e. $\mathbf{x} = \mathbf{x}(\eta) = \mu^{\mathsf{T}}\mathbf{B}_{\mathbf{p},\mathbf{T}}(\eta)$ and $y = y(\eta) = \mathbf{c}^{\mathsf{T}}\mathbf{B}_{\mathbf{p},\mathbf{T}}(\eta)$, using the same knot vector $\mathbf{T}$ and degree $\mathbf{p}$. The new coefficients are called *knot averages*, and are here denoted by $\mu_i \in \mathbb{R}^N$, for $i = 1, \ldots, d$. To leave y unaltered μ is calculated so that $\mathbf{x}(\eta) = \eta$, that is

$$\mathbf{x}(\eta) = [\mu_1, \ldots, \mu_d]^{\mathsf{T}}\mathbf{B}_{\mathbf{p},\mathbf{T}}(\eta) = \eta. \tag{2.9}$$

The knot averages may be computed explicitly from the knot vectors, or implicitly by evaluating $\mathbf{B}_{\mathbf{p},\mathbf{T}}$ at $d \cdot N$ points and solving a linear system of (2.9). In the univariate case, $d = 1$, the knot averages are simply $\mu_j = (t_{j+1} + \ldots + t_{j+p})/p$.

With the knot averages computed, the parametric form of (2.7) can be written as

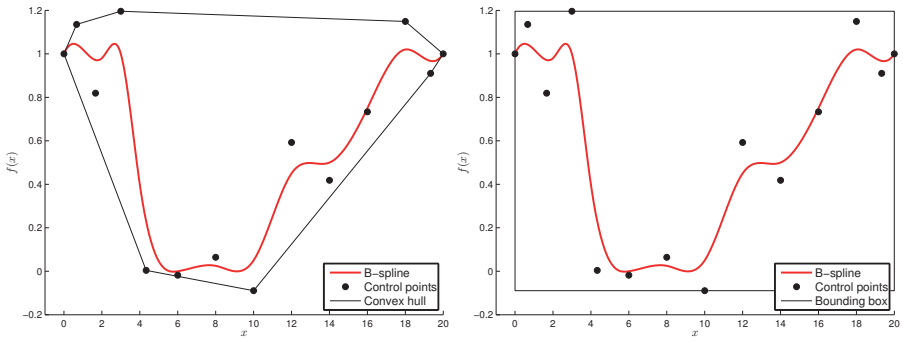
$$\begin{bmatrix}\mathbf{x}(\eta)\\ y(\eta)\end{bmatrix} = [\mu_1, \ldots, \mu_d, \mathbf{c}]^{\mathsf{T}}\mathbf{B}_{\mathbf{p},\mathbf{T}}(\eta) = \mathsf{P}\mathbf{B}_{\mathbf{p},\mathbf{T}}(\eta). \tag{2.10}$$

The control points of the B-spline in (2.7) are the N points with $(\mathbf{x}, y)$-coordinates equal to the columns $\{\mathsf{P}_j\}_{j=0}^{N-1}$ of $\mathsf{P} \in \mathbb{R}^{d+1 \times N}$.

Recalling the basis function properties of nonnegativity and partition of unity, it follows from (2.10) that any point $(\mathbf{x}, y)$ on the B-spline surface is given as a convex combination of the control points $\{\mathsf{P}_j\}$. Consequently, the B-spline must be in the convex hull of the control points. This property of the B-spline is summarized in the next lemma and illustrated in Fig. 2.1a.

Lemma 2.1 (Convex hull property)**.** *Let* $\mathbb{C} = conv(\{\mathsf{P}_j \mid j = 0, \ldots, N-1\})$ *and* $\mathbb{D} = \{(\mathbf{x}, f(\mathbf{x})) \mid \mathbf{x} \in X\}$*, then* $\mathbb{D} \subseteq \mathbb{C}$.

The convex hull of a set of points is contained inside the *axis-aligned minimum bounding box* of the points. The minimum bounding box of a set of points is the smallest (by some metric) hyperrectangle that contains all points. It is said to be axis-aligned if its edges are parallel to the coordinate axis. Let the $(d+1)$-dimensional points of the set be given as $\mathsf{P}_j = [p_{1,j}, \ldots, p_{d+1,j}]^{\mathsf{T}}$, and the minimum and maximum element in each dimension be given as $\underline{p}_i = \min\{p_{i,0}, \ldots, p_{i,N-1}\}$ and $\overline{p}_i = \max\{p_{i,0}, \ldots, p_{i,N-1}\}$. The axis-aligned minimum bounding box is then the hyperrectangle $\mathbb{H} = [\underline{p}_1, \overline{p}_1] \times \cdots \times [\underline{p}_{d+1}, \overline{p}_{d+1}]$. The next corollary follows directly from Lemma 2.1.

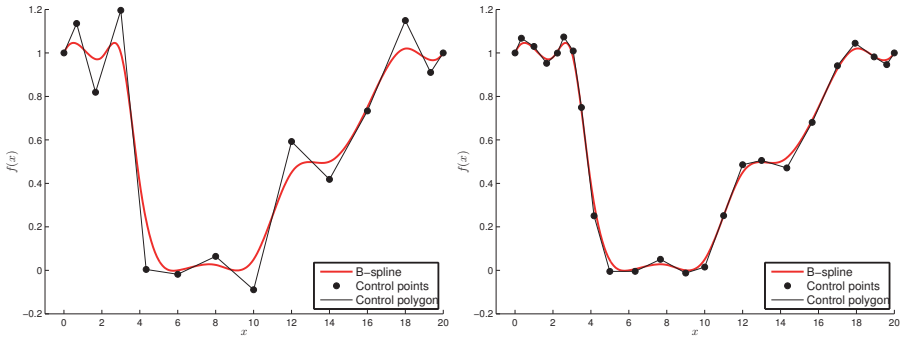


(a) Convex hull (b) Minimum bounding box

Figure 2.1.: The convex hull (a) and bounding box (b) property illustrated with a univariate, cubic B-spline.

Corollary 2.1 (Minimum bounding box property)**.** *From Lemma 2.1 it follows that* $\mathbb{D} \subseteq \mathbb{C} \subseteq \mathbb{H}$*, where* $\mathbb{H}$ *is the axis-aligned minimum bounding box of the control points* $\{\mathsf{P}_j \mid j = 0, \ldots, N-1\}$.

Closely related to the control points is the *B-spline control structure*. It is the linear interpolant of the control points and may itself be written as a B-spline of degree one ($p = 1$). For $d = 1$ it is called the control polygon since it is the polygon formed by the control points $\{\mathsf{P}_j\}$. The control polygon of a cubic spline is shown in Fig. 2.2.



(a) (b)

Figure 2.2.: A univariate, cubic B-spline with 18 knots (a) and 28 knots (b). The two splines are geometrically identical, but have different knot spacing. Notably, the control polygon of (b) is closer to the spline than the control polygon of (a).

For $\mathbf{x}$ in the interval $X = X_1 \times \ldots \times X_d$, let $h(\mathbf{x}) = \mathbf{c}^{\mathsf{T}}\mathbf{B}_{\mathbf{1},\mathbf{T}}(\mathbf{x})$ be the piecewise linear function that interpolates the control points $\{\mathsf{P}_j\}$. Furthermore, let $\Delta_i = \max\{t_{j,i} - t_{j-1,i} \mid j = 1, \ldots, n_i + p_i\}$ be the largest knot span in the knot vector $\mathbf{t}_i$,

and let $D_i^2 f$ be the ith-directional second derivative of f. Then, it can be shown that (cf. Schumaker 2007; Cohen and Schumaker 1985; Prautzsch and Kobbelt 1994; Reif 2000)

$$||f(\mathbf{x}) - h(\mathbf{x})||_{X,\infty} \leq \sum_{i=1}^{d} C_i \Delta_i^2 ||D_i^2 f||_{X_i,\infty}, \tag{2.11}$$

where C_i are constants and $||\cdot||_{I,\infty}$ denotes the L_∞-norm on the interval I. Because of the factors Δ_i^2 in (2.11) the control structure is said to converge quadratically to the spline. Degree elevation and knot refinement are two methods that increase the number of knots (shorten the knot spans), and effectively lower the upper bound in (2.11). Knot refinement is the preferred method in this paper (see Cohen and Schumaker 1985 for a comparison of the two methods). Before presenting the knot insertion and knot refinement method, consider the following comments to (2.11):

- As the largest knot span of *all* knot vectors goes to zero, so does the upper bound.
- Decreasing the largest knot span of a knot vector will lower the upper bound. This motivates knot refinement on a largest span first basis. However, the upper bound does not guarantee that this approach is optimal. Reif (2000) discusses a lower bound that may be better suited to device an efficient refinement procedure.
- As the knot spacing goes to zero the control structure converges to the spline, and the convex hull of Lemma 2.1 approaches the convex envelope of the spline.

2.2.4. Knot insertion

Knot insertion is the procedure of augmenting the knot vector with new knots. It is one of the most important B-spline algorithms and has many applications; one of them being subdivision. In essence, knot insertion is a change of vector space basis, e.g. the curve is not changed geometrically or parametrically (Boehm, 1980; Lyche et al., 1985; Piegl and Tiller, 1997).

Definition 2.2 (Refinement)**.** *A knot vector τ is said to be a* refinement *of a knot vector $\mathbf{t}$ if any real number occurs at least as many times in τ as in $\mathbf{t}$.*

Lemma 2.2. *Let p be a positive integer and let $\mathbf{t}$ be a knot vector with at least $p+2$ knots. If τ is a refinement of $\mathbf{t}$ then $\mathbb{S}_{p,\mathbf{t}} \subseteq \mathbb{S}_{p,\tau}$.*

Let $\tau = [\tau_i]_{i=0}^{\tilde{n}+p}$ be a refinement of $\mathbf{t} = [t_i]_{i=0}^{n+p}$, so that $n \leq \tilde{n}$. In practice τ is a knot vector obtained by adding knots to $\mathbf{t}$. Let

$$\mathbb{S}_{p,\mathbf{t}} = \text{span}\{B_{j,p,\mathbf{t}}\}_{j=0}^{n-1} \text{ and } \mathbb{S}_{p,\tau} = \text{span}\{N_{j,p,\tau}\}_{j=0}^{\tilde{n}-1}$$

be the spline spaces of degree p spanned by the basis functions defined by $\mathbf{t}$ and τ, respectively. From Lemma 2.2, $\mathbb{S}_{p,\mathbf{t}} \subseteq \mathbb{S}_{p,\tau}$ and there exists an $\tilde{n} \times n$ matrix A so that

$$\mathbf{B}_{p,\mathbf{t}} = \mathsf{A}^{\mathsf{T}}\mathbf{N}_{p,\tau},$$

i.e. $\mathbb{S}_{p,\mathbf{t}}$ is invariant to the linear transformation of knot insertion. The change of basis from $\mathbf{B}_{p,\mathbf{t}}$ to $\mathbf{N}_{p,\tau}$ is given by the *knot insertion matrix* A. A can be efficiently calculated using the *Oslo algorithm 1* (Cohen et al., 1980; Lyche et al., 1985). Selecting the new coefficients as $\mathbf{d} = \mathsf{A}\mathbf{c}$ gives

$$f(x) = \mathbf{c}^{\mathsf{T}}\mathbf{B}_{p,\mathbf{t}}(x) = \mathbf{c}^{\mathsf{T}}\mathsf{A}^{\mathsf{T}}\mathbf{N}_{p,\tau}(x) = \mathbf{d}^{\mathsf{T}}\mathbf{N}_{p,\tau}(x),$$

where the last term is called the B-spline expansion of f on τ.

In the multivariate case, consider the B-spline

$$f(\mathbf{x}) = \mathbf{c}^{\mathsf{T}}\left(\mathbf{B}_{p_1,\mathbf{t}_1}(x_1) \otimes \ldots \otimes \mathbf{B}_{p_d,\mathbf{t}_d}(x_d)\right)$$

with knot vectors $\{\mathbf{t}_1, \ldots, \mathbf{t}_d\}$. A B-spline expansion $\tilde{f}(\mathbf{x})$ of $f(\mathbf{x})$ can be written as

$$\tilde{f}(\mathbf{x}) = \mathbf{d}^{\mathsf{T}}\left(\mathbf{N}_{p_1,\tau_1}(x_1) \otimes \ldots \otimes \mathbf{N}_{p_d,\tau_d}(x_d)\right),$$

where the knot vector τ_i is a refinement of $\mathbf{t}_i$ for $i = 1, \ldots, d$. Let the matrices $\{\mathsf{A}_i\}_{i=1}^{d}$ be the corresponding univariate knot insertion matrices such that

$$\mathbf{B}_{p_i,\mathbf{t}_i}(x_i) = \mathsf{A}_i^{\mathsf{T}}\mathbf{N}_{p_i,\tau_i}(x_i), \text{ for } i = 1, \ldots, d.$$

Note that $\mathsf{A}_i = \mathbf{I}$ when $\tau_i = \mathbf{t}_i$, that is, when there is no refinement of $\mathbf{t}_i$. The transformation

$$\mathbf{d} = \left(\mathsf{A}_1 \otimes \ldots \otimes \mathsf{A}_d\right)\mathbf{c} \tag{2.12}$$

then gives

$$\begin{aligned}
\tilde{f}(\mathbf{x}) &= \mathbf{d}^{\mathsf{T}}\left(\mathbf{N}_{p_1,\tau_1}(x_1) \otimes \ldots \otimes \mathbf{N}_{p_d,\tau_d}(x_d)\right) \\
&= \mathbf{c}^{\mathsf{T}}\left(\mathsf{A}_1^{\mathsf{T}} \otimes \ldots \otimes \mathsf{A}_d^{\mathsf{T}}\right)\left(\mathbf{N}_{p_1,\tau_1}(x_1) \otimes \ldots \otimes \mathbf{N}_{p_d,\tau_d}(x_d)\right) \\
&= \mathbf{c}^{\mathsf{T}}\left(\mathsf{A}_1^{\mathsf{T}}\mathbf{N}_{p_1,\tau_1}(x_1) \otimes \ldots \otimes \mathsf{A}_d^{\mathsf{T}}\mathbf{N}_{p_d,\tau_d}(x_d)\right) \\
&= \mathbf{c}^{\mathsf{T}}\left(\mathbf{B}_{p_1,\mathbf{t}_1}(x_1) \otimes \ldots \otimes \mathbf{B}_{p_d,\mathbf{t}_d}(x_d)\right) \\
&= f(\mathbf{x}),
\end{aligned}$$

and $f(\mathbf{x})$ is left unaltered by the change of basis. Eq. (2.12) is utilized in the following procedure for knot insertion.

The knot insertion procedure in Algorithm 1 has many applications. In the following *knot refinement* procedure (see Algorithm 2) it is used to refine the knot vectors to a desired number of knots. The procedure inserts knots iteratively at the midpoint of the largest knot span of a knot vector $\mathbf{t}_i$ until a desired number of knots r_i is achieved.

An important note to the knot refinement procedure is that, because it always inserts knots at the midpoint of a knot span, it will never break knot vector regularity (see Definition 2.1).

Algorithm 1 Knot insertion

Require: Coefficients $\mathbf{c}$, knot vectors $\{\mathbf{t}_1, \ldots, \mathbf{t}_d\}$, and refined knot vectors $\{\tau_1, \ldots, \tau_d\}$
Ensure: New B-spline coefficients $\mathbf{d}$
1: Compute univariate knot insertion matrices $\{\mathsf{A}_i\}_{i=1}^{d}$
2: Compute new B-spline coefficients as $\mathbf{d} = (\mathsf{A}_1 \otimes \ldots \otimes \mathsf{A}_d)\,\mathbf{c}$

Algorithm 2 Knot refinement

Require: Coefficients $\mathbf{c}$, knot vectors $\{\mathbf{t}_1, \ldots, \mathbf{t}_d\}$, and desired number of knots $\{r_1, \ldots, r_d\}$
Ensure: Refined knot vectors $\{\tau_1, \ldots, \tau_d\}$ and coefficients $\mathbf{d}$
1: Initialize new knot vectors as $\tau_i \leftarrow \mathbf{t}_i$, for $i = 1, \ldots, d$
2: **for** $i = 1$ to d **do**
3: **while** number of knots in τ_i is less than r_i **do**
4: $j \leftarrow$ index of largest knot interval $[\tau_j, \tau_{j+1}]$ in τ_i
5: $s \leftarrow (\tau_j + \tau_{j+1})/2$
6: Insert s in τ_i
7: **end while**
8: **end for**
9: With the refined knot vectors $\{\tau_1, \ldots, \tau_d\}$ compute new B-spline coefficients $\mathbf{d}$ using the knot insertion procedure in Algorithm 1

2.2.5. Representing polynomials with B-splines

Consider a univariate polynomial

$$p_p(x) = \lambda_0 + \lambda_1 x + \ldots + \lambda_p x^p = \lambda^{\mathsf{T}} \mathbf{x}_p \tag{2.13}$$

defined on $x \in [a, b]$, where $\mathbf{x}_p = [x^i]_{i=0}^{p}$ is the power basis to degree p, and $\lambda = [\lambda_i]_{i=0}^{p}$ are the coefficients. It is well known that any polynomial can be expressed by the Bernstein polynomials. Furthermore, the Bernstein polynomials are a B-spline basis with a special knot vector, as described by Property 2.4. Thus, it is clear that any polynomial can be expressed by a B-spline, as expected from the fact that $\mathbb{P}_p \subseteq \mathbb{S}_p$. With degree p and a knot vector defined as in Property 2.4, a B-spline is defined on the segment $[a, b]$ as

$$f(x) = \sum_{j=0}^{p} c_j B_{j,p,\mathbf{t}}(x) = \mathbf{c}^{\mathsf{T}} \mathbf{B}_{p,\mathbf{t}}(x) = \mathbf{c}^{\mathsf{T}} \mathsf{T}_p^{\mathsf{T}} \mathbf{x}_p. \tag{2.14}$$

The last term in (2.14) states that the B-spline basis can be written as a linear transformation of the power basis, i.e., $\mathbf{B}_p = \mathsf{T}_p^{\mathsf{T}} \mathbf{x}_p$. Following Piegl and Tiller (1997), the linear transformation is considered in two operations; by letting $\mathsf{T}_p = \mathsf{R}_p \mathsf{M}_p$, so that $\mathbf{B}_p = \mathsf{M}_p^{\mathsf{T}} \mathsf{R}_p^{\mathsf{T}} \mathbf{x}_p$. The first transformation, given by R_p, is a reparameterization of the polynomial from the interval $[a, b]$ to $[0, 1]$, as required by the Bernstein

polynomial. With $c = 1/(b-a)$ and $d = -a/(b-a)$ the reparameterization matrix $\mathsf{R}_p \in \mathbb{R}^{(p+1)\times(p+1)}$ is given as

$$\mathsf{R}_p(i,j) = \begin{cases} 0 & , \quad \text{for } i > j, \\ \binom{j}{i} c^i d^{j-i} & , \quad \text{for } i \leq j. \end{cases} \tag{2.15}$$

The second transformation matrix, $\mathsf{M}_p \in \mathbb{R}^{(p+1)\times(p+1)}$, takes the reparameterized power basis to the Bernstein polynomials and is given as

$$\mathsf{M}_p(i,j) = \begin{cases} 0 & , \quad \text{for } i < j, \\ (-1)^{i-j}\binom{p}{j}\binom{p-j}{i-j} & , \quad \text{for } i \geq j. \end{cases} \tag{2.16}$$

In the constructions above, zero-indexed matrices are assumed. As seen by comparing (2.13) and (2.14), a degree p B-spline with coefficients $\mathbf{c}$ calculated as

$$\mathsf{R}_p\mathsf{M}_p\mathbf{c} = \lambda \implies \mathbf{c} = (\mathsf{R}_p\mathsf{M}_p)^{-1}\lambda, \tag{2.17}$$

is equal to the polynomial in (2.13). That is, $f(x; \mathbf{c}, p, \mathbf{t}) = p_p(x)$, for $x \in [a, b]$.

The above procedure for taking a polynomial to a B-spline form can be used in the multivariate case with only a few adjustments. Consider a multivariate polynomial in $\mathbb{P}_{\mathbf{p}}$, here written as the Kronecker products of univariate power basis vectors:

$$p_{\mathbf{p}}(\mathbf{x}) = \lambda^{\mathsf{T}} \bigotimes_{i=1}^{d} \mathbf{x}_{i,p_i}, \tag{2.18}$$

where $\mathbf{x} \in [a_1, b_1] \times \cdots \times [a_d, b_d] = X$. Here $\mathbf{x}_{i,p_i}$ denotes the power basis vector to degree p_i in the variable x_i. Let $\mathsf{T}_i = (\mathsf{R}_{p_i}\mathsf{M}_{p_i})^{-1}$ be the inverse of the transformation matrix that takes a univariate, degree p_i power basis on $[a_i, b_i]$ to a B-spline basis. Then, the $\mathbf{p}$-th degree multivariate B-spline $f(\mathbf{x}; \mathbf{c}, \mathbf{p}, \mathbf{T})$, with knot vectors $\mathbf{T}$ defined as in Property 2.4, and coefficients

$$\mathbf{c} = \left(\bigotimes_{i=1}^{d} \mathsf{T}_i \right) \lambda, \tag{2.19}$$

is equal to (2.18) on X. That is, $f(\mathbf{x}; \mathbf{c}, \mathbf{p}, \mathbf{T}) = p_{\mathbf{p}}(\mathbf{x})$, for $\mathbf{x} \in X$.

2.2.6. Function approximation with B-splines

In this section we briefly introduce the widely used approximation scheme known as cubic spline interpolation. The cubic spline offers great flexibility and may fit difficult shapes without suffering from Runge's phenomenon – the oscillation that may occur in high-degree polynomial interpolation.

Let any function $f : \mathbb{R}^d \to \mathbb{R}$ be sampled on a regular grid to yield M data points $\{\mathbf{x}^i, y^i\}_{i=1}^{M}$, where $f(\mathbf{x}^i) = y^i$. A B-spline that approximates f by interpolating the M data points is obtained by solving the following linear system:

$$\underbrace{\left[\mathbf{B}_p(\mathbf{x}^1)\ \mathbf{B}_p(\mathbf{x}^2)\ \ldots\ \mathbf{B}_p(\mathbf{x}^M)\right]^{\mathsf{T}}}_{\mathsf{B}_c} \mathbf{c} = \mathbf{y} \tag{2.20}$$

where $\mathbf{y} = [y^i]_{i=1}^M$ and $\mathsf{B}_c \in \mathbb{R}^{M\times N}$ is called the *B-spline collocation matrix*. It is customary to select a knot vector that gives a square collocation matrix ($M = N$). An example of such a knot vector is the *free end conditions* knot vector for cubic spline interpolation ($p = 3$):

$$\mathbf{t}_F = \{ \underbrace{x^1, \ldots, x^1}_{p+1 \text{ repititions}}, x^3, \ldots, x^{M-2}, \underbrace{x^M, \ldots, x^M}_{p+1 \text{ repititions}} \}.$$

Notice that the second and second last knot is omitted from $\mathbf{t}_F$ to obtain $M = N$. For square B_c, the conditions under which B_c is invertible are known as the *Schoenberg-Whitney nesting conditions*: $t_i < x^i < t_{i+p+1}$ for $i = 1, 2, \ldots, M$, allowing $x^i = t_i$ only if $t_i = t_{i+p} < t_{i+p+1}$. These conditions are fulfilled for $\mathbf{t} = \mathbf{t}_F$, and the B-spline coefficients can readily be computed by solving $\mathsf{B}_c\mathbf{c} = \mathbf{y}$.

The $M \times M$ linear system in (2.20) can be solved efficiently by a sparse solver on a modern desktop computer for $M \leq 100,000$. This is sufficient to achieve reasonably accurate approximations of most functions of 5 or less variables. The approximation error of a spline can be made arbitrarily small for continuous functions by increasing the sampling density. Furthermore, when the system in (3.20) is augmented with *natural* boundary conditions, the resulting spline is the interpolating $\mathcal{C}^2$-function that minimizes the second derivative.

Example 3. *Consider the Michalewicz function*

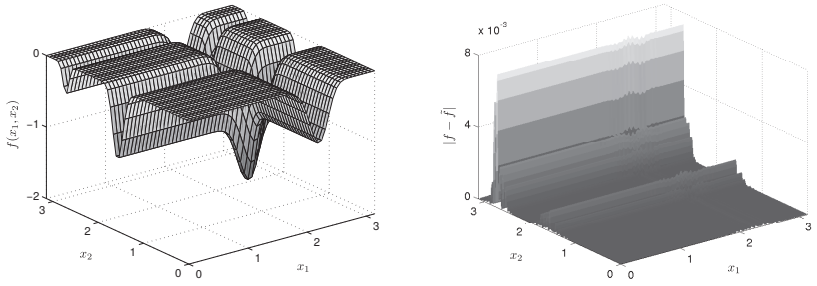
$$f_m(\mathbf{x}) = -\sum_{i=1}^{d} \sin(x_i) \sin^{2m}\left(\frac{ix_i^2}{\pi}\right),$$

on $\mathbf{x} \in X = [0, \pi]^2$. *The function is plotted in Fig. 2.3a for* $d = 2$ *and* $m = 10$. *An interpolating cubic spline* $\tilde{f}_m$ *is constructed by sampling* f_m *on a* 50×50 *grid and solving* (2.20). *The absolute error* $|f_m - \tilde{f}_m|$ *is plotted in Fig. 2.3b.*

2.3. Global optimization with B-spline constraints

Methods for solving $\mathbf{P}'$, such as the spatial branch-and-bound, generate a sequence of lower bounding problems that measure the progress towards closing the optimality gap. A lower bounding problem is required to be a convex relaxation of $\mathbf{P}'$ to assure that it can be solved to (global) optimality and give a valid lower bound. A convex relaxation of $\mathbf{P}'$ is here constructed in two steps: *reformulation* and *convexification*. The reformulation translates the problem to an equivalent form, mapping all local and global optima of the original problem. Reformulation is done only once with the purpose of obtaining a formulation suitable for convexification. The convexification step subsequently replaces all nonconvex constraints with convex outer-estimating constraints, producing a convex lower bounding problem.

A subclass of $\mathbf{P}'$ are problems where the objective and constraint functions are splines. The remainder of this section treats this subclass of problems using the



(a) Michalewicz function (b) Approximation error

Figure 2.3.: In (a), the Michalewicz function and a 50×50 regular grid of equidistant points. In (b), the cubic spline approximation error $|f_m - \tilde{f}_m|$, which coincide with the grooves in the Michalewicz function. The maximum absolute error on X is $||f_m - \tilde{f}_m||_{X,\infty} = 7.6 \cdot 10^{-3}$.

spline theory in Sec. 2.2. The proposed reformulation-convexification is put into an sBB setting in Sec. 2.4.

Reformulation of $\mathbf{P}'$ begins by renaming the constraint and objective functions to $v_{n+i}(\mathbf{x}) = g_i(\mathbf{x})$, for $i = 1, \ldots, m$, and $v_{n+m+1}(\mathbf{x}) = f(\mathbf{x})$. By introducing $m+1$ *auxiliary* variables, $\mathbf{P}'$ is recast into the equivalent MINLP problem:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & x_{n+m+1} \\ \text{subject to} \quad & x_i = v_i(\mathbf{x}), \quad i = n+1, \ldots, n+m+1 \\ & \mathbf{x} \in X \cap \mathbb{Z}^{n_d} \times \mathbb{R}^{n+m+1-n_d} = X^d \end{aligned} \tag{$\mathbf{P}$}$$

where, for $i = n+1, \ldots, n+m+1$, $v_i : \mathbb{R}^n \to \mathbb{R}$ is a B-spline function in a respective spline space $\mathbb{S}^i$ of appropriate degree (larger than zero). If v_i is a polynomial it is brought to B-spline form (2.7) as described in Sec. 2.2.5. The set X is a convex polyhedron that bounds all *complicating* variables (variables that are discrete and/or participate in non-convex constraints). In the reformulation $\mathbf{P}$, X incorporates the right hand side of the inequality constraints in $\mathbf{P}'$, i.e. $X = X' \cup \{x_i \leq 0, i = n+1, \ldots, n+m\}$. In the rest of this paper it is assumed that X includes box constraints on the variables, denoted as $\mathbf{x}^l \leq \mathbf{x} \leq \mathbf{x}^u$.

Remark 2.1. *In the reformulation* $\boldsymbol{P}$ *there is* one *auxiliary variable for each of the objective and constraint functions, and a total of* $m+1$ *auxiliary variables. However, if a function is additively separable it may be advantageous to separate it by introducing additional auxiliary variables. In particular, separation can reduce the dimension of the B-spline(-s) affiliated with the constraint.*

In the convexification step a convex relaxation of $\mathbf{P}$ is obtained by removing the integrality constraints on $\mathbf{x}$ and replacing all nonconvex constraints with convex outer-estimating functions. A nonconvex constraint $v_i(\mathbf{x}) = x_i$ can be relaxed

by introducing a convex function $v_i^l(\mathbf{x})$ and a concave function $v_i^u(\mathbf{x})$, such that $v_i^l(\mathbf{x}) \leq x_i \leq v_i^u(\mathbf{x})$. The constraints $v_i^l(\mathbf{x}) - x_i \leq 0$ and $-v_i^u(\mathbf{x}) + x_i \leq 0$ will then serve as a convex relaxation of $v_i(\mathbf{x}) = x_i$ (cf. Boyd and Vandenberghe 2004).

The relaxation can generally be strengthened by introducing new variables λ, lifting the problem to a higher-dimensional space, and allowing v^l and v^u to be functions of $\mathbf{x}$ and λ. The relaxed problem can be written in the following form:

$$\begin{aligned} \underset{\mathbf{x},\lambda}{\text{minimize}} \quad & x_{n+m+1} \\ \text{subject to} \quad & \tilde{v}_i(\mathbf{x}, \lambda) \leq 0, \quad i = 1, \ldots, \tilde{m}, \\ & \mathbf{x} \in X, \lambda \in \Lambda, \end{aligned} \tag{$\mathbf{R}$}$$

where $\tilde{v}_i$, for $i = 1, \ldots, \tilde{m}$, are convex functions, and λ are auxiliary variables constrained to the convex set Λ. Note that there is no restriction on $\tilde{m}$; a convex relaxation of $\mathbf{P}$ can be obtained by either removing a nonconvex constraint or replacing it with one or more convex constraints. When all $\tilde{v}_i$ are affine, $\mathbf{R}$ is an LP problem. This is the case for the convex relaxations of B-spline constraints presented in the following subsection.

For later reference, let G_P and G_R denote the constraint feasible sets of $\mathbf{P}$ and $\mathbf{R}$ in the variables $\mathbf{x}$, i.e.,

$$\begin{aligned} G_P &= \{\mathbf{x} \mid v_i(\mathbf{x}) = x_i,\ i = 1, \ldots, m\} \text{ and} \\ G_R &= \{\mathbf{x} \mid \tilde{v}_i(\mathbf{x}, \lambda) \leq 0,\ i = 1, \ldots, \tilde{m};\ \lambda \in \Lambda\}. \end{aligned} \tag{2.21}$$

The feasible sets of $\mathbf{x}$ in $\mathbf{P}$ and $\mathbf{R}$ are then $F_P = X^d \cap G_P$ and $F_R = X \cap G_R$, respectively. A convex relaxation of the constraints is said to be *valid* if $G_P \subseteq G_R$. A consequence of a valid constraint relaxation is that $F_P \subseteq F_R$, since $X^d \subseteq X$, and the relaxed problem in $\mathbf{R}$ gives a lower bound on $\mathbf{P}$.

Remark 2.2. *Many different reformulation-convexification approaches can be found in the literature (Sherali and Tuncbilek, 1995; Smith and Pantelides, 1999; Gatzke et al., 2002; Liberti and Pantelides, 2006). One approach is the* reformulation-linearization *technique (RLT) described by Belotti et al. (2009), among others. RLT follows a two-step construction with similarities to the one described above. In particular, the reformulation in RLT expands non-convex functions by replacing all nonlinear operators with auxiliary variables. The auxiliary variables are then linked to the original variables through simple constraints with the replaced (unary or binary) nonlinear operators. The convexification/linearization step relaxes these constraints using known linear, outer-estimating functions of the nonlinear operators, such as McCormick's linear outer-estimators of bilinear terms (McCormick, 1976).*

The RLT reformulation of a nonlinear function is often represented by a binary tree of nonlinear operations. It resembles the reformulation obtained by writing the function in B-spline form. Recalling (2.1) *and* (2.2)*, the B-spline basis functions are constructed from recursive convex combinations. This reformulation can also be represented as a tree structure of convex combinations, with the leaf nodes being the piecewise constant basis functions of degree zero.*

2.3.1. Convex relaxations of B-spline constraints

A B-spline constraint in $\mathbf{P}$ is written as

$$v_i(\mathbf{x}) = x_i, \tag{2.22}$$

where $v_i(\mathbf{x})$ is a B-spline function in the form of (2.7). Since $v_i(\mathbf{x})$ in general is a nonlinear function the equality constraint in (2.22) is nonconvex. Two possible valid, convex relaxations of (2.22) are described next.

Bounding box relaxation a B-spline constraint

From the bounding box property in Corollary 2.1 it follows that

$$c_{\min} = \min\{c_i\}_{i=0}^{N-1} \leq x_i \leq \max\{c_i\}_{i=0}^{N-1} = c_{\max} \tag{2.23}$$

is a valid convex relaxation of (2.22). Eq. (2.23) describes a simple box constraint on x_i. Note that no auxiliary variables are required in this relaxation and (2.23) can therefore serve as a lightweight convex relaxation of the B-spline constraint in (2.22).

Convex hull relaxation of a B-spline constraint

From Lemma 2.1 it follows that $(\mathbf{x}, v_i(\mathbf{x}))$ is in the convex hull of the control points $\{\mathsf{P}_i\}_{i=0}^{N-1}$. In other words, a pair $(\mathbf{x}, x_i)$ can be expressed as a convex combination of the control points, which makes the B-spline surface a subset of the possible convex combinations. A convex relaxation can be made by allowing *all* possible convex combinations. By introducing N new variables $\{\lambda_i\}_{i=0}^{N-1}$, a lifted polyhedral relaxation is obtained as:

$$\sum_{i=0}^{N-1} \mathsf{P}_i \lambda_i = \begin{bmatrix} \mathbf{x} \\ x_i \end{bmatrix}, \quad \sum_{i=0}^{N-1} \lambda_i = 1, \quad \text{and } \lambda_i \geq 0 \text{ for } i = 0, \ldots, N-1. \tag{2.24}$$

The convex set $\mathbb{C}$ in Lemma 2.1 is expressed by the equations in (2.24). Since $\mathbb{D} \subseteq \mathbb{C}$ it is a valid convex relaxation of (2.22). Obviously, this is the tightest convex relaxation that can be obtained from the control points alone.

As shown in Appendix 2.B, the relaxation in (2.24) of a B-spline representing a bilinear term $x_1 x_2$ is equivalent to the McCormick relaxation.

Remark 2.3. *The convex relaxation in* (2.24) *may be computationally heavy since it introduces a total of N auxiliary variables to the relaxed problem. Recalling that N grows exponentially with the dimension of the B-spline, see* (2.6)*, it is clear that the relaxation is unsuited for high-dimensional B-splines. For example, with d dimensions and* 10 *basis functions in each dimension $N = 10^d$. Three remedies to lessen the computation load are suggested below:*

1. *As pointed out in Remark 2.1, if a constraint is additively separable it can be separated and represented by several B-splines. This can reduce the dimension of the related B-splines and the total number of auxiliary variables needed to relax the constraint.*

2. *When N is prohibitively large the bounding box relaxation in* (2.23) *can be used at the cost of a looser relaxation.*

3. *The number of auxiliary variables in* (2.24) *can be reduced by realizing that many of the points in $\{\mathsf{P}_i\}$ may be in the interior of the convex hull and are thus redundant. The latter approach may be implemented by computing the convex hull of the points $\{\mathsf{P}_i\}$ and then introducing auxiliary variables as in* (2.24) *only for the points that participate as vertices in the convex hull. Since it is expensive to calculate the convex hull of a large number of points (especially in higher dimensions) an alternative approach is to apply the Akl-Toussaint heuristic (Akl and Toussaint, 1978) to quickly exclude points that are in the interior of the convex hull.*

Example 4 (Convex relaxations of monomials of odd degree)**.** *In this example a B-spline is configured to represent a monomial of odd degree $2k+1$ on the interval $x \in [a, b]$. The convex relaxation of the B-spline is then compared with two other relaxation methods: the general αBB relaxation (Adjiman et al., 1997) which can be applied to any twice differentiable functions, and the (tight) convex envelope of odd-degree monomials by Liberti and Pantelides (2003). For simplicity, the monomial with $k = 1$ on the interval $[-1, 1]$ is used in the example.*

First, consider the monomial and its convex envelope: $l_k(x) \le x^{2k+1} \le u_k(x)$. Liberti and Pantelides show that the convex envelope is given by the functions:

$$l_k(x) = \begin{cases} \left\{ \begin{array}{lll} a^{2k+1}(1 + R_k(x/a - 1)) & , & \text{if } x < c \\ x^{2k+1} & , & \text{if } x \ge c \end{array} \right\} & \text{, if } c < b \\ a^{2k+1} + \frac{b^{2k+1} - a^{2k+1}}{b-a}(x - a) & & \text{, otherwise} \end{cases}$$

$$u_k(x) = \begin{cases} \left\{ \begin{array}{lll} x^{2k+1} & , & \text{if } x \le d \\ b^{2k+1}(1 + R_k(x/b - 1)) & , & \text{if } x > d \end{array} \right\} & \text{, if } d > a \\ a^{2k+1} + \frac{b^{2k+1} - a^{2k+1}}{b-a}(x - a) & & \text{, otherwise} \end{cases}$$

where $R_k \equiv (r_k^{2k+1} - 1)/(r_k - 1)$, $c = r_k a$, $d = r_k b$, and r_k is a root of a $(2k-1)$-degree polynomial. For the case $k = 1$, they provide $r_k = r_1 = -0.5$. Next, consider the relaxation $\mathcal{L}_k(x) \le x^{2k+1} \le \mathcal{U}_k(x)$ of the αBB method. The upper and lower bounding functions are on the form:

$$\mathcal{L}_k(x) = x^{2k+1} + \alpha_k(x - a)(x - b),$$
$$\mathcal{U}_k(x) = x^{2k+1} - \beta_k(x - a)(x - b),$$

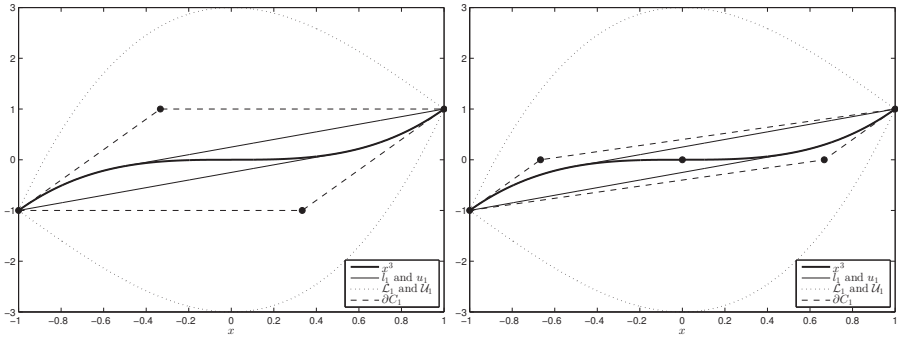
where α_k is a positive constant that is sufficiently large to render the second derivative $d^2\mathcal{L}/dx^2$ positive, and β_k is sufficiently large to render $d^2\mathcal{U}/dx^2$ negative, for all $x \in [a, b]$. For the monomial x^{2k+1}, $\alpha_k = k(2k+1)|a|^{2k+1}$ and $\beta_k = k(2k+1)b^{2k+1}$.

Last, the B-spline relaxation is constructed by writing the monomial as a $p = 2k + 1$ *degree B-spline using* (2.18)*. For* $k = 1$ *and* $x \in [-1, 1]$*, the resulting B-spline has the following knots, knot averages, and coefficients:*

$$\begin{aligned}\mathbf{t} &= [-1, -1, -1, -1, 1, 1, 1, 1],\\ \mu &= [-1, -0.333, 0.333, 1]^T, \text{ and}\\ \mathbf{c} &= [-1, 1, -1, 1]^T.\end{aligned}$$

The convex hull of the control points, $C_k = conv(\{(\mu_i, c_i)\}_{i=0}^{p})$*, can then be expressed as in* (2.24)*. In Figure 2.4 the boundary of* C_k*, denoted* ∂C_k*, is plotted with the other relaxations. The figure also shows the improvement of the B-spline relaxation when inserting a single knot at* $x = 0$*, giving*

$$\begin{aligned}\mathbf{t} &= [-1, -1, -1, -1, 0, 1, 1, 1, 1],\\ \mu &= [-1, -0.667, 0, 0.667, 1]^T, \text{ and}\\ \mathbf{c} &= [-1, 0, 0, 0, 1]^T.\end{aligned}$$



(a) (b)

Figure 2.4.: Comparison of convex relaxations of the monomial x^3 for $x \in [-1, 1]$. Control points are marked with filled circles. The B-spline relaxation with a minimal number of knots is shown in (a). In (b) a single knot is inserted at $x = 0$ to improve the relaxation.

Example 5 (Convex relaxation of the six-hump camelback function)**.** *This example studies how the number of variables in a B-spline relaxation, which can be increased by knot insertion, is related to the tightness of the relaxation. To measure this relation the minimization of the six-hump camelback function is considered:*

$$\begin{aligned}\underset{\mathbf{x}}{\text{minimize}}\ f(\mathbf{x}) &= (4 - 2.1x_1^2 + \frac{1}{3}x_1^4)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2\\ &= 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4,\end{aligned} \tag{2.25}$$

on the domain $\mathbf{x} \in [-3,3] \times [-3,3]$. *The problem has a total of six local minima, two of them being the global minima:* $\mathbf{x}^* = (-0.0898, 0.7126)$ *and* $\mathbf{x}^* = (0.0898, -0.7126)$, *at which* $f^* = f(\mathbf{x}^*) = -1.0316$.

With degrees $p_1 = 6$ *and* $p_2 = 4$, *a bivariate B-spline representing* $f(\mathbf{x})$ *is constructed as outlined in Sec. 2.2.5. The resulting B-spline has a total of* $(p_1+1)(p_2+1) = 35$ *control points (and basis functions). Using the convex hull relaxation in* (2.24) *the problem in* (2.25) *is relaxed to an LP problem with* $\tilde{n} = 38$ *variables and* $\tilde{m} = 4$ *constraints. The 38 variables are: the original variables* x_1 *and* x_2*; the auxiliary variable* x_3 *resulting from the reformulation* $x_3 = f(x_1, x_2)$*; and, the 35 auxiliary variables related to the control points. The 4 constraints are given in* (2.24). *The relaxation is improved by refining the regular knot vectors with Alg. 2. The desired number of knots,* r_1 *and* r_2, *are calculated as:* $r_1 = 2(p_1 + 1) + s$ *and* $r_2 = 2(p_2 + 1) + s$, *where* s *is the number of knots to insert into the knot vectors. With* $s = 0$, *zero knots are inserted. With* $s = 1$, *one knot is inserted at the midpoint of the largest knot span of each knot vector. And so on.*

The B-spline relaxation is compared with the relaxation that results from a symbolic reformulation. The problem is brought to standard form *by a binary tree function expansion (Smith and Pantelides, 1999), giving:*

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & f(\mathbf{w}) = 4w_3 - 2.1w_4 + \frac{1}{3}w_5 + w_6 - 4w_7 + 4w_8 \\ \text{subject to} \quad & w_3 = w_1^2, w_4 = w_3^2, w_5 = w_3 w_4 \\ & w_6 = w_1 w_2, w_7 = w_2^2, w_8 = w_7^2 \\ & w_1 \in [-3,3], w_2 \in [-3,3] \end{aligned} \tag{2.26}$$

The variable bounds on $w_3, \ldots, w_8$ *are inferred from the bounds on* $w_1 = x_1$ *and* $w_2 = x_2$ *during the reformulation procedure – no further domain reduction is applied. The reformulated problem in* (2.26) *is relaxed to a convex NLP using McCormick's relaxation of bilinear terms and the nonlinear convex envelope of univariate quadratic terms.*

The relaxations are compared by computing the optimality gap $f^* - \bar{f}$, *where* $\bar{f}$ *is the optimal objective function value of the relaxed problem. The results are plotted in Figure 2.5 and listed in Table 2.1 for various knot refinement values* s. *As expected the B-spline relaxation improves as the number of knots is increased, but at the cost of introducing more variables.*

Several observations can be made from the reported results for problem (2.25). *First of all, it is clear that with no knot refinement the B-spline relaxation gives a larger optimality gap than the standard form relaxation. However, it surpasses the latter when 4 or more knots are inserted. Further knot refinement yields diminishing returns in closing the optimality gap. This is not surprising since the effect of knot insertion becomes increasingly local as more knots are added: i.e., it will only affect neighbours in a condensing set of control points. This is illustrated in Fig. 2.5 where the reader can observe a constant optimality gap for* $s = 14$ *to* $s = 20$. *This happens because the additional knots are being placed too far away from the global optimum to affect the lower bound.*

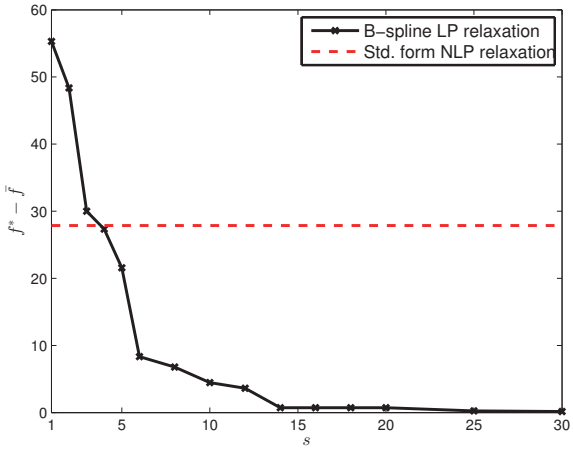


Figure 2.5.: Comparison of different convex relaxations of the six-hump camelback function.

Table 2.1.: Comparison of standard form relaxation and refined B-spline relaxations

Relaxation	Std. form	Refined B-spline relaxation						
		$s=0$	$s=1$	$s=2$	$s=5$	$s=10$	$s=20$	$s=50$
$\tilde{n}$	8	38	51	66	123	258	678	3138
$\tilde{m}$	16	4	4	4	4	4	4	4
N	-	35	48	63	120	255	675	3135
$f^* - \bar{f}$	27.87	607.2	55.29	48.36	21.87	4.46	0.73	0.08

$\tilde{n}$: number of variables, $\tilde{m}$: number of constraints, N: number of control points, $f^* - \bar{f}$: optimality gap

In unconstrained programming successive knot refinement (increasing s) is sufficient to close the optimality gap. This is a direct consequence of the fact that the control structure will converge to the B-spline as the largest knot interval goes towards zero. Thus, in the limit the control points with the lowest B-spline coefficient, i.e. $\{\mathsf{P}_i \mid c_i \leq c_j, \forall i, j = 0, \ldots, N-1\}$, will coincide with the global minima $\{(\mathbf{x}^*, f(\mathbf{x}^*))\}$. Unfortunately, as Table 2.1 shows, the number of variables quickly rises as knots are inserted, and, it is questionable if knot refinement alone is an efficient way to close the gap. In constrained nonlinear programming knot refinement alone is not enough to close the optimality gap and spatial branching is required.

2.4. A spatial branch-and-bound for spline-constrained MINLP problems

An algorithm that solves problem $\mathbf{P}$ using the reformulation-convexification in Sec. 2.3 is presented here. The algorithm is inspired by the prototypical spatial branch-and-bound framework by Belotti et al. (2009), which was implemented and presented as the global MINLP solver COUENNE in the same paper. The algorithmic framework is reproduced in Algorithm 3. Since the spatial branch-and-bound and its components have been examined in a great number of publications (see Belotti et al. 2009; Smith and Pantelides 1999 and the references therein), it is briefly described here to introduce the necessary terminology.

The sBB algorithm maintains a list of *nodes* that represents the *sBB tree*. Initially, the list only contains the root node $\mathbf{P}$, also denoted $\mathbf{P}_0$. At iteration k a problem $\mathbf{P}_k$ is selected for processing. The processing begins with an optional bounds tightening procedure that attempts reduce the variable ranges. Next, a convex, lower bounding problem $\mathbf{R}_k$ is generated from $\mathbf{P}_k$ and solved. The optimal solution of $\mathbf{R}_k$ is denoted $\bar{\mathbf{x}}^k$ at which $\bar{z}_k$ is the objective function value. The current upper bound on $\mathbf{P}$, denoted z^u, is updated if $\bar{\mathbf{x}}^k$ is a feasible solution to $\mathbf{P}_k$ and $\bar{z}_k < z^u$. $\mathbf{P}_k$ is then *fathomed* (deleted) if: the lower bound has surpassed the upper bound ($\bar{z}_k > z^u$); or the node has converged ($z^u - \bar{z}_k < \epsilon$); or $\mathbf{R}_k$ is infeasible. Otherwise, the node requires further processing. $\mathbf{P}_k$ can then be locally solved to possibly improve the upper bound z^u. Finally, a branching procedure subdivides $\mathbf{P}_k$ into two subproblems $\mathbf{P}_{k-}$ and $\mathbf{P}_{k+}$ which are added to the list for later processing. The algorithm continues in the same fashion by selecting a new problem for processing until the list is empty. Upon termination the global optimum can be certified with ϵ-precision.

Algorithm 3 has been implemented in CENSO (Convex ENvelopes for Spline Optimization), a publicly available C++ code for global optimization of spline-constrained MINLP problems (Grimstad et al., 2015a). CENSO contains a simple sBB implementation which make use of three different solvers: GUROBI (Gurobi Optimization, Inc., 2014) to solve lower bounding LP problems; and, IPOPT (Wächter and Biegler, 2006) and BONMIN (Bonami et al., 2008) to solve upper bounding NLP and MINLP problems, respectively. All spline related computations are handled by SPLINTER (Grimstad et al., 2015b), a spline library which implements the theory presented in Sec. 2.2 using the linear algebra library EIGEN (Guennebaud et al., 2010). The library is used in four steps of the algorithm to:

- reformulate the problem to the form in $\mathbf{P}$,
- generate the lower bounding problems $\mathbf{R}_k$,
- perform the subdivision that creates subproblems $\mathbf{P}_{k-}$ and $\mathbf{P}_{k+}$, and
- evaluate the splines and their derivatives when solving the upper bounding problems $\mathbf{P}_k$.

Algorithm 3 Spatial branch-and-bound

Require: Problem $\mathbf{P}$
Ensure: The value z_{opt} of an optimal solution of $\mathbf{P}$
  Initialize list of subproblems: $\mathcal{L} \leftarrow \{\mathbf{P}\}$
  Set initial upper bound on $\mathbf{P}$: $z^u \leftarrow \infty$
  while $\mathcal{L} \neq \emptyset$ **do**
    Select and remove problem $\mathbf{P}_k$ from $\mathcal{L}$ ▷ (Sec. 2.4.1)
    (Optional) Apply bounds tightening to $\mathbf{P}_k$ ▷ (Sec. 2.4.4)
    if bounds tightening proved $\mathbf{P}_k$ infeasible **then**
      continue (subproblem is fathomed)
    end if
    Generate a convex relaxation $\mathbf{R}_k$ of $\mathbf{P}_k$ ▷ (Sec. 2.3)
    Solve $\mathbf{R}_k$; let $\bar{\mathbf{x}}^k$ be an optimum and $\bar{z}_k$ its objective value
    if $\bar{\mathbf{x}}^k$ is feasible for $\mathbf{P}_k$ **then**
      Let $z^u \leftarrow \min\{z^u, \bar{z}_k\}$
    end if
    if $\bar{z}_k \geq z^u$ or $z^u - \bar{z}_k \leq \epsilon$ or $\mathbf{R}_k$ infeasible **then**
      continue (subproblem is fathomed)
    else
      (Optional) Solve $\mathbf{P}_k$ for possible $\hat{z}_k$ ▷ (Sec. 2.4.5)
      $z^u \leftarrow \min\{z^u, \hat{z}_k\}$
      Choose a branching variable x_i and a branching point x_i^b ▷ (Sec. 2.4.2)
      Create subproblems: $\mathbf{P}_{k-}$ $(x_i \leq x_i^b)$ and $\mathbf{P}_{k+}$ $(x_i \geq x_i^b)$ ▷ (Sec. 2.4.3)
      $\mathcal{L} \leftarrow \mathcal{L} \cup \{\mathbf{P}_{k-}, \mathbf{P}_{k+}\}$
    end if
  end while
  Output $z_{opt} = z^u$

The two first steps above have already been discussed in Sec. 2.3. A description of the two last steps can be found in the subsequent Sec. 2.4.3 and Sec. 2.4.5, respectively. Note that these, and the other steps of algorithm, are labelled in Alg. 3 with the sections that treat them.

Remark 2.4 (Default relaxation)**.** *The default convex relaxation in* CENSO *is the convex hull relaxation in Sec. 2.3.1.*

Remark 2.5. *The B-spline module does not require any user input unless knot refinement is turned on (the desired number of knots in Alg. 2 is then required).*

2.4.1. Selection operation

A subproblem is selected with a *best-bound-first* policy:

$$\text{Select a subproblem } \mathbf{P}_k \text{ with } k \in \underset{j:\mathbf{P}_j \in \mathcal{L}}{\arg\min}\{\bar{z}_j\}. \tag{2.27}$$

With this selection policy the subproblem with the lowest lower bound is selected for processing in the hope of maximizing the reduction of the optimality gap. Due to its simplicity and *bound improving* property it is a popular selection policy in sBB algorithms.

2.4.2. Branching procedure

Branching-variable selection. Branching variables are selected by prioritizing integer variables over continuous variables. When all integer variables have been fixed spatial branching begins. Among all continuous variables that participate in non-convex constraints, the variable with the largest bound range is selected for branching. The branching rule can be expressed as:

$$\text{Select a branching variable } x_i \text{ with } i = \arg\max_{i \in I_c} \{x_i^u - x_i^l\}, \tag{2.28}$$

where I_c denotes the index set of continuous variables that participate in non-convex constraints (and require branching). This simple branching rule is motivated by the comments to the B-spline control structure convergence given in Sec. 2.2.3.

Branching-point selection. After a variable x_i has been selected for branching one has to decide a branching point $x_i^b \in (x_i^l, x_i^u)$. To improve the lower bounds on the subproblems it is desirable to branch at a point that makes $\bar{x}^k$ infeasible. This is usually done by branching at a point near $\bar{x}_i^k$. In some cases this point may be close to the variable bounds and branching is likely to create a very easy subproblem and a very hard one. To prevent this and keep the BB tree balanced, the branching point may be selected as a convex combination of $\bar{x}_i^k$ and the midpoint $x_i^m = (x_i^l + x_i^u)/2$. The following strategy is adopted from (Belotti et al., 2009) and guarantees a minimum distance from the variable bounds:

$$x_i^b = \max\left\{x_i^l + b, \min\{x_i^u - b, \alpha\bar{x}_i^k + (1+\alpha)x_i^m\}\right\}, \tag{2.29}$$

where $0 < \alpha < 1$ and $b = \beta(x_i^u - x_i^l)/2$ for $0 < \beta < 1/2$. In CENSO, the default value for α and β is 0.25 and 0.2, respectively.

Branching at a point close to $\bar{x}_i$ is advantageous when x_i participates in a B-spline constraint. In the subdivision of a B-spline the control structure will interpolate the constraint at the branching point. The relaxation will also improve in the neighbourhood of x_i^b due to the local support of B-splines. If knot refinement is applied the relaxation tighten at points away from x_i^b. These notions are described in the following section on B-spline subdivision.

2.4.3. B-spline subdivision

This section will shed light on how B-splines are affected by subdivision and how they should be treated to ensure properties required for a consistent bounding

procedure. A consistent bounding procedure requires a non-decreasing sequence of lower bounds. This translates into a requirement on the convex relaxations: they must get tighter as subdivision shrinks the search space. Two B-spline relaxations were introduced in Sec. 2.3.1. This section will focus on the convex hull based relaxation in Sec. 2.3.1, but the arguments are valid for the bounding box relaxation as well.

Throughout this section all steps are illustrated on a quadratic, univariate B-spline with knots $\mathbf{t} = [0\ 0\ 0\ 1\ 1\ 1]$ and coefficients $\mathbf{c} = [1\ -1\ 1]$, which is a B-spline expansion of $4x^2 - 4x + 1$ on $x \in [0, 1]$. The B-spline, with its control points and convex hull relaxation, is illustrated in Fig. 2.6. Bisection at $x^b = 0.5$ gives two subproblems: *Child A* with $x \in [0, x^b]$ and *Child B* with $x \in [x^b, 1]$.

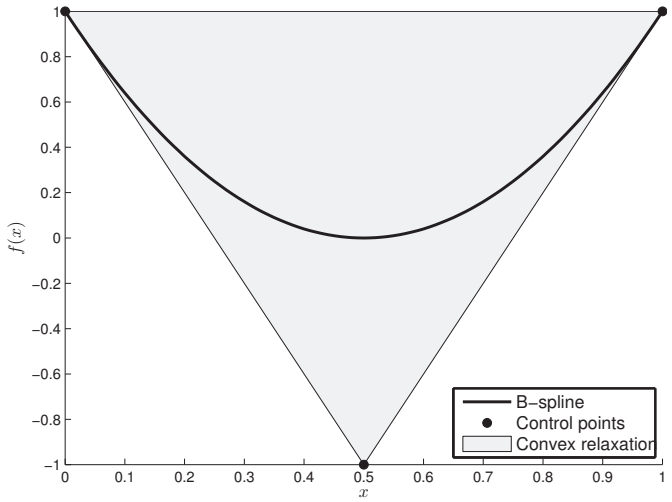
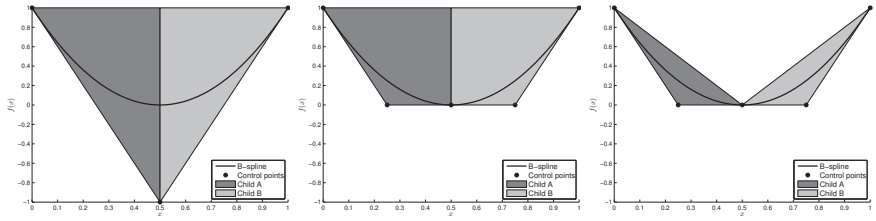


Figure 2.6.: A quadratic B-spline representing $4x^2 - 4x + 1$, its control points and convex hull relaxation. It can easily be verified that the knot averages, the x-coordinates of the control points, are $\mathbf{t}^* = [0\ 0.5\ 1]$.

To make a convex hull relaxation tighter the control points must either be moved (to lie closer to the relaxed function) or simply removed. As stated in Property 2.2, a B-spline basis function $B_{j,p,\mathbf{t}}(x)$ evaluates to zero if x is outside the interval $[t_j, t_{j+p+1})$. If a subproblem k is created such that $X_k \cap [t_j, t_{j+p+1}) = \emptyset$, then $B_{j,p,\mathbf{t}}(x)$ is unsupported inside the search space of the problem and can be ignored during function evaluation. The control points associated with unsupported basis functions are not part of the convex combination that creates the function surface. Thus, they are not needed in the relaxation and can be removed. As the search space is partitioned the convex relaxations will shrink until a minimum number of supported basis functions is reached (a B-spline of degree p will have $p + 1$ supported basis functions at any given point in its domain). Because of this, it is not possible to achieve complete convergence of the convex relaxations by removal of unsupported basis functions alone. This is illustrated in Figure 2.6, where all basis functions

are supported over the entire domain (this is the case with all polynomials created by the procedure given in section 2.2.5). The result of subdivision is illustrated in Figure 2.7a; the union of the child node relaxations is equal to the parent node relaxation.

Knot insertion can be used to ensure tighter relaxations. Inserting knots creates new basis functions with new control points. It will also reduce the support for existing basis functions and draw existing control points closer to the function surface according to (2.11). Figure 2.7b illustrates the effect of inserting additional knots at the bisection point. The adjusted control points are closer to the curve, and the convex relaxation is improved. In addition, since the new basis functions have reduced support, they become susceptible to the removal argument in the previous paragraph. This is illustrated in Figure 2.7c, where, after increasing the number of basis functions from three to six, the first three basis functions are supported in Child A and unsupported in Child B, and the other way around for the last three basis functions. When this is accounted for, the relaxations are improved yet again.



(a) (b) (c)

Figure 2.7.: The quadratic B-spline is divided into two subproblems A and B at the bisection point $x^b = 0.5$. (a) Each subproblem holds a copy of the parent B-spline, but with a reduced domain. The union of the two convex relaxations is identical to the original convex relaxation, because all basis functions are supported in both subproblems. (b) Three knots are inserted at the bisection point $x^b = 0.5$. The knot vectors become $\mathbf{t}_A = \mathbf{t}_B = [0\ 0\ 0\ 0.5\ 0.5\ 0.5\ 1\ 1\ 1]$. (c) After unsupported basis functions and corresponding control points are removed, the knot vectors of the two subproblems become $\mathbf{t}_A = [0\ 0\ 0\ 0.5\ 0.5\ 0.5]$ and $\mathbf{t}_B = [0.5\ 0.5\ 0.5\ 1\ 1\ 1]$.

Knot insertion is used to guarantee improvement of the convex relaxations by subdivision. The subdivision procedure presented here inserts knots at the branching point in order to maintain regular knot sequences for all subspaces. With regular knot vectors, the control points associated with the first and last basis function will coincide with the function surface, which makes the convex relaxation as tight as possible at the subspace boundaries. Additional knots can be inserted in between the remaining knots to improve the convex relaxation even further. Inserting additional knots will increase the computational load because it will increase the number of variables in the lower bounding problem. It also takes a small computational effort to produce the new control points. The number of additional knots is therefore a trade-off between the quality of the relaxation and the computational time, which in a branch and bound setting translates to the number of nodes processed

and the time spent processing each individual node. Figure 2.8 illustrates the effect of inserting a knot at the midpoint of each knot vector. The control points are drawn closer to the function and the relaxation is improved. It is however worth noting that it is only the part of the relaxation that lies below the function that is affected by knot refinement in this particular case. The reason being that the part above the function is governed by the control points that already interpolate the function. The region above the function can only be dealt with by further subdivision (followed by knot insertion).

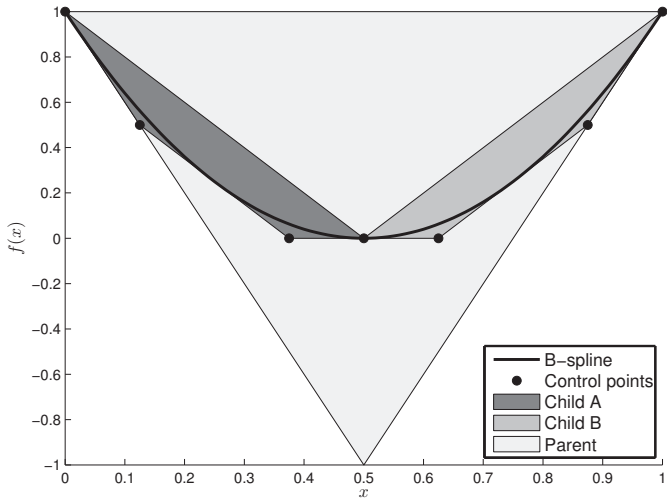


Figure 2.8.: Convex relaxations after knot refinement. Knots added at $x = 0.25$ and $x = 0.75$ so that $\mathbf{t}_A = [0\ 0\ 0\ 0.25\ 0.5\ 0.5\ 0.5]$ and $\mathbf{t}_B = [0.5\ 0.5\ 0.5\ 0.75\ 1\ 1\ 1]$. The convex hull relaxation of the parent problem is included for comparison.

To summarize, B-splines affected by the subdivision of branching will have knots inserted at the branching point until the knot multiplicity equals the spline order. Each subproblem then receives a "copy" of the B-splines and removes the basis functions that are unsupported inside the subspace. The knot sequences can then be further refined by inserting additional knots to maintain a minimum number of basis functions inside each subspace. This B-spline subdivision procedure is described by Algorithm 4.

2.4.4. Bounds tightening

Bounds tightening (BT) are techniques that reduce the variable intervals $[\mathbf{x}^l, \mathbf{x}^u]$ of a problem without removing its optimal point. A BT method that does this is said to produce *valid inequalities*. BT may expedite the BB search by shrinking the feasible set of the subproblems and, consequently, strengthening their convex relaxations. It may also prove a subproblem infeasible so that it can be fathomed.

Algorithm 4 B-spline subdivision

Require: Branching point x_i^b, spline degree p_i, knot vector $\mathbf{t}_i$

1: Split range of variable $x_i \in \left[x_i^l, x_i^u\right]$ at x_i^b to produce $X_A = \left[x_i^l, x_i^b\right]$ and $X_B = \left[x_i^b, x_i^u\right]$
2: **for** subproblem A and B **do**
3: **repeat**
4: Insert x_i^b in $\mathbf{t}_i$ using knot insertion procedure in Algorithm 1
5: **until** x_i^b occurs $p_i + 1$ times in $\mathbf{t}_i$
6: Remove unsupported basis functions
7: (Optional) Refine knot vectors using Algorithm 2
8: **end for**

CENSO employs a simple BT method known as *reduced-cost* bounds tightening (RCBT). It was originally introduced for MILP problems, but is applicable also to MINLPs (Ryoo and Sahinidis, 1996; Belotti et al., 2009). RCBT utilizes the following optimality argument, relating the reduced costs of a relaxed problem $\mathbf{R}_k$ to the upper and lower bound on $\mathbf{P}_k$, to tighten the variable bounds.

$$\begin{aligned} \bar{x}_i = x_i^l \text{ and } d_i > 0 \Rightarrow x_i \leq x_i^l + (\hat{z} - \bar{z})/d_i \\ \bar{x}_i = x_i^u \text{ and } d_i > 0 \Rightarrow x_i \geq x_i^u + (\hat{z} - \bar{z})/d_i \end{aligned} \tag{2.30}$$

In addition to RCBT, CENSO use another type of computationally cheap techniques known as *feasibility-based* BT (FBBT) (Hansen, 2003; Carrizosa et al., 2004; Messine, 2004; Belotti et al., 2009; Hooker, 2000). FBBT is a form of interval analysis where the variable intervals are propagated through the constraints. In CENSO, both linear and B-spline constraints are used in FBBT with the purpose of shrinking the variable intervals as much as possible.

Remark 2.6. *RCBT seldom produces deep cuts and FBBT will not always find the minimal variable intervals (since it is performed on one variable interval and one constraints at a time). More efficient BT methods can be found in the literature, such as the* optimality-based *BT (Zamora and Grossmann, 1999; Sahinidis, 2003). This particular technique is, however, more computationally demanding than the BT techniques used by* CENSO*. Also worth noting are the box and hull consistency BT techniques developed by Nataraj and Arounassalame (2011). These methods exploit the properties of the Bernstein coefficients related to a polynomial constraint to tighten the variable bounds.*

2.4.5. Upper bounding problems

If an optimal solution $\bar{\mathbf{x}}^k$ to $\mathbf{R}_k$ is infeasible for $\mathbf{P}_k$, a local solver can be initiated to search for a feasible solution to $\mathbf{P}_k$. The motivation for doing a local search is that finding a good upper bound on $\mathbf{P}$ early in the BB tree may result in fathoming of branches – and hence reduce the tree size. In CENSO a local search is performed

at every node down to depth $L_{\text{NLP}} = 2$, and then at every Lth depth after that. When the subproblem has free integer variables the convex MINLP solver BONMIN (Bonami et al., 2008) is used to perform a heuristical search. The maximum number of iterations is then restricted to avoid an exhaustive search. When all integer variables have been fixed to integer values the local search is performed by the NLP solver IPOPT (Wächter and Biegler, 2006).

As discussed in Sec. 2.2, the B-spline functions in $\mathbf{P}$, and their derivatives, can be evaluated efficiently. A small issue that should be considered however, is that some loss of smoothness can occur. E.g., a B-spline with degree $p \leq 2$ will have discontinuous second derivatives at the knots, even when all knots are distinct. If $p = 1$ its first derivative is discontinuous at the knots. This may fool the NLP solver to search in poor directions, degrading the performance of the local search.

2.4.6. Convergence of the sBB algorithm

An (infinite) sBB algorithm is said to be convergent if $\lim_{k\to\infty} |z_k^u - z_k^l| = 0$, where z_k^u and z_k^l is the upper and lower bound on $\mathbf{P}$ at iteration k. According to Horst and Tuy (1996) the convergence can be asserted by checking the following criteria:

1. The selection operation is *bound improving*.

2. The bounding operation is *consistent*. That is, at any step in the algorithm, a non-decreasing sequence of lower bounds can be generated for any unfathomed partition. Consistency of a partition i is implied by convergence of the lower bound to the upper bound under further partitioning: $\lim_{q\to\infty} |\hat{z}_{i_q} - \bar{z}_{i_q}| = 0$ (Ryoo and Sahinidis, 1996).

3. The bounds tightening must produce *valid inequalities*. That is, it may only eliminate feasible solutions which are not optimal.

The selection operation in Sec. 2.4.1 is bound improving by definition. The BT methods in Sec. 2.4.4 produce valid inequalities. Thus, requirement 1 and 3 are fulfilled. Requirement 2, on the consistency of the bounding operation follows from:

1. The branching procedure in Sec. 2.4.2 guarantees that $X_{i_q} \subset X_i$ for all q, where X_{i_q} are subsequent partitions of X_i.

2. The convex relaxation of the B-spline is valid according to Lemma 2.1. Together with the relaxation of the integer constraints it follows that $F_{P_k} = G_{P_k} \cap X_{P_k}^d \subseteq G_{R_k} \cap X_{R_k} = F_{R_k}$ (see Sec. 2.3).

3. Finally, it follows from the convergence of the control structure to the spline (2.11), and the subdivision procedure in Algorithm 4, that $\mathcal{F}_{R_{k_q}} \subseteq \mathcal{F}_{R_k}$ for all q descendants of $\mathbf{R}_k$. Furthermore, $\hat{z}_k = \bar{z}_k$ at a terminal node $\mathbf{P}_k$ with $x_j^u = x_j^l$ for all variables x_j participating in spline constraints.

2.5. Computational results

A computational study comparing CENSO with three state-of-the-art global solvers is presented here. The purpose of the study is to investigate the performance and display the versatility of a spline-based branch-and-bound algorithm. The following three global solvers were selected for comparison:

- BARON version 12.7.3 (Tawarmalani and Sahinidis, 2005),
- COUENNE version 0.4 (Belotti et al., 2009), and
- LINDO (LINDOGLOBAL) version 8.0 (Lin and Schrage, 2009).

The selected solvers handle the general class of non-convex MINLPs with few restrictions on the nonlinear terms in the problems. Common for all three solvers is that they are BB-based reformulation-convexification algorithms that rely on symbolic operations on the problem. They employ advanced linearization and branching techniques, as well as various domain reduction techniques, to accelerate convergence. The three solvers were run from the General Algebraic Modelling System, GAMS, of version 24.2.1 (GAMS Development Corporation, 2013). (CENSO does not presently have an interface to GAMS.)

All solvers were run with an absolute ϵ-convergence termination criteria of $\epsilon = 1 \cdot 10^{-6}$ and a feasibility tolerance of $1 \cdot 10^{-6}$. Otherwise all settings were left on default values. The tests were run on a machine equipped with a 2.7 GHz processor and 8 GB of RAM memory.

2.5.1. Small-sized polynomial problems

The solvers were benchmarked on a set of non-convex, polynomially constrained NLP problems. The test problems are given in Appendix 2.A and summarized in Table 2.2. The same problem set was used to test the Bernstein branch-and-prune algorithm in (Nataraj and Arounassalame, 2011). This specialized set of relatively small continuous problems was deemed appropriate for several reasons: 1) polynomially constrained problems are easily reformulated to spline constrained problems; 2) problems with few constraints were considered because CENSO does not exploit the domain reduction, constraint and variable elimination techniques made possible by an algebraic modelling system such as GAMS. Thus, the benchmark should be interpreted only as an indicator of the performance one could expect from CENSO on the aforementioned type of problems.

Several specialized solvers exist for the special class of polynomially constrained problems, e.g. GloptiPoly (Lasserre, 2001) and SparsePOP (Waki et al., 2006). We included SparsePOP in our study to assess how the speed of a specialized solver would compare to the aforementioned solvers. In particular, we ran SparsePOP version 300, configured to use SeDuMi version 1.3 to solve the resulting semidefinite programming problems (Sturm, 1999). Note that GloptiPoly was not included in

this study due to its inferior performance on optimization problems with multivariate polynomials (see Henrion et al. 2009).

Table 2.2.: Overview of the non-convex NLP test problems.

Problem	n	m	N_p	$\bar{p}$	$\bar{d}$
P1	2	2	2	4	1
P2	2	2	3	3	2
P3	2	2	2	3	2
P4	3	3	1	2	3
P5	3	4	4	3	3
P6	4	4	2	3	4
P7	4	7	1	8	3
P8	4	7	4	5	4
P9	5	1	1	2	5
P10	6	2	1	2	6
P11	6	6	3	2	6
P12	7	6	4	2	7
P13	7	11	10	8	7

n: number of variables, m: number of constraints, N_p: number of polynomial functions, $\bar{p}$: highest polynomial order, $\bar{d}$: highest polynomial dimension
($\bar{p}$ and $\bar{d}$ may be related to different polynomials)

The computational results are recorded in Table 2.3. The reported numbers are the solvers' CPU times in milliseconds, averaged over 10 runs for each problem. The number of iterations is not reported since some of the solvers run subroutines (e.g. preprocessing and bounds tightening) that does a lot of work without incrementing the branch-and-bound iteration counter.

As reported, CENSO has the lowest CPU time on 8 of the 13 problems. On these problems CENSO needs only a handful of iterations – in particular, on problems P6 and P10 it converges in one iteration. The explanation to this is that the B-spline relaxation is tight at the global solution. For example, B-spline constraints are interpolated by the B-spline convex hull relaxation at points in the corners of the bounding box $[\mathbf{x}^l, \mathbf{x}^u]$.

The reported results are inadequate to evaluate the performance of CENSO on larger problems. However, the results from P12 and P13 do suggest that it becomes less competitive as the number of variables and constraints grow. This is expected since CENSO lacks some of the preprocessing and domain reduction techniques that accelerate the other solvers.

2.5.2. Pump network synthesis problem

The following pump network synthesis problem from (Westerlund et al., 1994) is solved. The same problem is used by Adjiman et al. (2000) to test the GMIN-αBB

Table 2.3.: Comparison of solvers on nonconvex NLP problems.

	CPU time (ms)				
Problem	SparsePOP	BARON	COUENNE	LINDO	CENSO
P1	505	79	122	644	**33**
P2	771	61	61	445	**53**
P3	×	57	58	437	**29**
P4	821	**67**	192	493	69
P5	1261	45	52	501	**37**
P6	1269	43	51	502	**2***
P7	41978	**57**	67	519	136
P8	3264	104	**87**	1172	199
P9	1323	68	108	599	**15**
P10	896	55	54	1183	**1***
P11	1156	59	71	477	**39**
P12	2765	**88**	509	633	573
P13	×	**58**	86	1051	261

× Failed to solve to global optimality within 1 minute
* CENSO converged after one iteration

algorithm. The problem belongs to the class of nonconvex MINLP problems, where integer variables participate in nonlinear terms.

The problem is to find the least costly configuration of pumps that provide the required pressure rise and total flow. The pump configuration is restricted to an L level superstructure, where pumps may be installed in series and/or parallel at each level. The existence of each level $i \in 1, \ldots, L$ is decided by a binary variable z_i. On a given level i there are N_i^p parallel pumping lines with N_i^s pumps of the same type in series. The pumps at level i operate with rotational speed ω_i and power P_i to supply a pressure rise of Δp_i and a fraction x_i of the total flow v_{tot}. The problem is described by the following MINLP formulation with data in Table 2.4.

$$
\begin{aligned}
\text{min.} \quad & \sum_{i=1}^{L} \left(C_i + C_i' P_i\right) N_i^p N_i^s z_i \\
\text{s.t.} \quad & \sum_{i=1}^{L} x_i = 1, \\
& \left.\begin{aligned}
& P_i - f_{1,i}(\omega_i, v_i) = 0, \\
& \Delta p_i - f_{2,i}(\omega_i, v_i) = 0, \\
& v_i N_i^p - x_i v_{\text{tot}} = 0, \\
& \Delta p_{\text{tot}} z_i - \Delta p_i N_i^s = 0, \\
& P_i - P_i^{\max} z_i \leq 0, \\
& \Delta p_i - \Delta p_{\text{tot}} z_i \leq 0, \\
& v_i - v_{\text{tot}} z_i \leq 0, \\
& x_i - z_i \leq 0, \\
& \omega_i - \omega_{\max} z_i \leq 0, \\
& N_i^p - N^p z_i \leq 0, \\
& N_i^s - N^s z_i \leq 0, \\
& 0 \leq x_i \leq 1, \\
& 0 \leq v_i \leq v_{\text{tot}}, \\
& 0 \leq \omega_i \leq \omega_{\max}, \\
& 0 \leq P_i \leq P_i^{\max}, \\
& 0 \leq \Delta p_i \leq \Delta p_{\text{tot}}, \\
& N_i^p \in \{1, \ldots, N^p\}, \\
& N_i^s \in \{1, \ldots, N^s\}, \\
& z_i \in \{0, 1\},
\end{aligned} \right\} i = 1, \ldots, L
\end{aligned}
\tag{PNSP}
$$

where

$$
f_{1,i}(\omega_i, v_i) = \alpha_i \left(\frac{\omega_i}{\omega_{\max}}\right)^3 + \beta_i \left(\frac{\omega_i}{\omega_{\max}}\right)^2 v_i + \gamma_i \left(\frac{\omega_i}{\omega_{\max}}\right) v_i^2, \tag{2.31}
$$

$$
f_{2,i}(\omega_i, v_i) = a_i \left(\frac{\omega_i}{\omega_{\max}}\right)^2 + b_i \left(\frac{\omega_i}{\omega_{\max}}\right) v_i + c_i v_i^2, \tag{2.32}
$$

are the power and pressure drop characteristics of the pumps, respectively. As pointed out by Westerlund et al. (1994), these functions are usually not known explicitly. They are typically given by the pump manufacturer in the form of lookup tables. (The lookup tables may be the results from running the pump in a test loop using water.)

The pump characteristics are polynomial functions of degree $\deg(f_{1,i}) = (3, 2)$ and $\deg(f_{2,i}) = (2, 2)$. These, and all bilinear terms in PNSP, may be represented as B-splines. However, assuming that $f_{1,i}$ and $f_{2,i}$ are unknown, the B-spline could be used to approximate the functions. Consider the approximation errors in Table 2.5 obtained by sampling the functions on regular grids. The errors are given in terms of $e(\mathbf{x}) = f(\mathbf{x}) - \tilde{f}(\mathbf{x})$, where $\tilde{f}$ is the B-spline approximation of f. The

Table 2.4.: Data for PNSP.

	Pump 1	Pump 2	Pump 3
Fixed cost (FIM)	38,900	15,300	20,700
C_i (FIM)	6,329.3	2,489.31	3,270.27
C_i' (FIM/kW)	1,800	1,800	1,800
α_i	19.9	1.21	6.52
β_i	0.161	0.0644	0.102
γ_i	-0.000561	-0.000564	-0.000232
a_i	629.0	215.0	361.0
b_i	0.696	2.950	0.530
c_i	-0.0116	-0.115	-0.00946
$P_i^{\max}$ (kW)	80	25	45
$v_{\text{tot}} = 350$ m^3/h,	$\Delta p_{\text{tot}} = 400$ kPa,		$\omega_{\max} = 2,950$ rpm

interpolating bicubic B-spline has degree $(3, 3)$, which is sufficient to approximate both functions without error, even on the coarsest grid. Similarly, the biquadratic B-spline has degree $(2, 2)$, and may approximate $f_{2,i}$ without error, but not $f_{1,i}$ which is approximated with small errors. The degree of the bilinear B-spline, $(1, 1)$, is insufficient to represent any of the functions. Even so, the approximation errors are small on the finest grid. On the coarser grids the errors of the bilinear B-spline become relevant.

PNSP was solved with BARON, COUENNE, LINDO, and CENSO. For CENSO, the pump characteristics were represented by exact and approximative B-splines on the various grids in Table 2.5. When using B-spline approximations, we configured CENSO to use the returned global optimum as the starting point of a local NLP search on the exact problem to obtain a feasible optimum. The results for PNSP with $L = 3$ are reported in Table 2.6. Note that the optimal solution is $128,894$ FIM (FIM stands for the now relinquished currency Finnish markka).

CENSO is able to solve PNSP in 34 seconds using exact B-spline representations of the pump characteristics. Both BARON and COUENNE are able to converge faster. The bicubic B-splines approximate the pump characteristics without error and, consequently, CENSO is able to locate the global optimum. The biquadratic B-splines are not exact, but accurate enough to guide CENSO to the global optimum. With bilinear B-splines CENSO finds the global optimum only when using the finer grids. On the 5×5 and 10×10 grids CENSO locates a suboptimal point with an optimality gap of 1.73%.

The solution times do not follow a clear pattern. However, there is a trade-off related to the number of samples (knots). Many samples (knots) give many auxiliary variables in the relaxation which demotes speed; but it also gives a tighter relaxation which promotes speed. A certain pattern, however, is that the solution time increases if the number of samples (knots) become too high. For example,

Table 2.5.: B-spline approximation errors for pump characteristics.

Function	Degree ($\mathbf{p}$)	Grid	e_1	e_∞
$f_{1,1}$	$(3,3)$	5×5	0	0
	$(3,3)$	10×10	0	0
	$(3,3)$	20×20	0	0
	$(3,3)$	30×30	0	0
	$(2,2)$	5×5	$4.72 \cdot 10^{-3}$	$7.28 \cdot 10^{-3}$
	$(2,2)$	10×10	$4.14 \cdot 10^{-4}$	$6.39 \cdot 10^{-4}$
	$(2,2)$	20×20	$4.40 \cdot 10^{-5}$	$6.79 \cdot 10^{-5}$
	$(2,2)$	30×30	$1.24 \cdot 10^{-5}$	$1.91 \cdot 10^{-5}$
	$(1,1)$	5×5	$2.75 \cdot 10^{-1}$	$1.08 \cdot 10^{+0}$
	$(1,1)$	10×10	$5.44 \cdot 10^{-2}$	$2.13 \cdot 10^{-1}$
	$(1,1)$	20×20	$1.22 \cdot 10^{-2}$	$4.78 \cdot 10^{-2}$
	$(1,1)$	30×30	$5.23 \cdot 10^{-3}$	$2.05 \cdot 10^{-2}$
$f_{2,1}$	$(3,3)$	5×5	0	0
	$(3,3)$	10×10	0	0
	$(3,3)$	20×20	0	0
	$(3,3)$	30×30	0	0
	$(2,2)$	5×5	0	0
	$(2,2)$	10×10	0	0
	$(2,2)$	20×20	0	0
	$(2,2)$	30×30	0	0
	$(1,1)$	5×5	$1.44 \times 10^{+2}$	$2.20 \cdot 10^{+2}$
	$(1,1)$	10×10	$2.85 \cdot 10^{+1}$	$4.35 \cdot 10^{+1}$
	$(1,1)$	20×20	$6.39 \cdot 10^{+0}$	$9.76 \cdot 10^{+0}$
	$(1,1)$	30×30	$2.74 \cdot 10^{+0}$	$4.19 \cdot 10^{+0}$

Error measures: $e_1 = \int_X |e(\mathbf{x})| d\mathbf{x}/\mathrm{Vol}(X)$, $e_\infty = \sup_{\mathbf{x} \in X} |e(\mathbf{x})|$, where $e = f - \tilde{f}$.

when we approximated with bicubic B-splines on 50×50 grids, CENSO required 211 seconds to converge.

2.6. Conclusion

A novel algorithm based on B-splines has been presented for global optimization of spline constrained MINLP problems. Various theoretical aspects of the algorithm have been analysed in discussions and examples throughout this paper. Furthermore, the algorithm has been implemented in the sBB solver CENSO, described in Sec. 2.4.

In Sec. 2.5 CENSO was compared to three state-of-the-art global solvers, and one specialized polynomial solver, with promising results. With the alternative refor-

Table 2.6.: Results for PNSP.

Solver	Degree ($\mathbf{p}$)	Grid	Time (sec)	Gap (%)
COUENNE	exact	–	7	0
BARON	exact	–	18	1.73
LINDO	exact	–	101	0
CENSO	exact	–	34	0
CENSO	$(3,3)$	5×5	53	0
CENSO	$(3,3)$	10×10	108	0
CENSO	$(3,3)$	20×20	84	0
CENSO	$(3,3)$	30×30	75	0
CENSO	$(2,2)$	5×5	43	0
CENSO	$(2,2)$	10×10	97	0
CENSO	$(2,2)$	20×20	73	0
CENSO	$(2,2)$	30×30	90	0
CENSO	$(1,1)$	5×5	21	1.73
CENSO	$(1,1)$	10×10	62	1.73
CENSO	$(1,1)$	20×20	44	0
CENSO	$(1,1)$	30×30	71	0

mulation based on B-splines, and the convex relaxations that follow, the algorithm performed well on the polynomially constrained test problems. This is exciting, remembering that CENSO lacks some of the sophisticated preprocessing and domain reduction techniques employed by the other solvers. It is expected that, in combination with these techniques, the B-spline-based reformulation-convexification can be competitive also on large-scale problems.

The versatility of spline constraints was displayed in Sec. 2.5.2 where a pump network synthesis MINLP problem was solved by approximating several pump characteristics with B-splines. The effect of sample density and spline degree on the results was investigated. The example shows that the B-spline is a powerful tool for approximating nonlinear functions. For this very reason, modelling with splines has a wide range of applications and may be considered an alternative to the often used piecewise linear functions. Arguably, the biggest contribution of this paper is a convenient and sBB-compatible framework for optimization with spline constraints.

The authors would like to conclude by suggesting a few directions in which the proposed algorithm could be developed:

- A B-spline can be used to approximate any black-box function by sampling. If the upper bound in (2.11) can be calculated, it can be used to measure the optimality gap in the BB algorithm. By requiring the black-box function to be Lipschitz with a known Lipschitz coefficient, the unknown second derivatives in the upper bound (2.11) could be removed. This approach would require

additional sampling of the black-box function during optimizing to ensure ϵ-convergence. The algorithm outlined above resembles the existing response surface methods used in global, black-box optimization (Jones, 2001). Note that several works on Lipschitz optimization can be found in the literature (Pinter, 1997).

- Several bounds tightening techniques can be imagined for the B-spline constraints. For example, by exploiting the non-negativity of the basis functions, it is straightforward to check the negativity of a spline in a given domain since it must come from the control points alone. Two other techniques could come from generalizing the Bernstein hull and box consistency algorithms to spline constraints (Nataraj and Arounassalame, 2011).

- A fundamental disadvantage of the presented method is that the number of auxiliary variables in the relaxation grows exponentially with number of dimensions/variables of a spline function. For high-dimensional functions it would be useful to use decomposition to reduce the number of auxiliary variables.

Acknowledgements

This work was supported by the Center for Integrated Operations in the Petroleum Industry, Trondheim, Norway.

2.A. Test problems

This appendix holds a collection of nonconvex NLP problems used in the computational study in this paper. Most of the problems can be found in the GLOBALLib library Meeraus, A. (2013) and in the test problem handbooks of Floudas Floudas and Pardalos (1990); Floudas et al. (1999). The same problem set was used in Nataraj and Arounassalame (2011).

Problem 1 (Floudas et al., 1999, Ch. 4.10).

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = -x_1 - x_2 \\ \text{subject to} \quad & x_2 \leq 2 + 8x_1^2 - 8x_1^3 + 2x_1^4 \\ & x_2 \leq 36 - 96x_1 + 88x_1^2 - 32x_1^3 + 4x_1^4 \\ & x_1 \in [0,3], x_2 \in [0,4] \end{aligned} \tag{P1}$$

The global optimum of (P1) is at $\mathbf{x}^* = [2.3295, 3.1785]^\mathsf{T}$ with $f(\mathbf{x}^*) = -5.5080$.

Problem 2. Problem G6 in Koziel and Michalewicz (1999).

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = (x_1 - 10)^3 + (x_2 - 20)^3 \\ \text{subject to} \quad & 100 - (x_1 - 5)^2 - (x_2 - 5)^2 \leq 0 \\ & -82.81 + (x_1 - 6)^2 + (x_2 - 5)^2 \leq 0 \\ & x_1 \in [13, 100], x_2 \in [0, 100] \end{aligned} \tag{P2}$$

Global optimum at $\mathbf{x}^* = [14.0950, 0.8430]^\mathsf{T}$ with $f(\mathbf{x}^*) = -6961.815$.

Problem 3 Lebbah et al. (2007).

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = x_1 \\ \text{subject to} \quad & x_1^2 - x_2 \leq 0 \\ & x_2 - x_1^2(x_1 - 2) + 10^{-5} \leq 0 \\ & \mathbf{x} \in [-10, 10]^2 \end{aligned} \tag{P3}$$

Global optimum at $\mathbf{x}^* = [3.0, 9.00001]^\mathsf{T}$ with $f(\mathbf{x}^*) = 3$.

Problem 4 (Floudas et al., 1999, Ch. 3.5). Problem ex3.1.4 in GlobalLib.

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = -2x_1 + x_2 - x_3 \\ \text{subject to} \quad & \mathbf{x}^\mathsf{T}\mathsf{A}^\mathsf{T}\mathsf{A}\mathbf{x} - 2\mathbf{y}^\mathsf{T}\mathsf{A}\mathbf{x} + ||\mathbf{y}||^2 - 0.25||\mathbf{b} - \mathbf{z}||^2 \geq 0 \\ & x_1 + x_2 + x_3 - 4 \leq 0 \\ & 3x_2 + x_3 - 6 \leq 0 \\ & x_1 \in [0, 2], x_2 \in [0, 10], x_3 \in [0, 3] \\ \text{with data} \quad & \\ & \mathsf{A} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ -2 & 1 & -1 \end{bmatrix} \\ & \mathbf{b} = [3, 0, -4]^\mathsf{T} \\ & \mathbf{y} = [1.5, -0.5, -5]^\mathsf{T} \\ & \mathbf{z} = [0, -1, -6]^\mathsf{T} \end{aligned} \tag{P4}$$

The global optimum of P4 is at $\mathbf{x}^* = [0.5, 0, 3]^\mathsf{T}$ with $f(\mathbf{x}^*) = -4$.

Problem 5. Himmelblau problem from Floudas et al. (1999). Problem ex14.1.1 in GlobalLib.

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = x_3 \\ \text{subject to} \quad & 2x_1^2 + 4x_1x_2 - 42x_1 + 4x_1^3 - x_3 \leq 14 \\ & -2x_1^2 - 4x_1x_2 + 42x_1 - 4x_1^3 - x_3 \leq -14 \\ & 2x_1^2 + 4x_1x_2 - 26x_2 + 4x_2^3 - x_3 \leq 22 \\ & -2x_1^2 - 4x_1x_2 + 26x_2 - 4x_2^3 - x_3 \leq -22 \\ & \mathbf{x} \in [-5, 5]^3 \end{aligned} \tag{P5}$$

This is a root finding problem with $f(\mathbf{x}^*) = 0$. $\mathbf{x}^* = [-0.3050690, -0.9133455, 0]^\mathsf{T}$ is a known solution to P5.

Problem 6. An optimal design problem for a pressure vessel Li and Chang (1998); Nataraj and Arounassalame (2011).

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & 0.6224x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\ \text{subject to} \quad & -x_1 + 0.0193x_3 \leq 0 \\ & -x_2 + 0.00954x_3 \leq 0 \\ & -\pi x_3^2x_4 - (4/3)\pi x_3^3 + 750.1728 \leq 0 \\ & -240 + x_4 \leq 0 \\ & x_1 \in [1, 1.375], x_2 \in [0.625, 1], \\ & x_3 \in [47.5, 52.5], x_4 \in [90, 112] \end{aligned} \tag{P6}$$

Best known solution is $\mathbf{x}^* = [1, 0.625, 47.5, 90]^\mathsf{T}$ with $f(\mathbf{x}^*) = 6395.5$.

Problem 7 Floudas et al. (1999). Problem ex7.3.2 in GlobalLib.

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = x_4 \\ \text{subject to} \quad & x_1^4x_2^4 - x_1^4 - x_2^4x_3 = 0 \\ & 1.4 - x_1 - 0.25x_4 \leq 0 \\ & -1.4 + x_1 - 0.25x_4 \leq 0 \\ & 1.5 - x_2 - 0.2x_4 \leq 0 \\ & -1.5 + x_2 - 0.2x_4 \leq 0 \\ & 0.8 - x_3 - 0.2x_4 \leq 0 \\ & -0.8 + x_3 - 0.2x_4 \leq 0 \\ & \mathbf{x} \in [0, 5]^4 \end{aligned} \tag{P7}$$

The global optimum of P7 is at $\mathbf{x}^* = [1.1275, 1.2820, 1.0179, 1.0899]^\mathsf{T}$ with $f(\mathbf{x}^*) = 1.0899$.

Problem 8. Mechanical design problem from Venkataraman (2009).

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = 27.264(2x_2x_4 + x_1x_3 - 2x_3x_4) \\ \text{subject to} \quad & 61.01627586 - I(\mathbf{x}) \leq 0 \\ & 8x_1 - I(\mathbf{x}) \leq 0 \\ & x_1x_2x_4 - x_2x_4^2 + x_1^2x_3 + x_3x_4^2 - 2x_1x_3x_4 - 3.5x_3I(\mathbf{x}) \leq 0 \\ & x_1 - 3x_2 \leq 0 \\ & 2x_2 - x_1 \leq 0 \\ & x_3 - 1.5x_4 \leq 0 \\ & 0.5x_4 - x_3 \leq 0 \\ & x_1 \in [3, 20], x_2 \in [2, 15], x_3 \in [0.125, 0.75], x_4 \in [0.25, 1.25] \end{aligned} \tag{P8}$$

where $I(\mathbf{x}) = 6x_1^2x_2x_3 - 12x_1x_2x_3^2 + 8x_2x_3^3 + x_1^3x_4 - 6x_1^2x_3x_4 + 12x_1x_3^2x_4 - 8x_3^3x_4$. The global optimum is attained at $\mathbf{x}^* = [4.9542, 2, 0.125, 0.25]^\mathsf{T}$ with $f(\mathbf{x}^*) = 42.444$.

Problem 9. Test problem 1 in (Floudas and Pardalos, 1990, Ch. 2.2.1). Problem ex2.1.1 in GlobalLib.

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = \mathbf{c}^\mathsf{T}\mathbf{x} - \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x} \\ \text{subject to} \quad & 20x_1 + 12x_2 + 11x_3 + 7x_4 + 4x_5 \leq 40 \\ & \mathbf{x} \in [0, 1]^5 \end{aligned} \tag{P9}$$

where $\mathbf{c} = [42, 44, 45, 47, 47.5]^\mathsf{T}$ and $\mathbf{Q} = 50\mathbf{I}$ ($\mathbf{I}$ is the identity matrix). The global optimum is attained at $\mathbf{x}^* = [1, 1, 0, 1, 0]^\mathsf{T}$ with $f(\mathbf{x}^*) = -17$.

Problem 10. Test problem 2 in (Floudas and Pardalos, 1990, Ch. 2.2.1). Problem ex2.1.2 in GlobalLib

$$\begin{aligned} \underset{\mathbf{x},y}{\text{minimize}} \quad & f(\mathbf{x}, y) = \mathbf{c}^\mathsf{T}\mathbf{x} - 0.5\mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x} - 10y \\ \text{subject to} \quad & 6x_1 + 3x_2 + 3x_3 + 2x_4 + x_5 \leq 6.5 \\ & 10x_1 + 10x_3 + y \leq 20 \\ & y \geq 0 \\ & \mathbf{x} \in [0, 1]^5 \end{aligned} \tag{P10}$$

where $\mathbf{c} = -[10.5, 7.5, 3.5, 2.5, 1.5]^\mathsf{T}$ and $\mathbf{Q} = \mathbf{I}$ ($\mathbf{I}$ is the identity matrix). The global optimum is attained at $\mathbf{x}^* = [0, 1, 0, 1, 1]^\mathsf{T}$ and $y^* = 20$ with $f(\mathbf{x}^*, y^*) = -213$.

Problem 11 (Floudas and Pardalos, 1990, Ch. 3.3.1).

$$
\begin{aligned}
\underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = -25(x_1-2)^2 - (x_2-2)^2 - (x_3-1)^2 \\
& \qquad - (x_4-4)^2 - (x_5-1)^2 - (x_6-4)^2 \\
\text{subject to} \quad & (x_3-3)^2 + x_4 \geq 4 \\
& (x_5-3)^2 + x_6 \geq 4 \\
& x_1 - 3x_2 \leq 2 \\
& -x_1 + x_2 \leq 2 \\
& x_1 + x_2 \leq 6 \\
& x_1 + x_2 \geq 2 \\
& x_1 \in [0,6], x_2 \in [0,6], x_3 \in [1,5], \\
& x_4 \in [0,6], x_5 \in [1,5], x_6 \in [0,10]
\end{aligned}
\tag{P11}
$$

The global optimum of P11 is at $\mathbf{x}^* = [5, 1, 5, 0, 5, 10]^\mathsf{T}$ with $f(\mathbf{x}^*) = -310$.

Problem 12 (Floudas et al., 1999, Ch. 5.2.4).

$$
\begin{aligned}
\underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = -x_4(9 - 6x_1 - 16x_2 - 15x_3) \\
& \qquad - x_5(15 - 6x_1 - 16x_2 - 15x_3) + x_6 - 5x_7 \\
\text{subject to} \quad & x_3x_4 + x_3x_5 \leq 50 \\
& x_4 + x_6 \leq 100 \\
& x_5 + x_7 \leq 200 \\
& x_4(3x_1 + x_2 + x_3 - 2.5) - 0.5x_6 \leq 0 \\
& x_5(3x_1 + x_2 + x_3 - 1.5) + 0.5x_7 \leq 0 \\
& x_1 + x_2 + x_3 = 1 \\
& x_1 \in [0,1], x_2 \in [0,1], x_3 \in [0,1], \\
& x_4 \in [0,100], x_5 \in [0,200], \\
& x_6 \in [0,100], x_7 \in [0,200]
\end{aligned}
\tag{P12}
$$

The global optimum of P12 is at $\mathbf{x}^* = [0, 0.5, 0.5, 0, 100, 0, 100]^\mathsf{T}$ with $f(\mathbf{x}^*) = -450$.

Problem 13 (Floudas and Pardalos, 1990, Ch. 11.3.1).

$$
\begin{aligned}
\underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) \\
& \quad - 1.508x_1(x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) \\
& \quad + 0.7854(x_4x_6^2 + x_5x_7^2) \\
\text{subject to} \quad & x_1x_2^2x_3 \geq 27 \\
& x_1x_2^2x_3^2 \geq 397.5 \\
& x_2x_6^4x_3x_4^{-3} \geq 1.93 \\
& x_2x_7^4x_3x_5^{-3} \geq 1.93 \\
& [(745x_4x_2^{-1}x_3^{-1})^2 + 16.911 \cdot 10^6]^{0.5}/(0.1x_6^3) \leq 1100 \\
& [(745x_5x_2^{-1}x_3^{-1})^2 + 157.51 \cdot 10^6]^{0.5}/(0.1x_7^3) \leq 850 \\
& x_2x_3 \leq 40 \\
& x_1/x_2 \geq 5 \\
& x_1/x_2 \leq 12 \\
& 1.5x_6 - x_4 \leq -1.9 \\
& 1.1x_7 - x_5 \leq -1.9 \\
& x_1 \in [2.6, 3.6], x_2 \in [0.7, 0.8], x_3 \in [17, 28], \\
& x_4 \in [7.3, 8.3], x_5 \in [7.3, 8.3], x_6 \in [2.9, 3.9] \\
& x_7 \in [5, 5.5]
\end{aligned}
\tag{P13}
$$

The best known solution for P13 is the point $\mathbf{x}^* = [3.5, 0.7, 17, 7.3, 7.71, 3.35, 5.287]^\mathsf{T}$ with $f(\mathbf{x}^*) = 2994.47$. The problem can be written as a polynomially constrained problem by multiplying to remove all fractional terms in the constraints. This is possible because all variables are positively bounded.

2.B. Proofs

Proposition 2.1 (Relaxation of bilinear terms)**.** *Consider the bilinear term $y = x_1x_2$, for $x_1 \in [x_1^l, x_1^u]$ and $x_2 \in [x_2^l, x_2^u]$. Let f be a B-spline representing the bilinear term, i.e. $f = y$. Then, the convex combination relaxation* (2.24) *of f is equivalent to McCormick's linear relaxation of bilinear terms (see McCormick 1976; Adjiman et al. 1998b).*

Proposition 2.1. Let $\mathbf{x}_{1,1} = [1, x_1]^\mathsf{T}$ and $\mathbf{x}_{2,1} = [1, x_2]^\mathsf{T}$ be the first degree power bases of x_1 and x_2. The the bilinear term can be written as $y = \lambda^\mathsf{T}(\mathbf{x}_{1,1} \otimes \mathbf{x}_{2,1}) = \lambda^\mathsf{T}[1, x_1, x_2, x_1x_2]^\mathsf{T} = x_1x_2$, for $\lambda^\mathsf{T} = [0, 0, 0, 1]$. Using the procedure in Sec. 2.2.5 one obtains the B-spline form f of y, which has four control points

$$
\mathsf{P} = \begin{bmatrix} x_1^l & x_1^l & x_1^u & x_1^u \\ x_2^l & x_2^u & x_2^l & x_2^u \\ x_1^lx_2^l & x_1^lx_2^u & x_1^ux_2^l & x_1^ux_2^u \end{bmatrix}. \tag{2.33}
$$

The relaxation in (2.24) requires four variables $\lambda = [\lambda_1, \lambda_2, \lambda_3, \lambda_4]^\mathsf{T}$, and is given by the equations

$$\underbrace{\begin{bmatrix} x_1^l & x_1^l & x_1^u & x_1^u \\ x_2^l & x_2^u & x_2^l & x_2^u \\ x_1^l x_2^l & x_1^l x_2^u & x_1^u x_2^l & x_1^u x_2^u \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{\mathsf{A}} \lambda = \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ y \\ 1 \end{bmatrix}}_{\mathbf{b}}, \lambda \geq \mathbf{0}. \tag{2.34}$$

A is a square matrix of full rank as long as $x_1^l < x_1^u$ and $x_2^l < x_2^u$, and it is possible to solve $\lambda = \mathsf{A}^{-1}\mathbf{b}$ analytically. This yields

$$\begin{aligned} \lambda_1 &= \frac{1}{\gamma}\left(y - x_2^u x_1 - x_1^u x_2 + x_1^u x_2^u\right), \\ \lambda_2 &= \frac{1}{\gamma}\left(-y + x_2^l x_1 + x_1^u x_2 - x_1^u x_2^l\right), \\ \lambda_3 &= \frac{1}{\gamma}\left(-y + x_2^u x_1 + x_1^l x_2 - x_1^l x_2^u\right), \\ \lambda_4 &= \frac{1}{\gamma}\left(y - x_2^l x_1 - x_1^l x_2 + x_1^l x_2^l\right), \end{aligned} \tag{2.35}$$

where $\gamma = (x_1^u - x_1^l)(x_2^u - x_2^l)$. Utilizing $\lambda \geq \mathbf{0}$, and the fact that $\gamma > 0$, one obtains

$$\begin{aligned} y &\geq x_2^u x_1 + x_1^u x_2 - x_1^u x_2^u, \\ y &\leq x_2^l x_1 + x_1^u x_2 - x_1^u x_2^l, \\ y &\leq x_2^u x_1 + x_1^l x_2 - x_1^l x_2^u, \\ y &\geq x_2^l x_1 + x_1^l x_2 - x_1^l x_2^l, \end{aligned} \tag{2.36}$$

which are precisely the linear constraints of the McCormick relaxation of $y = x_1 x_2$. □

Chapter 3

Global optimization of multiphase flow networks using spline surrogate models

Grimstad, B., Foss, B., Heddle, R., and Woodman, M. (2016). Global optimization of multiphase flow networks using spline surrogate models. *Computers & Chemical Engineering*, 84:237 – 254.

Summary

A general modelling framework for optimization of multiphase flow networks with discrete decision variables is presented. The framework is expressed with the graph and special attention is given to the convexity properties of the mathematical programming formulation that follows. Nonlinear pressure and temperature relations are modelled using multivariate splines, resulting in a mixed-integer nonlinear programming (MINLP) formulation with spline constraints. A global solution method is devised by combining the framework with a spline-compatible MINLP solver, recently presented in the literature. The solver is able to globally solve the nonconvex optimization problems. The new solution method is benchmarked with several local optimization methods on a set of three realistic subsea production optimization cases provided by the oil company BP.

3.1. Introduction

Multiphase flow networks appear in many application areas. In this paper we are particularly interested in multiphase flow networks for subsea oil and gas production. Such networks consist of wells, collection systems, pipelines, and in some cases processing units such as pumps and separators. In recent years real-time data capture and storage capabilities have become an industry standard, thus paving the way for the use of model-based techniques to improve operations. In practice, the use of model-based methods translates into advisory systems for production engineers. Such systems use real-time data in combination with calibrated mathematical models and optimization to improve economics of an oil field by increasing oil throughput. It can be hard to measure the true value of model-based advisory

systems since they have impact on profit, cost of operating, HSE and operating risk, and possibly other non-economic values. This may explain why operators tend to prefer maximization of hydrocarbons (oil and gas): hydrocarbon production can be measured, and sometimes must be measured to conform with legislation and fiscal systems. Some claims to a production increase of up to 4% due to use of model-based tools can be found in the literature (Stenhouse et al., 2010; Teixeira et al., 2013). In the latter testimonial, a 1.2% production increase on a medium size offshore production vessel is claimed, amounting to $35 mill per year. Thus, the economic potential is clearly significant. Despite this, real-time decision support tools as alluded to above are rarely used in the upstream petroleum industries.

Two key reasons for limited use are lack of tools for model maintenance, and robust and efficient solvers, respectively. First, models must be updated periodically due to the time-varying nature of the production system; in particular, reservoir conditions change with time due to reservoir pressure decline and changing fluid compositions. Second, the optimization problem itself is hard to solve since models are nonlinear and often available only as black box calculators. In fact, oil and gas production systems are typically modelled in proprietary process simulators, not offering gradient information. Thus, there are several factors that contribute to long solution times, including the following: a lack of analytical derivatives, computational expensive evaluations of the process simulator, and slow IO operations in the communication between process simulator and optimization solver. It may be added that different parts of the flow network, in particular well models and pipeline models, may be available in different simulator applications, thus complicating matters even more. Moreover, decision variables are both continuous and discrete. Thus, we are faced with mixed-integer nonlinear (MINLP) problems that may include black-box constraints. Long solution times prevent efficient use of decision support tools and break the natural workflow of the production engineers. When it takes several hours to arrive at an optimization the result is often "out of date" before it is available to the them.

This paper suggests a methodology to overcome the challenges related to the optimization part as presented above. This is done in three steps. First, we adapt a well known, graph-based modelling scheme to oil and gas networks. Second, we propose the use of spline-based surrogate models to represent the nonlinear parts of the system. This implies that models, which are available as proprietary (black-box) simulators, explicit model equations or look-up tables, are approximated with splines through a sampling and interpolation scheme. By performing this substitution for each item of equipment in the network *a priori* optimization, the solver can be decoupled from the process simulator during the optimization run, resulting in a considerable reduction in solution times. Third, we introduce a global branch-and-bound based MINLP solver that exploits the facts that all nonlinearities are described by splines and takes advantage of the structural properties of oil and gas networks. In order to evaluate our approach it was deemed necessary to use a comprehensive and realistic test bench rather than simplistic cases. Thus, three industrial cases are used where all relevant models and constraints are included.

The remainder of this paper is organized as follows. A short description of the production optimization problem for subsea production systems is given in Section 3.2. A brief report on recent works on this topic follows in Section 3.3. A new mathematical programming framework for optimizing general flow networks is derived in detail in Section 3.4. In Section 3.5, we give a description of B-splines, which are used as surrogate models for the nonlinear functions in the problem formulation presented in Section 3.4. The solution method is presented in Section 3.6 and benchmarked on several realistic cases in Section 3.7. Finally, some concluding remarks are given in Section 3.8.

3.2. Problem description

Consider the subsea production system illustrated in Figure 3.1, consisting of reservoirs, wells, manifolds, flowlines, risers, and separators. The system is built to allow a safe and efficient transportation of reservoir fluid to the surface. At the surface the fluid is separated before it is further treated in processing facilities. The fluid flow through the system is controlled with valves, e.g. chokes (adjustable valves) and manifold valves (on/off valves). The valve settings decide the production and operational status of the wells: that is, the flow rate of each well, the routing of the flows through the network, and the allocation of lift gas.

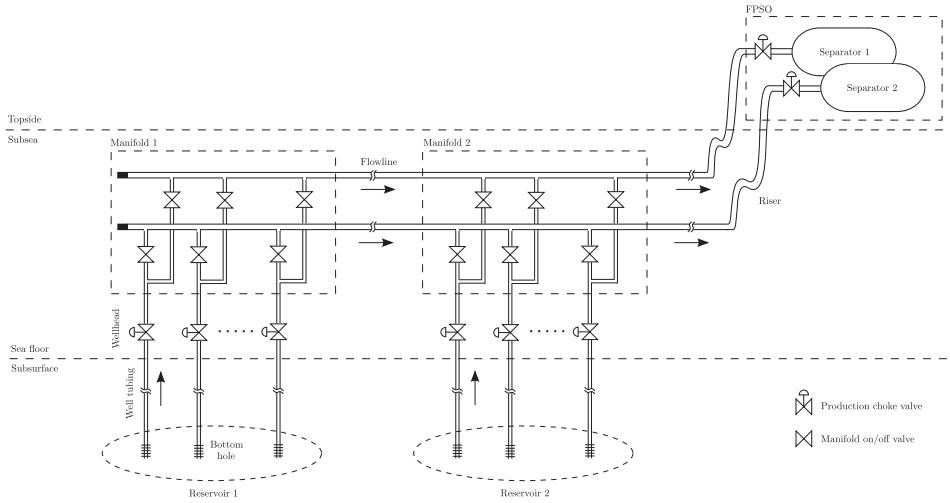


Figure 3.1.: A subsea production system with two daisy-chained manifolds.

The *daily production optimization problem* is the search for valve settings that maximize the production of oil (or profit) while respecting physical laws and operational constraints. Physical laws that must be abided include: mass, momentum, and energy conservation laws; and well inflow relations. Some physical laws may be empirically modelled because of their high complexity. For example, pipeline

pressure drops are often modelled with the empirical Beggs and Brill correlation (Beggs and Brill, 1973). Typical operational constraints may include: upper and lower rate constraints; draw-down (minimum) pressure constraints; oil, gas, and water handling capacity constraints; upper bound on gas-lift availability; and number of allowed routing changes. The operational constraints are typically provided by the user to be in accordance with the current production plan.

A requirement for solving the daily production optimization problem, and for it to provide applicable and optimal solutions, is that an accurate model of the production system is available. The model should accurately predict flow rates for any valve setting in the search space of interest. To reduce the modelling effort, we present a mathematical programming framework that includes the above-mentioned physical laws and operational constraints of a generic production system. The flexible framework, presented in the next section, allows for modelling of most common subsea production system topologies.

3.3. Previous work

Network flow and design problems lie in the intersection of several domains, including: operations research, applied mathematics, engineering, and computer science (Ahuja et al., 1993). A diverse set of problems can be formulated as network problems, for example: optimization of urban public transportation networks (Mandl, 1980), train routing and scheduling (Cordeau et al., 2009), and design of optimal water distribution systems (Alperovits and Shamir, 1977).

The recent works of Luathep et al. (2011) and Raghunathan (2013) look at global optimization of the network design problem using a MILP and MINLP approach, respectively. These methods have features similar to the method presented later in this paper. For instance, the graph is used as a modelling tool, convex formulations are obtained by using the big-M relaxation, nonlinear relations are approximated with piecewise functions, and a MILP or MINLP problem is solved using a specialized solver.

In the following we provide a brief report on works that address the petroleum production optimization problem described in Section 3.2. To the authors' knowledge, the works of Kosmidis et al. (2004, 2005) were the first to address well oil rate allocation, gas lift allocation, and well routing with a single problem formulation. In these works the complete production optimization problem is posed as a MINLP problem.[1] Many earlier works have considered optimization of individual network components, for instance optimal gas lift allocation on a well basis (Wang, 2003; Rashid et al., 2012). A survey on these early works is provided by Kosmidis et al. (2005) and Bieker et al. (2007). From 2006 and onwards, several works have emerged that build on the contributions of Kosmidis et al. (2005) or use a similar approach, see for example Martin et al. (2006); Misener et al. (2009); Gunnerud and Foss (2010); Codas and Camponogara (2012). These works use piecewise linear

[1]The problem is also referred to as the *daily well scheduling problem* in the literature.

formulations to approximate nonlinear relations in the network such as nonlinear pressure drop functions. This modelling approach results in a MILP that scales poorly when nonseparable functions of four or more variables are approximated (Misener and Floudas, 2010; Vielma et al., 2010; Vielma and Nemhauser, 2011). This may be one reason for the relatively simple formulations used in these works. For example, temperatures are not considered and pressure drops are modelled as functions of flow rates only. Interestingly, the MINLP formulation of Kosmidis et al. (2005) did include linear temperature drop models. A computational analysis of different multidimensional piecewise linear models was recently provided by Silva and Camponogara (2014). The analysis shows that SOS2 models and MILP models with a logarithmic number of binary variables have the best performance. These formulations may allow modelling of (nonseparable) functions of four or five variables, for which separation to multiple lower-dimensional functions is not possible.

The spline-based approach presented in this paper can be viewed as an alternative to the piecewise linear approaches mentioned above. The approach to be presented results in a global NLP problem (MINLP if routing is included). The main difference is that the solver must branch on continuous variables, instead of SOS2 or binary variables as is the case in the MILP approaches. The approach allows us to accurately approximate nonlinear functions in up to five variables. This enables us to model pressure and temperature drops as functions of flow rates, pressure, and temperature.

3.4. Multiphase flow network modelling

In this section we present a general mathematical programming formulation for multiphase flow networks. The goal is to achieve a formulation that is as simple as possible, while capturing important physics with sufficient accuracy. This goal reflects our desire to obtain a problem formulation that we can solve in reasonable time to get applicable solutions.

The problem formulation is based on the following assumptions:

A1. The system operates at steady-state conditions.

A2. Continuous and differentiable multiphase pressure drop and temperature drop models.

A3. The thermodynamics can be modelled under the assumptions in Section 3.4.3.

A4. No uncertainty is considered in the model structure or its parameters.

An argument for assumption A1 follows: The daily production optimization problem has a horizon spanning several hours to one day. In general, the fluid dynamics in the network (wells and pipelines) have time constants in the order of

minutes, and will appear instant on this horizon. Similarly, the dynamics at the system boundaries appear constant on this horizon: the reservoir dynamics have time constants in the order of weeks to months, and the surface facility dynamics have time constants in the order of seconds to minutes. One may note that there are exceptions where A1 does not hold, examples include dynamic phenomena like slugging or casing-heading instability which are highly influential on the production and act in the relevant horizon of hours.

Assumption A2 ensures continuous and differentiable constraint functions, which is a prerequisite for most gradient-based optimization solvers. As will be discussed, spline surrogate models have these properties by construction, even when the function they approximate do not. Assumption A4 is included since uncertainty is not structurally treated in the proposed framework.

A directed graph $G = (\mathbf{N}, \mathbf{E})$, with nodes $\mathbf{N}$ and edges $\mathbf{E}$, is used to represent the flow network (Ahuja et al., 1993). A node in $\mathbf{N}$ represents a junction or simply a point of interest in the network. An edge in $\mathbf{E}$ connects two nodes and represents a pipe segment (e.g. a wellbore, jumper, flowline, or riser), a valve (e.g. a production choke or manifold valve), or any item of equipment (e.g. a subsea multiphase pump). Valves represent special edges since they can be closed to disjoint the neighbouring nodes. To make this distinction clear we introduce a subset of edges, $\mathbf{E}^{\mathrm{d}}$, that represent the valves. An edge in $\mathbf{E}^{\mathrm{d}}$ is referred to as a *discrete* edge since it has two states: it is either open or closed. Associated with each discrete edge is a binary variable which is used to model the switching between the open and closed state. The discrete edges are used to route the flow through the network by restricting flow through certain valves. All other edges ($\mathbf{E} \backslash \mathbf{E}^{\mathrm{d}}$) represent pipes or equipment.

Table 3.1 gives the various sets used to describe the flow network. Some utility sets that simplify the notation are given in Table 3.2. In the rest of this paper the terms *graph*, *network*, and *system* are used interchangeably.

The following requirements are placed on the graph structure:

R1. A source node $i \in \mathbf{N}^{\mathrm{src}}$ has *zero* entering edges and *one* leaving edge, i.e. $\mathbf{E}_i^{\mathrm{in}} = \emptyset$ and $|\mathbf{E}_i^{\mathrm{out}}| = 1$.

R2. A sink node $i \in \mathbf{N}^{\mathrm{snk}}$ has *zero* leaving edges, i.e. $\mathbf{E}_i^{\mathrm{out}} = \emptyset$.

R3. An internal node $i \in \mathbf{N}^{\mathrm{int}}$ has *one or more* leaving edges. It may have *more than one* leaving edges iff all of them are discrete edges and at most one of them can be open at any time.

The first and second requirement follow the normal definition of source and sink nodes. The additional requirement that a source node may have only one leaving edge is made without loss of generality (an equivalent graph fulfilling this requirement can always be obtained by adding nodes). The third requirement on the internal nodes is needed because splitting of fluids is not modelled; this simplifying requirement is commonly applied in works on flow network modelling (Codas

Table 3.1.: Sets

Set	Description
$\mathbf{N}$	Set of nodes in the network.
$\mathbf{N}^{\text{src}}$	Set of source nodes in the network. $\mathbf{N}^{\text{src}} \subset \mathbf{N}$.
$\mathbf{N}^{\text{snk}}$	Set of sink nodes in the network. $\mathbf{N}^{\text{snk}} \subset \mathbf{N}$.
$\mathbf{N}^{\text{int}}$	Set of internal nodes in the network. $\mathbf{N}^{\text{int}} = \mathbf{N} \backslash \{\mathbf{N}^{\text{src}} \cup \mathbf{N}^{\text{snk}}\}$.
$\mathbf{E}$	Set of edges in the network. An edge $e = (i, j)$ connects node i to node j, where $i, j \in \mathbf{N}$.
$\mathbf{E}^{\text{d}}$	Set of *discrete* edges that can be *open* or *closed*. $\mathbf{E}^{\text{d}} \subset \mathbf{E}$.
$\mathbf{S}$	Set of flow phases in the network. For three-phase petroleum flow the phases are denoted $\mathbf{S} = \{\mathsf{oil}, \mathsf{gas}, \text{wat}\}$, where oil denotes the hydrocarbon liquid phase, gas the hydrocarbon gas phase, and wat the water liquid phase. A compositional model may have more than three phases/components.

et al., 2012). This requirement is enforced by manifold routing constraints, to be presented later in this section.

The sets $\mathbf{E}^{\text{src}}$ and $\mathbf{E}^{\text{snk}}$ in Table 3.2 are *cut-sets*. A cut-set is a set of edges that, if removed, partitions the graph nodes into two disconnected subsets. These sets are useful because the net flow through the graph can be measured as the net flow over the edges in a cut-set.

Table 3.2.: Utility sets

Set	Description
$\mathbf{E}_i^{\text{in}}$	Set of edges entering node i, i.e. $\mathbf{E}_i^{\text{in}} = \{e : e = (j, i) \in \mathbf{E}\}$.
$\mathbf{E}_i^{\text{out}}$	Set of edges leaving node i, i.e. $\mathbf{E}_i^{\text{out}} = \{e : e = (i, j) \in \mathbf{E}\}$.
$\mathbf{E}^{\text{src}}$	Set of edges leaving a source node in $\mathbf{N}^{\text{src}}$, i.e. $\mathbf{E}^{\text{src}} = \bigcup_{i \in \mathbf{N}^{\text{src}}} \mathbf{E}_i^{\text{out}}$.
$\mathbf{E}^{\text{snk}}$	Set of edges entering sink node in $\mathbf{N}^{\text{snk}}$, i.e. $\mathbf{E}^{\text{snk}} = \bigcup_{i \in \mathbf{N}^{\text{snk}}} \mathbf{E}_i^{\text{in}}$.
$\mathbf{N}^{\text{d}}$	Set of nodes with discrete leaving edges, i.e. $\mathbf{N}^{\text{d}} = \{i : i \in \mathbf{N}, \mathbf{E}_i^{\text{out}} \subset \mathbf{E}^{\text{d}}\} \subset \mathbf{N}$.

The variables of the problem, listed in Table 3.3, are related to the nodes and edges of the graph. The flow rates are given as mass flow rates or as volumetric flow rates in standard conditions. In the latter case, the flow rates must be properly scaled with the phases' standard condition densities, denoted with ρ_s for $s \in \mathbf{S}$. For brevity, the phase flow rates on an edge $e \in \mathbf{E}$ are collectively denoted $\mathbf{q}_e$, that is, with an oil, gas, and water phase, $\mathbf{q}_e = [q_{e,\mathsf{oil}}, q_{e,\mathsf{gas}}, q_{e,\text{wat}}]^{\mathsf{T}}$. Furthermore, we denote all the flow rates, pressures, and pressure drops in the network with $\mathbf{q}$, $\mathbf{p}$, and $\mathbf{\Delta p}$, respectively. We use the same notation for vectors containing the temperature and

enthalpy variables.

Table 3.3.: Variables

Variable	Description
p_i	Pressure at node $i \in \mathbf{N}$.
Δp_e	Pressure drop over edge $e = (i, j) \in \mathbf{E}$, e.g. $\Delta p_e = p_i - p_j$.
t_e	Temperature of fluid entering edge $e \in \mathbf{E}$.
Δt_e	Temperature drop over edge $e \in \mathbf{E}$. The fluid leaving e has a temperature of $t_e - \Delta t_e$.
h_e	Enthalpy of fluid entering edge $e \in \mathbf{E}$.
Δh_e	Enthalpy drop over edge $e \in \mathbf{E}$. The fluid leaving e has an enthalpy of $h_e - \Delta h_e$.
$q_{e,s}$	Flow rate of phase $s \in \mathbf{S}$ on edge $e \in \mathbf{E}$.
y_e	Binary variable associated with an edge $e \in \mathbf{E}^{\mathrm{d}}$. If $y_e = 1$ the edge is open, allowing a nonzero flow; otherwise, $y_e = 0$ and the edge is closed with zero flow.

Table 3.4.: Parameters

Parameter	Description
$q^L_{e,s}, q^U_{e,s}$	Lower and upper bound, respectively, for flow rate $q_{e,s}$ of phase $s \in \mathbf{S}$ on edge $e \in \mathbf{E}$. It is assumed that $0 \leq q^L_{e,s} \leq q^U_{e,s}$.
p^L_i, p^U_i	Lower and upper bound, respectively, for pressure p_i in node $i \in \mathbf{N}$. It is assumed that $0 \leq p^L_i \leq p^U_i$.
t^L_e, t^U_e	Lower and upper bound, respectively, for temperature t_i on edge $e \in \mathbf{E}$. It is assumed that $0 \leq t^L_e \leq t^U_e$.

With the network topology represented by the graph, and the variables and parameters associated with the nodes and edges, the flow network is modelled by placing control volumes around each node and edge. In each control volume *mass*, *momentum*, and *energy* conservation laws are enforced. In the following, we present the equations/constraints for the conservation laws, as well as some operational constraints. Together with an objective, they form the basis of the proposed mathematical programming problem formulation, or framework, for production optimization. The complete formulation is given towards the end of this section.

3.4.1. Mass balances

In steady-state, the mass flow into a node must equal the mass flow out of it, i.e. there is no accumulation of fluid in the node (or in the network). Using the sets

$\mathbf{E}_i^{\text{in}}$ and $\mathbf{E}_i^{\text{out}}$, the mass balances for the nodes may be expressed as:

$$\sum_{e \in \mathbf{E}_i^{\text{in}}} q_{e,s} - \sum_{e \in \mathbf{E}_i^{\text{out}}} q_{e,s} = 0, \quad \forall s \in \mathbf{S}, i \in \mathbf{N}^{\text{int}}. \tag{3.1}$$

Note that the mass balances are defined only for the *internal* nodes in the network ($\mathbf{N}^{\text{int}}$). Since a source (sink) node have leaving (entering) edges only, its mass balance would enforce zero net flow out (in) of the node. Hence, source and sink nodes are excluded from Eq. (3.1).

3.4.2. Momentum balances

The multiphase flows in the network are driven by the node pressures (potentials) p_i. The pressure drop over an edge $e = (i, j)$ is defined as $\Delta p_e \triangleq p_i - p_j$, and relates the two node pressures p_i and p_j. For edges $e \in \mathbf{E}^{\text{d}}$ that represent choke or on/off valves, Δp_e is a free/adjustable variable as discussed in 3.A. For edges $e \in \mathbf{E} \backslash \mathbf{E}^{\text{d}}$ that represent pipes, Δp_e is given by some pressure drop correlation $\Delta p_e = f_e(\mathbf{q}_e, p_i, t_e)$. The function $f_e(\cdot)$ maps the upstream conditions (flow rates, pressure, and temperature) to the pressure drop Δp_e. When it is more convenient to express f_e in terms of the downstream pressure, p_i can simply be replaced with p_j. The pressure drops in the network are modelled with the following constraints:

$$\Delta p_e = f_e(\mathbf{q}_e, p_i, t_e), \quad \forall e \in \mathbf{E} \backslash \mathbf{E}^{\text{d}}. \tag{3.2}$$

Notice that Eq. (3.2) does not apply to edges with an adjustable pressure drop ($\mathbf{E}^{\text{d}}$). A pressure correlation may be insensitive to temperature for certain flow conditions, e.g. liquid dominated flows. In this case the correlation can be simplified to $\Delta p_e = f_e(\mathbf{q}_e, p_i)$ without any significant loss of accuracy. Another special case occurs for edges representing short pipes with negligible pressure drop, i.e. with $f_e(\cdot) \approx 0$, giving $\Delta p_e = p_i - p_j \approx 0$. In the rest of this paper $f_e(\cdot)$ will be used to denote the pressure correlation of edge e, even if $f_e(\cdot) = 0$.

For a discrete edge $e \in \mathbf{E}^{\text{d}}$, the momentum balance needs to be deactivated when the edge is closed. This logic is accurately expressed by the following disjunction:

$$\big[y_e = 0\big] \vee \begin{bmatrix} y_e = 1 \\ \Delta p_e = p_i - p_j \end{bmatrix}. \tag{3.3}$$

The disjunction in Eq. (3.3) can be interpreted as follows: if an edge e is closed ($y_e = 0$), then there is no *direct* relation between the pressures in the adjacent nodes i and j (the node pressures may still be *indirectly* related through other paths in the network); if the edge is open ($y_e = 1$), then the two pressures must satisfy the relation $\Delta p_e = p_i - p_j$, where Δp_e is given by $f_e(\cdot)$ in Eq. (3.2).

Although the disjunction in Eq. (3.3) captures the desired logic for the momentum balance its form is not widely supported by commercial solvers. A straightforward way to deal with the disjunction without using logical expressions is to approximate it with

$$y_e(p_i - p_j - \Delta p_e) = 0. \tag{3.4}$$

This formulation introduces an additional (and undesired) nonlinearity to the problem through the multiplication with y_e. This nonlinearity can be relaxed using linear big-M constraints.[2] Notice that the pressures p_i and p_j are constrained to $0 \leq p_i^L \leq p_i \leq p_i^U$ and $0 \leq p_j^L \leq p_j \leq p_j^U$. In practice p_j^L and p_j^U may be inferred from p_i^L and p_i^U, and the image of $f_e(\cdot)$. These bounds imply that $-M_e \leq p_i - p_j - \Delta p_e \leq M_e$, where $M_e = (p_i^U - p_i^L) + (p_j^U - p_j^L)$. Using M_e, the disjunction in Eq. (3.3) may be approximated with the big-M constraints

$$-M_e(1 - y_e) \leq p_i - p_j - \Delta p_e \leq M_e(1 - y_e). \tag{3.5}$$

For $y_e = 1$, Eq. (3.5) yields $0 \leq p_i - p_j - \Delta p_e \leq 0$, and the constraint $\Delta p_e = p_i - p_j$ in Eq. (3.3) is retrieved. For $y_e = 0$, Eq. (3.5) yields two constraints which are inactive in the feasible set: e.g. the inactive constraints allow Δp_e to take on any value in $[(p_i^L - p_j^U), (p_i^U - p_j^L)]$, effectively disconnecting p_i and p_j. Thus, the relaxation do not alter the optimal solution of the problem. A drawback with using big-M constraints is that they often produce a weak relaxation of the disjunction.[3] However, reasonably tight values for M_e can easily be derived from the pressure drop functions. Thus, we accept Eq. (3.5) as an alternative model to Eq. (3.3) and use it to model the momentum balances. Before proceeding, we note that the same big-M constraints were used by Codas et al. (2012). We also note that an alternative relaxation could have been achieved by using a convex hull formulation (Grossmann, 2002).

3.4.3. Energy balances

The thermodynamic potentials (enthalpies), h_e, of the fluids in the network are modelled using the temperature variables t_e. At the source nodes, fluid enters the network with a specified temperature (typically close to the reservoir temperature). As the fluid flows through the network, its temperature changes due to mixing with other fluids and energy loss through the pipe walls to the surroundings. In the following we will assume that the mixing happens at the nodes, and the energy loss occurs at the edges (pipes). To simplify the modelling we make the following assumptions:

- Instant mixing. At any point in the network all fluid phases are assumed to have the same temperature.
- No work is done by the system. However, the model can easily be extended to include energy generation or loss through work, allowing for active components such as pumps and compressors.

[2] The big-M constraints can easily be derived by applying McCormick's relaxation of bilinear terms to Eq. (3.4).

[3] In theory it is possible to let $M_e \to \infty$ and still obtain a valid relaxation. This will however give an increasingly poor relaxation and produce ill-conditioned systems of equations in the solver, causing numerical problems.

- The heat transfer between the system and its surroundings may be completely determined from internal states. Consequently, the heat transfer properties and surrounding temperatures are assumed to be constant.
- The enthalpy is equal to the internal energy of a fluid, that is, no pV-work is done. This assumption is reasonable for a stationary process without fluid accumulation.
- Constant heat capacities c_s for all phases $s \in \mathbf{S}$.

Regarding the last assumption above: In general, the heat capacity of a fluid is a function of pressure and temperature, i.e. $c_s = c_s(p_i, t_e)$. In practice, this relation is available in a compositional PVT model or a black oil model (Aziz and Settari, 1979). To simplify the model we assume c_s to be constant in this work. This simplification is reasonable for liquids, but may give rise to large errors for gases. However, in the enthalpy calculations below, the contribution from gas is generally much smaller than that of liquids, mitigating the erroneous heat capacity of gas. According to the above assumptions, we next present the equations in the thermodynamic model.

The temperature drop over the edges are modelled as

$$\Delta t_e = g_e(\mathbf{q}_e, p_i, t_e), \quad \forall e \in \mathbf{E}. \tag{3.6}$$

The relation gives the temperature change due to heat transfer through the pipe walls to the surroundings. In short, insulated, non-restrictive pipes the temperature drop can usually be ignored by setting $\Delta t_e = 0$.

The enthalpy of the fluid entering edge e, and the change in enthalpy across edge e, are calculated as

$$\begin{aligned} h_e &= t_e \sum_{s \in \mathbf{S}} c_s \cdot q_{e,s}, \qquad \forall e \in \mathbf{E}, \\ \Delta h_e &= \Delta t_e \sum_{s \in \mathbf{S}} c_s \cdot q_{e,s}, \qquad \forall e \in \mathbf{E}, \end{aligned} \tag{3.7}$$

where c_s is the constant heat capacity of fluid $s \in \mathbf{S}$.[4] As previously mentioned, nonlinear heat capacities on the form $c_s(p_i, t_e)$ may be used in Eq. (3.7) to increase the accuracy of the model. However, for liquid dominated flow the contribution to enthalpy from gas is relatively small.

Similar to the mass balances, we enforce an energy balance at each internal node (conjunction) in the network. With the enthalpy variables available the energy balances are easily expressed as:

$$\sum_{e \in \mathbf{E}_i^{\text{in}}} (h_e - \Delta h_e) = \sum_{e \in \mathbf{E}_i^{\text{out}}} h_e, \quad \forall i \in \mathbf{N}^{\text{int}}. \tag{3.8}$$

[4] In Eq. (3.7) the heat capacities c_s are given in [J/kg K], the rates $q_{e,s}$ in [kg/s], the temperatures t_e and Δt_e in [K], and the enthalpies h_e and Δh_e in [J/s] = [W].

Note that the downstream enthalpies ($h_e - \Delta h_e$) are used in the left-hand side of Eq. (3.8) to summarize the energy entering the node. According to the assumptions, the energy balances above are correct if: 1) no work is performed, 2) no heat is added, and 3) that the net change in kinetic and potential energy is zero.

The inclusion of the above energy model may increase the accuracy of the overall problem formulation. However, the increased accuracy comes at the cost of a computational heavier formulation since $3|\mathbf{E}|$ nonconvex constraints (nonlinear equality constraints) are added to the problem. One upside with the energy model is that it does not involve any binary routing variables. To see why, consider the case when a discrete edge is closed ($y_e = 0$). The flow rate is then forced to zero, which in turn forces the enthalpy on the edge to zero. Consequently, it does not contribute to the energy balance in Eq. (3.8). Thus, there is no need to involve binary variables in the energy balances.

3.4.4. Flow routing

Flows can be routed through certain parts of the network by opening and closing discrete edges: closing a discrete edge forces its mass flow to zero. This behaviour is expressed by combining the binary variable y_e with the lower ($q^L_{e,s}$) and upper ($q^U_{e,s}$) bounds on the flow rate as:

$$y_e q^L_{e,s} \leq q_{e,s} \leq y_e q^U_{e,s}, \quad \forall s \in \mathbf{S}, e \in \mathbf{E}^{\mathrm{d}}. \tag{3.9}$$

Note that Eq. (3.9) may force the flow rates $\mathbf{q}_e$ to zero. Thus, the domains of the nonlinear functions $f_e(\cdot)$ and $g_e(\cdot)$ should contain $\mathbf{q}_e = \mathbf{0}$; otherwise, $y_e = 0 \implies \mathbf{q}_e = \mathbf{0}$ is infeasible.

Depending on the network topology, some binary variables may be redundant. For example, the mass balance of a node with one entering and one leaving discrete edge will enforce equal flow rates. Thus, closing any one edge will force the flow rate on the other edge to zero. In this case one discrete edge (a single binary variable) is sufficient to model the on/off logic.

In general, the discrete edges may be configured to model any routing problem. Next we discuss a common routing configuration called a manifold.

The manifold: a special routing structure

A manifold is a collection of pipes and on/off valves designed so that its inlets can be routed to its outlets in various configurations, possibly by commingling the inlet streams. In a graph, the analogue to a manifold is a set of discrete edges connected as shown in Figure 3.2.

A normal operational constraint on subsea manifolds is that an inlet stream can be routed to at most one of the outlets. This constraint enforces requirement R2; that a node may have at most one open outlet. The manifold routing constraints

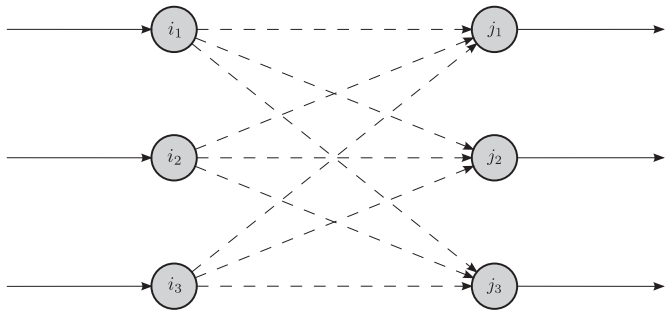


Figure 3.2.: A manifold modelled with discrete edges (dashed lines). The manifold can route each of the three inlet streams to any of the three outlets.

are easily expressed with the binary variables of the discrete edges as

$$\sum_{e \in \mathbf{E}_i^{\text{out}}} y_e \leq 1, \quad \forall i \in \mathbf{N}^{\text{d}}, \tag{3.10}$$

where $\mathbf{N}^{\text{d}}$ are nodes with discrete leaving edges.[5] The constraints in Eq. (3.10) allow flow to none or one of the edges leaving a node. Routing to zero edges forces the phase flow rates to zero via the mass balances in Eq. (3.1) and flow routing constraints in Eq. (3.9). If the inlet stream is required to flow to exactly *one* outlet the inequality in Eq. (3.10) is replaced with equality.

A manifold with 9 discrete edges configured as in Figure 3.2 has a total of $2^9 = 512$ possible routing combinations. The cut in Eq. (3.10) reduces the number of feasible routing combinations to $2^6 = 64$. A quick way to calculate the number of feasible combinations is to find the number of feasible combinations for each node, which is $n + 1$ for a node with n discrete leaving edges, and then multiply these numbers together. As calculated above we obtain $n+1 = 4$ for all three nodes, and $4 \cdot 4 \cdot 4 = 4^3 = 2^6 = 64$ feasible combinations.

Note that a manifold is constructed to minimize pressure loss across its pipes and valves, hence the pressure drop over the discrete edges may be fixed to zero.

3.4.5. Boundary conditions

To obtain a well-posed flow network problem it is necessary to specify boundary conditions for the network. The boundary conditions are usually related to the source and sink nodes, and specify the interaction between the network and its neighbouring systems. Next we discuss a few upstream and downstream boundary conditions, commonly used in models of subsea petroleum production networks.

[5]The constraint in Eq. (3.10) is redundant for nodes with only one discrete leaving edge: the constraint would be $y_e \leq 1$, which is always true. These redundant constraints can easily be omitted by altering Eq. (3.10) to apply only to nodes $i \in \{j : j \in \mathbf{N}^{\text{d}},\ |\mathbf{E}_j^{\text{out}}| > 1\}$. However, to keep the notation simple the manifold routing constraint is applied to all nodes in $\mathbf{N}^{\text{d}}$.

Upstream boundary conditions

At a source node $i \in \mathbf{N}^{\text{src}}$ we assume that the following relation between the pressure p_i and flow rates $\mathbf{q}_e$ exists:

$$\zeta_{i,s}(\mathbf{q}_e, p_i) = 0, \quad \forall s \in \mathbf{S}, i \in \mathbf{N}^{\text{src}}, \tag{3.11}$$

where $\zeta_{i,s} : \mathbb{R}^{|\mathbf{S}|+1} \to \mathbb{R}$, and $e = (i, j)$ is the (only) edge leaving source node i.

A common class of inflow rate boundary conditions in subsea production networks is the *inflow performance relationship* (IPR). It describes the mass flow from the reservoir into the well as a function of the measured bottom-hole pressure (also known as draw-down pressure). Wells without bottom-hole pressure sensors are usually modelled with a *well performance curve* (WPC), relating the flow rate to the wellhead pressure.

Two widely used IPRs are the linear (straight line) IPR and Vogel's quadratic IPR (Ahmed, 2006). With a linear IPR, a well $i \in \mathbf{N}^{\text{src}}$ can be modelled with the linear constraints:

$$\begin{aligned} q_{e,\mathsf{oil}} &= c_{i,\text{PI}}(p_{i,\text{res}} - p_i), \\ q_{e,\mathsf{gas}} &= c_{i,\text{GOR}} \cdot q_{e,\mathsf{oil}}, \\ q_{e,\text{wat}} &= \frac{c_{i,\text{WCT}}}{100 - c_{i,\text{WCT}}} \cdot q_{e,\mathsf{oil}}. \end{aligned} \tag{3.12}$$

There are four constants in Eq. (3.12) that characterizes a well: the reservoir block pressure $p_{i,\text{res}}$ (which is considered constant according to assumption A1), the productivity index $c_{i,\text{PI}}$, the gas-oil ratio $c_{i,\text{GOR}} \geq 0$, and the water cut $c_{i,\text{WCT}} \in [0, 100)$.

The linear model in Eq. (3.12) does not hold for all reservoirs. For instance, it does not hold for reservoirs with a thin oil rim overlaid by a large gas cap, where wells are subject to gas coning. In coning wells, the gas rate varies nonlinearly with the oil flow rate, and Eq. (3.12) should be substituted with nonlinear relations. These relations may be generated from near-well simulations performed by a reservoir simulator (Mjaavatten et al., 2008).

When temperatures are included in the model, it is customary to assume that the temperature of the entering fluid is constant and equal to the reservoir temperature, that is

$$t_e = \text{const.}, \quad e \in \mathbf{E}^{\text{src}}. \tag{3.13}$$

Reasonably accurate inflow models is a prerequisite for an accurate network model. This part of the model is, however, hard to calibrate. In practise, experiments need to be performed to collect data for inflow model calibration. This usually involves disruptive well testing, where a single well is routed to a test header to allow the measuring of flow rates over a time span of hours.

Downstream boundary conditions

In line with Assumption A1 it is reasonable to assume a constant downstream (separator) pressure when modelling a subsea production system. The constraints

are easily expressed as:

$$p_i = \text{const.}, \quad i \in \mathbf{N}^{\text{snk}}, \tag{3.14}$$

where a constant pressure is specified for the sink (separator) nodes.

3.4.6. Operational constraints

In daily production optimization the production engineers must consider many operational constraints. To obtain solutions with practical value these constraints must be included in the optimization problem. Here we mention two very common operational constraints: namely the capacity and draw-down constraint.

Capacity constraints

In daily production optimization, the topside separator is typically considered to be the downstream boundary of the network. Hence, the amount of fluid entering the separator must honour the water and gas handling capacity of the downstream process facility. The capacity constraints on the total production of gas and water are easily expressed by cut sets (here we have used the set of sink edges $\mathbf{E}^{\text{snk}}$):

$$\sum_{e \in \mathbf{E}^{\text{snk}}} q_{e,\mathsf{gas}} \leq C_{\mathsf{gas}} \text{ and } \sum_{e \in \mathbf{E}^{\text{snk}}} q_{e,\text{wat}} \leq C_{\text{wat}}, \tag{3.15}$$

where the total gas (water) flowing into the separator/sink nodes is limited by the gas (water) handling capacity C_{gas} (C_{wat}).

Draw-down constraints

A draw-down constraint is a lower limit on the bottom hole pressure of a well. The constraint prevents operation at pressures and thereby rates that potentially can damage the well and near-well reservoir. Let $i \in \mathbf{N}$ be a node representing the bottom hole of a well. Then a draw-down constraint on i is expressed with the bounds on p_i: i.e. $p_i^L \leq p_i \leq p_i^U$, where the lower bound p_i^L specifies the draw-down limit.

3.4.7. Objective function

As discussed in the introduction, the main objective when optimizing a petroleum network is typically the maximization of oil production. This objective is easily expressed by summing the oil rates of all edges in a cut set. Two obvious cut sets are the edges leaving a source node ($\mathbf{E}^{\text{src}}$) or the edges entering a sink node ($\mathbf{E}^{\text{snk}}$). Below we express the objective function using the latter.

$$\text{maximize} \quad z = \sum_{e \in \mathbf{E}^{\text{snk}}} q_{e,\mathsf{oil}}, \tag{3.16}$$

Sometimes it makes sense to include contributions to the cost of operating in the objective function; for example the cost of utilizing gas lift or processing produced water. In this framework it is straightforward to include these in the objective.

3.4.8. Flow network: a MINLP formulation

With the complete flow network modelled, the daily production optimization problem is posed as the following mixed-integer nonlinear programming problem:

$$
\begin{aligned}
\underset{\mathbf{y},\mathbf{q},\mathbf{p},\boldsymbol{\Delta}\mathbf{p},\mathbf{t},\boldsymbol{\Delta}\mathbf{t},\mathbf{h},\boldsymbol{\Delta}\mathbf{h}}{\text{maximize}} \quad & z = \sum_{e\in\mathbf{E}^{\text{snk}}} q_{e,\mathsf{oil}} && \\
\text{subject to} \quad & \sum_{e\in\mathbf{E}_i^{\text{in}}} q_{e,s} - \sum_{e\in\mathbf{E}_i^{\text{out}}} q_{e,s} = 0, && \forall s\in\mathbf{S}, i\in\mathbf{N}^{\text{int}} \\
& \Delta p_e = f_e(\mathbf{q}_e, p_i, t_e), && \forall e\in\mathbf{E}\backslash\mathbf{E}^{\text{d}} \\
& \Delta p_e = p_i - p_j, && \forall e\in\mathbf{E}\backslash\mathbf{E}^{\text{d}} \\
& -M_e(1-y_e) \le p_i - p_j - \Delta p_e \le M_e(1-y_e), && \forall e\in\mathbf{E}^{\text{d}} \\
& \Delta t_e = g_e(\mathbf{q}_e, p_i, t_e), && \forall e\in\mathbf{E} \\
& h_e = t_e \sum_{s\in\mathbf{S}} c_s\cdot q_{e,s}, && \forall e\in\mathbf{E} \\
& \Delta h_e = \Delta t_e \sum_{s\in\mathbf{S}} c_s\cdot q_{e,s}, && \forall e\in\mathbf{E} \\
& \sum_{e\in\mathbf{E}_i^{\text{in}}} (h_e - \Delta h_e) = \sum_{e\in\mathbf{E}_i^{\text{out}}} h_e, && \forall i\in\mathbf{N}^{\text{int}} \\
& \sum_{e\in\mathbf{E}_i^{\text{out}}} y_e \le 1, && \forall i\in\mathbf{N}^{\text{d}} \\
& y_e q_{e,s}^L \le q_{e,s} \le y_e q_{e,s}^U, && \forall s\in\mathbf{S}, e\in\mathbf{E}^{\text{d}} \\
& q_{e,s}^L \le q_{e,s} \le q_{e,s}^U, && \forall s\in\mathbf{S}, e\in\mathbf{E}\backslash\mathbf{E}^{\text{d}} \\
& p_i^L \le p_i \le p_i^U, && \forall i\in\mathbf{N} \\
& t_e^L \le t_e \le t_e^U, && \forall e\in\mathbf{E} \\
& \zeta_{i,s}(\mathbf{q}_e, p_i) = 0, && \forall s\in\mathbf{S}, i\in\mathbf{N}^{\text{src}} \\
& p_i = \text{const.}, && \forall i\in\mathbf{N}^{\text{snk}} \\
& t_e = \text{const.}, && \forall e\in\mathbf{E}^{\text{src}} \\
& y_e \in \{0,1\}, && \forall e\in\mathbf{E}^{\text{d}} \qquad (\mathbf{P})
\end{aligned}
$$

In the rest of this work we denote an optimal value of $\mathbf{P}$ by z^*, obtained at an optimal solution $(\mathbf{x}^*, \mathbf{y}^*)$, where $\mathbf{x}$ is a vector containing all the continuous variables in $\mathbf{P}$. Notice that the problem is nonconvex due to the integer variables and the nonlinear equality constraints Eqs. (3.2), (3.6), (3.7), and (3.11). Consequently, we

cannot expect to find a global optimum, unless the problem is solved with a global solver.

The formulation in $\mathbf{P}$ can be used to model any nonlinear flow network under Assumptions A1-A4, and topology requirements R1-R3. For problems not requiring an accurate energy model, a cruder model can be obtained by removing from $\mathbf{P}$ the temperature and enthalpy variables, as well as the constraints for energy conservation. This will remove $3|\mathbf{E}|$ nonconvex constraints, simplifying the problem considerably. For a subsea production network, the framework allows for modelling of gas lifted wells and complex multi-branch wells by the addition of nodes and edges.

A key property of $\mathbf{P}$, which may not present itself immediately, is that the integer variables participate in linear constraints only. This is an advantageous property since the discrete logic may be exclusively handled by the solver. In some cases the nonlinearities are represented by process simulators without the capacity to handle discrete logic.

Another important aspect of the formulation is that it does not contain functions of more than $|\mathbf{S}| + 2$ variables (rates, pressure, and temperature). This allows the nonlinear functions to be replaced with approximations/surrogates of low dimension. In Section 3.5 we show how the nonlinear functions $f(\cdot)$ may be approximated with spline surrogate models. As will become clear later in Section 3.6, this allows us to solve $\mathbf{P}$ to global optimality with a spline-compatible solver.

Before continuing, we would like to remark on the fact that choke openings are not directly computed in $\mathbf{P}$. Chokes are usually modelled with nonlinear Cv curves, relating the choke opening and the differential pressure over the choke. To avoid the additional nonlinearity of the Cv curves when optimizing, the choke are represented by a differential pressure variable (Δp_e). The choke openings are back-calculated from the optimal differential pressures after solving $\mathbf{P}$.

3.5. Spline surrogate models

In this section we give a brief introduction to function approximation with splines. Our purpose is to motivate the use of splines as surrogates for the nonlinear functions in the optimization problem, $\mathbf{P}$. We will use a light notation and represent the splines as basis splines, or *B-splines*. For a detailed treatment of B-splines we refer the reader to the literature on spline theory; cf. the textbooks of Piegl and Tiller (1997) and Schumaker (2007).

3.5.1. Univariate and multivariate B-splines

A spline is a piecewise polynomial function which possesses a required degree of smoothness at the points where the polynomial pieces connect (which are called *knots*). First we consider the univariate B-spline, denoted as

$$\phi_p(x) = \mathbf{c}^\mathsf{T}\mathbf{b}_p(x), \tag{3.17}$$

where $\mathbf{c} \in \mathbb{R}^n$ is a vector of n coefficients and $\mathbf{b}_p \in \mathbb{R}^n$ is a vector of n *B-spline basis functions*.[6] The basis functions in $\mathbf{b}_p$ are (overlapping) p-th degree polynomial pieces in the variables x; see Figure 3.3 for an illustration. They are recursively constructed from a nondecreasing sequence of $n + p + 1$ real numbers $t_1 \leq \ldots \leq t_{n+p+1}$ known as *knots*. These numbers are often collected in a vector $\mathbf{t} = \{t_i\}_{i=1}^{n+p+1}$, called the *knot vector*. Note that with our notation the dependence of $\mathbf{b}_p$, and ϕ_p, on $\mathbf{t}$ is implied. We refer the reader to the literature for a description of the relation between the knots and the basis functions.

The B-spline ϕ_p is a linear combination of basis functions and consequently a piecewise polynomial with degree p. An important property of the B-spline is that it has local support, meaning that at most $p + 1$ basis functions are nonzero at a point x. This, in addition to several other advantageous properties, allow fast and numerically stable methods for manipulation and evaluation of splines; see for example De Boor (1972) and Cox (1972).

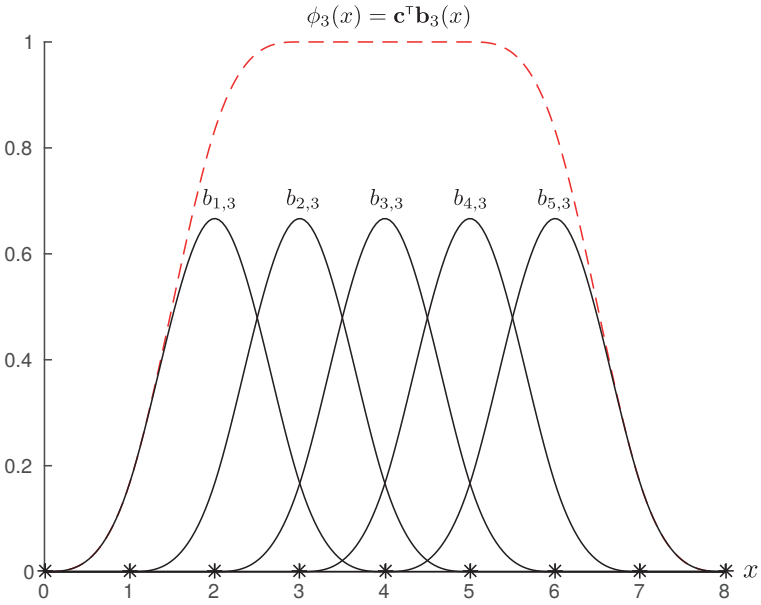


Figure 3.3.: Illustration of the univariate, cubic B-spline basis functions $\mathbf{b}_3 = [b_{i,3}]_{i=1}^5$ for the knot vector $\mathbf{t} = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ (marked with asterisks on the x-axis). The cubic B-spline $\phi_3(x) = \mathbf{c}^\mathsf{T}\mathbf{b}_3(x)$, with coefficients $\mathbf{c}^\mathsf{T} = [1, 1, 1, 1, 1]$, is also shown.

The B-spline generalizes nicely to the multivariate case. A degree p B-spline in the variables $\mathbf{x} \in \mathbb{R}^d$ may be compactly written as

$$\phi_p(\mathbf{x}) = \mathbf{c}^\mathsf{T}\mathbf{B}_p(\mathbf{x}), \tag{3.18}$$

[6] Vectors are assumed to be column vectors unless otherwise stated.

where $\mathbf{B}_p \in \mathbb{R}^N$ is a vector of N multivariate B-spline basis functions of degree p. The multivariate basis functions are obtained by taking the tensor product of univariate basis functions, i.e.

$$\mathbf{B}_p(\mathbf{x}) = \mathbf{b}_p^1(x_1) \otimes \ldots \otimes \mathbf{b}_p^d(x_d) = \bigotimes_{i=1}^{d} \mathbf{b}_p^i(x_i), \tag{3.19}$$

where $\otimes$ denotes the Kronecker product.[7] The Kronecker product produces a vector $\mathbf{B}_p$ that contains all possible combinations of the univariate bases: this results in a total of $N = n_1 \cdots n_d$ multivariate basis functions, where n_i is the number of univariate basis functions in $\mathbf{b}_p^i(x_i)$ in variable x_i. Consider the following example illuminating the Kronecker product: $[a, b] \otimes [c, d] = [ac, ad, bc, bd]$. Transposition is distributed over the Kronecker product so in the case of column vectors we obtain $[a, b]^\mathsf{T} \otimes [c, d]^\mathsf{T} = [ac, ad, bc, bd]^\mathsf{T}$.

Each basis function vector $\mathbf{b}_p^i$ in Eq. (3.19) is parametrized by its own knot vector $\mathbf{t}_i$. Note that a multivariate basis function is a product of d degree p univariate basis functions, making it a multivariate, piecewise polynomial of degree dp.[8] The domain of $\phi(\mathbf{x})$ is considered to be the box $X = X_1 \times \cdots \times X_d$, where X_i is the interval supported by at least one basis function in $\mathbf{b}_p^i(x_i)$. Consequently, $\phi(\mathbf{x}) = 0, \forall \mathbf{x} \notin X$. A bivariate B-spline is illustrated in Figures 3.4 and 3.5.

Most properties of the univariate B-spline carry over to the multivariate case. For example, the multivariate B-spline also enjoys local support and have fast algorithms for manipulation and evaluation (although their implementation require extra care to exploit sparsity patterns). The multivariate B-spline is a powerful modelling and approximation tool, as is testified by the numerous computer-aided design tools that use it. A broad application follows from the fact that the B-spline may represent any piecewise polynomial function exactly, that is, without any approximation error. Models containing non-polynomial functions, such as the transcendental functions, may only be approximated by a B-spline. The approximation error can then be controlled by changing the density of the samples. In the next section we show how to approximate a function that has been sampled on a grid with a B-spline.

3.5.2. Function approximation with B-splines

Let any function $f : \mathbb{R}^d \to \mathbb{R}$ be sampled on a regular grid to yield m data points $\{\mathbf{x}^i, y^i\}_{i=1}^m$, where $f(\mathbf{x}^i) = y^i$. Using only these data points a B-spline that approximates f is constructed. Several approximation methods exist and they are usually categorized as being interpolating or smoothing. Among the interpolating methods

[7] In the literature the multivariate B-spline is often referred to as tensor product B-spline since the basis functions are constructed using the tensor product.

[8] To ease the notation in Eqs. (3.18) and (3.19) we have assumed that all univariate basis functions vectors $\mathbf{b}_p^i$ are of the same degree p. This assumption can easily be removed without any consequences.

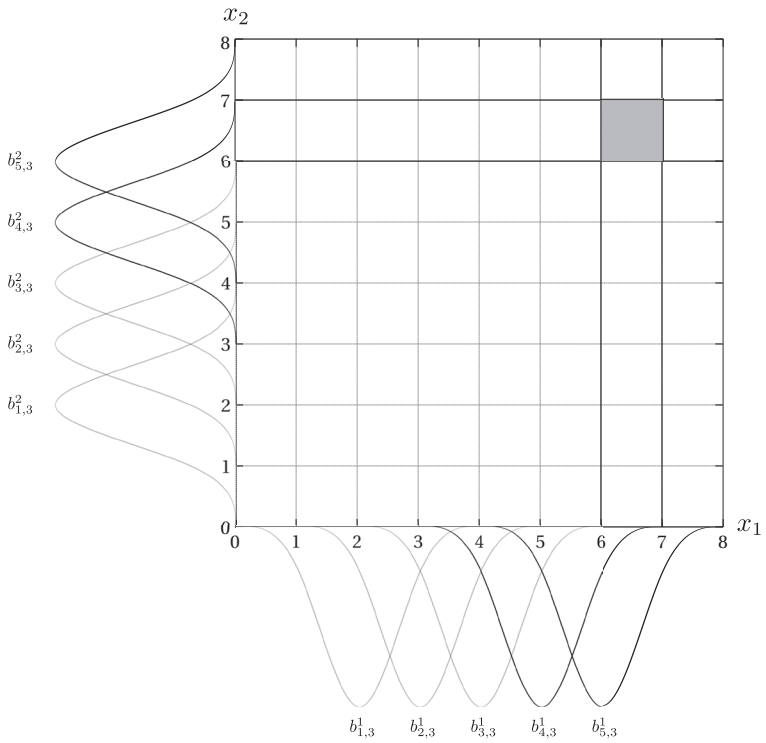


Figure 3.4.: Illustration of the univariate, cubic B-spline basis functions $\mathbf{b}^1_3 = [b^1_{i,3}]^5_{i=1}$ and $\mathbf{b}^2_3 = [b^2_{i,3}]^5_{i=1}$ for the knot vectors $\mathbf{t}_1 = \mathbf{t}_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$. Bivariate, bicubic basis functions are constructed by $\mathbf{B}_p = \mathbf{b}^1_3 \otimes \mathbf{b}^2_3$. The grey box $\mathbf{x} \in [6, 7]^2$ is supported by the following bivariate basis functions: $[b^1_{4,3}, b^1_{5,3}] \otimes [b^2_{4,3}, b^2_{5,3}] = [b^1_{4,3}b^2_{4,3}, b^1_{4,3}b^2_{5,3}, b^1_{5,3}b^2_{4,3}, b^1_{5,3}b^2_{5,3}]$.

the widely used cubic spline interpolation is most common. There are especially three reasons for the popularity of cubic spline interpolation: 1) it is fast to compute, 2) it offers a high degree of smoothness, and 3) it is a good approximation to a broad class of functions.

An interpolation method computes a B-spline that interpolates f at all of the m data points. Mathematically, the following linear system is solved for the coefficients $\mathbf{c}$:

$$\underbrace{\left[\mathbf{B}_p(\mathbf{x}^1)\ \mathbf{B}_p(\mathbf{x}^2)\ \ldots\ \mathbf{B}_p(\mathbf{x}^m)\right]^\mathsf{T}}_{\mathsf{B}_c}\mathbf{c} = \mathbf{y} \tag{3.20}$$

In Eq. (3.20) $\mathbf{y} = [y^i]^m_{i=1}$ and $\mathsf{B}_c \in \mathbb{R}^{m\times N}$ is the so-called *B-spline collocation matrix*: the matrix where row i corresponds to the vector of basis functions $\mathbf{B}_p(\mathbf{x}^i)$ evaluated at sample $\mathbf{x}^i$. It is customary to select a knot vector that gives a square collocation matrix ($N = m$). An example of such a knot vector is the *free end*

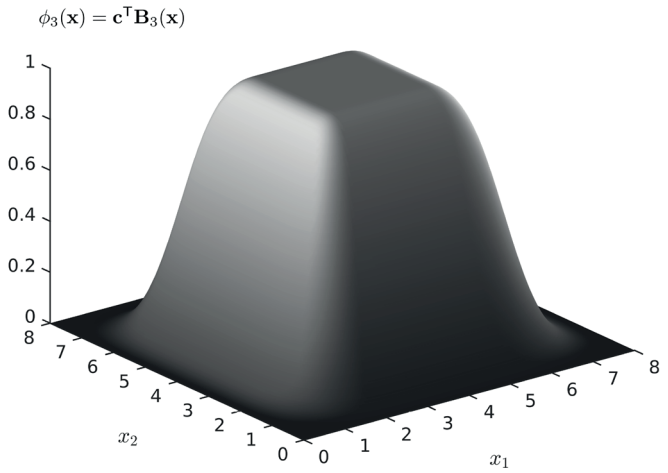


Figure 3.5.: A bivariate, cubic B-spline constructed from the basis functions in Figure 3.4 and the coefficients $\mathbf{c} = \mathbf{1}_{25}$, where $\mathbf{1}_{25} \in \mathbb{R}^{25}$ is a vector of $5 \times 5 = 25$ ones.

conditions knot vector for cubic spline interpolation ($p = 3$):

$$\mathbf{t}_F = \{ \underbrace{x^1, \ldots, x^1}_{p+1 \text{ repetitions}}, x^3, \ldots, x^{m-2}, \underbrace{x^m, \ldots, x^m}_{p+1 \text{ repetitions}} \}.$$

Notice that the second and second last knot is omitted from $\mathbf{t}_F$ to give $N = m$. For square B_c, the conditions under which B_c is invertible are known as the *Schoenberg-Whitney nesting conditions*: $t_i < x^i < t_{i+p+1}$ for $i = 1, 2, \ldots, m$, allowing $x^i = t_i$ only if $t_i = t_{i+p} < t_{i+p+1}$. These conditions are fulfilled for $\mathbf{t} = \mathbf{t}_F$. When B_c is square and invertible, the B-spline coefficients can readily be computed by solving $\mathsf{B}_c\mathbf{c} = \mathbf{y}$.

The $m \times m$ linear system in Eq. (3.20) can be solved efficiently by a sparse solver on a modern desktop computer for $m \leq 100,000$. For example, when approximating a function in 5 variables this practical limit allows a discretization with 10 values in each variables, resulting in a grid of $10^5 = 100,000$ sample points. For a "well-behaving", low-dimensional function such as a well lift curve, 100,000 samples is more than enough to achieve an accurate approximation. In this work we have utilized the SPLINTER library for function approximation (Grimstad et al., 2015b).

Before we illustrate cubic spline interpolation in the next subsection, we point out that the approximation error of a spline can be made arbitrarily small for continuous functions by increasing the sampling density. Furthermore, the approximation error is dependant on the knot placement. Optimal knot placement, however, is a difficult

and largely unresolved problem, particularly for multivariate B-splines (Natali and Pinto, 2009). Luckily, when samples are taken on a regular grid the knots can be set equal to the sample points as in $\mathbf{t}_F$.[9] With scattered (irregular) sample points it is not trivial to select the knot vectors. In either case, the problem of where to place the sample points still remains and is highly dependant on the function to be approximated.

3.5.3. An example: Beggs and Brill approximated with a B-spline

Figure 3.6 shows the Beggs and Brill pressure drop correlation for a slightly inclined pipe. With the given parameters, the correlation includes three different flow regimes on the domain. By inspecting the figure one may observe several bends in the correlation; the groove between the segregated and transition flow regime is conspicuous.

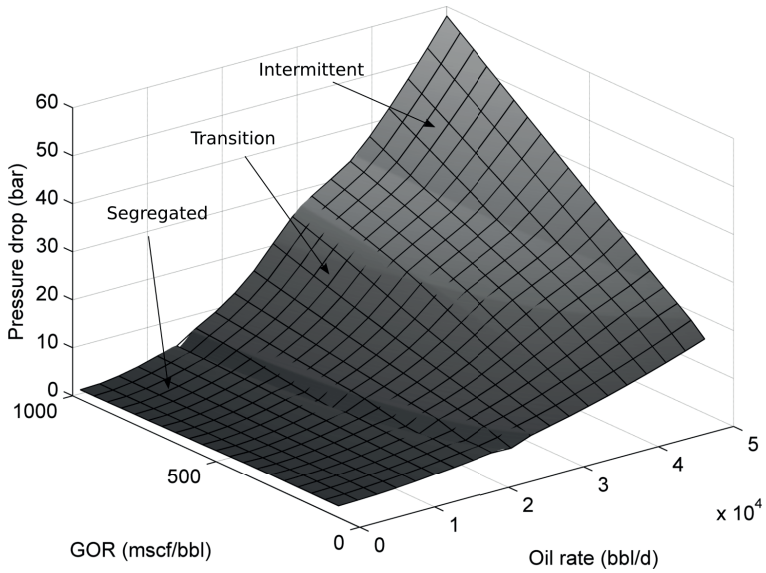


Figure 3.6.: Beggs and Brill pressure drop correlation for a 1 degree inclined, 1000 meter long, 12 inch pipe. The water-cut is fixed at 10% and the outlet pressure is 30 bar. As indicated, three different flow regimes occur as the oil rate increases. The overlaying grid shows the sample points used to build the spline approximations.

The 20x20 grid in Figure 3.6 shows the $m = 400$ points where the correlation was sampled. From these points two approximations are constructed: a linear spline ($p = 1$) and a cubic spline ($p = 3$). The approximation error of the two splines are plotted in Figure 3.7.

[9]By default, SPLINTER computes the knots by applying a moving average filter with window size $p + 2$ to the sample points. With equidistant samples this filter produces a knot vector equivalent to $\mathbf{t}_F$.

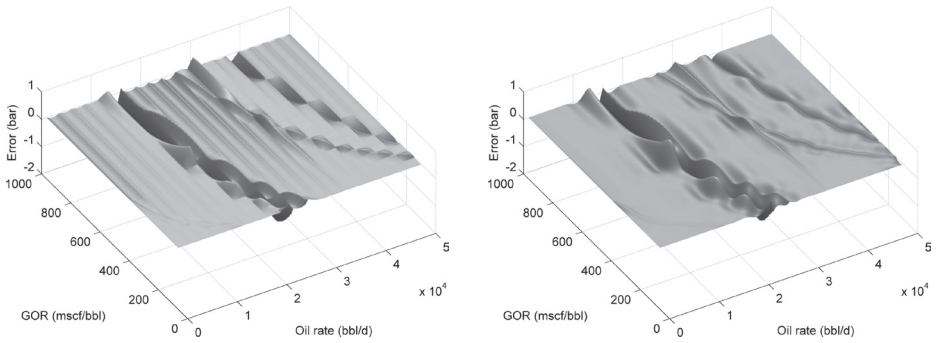


Figure 3.7.: The approximation error of the linear spline (a) and the cubic spline (b) interpolating the Beggs and Brill pressure drop correlation on the equidistant 20x20 grid in Figure 3.6.

By inspecting the figure we see that qualitatively the approximation errors of the two interpolating splines are similar. The error increases along the diagonal grooves/bends where the rectangular grid fails to capture the geometry. The most notable difference between the two splines are the ripples in Figure 3.7a. The ripples show that the linear spline fails to capture the curvature between the grid lines; this effect is not present to the same degree in Figure 3.7b.

To quantitatively compare the interpolating splines we measure the relative approximation error with $\epsilon_{\text{rel},2} = ||1 - \phi(\mathbf{x})/f(\mathbf{x})||_{2,X}$ and $\epsilon_{\text{abs},\infty} = ||f(\mathbf{x}) - \phi(\mathbf{x})||_{\infty,X}$ for $\mathbf{x} \in X$, where $||\cdot||_{p,X}$ denotes the L^p-norm on the domain X. The errors for the two splines are given in Table 3.5.

Table 3.5.: Spline approximation errors.

	$\epsilon_{\text{rel},2}$	$\epsilon_{\text{abs},\infty}$
Linear spline	0.0044	1.1193
Cubic spline	0.0039	1.0545

3.6. Solution method

In this section we present the proposed method for solving $\mathbf{P}$. The main assumption is that *all* the nonlinear functions in $\mathbf{P}$ are B-spline functions on the form in Eq. (3.18), that is, we assume that $f_e(\cdot)$, $g_e(\cdot)$, and $\zeta_{i,s}(\mathbf{q}_e, p_i)$ are B-spline functions. The bilinear terms in the enthalpy constraints in Eq. (3.7) may also be represented exactly with splines.[10] With this assumption problem $\mathbf{P}$ falls into the category

[10] A spline may represent a bilinear term exactly. In fact, the convex hull relaxation of the B-spline is identical to the McCormick relaxation for bilinear terms (Grimstad and Sandnes, 2015).

of *spline-constrained* MINLP problems, which may be solved to global optimality by the spline-compatible optimization framework CENSO (Convex ENvelopes for Spline Optimization), recently presented by Grimstad and Sandnes (2015). CENSO is publicly available as open-source C++ code (Grimstad et al., 2015a). A description of the algorithm is given in the next section. Please note that without loss of generality we assume $\mathbf{P}$ to be a minimization problem in this description. After the description, we present two improvements that may speed up the algorithm when solving production optimization problems.

3.6.1. Description of CENSO

CENSO is a framework for optimization with spline constraints. It contains a spatial branch-and-bound (sBB) algorithm that partitions the problem domain by branching on continuous variables as well as integer variables; this produces subproblems $\mathbf{P}_k$ of $\mathbf{P}$. Spline (equality) constraints are generally nonlinear, thus nonconvex, and must be relaxed during the solution process. CENSO employs lifted polyhedral sets to relax spline constraints, producing relaxed LP subproblems, denoted $\mathbf{R}_k$.

Let $\hat{z}_k$ denote the solution to $\mathbf{P}_k$ and $\bar{z}_k$ the solution to $\mathbf{R}_k$. The fact that $\hat{z}_k \geq \bar{z}_k$ is used to process the search domain. The current best feasible solution found, known as the incumbent, is denoted z^u. The algorithm described next, terminates when it has proved that there cannot exist a solution better than $z^u - \epsilon$ (where $\epsilon > 0$ is a small number). This is known as ϵ-convergence.

The schematic in Figure 3.8 describes the sBB algorithm in CENSO. From the top: the algorithm is initialized by adding $\mathbf{P}$ to the list of problems $\mathcal{L}$, and setting the upper bound on $\mathbf{P}$ to $z^u = \infty$. The algorithm then enters a loop which terminates when $\mathcal{L}$ is empty. Upon termination there are two possible outcomes: a global optimum has been found (ϵ-convergence) or the problem is infeasible.

The first step inside the loop is to select and remove from $\mathcal{L}$ the next subproblem $\mathbf{P}_k$ to be processed. The sBB in CENSO uses a simple *best-bound-first* policy, selecting the subproblem with the lowest lower bound $\bar{z}_k$ (inherited from its parent node).

After selection, bounds tightening techniques are applied to $\mathbf{P}_k$. The purpose of these techniques is to reduce the domain of $\mathbf{P}_k$ and hence to accelerate the exploration of the search space. These methods may also prove $\mathbf{P}_k$ infeasible, in which case it is fathomed.

Next, the convex relaxation $\mathbf{R}_k$ is generated and solved to get $\bar{z}_k$. By default, the convex relaxation is solved by Gurobi (Gurobi Optimization, Inc., 2014). With the lower bound on $\mathbf{P}_k$ three fathoming rules are checked: (i) $\bar{z}_k \geq z^u$, (ii) $\bar{z}_k = \infty$ ($\mathbf{R}_k$ infeasible), and (iii) $z^u - \bar{z}_k \leq \epsilon$ (converged). If any of (i)–(iii) are true, the node is fathomed as it may not contain a solution better than $z^u - \epsilon$.

If the subproblem cannot be fathomed its domain needs further processing. First, the incumbent is updated by checking if the solution to $\mathbf{R}_k$ is feasible to $\mathbf{P}_k$. To further improve the incumbent an NLP or MINLP solver may be used to find a

feasible solution of $\mathbf{P}_k$ that is better than z^u. This heuristic is not required, but may speed up the convergence of the search.

Finally, at the end of the loop a continuous or integer branching variable is selected for $\mathbf{P}_k$. This variable is then branched on to create two new partitions $\mathbf{P}_{k-}$ and $\mathbf{P}_{k+}$. The two partitions are added to the list $\mathcal{L}$, completing one iteration of the loop. Note that after one loop iteration the list size $|\mathcal{L}|$ is either decremented by one (if $\mathbf{P}_k$ fathomed) or incremented by one (if $\mathbf{P}_k$ is branched on). If the list is empty, the search terminates with the optimal solution z^u.

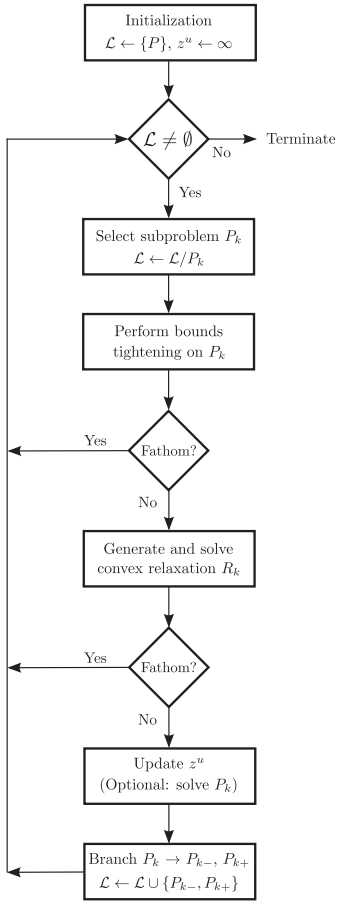


Figure 3.8.: The spatial bound-and-bound algorithm in CENSO.

3.6.2. Branching variables and bounds tightening

A requirement for the sBB algorithm to converge to a global optimum is that it may branch on all *complicating variables*. In a MINLP problem, the complicating

variables are the integer variables and any continuous variable that participates *nonlinearly* in a nonconvex constraint. For problem $\mathbf{P}$, the complicating variables are $\mathbf{y}$ and $\mathbf{x}_c$, where $\mathbf{x}_c = [x_i]_{i\in\mathcal{I}_c}$ are the complicating continuous variables. The index set $\mathcal{I}_c$ is given so that $\{x_i : i \in \mathcal{I}_c\} = \{\mathbf{q}_e, p_i, t_e, \Delta t_e : e = (i,j) \in \mathbf{E}\}$. Note that $\mathbf{x}_c$ does not contain the variables Δp_e, h_e, and Δh_e since they participate linearly in all constraints (the reader can verify this by looking at $\mathbf{P}$).

A continuous branching variable, like an integer branching variable, must be branched on a finite number of times to ensure ϵ-convergence. However, the required number of branches may be large depending on the value of ϵ and the convergence rate of the convex relaxations. At any rate, it is highly desirable to keep the number of continuous branching variables at a minimum.

Problem $\mathbf{P}$ has a relatively large number of nonconvex constraints and, as a result, $\mathbf{x}_c$ contains most of the continuous variables. From computational experience, we know that branching on all of the variables in $\mathbf{x}_c$ is detrimental to the efficiency of the algorithm, even for small network problems. To alleviate the computational load we employ so-called *bounds tightening*.[11]

Bounds tightening (BT) are techniques that reduce the variable bounds $[\mathbf{x}^L, \mathbf{x}^U]$ of a problem without removing its optimal point. BT techniques with this property are said to be *valid* or to produce *valid inequalities*. BT will shrink the feasible set of the primal problem and its convex relaxation. In some cases it may also prove a problem infeasible. All BT techniques utilize the constraints to, in some way, reduce the variable bounds.

Let us illustrate the advantage of bounds tightening with a simple example: Let $x_1 \in [x_1^L, x_1^U]$ and $x_2 \in [x_2^L, x_2^U]$ be continuous branching variables, related via the constraint $x_1 - x_2 = 0$. Bounds tightening will propagate the variable bounds through the constraint and ensure that $x_1, x_2 \in [\max\{x_1^L, x_2^L\}, \min\{x_1^U, x_2^U\}]$. Thus, when bounds tightening is applied, it is sufficient to branch on one of the two variables: e.g. branching on x_1 will reduce the feasible range of x_2, and vice versa.

Immediately, we understand that if we branch on the variables associated with the degree of freedom in $\mathbf{P}$, bounds tightening will ensure diminishing bounds on the remaining (branching) variables. Let $\bar{\mathbf{x}}_c = [p_i]_{e=(i,j)\in\mathbf{E}^{\mathrm{d}}}$ be the $|\mathbf{E}^{\mathrm{d}}|$ free, continuous variables in $\mathbf{P}$ (according to the DOF analysis in 3.A). Then, it is sufficient to branch on $\mathbf{y}$ and $\bar{\mathbf{x}}_c$. It is clear that $\bar{\mathbf{x}}_c \in \mathbb{R}^{|\mathbf{E}^{\mathrm{d}}|} \subset \mathbf{x}_c \in \mathbb{R}^{(|\mathbf{S}|+3)|\mathbf{E}|}$. To be more precise, with bounds tightening the number of continuous branching variables is reduced from $(|\mathbf{S}|+3)|\mathbf{E}|$ to $|\mathbf{E}^{\mathrm{d}}|$. The reduction in the number of continuous branching variables limits the tree size and accelerates the solution time of the sBB algorithm, even when accounting for the additional computational load of the bounds tightening techniques. Next, we briefly describe the bounds tightening capabilities of CENSO.

[11] Techniques that use the constraints to reduce the variable bounds have several names in the literature, including: bounds tightening, range reduction, and interval analysis.

Bounds tightening techniques in CENSO

CENSO employs the following BT techniques for MINLP problems: the *reduced-cost* BT (RCBT), originally introduced for MILP problems (Ryoo and Sahinidis, 1996; Belotti et al., 2009); and *feasibility-based* BT (FBBT) (Messine, 2004; Belotti et al., 2010). These are computationally cheap techniques that perform tightening by propagating variable bounds through the constraints. They can be solved at any node in the sBB tree, but generally produce shallow cuts. To improve upon the BT capabilities of CENSO we implement the optimality-based BT (OBBT) technique used by Zamora and Grossmann (1999) and Sahinidis (2003).

With OBBT a relaxed problem $\mathbf{R}_k$ is solved with the objective to minimize or maximize one variable. This is done for each complicating variable in $\mathbf{x}_c$. Let $F(\mathbf{R}_k)$ represent the (convex) feasible region of $\mathbf{R}_k$. Then the convex problems solved by the OBBT are

$$\underset{\mathbf{x}}{\text{minimize}} \ \{\pm x_i : \mathbf{x} \in F(\mathbf{R}_k),\, z \leq z^u\}, i \in \mathcal{I}_c. \tag{3.21}$$

Let $\tilde{x}_i^L$ and $\tilde{x}_i^U$ be the solutions for the two objectives in Eq. (3.21) for variable x_i. Then the new bounds on x_i are $[x_i^L, x_i^U] \cap [\tilde{x}_i^L, \tilde{x}_i^U]$. If any of the problems in Eq. (3.21) are infeasible, problem $\mathbf{R}_k$ must be infeasible and can therefore be fathomed.

The OBBT requires the solution of $2|\mathcal{I}_c|$ convex NLPs or LPs; when all non-linearities of $\mathbf{P}$ are represented with B-splines the relaxed problems $\mathbf{R}_k$ are LP problems. The OBBT may be run iteratively to achieve a greater tightening of the bounds: tighter variable bounds produce tighter convex relaxations, which in turn produce tighter variable bounds. Running OBBT iteratively is expensive and yields diminishing returns. However, it may greatly reduce the size of the BB tree. Therefore, it is typically used on every subproblem down to a certain depth, and to a limited extent deeper in the BB tree.

3.6.3. Primal heuristic

At the root node of the sBB tree, the MINLP solver BONMIN (Bonami et al., 2008) is evoked to search for a feasible solution to $\mathbf{P}$. If successful, the (primal) feasible solution, being an upper bound on the solution of $\mathbf{P}$, may help in cutting large portions of the sBB tree. BONMIN is a heuristic in this setting since it is used to find a local optimum to the nonconvex problem $\mathbf{P}$.

3.7. Case studies

In this section we present a benchmark study of the solution methods in Table 3.6. The study includes three realistic production optimization cases from two BP operated subsea production systems, referred to as BP subsea production system 1 and 2, from here on. Note that these cases do not necessarily correspond to the normal operation of the production system. The cases are based on models

implemented in the GAP software from Petroleum Experts (Petroleum Experts Ltd., 2014). The GAP models serve as reference models when comparing solutions generated from the four different solution methods.

The four solution methods in Table 3.6 may be described as follows. The first is a traditional approach where a proprietary gradient-based multi-start NLP solver treats the GAP model as a black-box model, calculating gradients by finite differences. The three other methods formulate problem **P** by approximating the nonlinear relations in the production system models with splines. Problem **P** is then solved using IPOPT, BONMIN, and CENSO, respectively. The branch-and-bound-based MINLP solvers BONMIN and CENSO may solve problem **P** with discrete edges (discrete variables). The proprietary solver and IPOPT, being NLP solvers, cannot handle discrete variables.

Table 3.6.: Solution methods.

Solver	Type	Routing	Global	Model
Proprietary solver	NLP	No	No	GAP
IPOPT (Wächter and Biegler, 2006)	NLP	No	No	**P**
BONMIN (Bonami et al., 2008)	MINLP*	Yes	No	**P**
CENSO (Grimstad and Sandnes, 2015)	MINLP**	Yes	Yes	**P**

*Convex MINLPs, **Spline constrained MINLPs.

CENSO solves problem **P** to global optimality and provides an optimality certificate with the solution, i.e. the optimality gap is less than ϵ upon termination. Solving a MINLP problem to global optimality is considerably harder, and more time consuming, than attempting a local solve. To illustrate the difference, the cases were solved to local optimality using BONMIN. To improve BONMIN's chances of finding good solutions of the nonconvex problems it was configured with the following options: algorithm set to "B-BB" (standard branch-and-bound mode), num_resolve_at_root set to 10, and num_resolve_at_node set to 2. This allows BONMIN to solve the nonconvex subproblems in the BB tree from several starting points; the starting points are naively drawn from a uniform distribution limited by the variable bounds. All other options were left at their default values.

There are a few key differences between the solution methods described above that complicates comparison of the methods. First of all, the proprietary solver and IPOPT cannot handle discrete variables. Thus, we include them in the comparison only when all discrete decisions are fixed. Second, the proprietary method solves a different model/optimization problem than the other methods since it uses the GAP model directly. To achieve a somewhat fair comparison, the optimal solutions are compared by evaluating GAP at the optimal valve settings.

The three last methods in Table 3.6 were run on a laptop computer equipped with an Intel 2.7 GHz dual-core processor and 8 GB of RAM memory. The proprietary solver was run on another computer with favourable performance.

Note that the reported solution times do not include the time it took to build the splines in $\mathbf{P}$. The timings for building the splines are reported at the end of this section.

3.7.1. Case 1: Production optimization of BP subsea production system 1

In this case we consider the subsea production system depicted in Figure 3.9. The production system consists of 10 wells, 4 daisy-chained manifolds, 4 flowlines and 1 riser. The system is modelled with three fluid phases, i.e. $\mathbf{S} = \{\mathsf{oil}, \mathsf{gas}, \mathrm{wat}\}$.

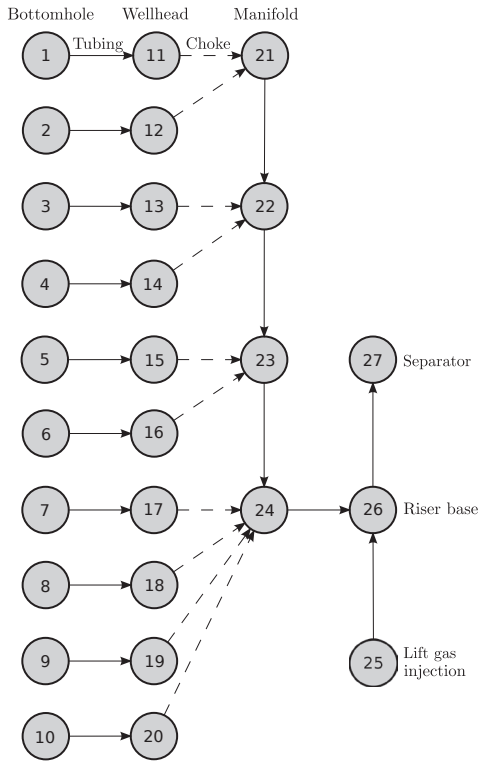


Figure 3.9.: Topology of BP subsea production system 1.

As shown in the figure (Node 25), lift gas can be injected into the riser base to increase production by lowering the density of the fluid column. This is achieved by modelling Node 25 with the rate boundary conditions $q_{e,\mathsf{oil}} = 0$ and $q_{e,\mathrm{wat}} = 0$. The amount of lift gas injected into the riser is given by $q_{e,\mathsf{gas}} \in [0, 20]$ mmscf/d. To simplify the model, the lift gas is assumed to have the same composition as the produced gas. The total gas production is limited to 340 mmscf/d, which is the gas handling capacity of the downstream processing facilities.

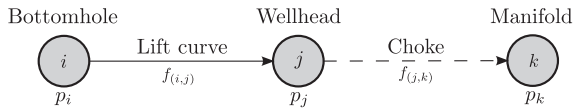


Figure 3.10.: A well modelled with three nodes and two edges.

The wells are modelled by connecting three nodes $\{i, j, k\}$ with two edges, (i, j) and (j, k). In this configuration, depicted in Figure 3.10, the nodes are labeled as follows: bottomhole (i), wellhead (j), and manifold (k). The bottomhole node (i) is a source node with no incoming edges. The inflow from the reservoir to the well is modelled by a nonlinear IPR (a piecewise function composed of a straight line and Vogel's equation), with a fixed GOR and WCT. The well parameters are listed in Table 3.7. The pressure drop over the wellbore, edge (i, j), is described by a nonlinear lift curve $f_{(i,j)}$ relating the flow rates $\mathbf{q}_{(i,j)}$ to the wellhead pressure p_j. The choke is described by the edge (j, k), with a related pressure drop $\Delta p_{(j,k)}$.

Table 3.7.: Well parameters.

Well	PI (rank #)	GOR (scf/STB)	WCT (%)	p_i^L (bara)
1	6	1100	15	190
2	3	800	25	200
3	4	800	40	110
4	1	800	55	120
5	10	600	55	120
6*	2	700	50	0
7*	8	700	25	0
8	5	700	30	210
9	7	700	0	210
10	9	800	0	170

* Well is offline. ** Values are rounded for commercial reasons.

The system is modelled without any energy considerations, i.e. temperature and enthalpy variables, and related constraints, are not included in the problem formulation **P**. This reduces the number of nonconvex constraints and (complicating) variables, and hence the complexity of the problem.

We divide Case 1 into two parts. In Case 1.1 and Case 1.2 the nonlinearities are represented with linear and cubic interpolating splines, respectively. The interpolating splines are constructed by solving Eq. (3.20), with degree $p = 1$ for Case 1.1 and $p = 3$ for Case 1.2. Since the case includes discrete edges (binary variables) it is only solved with BONMIN and CENSO.

The results for the cases are reported in Table 3.8. Evidently, the cases are solved efficiently by both solvers. The solution time of CENSO is strictly higher than that of BONMIN, as is it must be since it runs BONMIN as a primal heuristic. The

Table 3.8.: Results for Case 1.

Case, Solver	Iterations (#)	Time (s)	z^* (mSTB/d)
Case 1.1 (linear splines)			
BONMIN	0	2	77.483
CENSO	9	56	77.483
Case 1.2 (cubic splines)			
BONMIN	0	5	78.381
CENSO	17	191	78.381

number of iterations used by CENSO is kept low by intensive bounds tightening, while BONMIN terminates with the global optimum after examining the root node only; i.e. it uses 0 iterations in both cases. However, BONMIN does not terminate with an optimality certificate, like CENSO.

Notice that the optimal value of the two cases differ with almost 1 mSTB/d. The difference is due to the linear and cubic spline interpolation of the pressure drop curves since the pressure drop curves have a positive curvature (convex-like curves). This curvature is captured by the cubic spline, but is over-estimated by the linear spline (piecewise linear) interpolation. Consequently, the higher pressure drop causes a lower production for a fixed separator pressure. The two optimal solutions do however give the same optimal valve settings.

The 11 active constraints at the optimal solution are listed in Table 3.15. The case has 10 wells, and one additional source node for gas lift, giving 11 DOF (when all binary variables are fixed). Hence, there are 11 active constraints at the optimal solution (in addition to 10 fixed binary variables).

At the optimal solution Well 4-7 are offline. Well 6 and 7 are set offline. Well 4 and 5 have a WCT above 50% and it is not unexpected that they are offline at the optimal solution. All online wells operate at the minimum choke differential pressure, meaning that the system is pressure constrained – the gas capacity constraint is not active and maximum gas lift is used.

To investigate the approximation error of problem $\mathbf{P}$ to the GAP model we insert the optimal valve settings into GAP and record the pressures and rates it predicts. The relative errors between the variables in GAP and Case 1.2 (cubic splines) are reported in Table 3.9. Most of the errors are below 1%, which is satisfactory. We do observe some propagation of error along the flowlines, and for the riser the pressure loss error is almost 4%. This may be improved upon by sampling the flowline pressure drop more densely, and accepting a higher computation time.

Table 3.9.: Validation using GAP of solution from CENSO on Case 1.2.

Edge e	Error (%)		
	$q_{e,\mathsf{oil}}$	Δp_e	p_j
Wells			
(1,11)	0.26	0.18	0.14
(2,12)	0.23	0.13	0.11
(3,13)	0.21	0.10	0.09
(4,14)*	0	0	0
(5,15)*	0	0	0
(6,16)*	0	0	0
(7,17)*	0	0	0
(8,18)	0.91	0.42	0.40
(9,19)	0.47	0.38	0.37
(10,20)	1.18	0.44	0.42
Flowlines			
(21,22)	0.24	0.62	0.05
(22,23)	0.23	2.70	0.28
(23,24)	0.23	1.35	0.39
(24,26)	0.55	3.84	0.55
(26,27)	0.55	0.81	0**

* Edge is closed (well is offline) at the optimal solution.
** The separator pressure is fixed.

3.7.2. Case 2 and 3: Production optimization of BP subsea production system 2

In these cases we consider the production system drawn in Figure 3.11. The system has 13 wells, 5 flowlines, and 2 risers. Four of the wells can be routed to either of the risers. The two risers are named as follows: edge $(48, 51)$ is the east (E) riser and edge $(50, 51)$ is the west (W) riser. We refer to the flow path $44 \to 46 \to 48 \to 51$ as the E loop and the flow path $45 \to 47 \to 49 \to 50 \to 51$ as the W loop.

For brevity we assign numbers to the wells so that well i represents the well with bottomhole node index i, although the well consists of several edges and nodes. The wells are modelled using a nonlinear IPR (a piecewise function composed of a straight line and Vogel's equation), with a fixed GOR and WCT. The well parameters are listed in Table 3.10. As in the previous case, we model the system with three fluid phases, i.e. $\mathbf{S} = \{\mathsf{oil}, \mathsf{gas}, \mathrm{wat}\}$.

The network has a total of 17 discrete edges for routing and shutting in wells. To mimic the current field operation wells 9, 10, and 13 are set offline. This leaves $2^{14} = 16,384$ routing and well status combinations. By considering the manifold

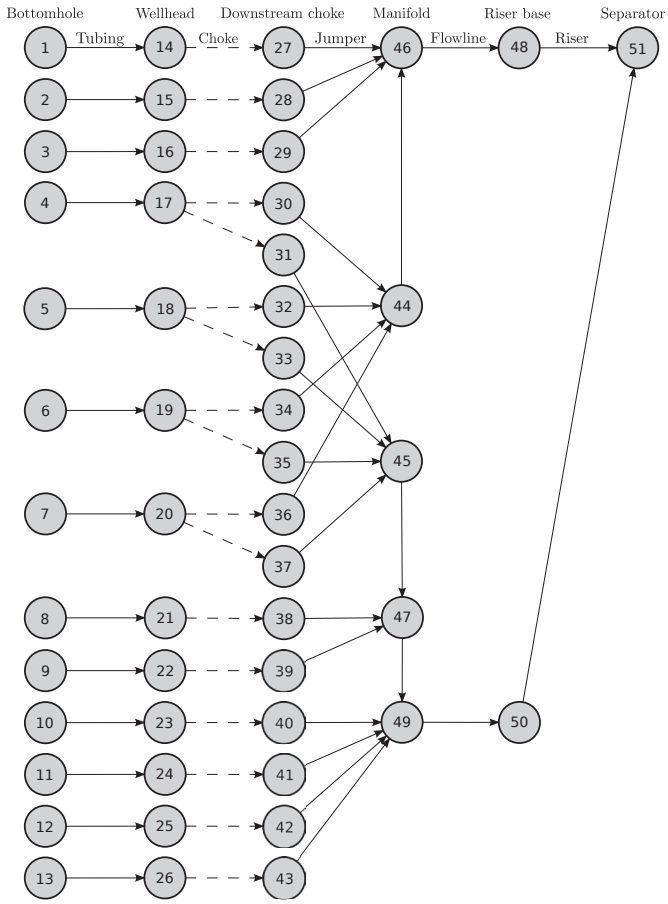


Figure 3.11.: Topology of BP subsea production system 2.

routing constraints in Eq. (3.10) we find that the number of feasible combinations is $2^6 \cdot 3^4 = 5,184$.

In addition to the common constraints described in Section 3.4, this case has two special constraints. Each riser has a maximum mix velocity constraint (the mix velocity is the sum of the in-situ liquid and gas velocity). These constraints limit the erosion of the risers' inner tube coating due to high velocity sand particles. The mix velocity of riser $e \in \{(48, 51), (50, 51)\}$ is modelled as $v_e(\mathbf{q}_e, p_j, t_e + \Delta t_e) \leq v_e^U$, where v_e^U is the upper velocity limit. Notice that the downstream pressure p_j and temperature $t_e + \Delta t_e$ are used since the velocity is calculated at the outlet, where it invariably attains its maximum value. To accurately express these important constraints, temperature and enthalpy variables are included in the formulation.

To benchmark how the model complexity added by the discrete decisions and temperature variables affect the computation time, we solve Case 2 and 3 with the

Table 3.10.: Well parameters for Case 2 and 3.

Well i	PI (rank #)	GOR (scf/STB)	WCT (%)	p_i^L (bara)
1	10	2400	45	280
2	5	900	0	190
3	8	4500	30	220
4	2	1500	65	0
5	3	1800	35	220
6	11	3300	0	210
7	6	2900	5	220
8	12	4200	0	220
9*	1	0	0	0
10*	7	0	0	0
11	9	900	20	200
12	4	900	0	200
13*	13	0	0	0

* Well is offline. ** Values are rounded for commercial reasons.

Table 3.11.: Configurations in Case 2 and 3.

Case	Well status	Well routing	Energy balances	Riser velocity constraints
Case 2.1	–	–	–	–
Case 2.2	✓	–	–	–
Case 2.3	✓	✓	–	–
Case 3.1	–	–	✓	✓
Case 3.2	✓	–	✓	✓
Case 3.3	✓	✓	✓	✓

Yes: ✓, No: –

various configurations described in Table 3.11. Note that in Case 2.1 and Case 3.1 the status and routing of the wells are set to the state of the current field operation.

Case 2: Optimization without energy balances

In Case 2 problem **P** is solved without temperature and enthalpy variables ($\mathbf{t}, \mathbf{\Delta t}, \mathbf{h}, \mathbf{\Delta h}$), and without the energy conservation constraints in Eqs. (3.6), (3.7), and (3.8). Since the riser mix velocity constraints cannot be modelled without temperature variables, they are also excluded from the problem. The nonlinear relations for pressure drop and boundary conditions are modelled with cubic splines.

The results of Case 2 are reported in Table 3.12. Several interesting observations can be made from the results. First of all, it is clear that allowing wells to be shut

in or re-routed may only increase the optimal value. The optimal value of Case 2.2 is therefore higher than Case 2.1, but lower than that of Case 2.3. Next, the local solvers seem to find good optimal solutions. In fact, in all cases except Case 2.1, all solvers are able to locate the global optimum. This may be attributed to the problem formulation in **P**, and the smoothness of the cubic splines. Finally, the exponential increase in computation time becomes distinct when globally solving Case 2.3 with CENSO.

Table 3.12.: Results for Case 2.

Case, Solver	Iterations (#)	Time (s)	z^* (mSTB/d)
Case 2.1			
Proprietary solver	–	9000	143.139
IPOPT	37	0.1	143.435
BONMIN	0*	5	143.435
CENSO	30	280	143.435
Case 2.2			
BONMIN	761	44	143.763
CENSO	19	314	143.763
Case 2.3			
BONMIN	4386	146	143.875
CENSO	89	1870	143.875

* Problem has no integer variables and is solved at the root node.

In Case 2.3, CENSO finds the solution $z^* = 143.875$ mSTB/d. This solution is verified by running the GAP model with the optimal valve settings. This gives a production of 143.936 mSTB/d; a relative difference of 0.04%. Compared to the optimal solution from the proprietary solver in Case 2.1, the increase in production is 0.56%.

The active constraints at the optimal solution of Case 2.3 are listed in Table 3.16. As indicated in the table, Well 3 is offline in the optimal solution. This is not surprising since it is a weak producer and the well with the highest GOR. Well 8, having the second highest GOR, is choked to hit the gas capacity constraint on the total gas production. The rest of the wells operate at maximum capacity, i.e. at their draw-down pressure or minimum choke differential pressure.

Case 3: Optimization with energy balances

The full Problem **P** is solved with temperature and enthalpy variables. The previously described riser mix velocity constraints are included in the problem formulation to guard against solutions susceptible to high erosion rates.

Note that Case 3 includes all constraints of Case 2, in addition to the energy balances and riser mix velocity constraints. Thus, the optimal values in Case 2 are necessarily lower bounds on the optimal values in Case 3.

The results for Case 3 are presented in Table 3.13. In Case 3.1 IPOPT, BONMIN, and CENSO finds the same (globally) optimal solution. The same solution is found by BONMIN and CENSO in Case 3.2. As reported, CENSO requires more than 1 hour to find and certify a global optimum. In Case 3.3, BONMIN fails to locate the same or a better optimum than Case 3.2 and 3.1. This happens because BONMIN mistakenly cuts away the optimum during its search. By comparing Case 3.1 and 3.2 we notice that, as in Case 2, the option to turn off wells does not seem to have a large impact on the solution time.

Table 3.13.: Results for Case 3.

Case, Solver	Iterations (#)	Time (s)	z^* (mSTB/d)
Case 3.1			
Proprietary solver	–	9000	136.400
IPOPT	124	0.6	140.674
BONMIN	0*	32	140.674
CENSO	59	2630	140.674
Case 3.2			
BONMIN	2432	387	140.674
CENSO	11	3670	140.674
Case 3.3			
BONMIN	16152	2328	140.462
CENSO	249	9000**	140.674

* Problem has no integer variables and is solved at the root node.
** CENSO was terminated after 9000 seconds with an optimality gap of 25.163 mSTB/d.

CENSO finds the same optimal solution in all three cases, with the active constraints listed in Table 3.17. As indicated, Well 3 and 8 act as "swing producers" and are adjusted to hit the E and W riser mix velocity constraint, respectively. Wells 9, 10, and 13 are shut in. The rest of the wells operate at maximum capacity, i.e. in this particular case at their draw-down pressure or minimum choke differential pressure. The results indicate that shutting in or re-routing wells does not increase oil production since it is limited by the riser velocity constraints.

The optimal solution found by CENSO on Case 3.2 is 140.674 mSTB/d. This solution is verified by implementing the optimal valve settings in GAP, to give 140.650 mSTB/d. The relative difference between these two solutions is -0.02%. It is not possible to assert the accuracy of which $\mathbf{P}$ approximates the GAP model based on a single point; however, this may indicate that the accuracy increases

when temperatures are included in **P**.

Prior to solving Case 3.1, the best known solution was 136.400 mSTB/d, found by the proprietary solver. The solution located by IPOPT, BONMIN, and CENSO gives a production of 140.650 mSTB/d in GAP. The potential increase in production is 4.25 mSTB/d, or 3.12%.

3.7.3. Pre-computations: building B-splines

Before solving **P**, B-spline approximations must be built from the samples taken from the nonlinear relations. In Table 3.14 we report the build times for various B-splines; the build time of a B-spline is the time it takes to solve the linear system in Eq. (3.20). The examples include the inflow curves, pressure drop relations, and temperature drop relations used in the cases presented previously. It is worth noting that a B-spline must be rebuilt only when the relation it approximates changes.

Table 3.14.: B-spline build times.

Samples (#)	Dimension (#)	Degree (#)	Time (s)
24	1	1	96×10^{-6}
24	1	3	127×10^{-6}
3773	4	1	0.2
3773	4	3	21.7
9800	4	1	3.0
9800	4	3	590.0
5184	5	1	1.4
5184	5	3	93.3

3.8. Concluding remarks

A framework for production optimization of multiphase flow networks has been presented. By modelling the network with a graph and the nonlinear relations in the network with B-splines, a fast solution method based on the spline-compatible MINLP solver in CENSO was devised. The solution method can solve problems formulated in the framework to global optimality. To accelerate solution times, CENSO was augmented with a primal heuristic (BONMIN) and an optimality-based bounds tightening technique from the literature. Together with a DOF analysis, this allowed us to reduce the number of sBB tree branches considerably.

In addition to the theoretical contributions outlined above, we have performed a benchmark study where the solution method is compared to several other nonlinear programming methods. The study involves three realistic cases defined using two subsea production system models provided by BP. The findings are summarized below.

- The formulation in **P** proved flexible and allowed us to model the cases in the benchmark study. The formulation includes nonlinear energy balances to model the transportation of energy with higher detail than previous works on petroleum production optimization. Features such as daisy-chained manifolds, lift gas injection, routing, and velocity constraints were easy to include.

- The B-spline surrogate models were sufficiently accurate to be used in production optimization of real cases. Inflow performance curves, pressure and temperature drop correlations, and velocity maps were successfully modelled with splines.

- CENSO was able to successfully solve several realistic cases to global optimality. However, as the solution time increases exponentially with the size of the problem (number of complicating variables), we found that the global solution method was not viable for daily production optimization of the largest case, namely Case 3.3.

- The local solvers IPOPT and BONMIN were able to successfully solve problems formulated with **P** to local optimality. In all cases except one, they located the global optimum certified by CENSO. This leads us to believe that the NLP relaxation of **P** is near convex in large portions of the feasible region. We attribute the consistency of the results to the smoothness and derivatives of the cubic B-splines, and to the linear participation of the integer variables in **P**.

- The local and global solvers are complementary in the sense that the local solvers provide fast results for complex problems and are thus suitable for daily production optimization. CENSO, however, can be used to certify local solutions from time to time, and also globally explore new production settings.

- In Case 3, the new methodology identified a potential increase in production of 4250 standard barrels of oil per day, or 3.12% more than the best, previously known solution. This solution was verified in the GAP simulator.

We believe that the above findings illustrate what any proficient practitioner of mathematical programming knows; that a "good" problem formulation is a requirement for fast solution times and consistency across solvers.

The speed of the new method would allow for parameter sensitivity analysis and stochastic optimization to include uncertainty in crucial model parameters (at least for small to moderately sized problems). With such approaches it would be possible to generate not only an optimal point, but an optimal operational plan for the user. This is an important step towards better decision support systems.

Acknowledgements

This work was supported by the Center for Integrated Operations in the Petroleum Industry, Trondheim, Norway.

The authors would like to thank Esmaeil Jahanshahi for his help in testing the thermodynamic model used in this paper.

3.A. Degree-of-freedom analysis

Here we give a degree-of-freedom (DOF) analysis for problem $\mathbf{P}$. We denote the DOF with D, and calculate it as $D = D_c + D_d$, where D_c and D_d is the number of free continuous and binary variables, respectively. It is straightforward to verify that $D_d = |\mathbf{E}^{\mathrm{d}}|$ so we focus on calculating D_c. We perform the analysis in two steps: first, we calculate the DOF for a network without discrete edges, i.e. with $\mathbf{E}^{\mathrm{d}} = \emptyset$; second, we calculate the DOF for discrete edges.

First, we consider a network without discrete edges. According to the requirements all nodes, except sink nodes, must have exactly one leaving edge. Consequently, for a network without discrete edges the following must be true: $|\mathbf{E}| = |\mathbf{N}| - |\mathbf{N}^{\mathrm{snk}}|$. This relation between the number of edges and nodes is useful when we next attempt to eliminate variables with equality constraints.

To calculate D_c we first count the number of continuous variables to

$$\overbrace{|\mathbf{S}| \cdot |\mathbf{E}|}^{\mathbf{q}} + \overbrace{|\mathbf{N}|}^{\mathbf{p}} + \overbrace{|\mathbf{E}|}^{\mathbf{\Delta p}} + \overbrace{|\mathbf{E}|}^{\mathbf{t}} + \overbrace{|\mathbf{E}|}^{\mathbf{\Delta t}} + \overbrace{|\mathbf{E}|}^{\mathbf{h}} + \overbrace{|\mathbf{E}|}^{\mathbf{\Delta h}}. \tag{3.22}$$

A quick glance at Table 3.3 verifies these numbers. Remark that when we now attempt to eliminate variables we must take care to count one elimination per constraint, and to only eliminate a variable that participate in the constraint.

Starting with the flow rates $\mathbf{q}$: we count equality constraints related to flow rates to $|\mathbf{S}| \cdot |\mathbf{N}^{\mathrm{int}}| + |\mathbf{S}| \cdot |\mathbf{N}^{\mathrm{src}}|$, which is the number of mass balances in Eq. (3.1) plus the number of rate boundary conditions in Eq. (3.11), respectively. Using the relation $|\mathbf{E}| = |\mathbf{N}| - |\mathbf{N}^{\mathrm{snk}}|$ we find that $|\mathbf{N}^{\mathrm{int}}| + |\mathbf{N}^{\mathrm{src}}| = |\mathbf{N}| - |\mathbf{N}^{\mathrm{snk}}| = |\mathbf{E}|$. Thus, there are $|\mathbf{S}| \cdot |\mathbf{E}|$ variables, $|\mathbf{S}| \cdot |\mathbf{E}|$ constraints, and zero DOF in the flow rates $\mathbf{q}$.

We continue by counting $2(|\mathbf{E}| - |\mathbf{E}^{\mathrm{d}}|) = 2|\mathbf{E}|$ pressure drop constraints $\Delta p_e = f_e(\cdot)$ and $\Delta p_e = p_i - p_j$ (remembering that $\mathbf{E}^{\mathrm{d}} = \emptyset$). We also count $|\mathbf{N}^{\mathrm{snk}}|$ pressure boundary conditions. In total we get $2|\mathbf{E}| + |\mathbf{N}^{\mathrm{snk}}| = 2|\mathbf{E}| + |\mathbf{N}| - |\mathbf{E}| = |\mathbf{N}| + |\mathbf{E}|$ constraints, which is the same as the number of pressure variables. Thus, we have zero DOF in the pressure variables $\mathbf{p}$ and $\mathbf{\Delta p}$.

In the same fashion we consider the $4|\mathbf{E}|$ temperature and enthalpy variables. From Eqs. (3.6), (3.7), and (3.8) we count $3|\mathbf{E}| + |\mathbf{N}^{\mathrm{int}}|$ constraints. We also have $|\mathbf{E}^{\mathrm{src}}| = |\mathbf{N}^{\mathrm{src}}|$ boundary constraints in Eq. (3.13) on the temperature variables. The total number of constraints is $3|\mathbf{E}| + |\mathbf{N}^{\mathrm{int}}| + |\mathbf{N}^{\mathrm{src}}| = 3|\mathbf{E}| + |\mathbf{N}| - |\mathbf{N}^{\mathrm{snk}}| = 4|\mathbf{E}|$. Thus, we find no DOF in the variables $\mathbf{t}$, $\mathbf{\Delta t}$, $\mathbf{h}$, and $\mathbf{\Delta h}$.

We conclude the first step of the analysis by establishing that there is no degree of freedom in $\mathbf{P}$ when $\mathbf{E}^{\mathrm{d}} = \emptyset$, i.e. $D = D_c + D_d = 0$.

In the second step of the analysis we let $\mathbf{E}^{\mathrm{d}} \neq \emptyset$, i.e. we allow discrete edges. We begin by considering a node with one leaving discrete edge. The discrete edge does not have the constraints $\Delta p_e = f_e(\cdot)$ and $\Delta p_e = p_i - p_j$. However, when $y_e = 1$, $\Delta p_e = p_i - p_j$ is recovered from the big-M constraint in Eq. (3.5). On the other hand, when $y_e = 0$, the flow rates are forced to zero ($\mathbf{q}_e = \mathbf{0}$) by the flow routing constraint in Eq. (3.9) (since we already have zero DOF in the flow rates we may use the boundary constraint $\zeta_{i,s}(\mathbf{q}_e, p_i) = 0$ to fix one pressure). In either case, one DOF remains. This DOF reflects different things for the two cases: for $y_e = 1$, Δp_e is free, but it relates the node pressures p_i and p_j, affecting the flow rate $\mathbf{q}_e$; for $y_e = 0$, Δp_e is free, but does not affect the flow rate since $\mathbf{q}_e = \mathbf{0}$ or the neighbouring pressures since $-M_e \leq p_i - p_j - \Delta p_e \leq M_e$ never become active. Note that there is a subtlety with the latter case ($y_e = 0$): since Δp_e cannot affect other variables it is not suited to be a branching variable (more importantly, Δp_e is not a complicating variable in $\mathbf{P}$). It is better to branch on p_i, which is a complicating variable that may affect other variables. We conclude that for each discrete edge we get one DOF in the continuous variables, and in total $D_c = |\mathbf{E}^{\mathrm{d}}|$.

Finally, we consider the special case where nodes may have multiple leaving discrete edges. The only change in $\mathbf{P}$ is the addition of the inequality constraints for manifold routing in Eq. (3.10). These constraints do not alter the DOF.

We conclude the analysis by establishing that $D_c = |\mathbf{E}^{\mathrm{d}}|$ and $D_d = |\mathbf{E}^{\mathrm{d}}|$, giving $D = 2|\mathbf{E}^{\mathrm{d}}|$. The DOF is associated with the discrete edges $e \in \mathbf{E}^{\mathrm{d}}$ representing (choke) valves.

3.B. Case results

The active constraints at the optimal solution of some of the cases are reported in this appendix.

Table 3.15.: Active constraints at optimal solution of Case 1.

Well	Online	Active constraint	Lower bound	Upper bound	Solution
1	Yes	Choke Δp (bar)	0	–	0
2	Yes	Choke Δp (bar)	0	–	0
3	Yes	Choke Δp (bar)	0	–	0
4	No	Oil rate (mSTB/d)	0	–	0
5	No	Oil rate (mSTB/d)	0	–	0
6	No	Oil rate (mSTB/d)	0	–	0
7	No	Oil rate (mSTB/d)	0	–	0
8	Yes	Choke Δp (bar)	0	–	0
9	Yes	Choke Δp (bar)	0	–	0
10	Yes	Choke Δp (bar)	0	–	0
–	–	Lift gas (mmSTB/d)	0	20	20

Table 3.16.: Active constraints for Case 2.3.

Well	Online	Active constraint	Lower bound	Upper bound	Solution
E loop					
1	Yes	Draw-down pressure (bara)	283.0	–	283.0
2	Yes	Choke Δp (bar)	0.5	–	0.5
3	No	Oil rate (mSTB/d)	0.0	–	0.0
5	Yes	Choke Δp (bar)	9.5	–	9.5
6	Yes	Choke Δp (bar)	10.0	–	10.0
W loop					
4	Yes	Choke Δp (bar)	1.0	–	1.0
7	Yes	Choke Δp (bar)	5.0	–	5.0
8*	Yes	Total gas (mmscf/d)	–	300.0	300.0
9	No	Oil rate (mSTB/d)	0.0	–	0.0
10	No	Oil rate (mSTB/d)	0.0	–	0.0
11	Yes	Choke Δp (bar)	5.5	–	5.5
12	Yes	Fixed oil rate (mSTB/d)	4.4715	4.4715	4.4715
13	No	Oil rate (mSTB/d)	0.0	–	0.0

* Well 8 is adjusted to hit the gas capacity constraint.

Table 3.17.: Active constraints for Case 3.2.

Well	Online	Active constraint	Lower bound	Upper bound	Solution
E loop					
1	Yes	Draw-down pressure (bara)	283.0	–	283.0
2	Yes	Choke Δp (bar)	0.5	–	0.5
3	Yes	Mixed velocity, E riser (m/s)*	–	v_e^U	v_e^U
5	Yes	Choke Δp (bar)	9.5	–	9.5
7	Yes	Choke Δp (bar)	5.0	–	5.0
W loop					
4	Yes	Choke Δp (bar)	1.0	–	1.0
6	Yes	Choke Δp (bar)	10.0	–	10.0
8	Yes	Mixed velocity, W riser (m/s)*	–	v_e^U	v_e^U
9	No	Oil rate (mSTB/d)	0.0	–	0.0
10	No	Oil rate (mSTB/d)	0.0	–	0.0
11	Yes	Choke Δp (bar)	5.5	–	5.5
12	Yes	Fixed oil rate (mSTB/d)	4.4715	4.4715	4.4715
13	No	Oil rate (mSTB/d)	0.0	–	0.0

* Velocities not displayed for commercial reasons.

Chapter 4

Virtual Flow Metering using B-spline Surrogate Models

Grimstad, B., Robertson, P., and Foss, B. (2015c). Virtual flow metering using B-spline surrogate models. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Florianópolis, Brazil.

Summary

Existing optimization-based virtual flow metering solutions use advanced, black-box process models directly in the optimization problem. This approach has many potential disadvantages, for example: non-smooth models and lack of derivative information may hamper the optimization solver. In this paper a new approach to optimization-based virtual flow metering using B-spline surrogate models is presented. In this approach the black-box process models are replaced with smooth B-spline approximations, with gradients readily available to the solver. We show that the approximation can be done without any significant loss of accuracy. By using surrogate models the optimization solver can be decoupled from the process simulator, saving I/O-operations and evaluations of the process model, resulting in reduced solution times. Another beneficial feature of the problem formulation is that poorly calibrated models may be identified and weighed less in the optimization problem. Some insight on how to select measurement noise and model error weights is shared with the reader.

4.1. Introduction

Model-based technologies are increasingly used to improve the operability and safety of subsea oil and gas production systems; several testimonials to this can be found in the literature, cf. (Stenhouse, 2008; Foss, 2012). By coupling sensor data with process models, operators may estimate the unknown flow rates in the system. This may aid them in: operating within safety and flow assurance limits, preventing unnecessary wear and tear on the equipment, identifying equipment failure, and in guiding the system to desired operating points.

In modern field developments, accurate pressure and temperature sensors are installed throughout the production system, from the bottom-hole of the wells to the separator. Flow meters are installed more sparingly due to high costs. For additional accuracy and redundancy, the systems are monitored with software that infer the flow rates by inserting available measurements into an advanced process model/simulator. This technology is known as *flow estimation*, *data reconciliation* or *virtual flow metering* (VFM). A survey and discussion on the use of flow estimation in subsea oil and gas production systems can be found in the recent work of Robertson (2014). The same work provides a list of existing commercial and in-house VFM solutions. One example from this list is FMC Technologies' FlowManager$^{\text{TM}}$(Holmås and Løvli, 2011).

A VFM system is an *online* system, running at real-time speed in intervals of seconds or minutes. For this reason, steady-state models have been prevalent in VFM systems to obtain the required solution times. Once within each interval a steady-state flow estimation problem, or data reconciliation problem, is solved to obtain the estimated rates. For a linear process model, this problem is a special case of the Kalman filter (Narasimhan and Jordache, 1999). This relation becomes less clear when a nonlinear model is used and operational constraints are included. The resulting optimization problem is then non-convex and difficult to solve. The situation is not improved by the fact that the process model is considered to be a black-box model without available gradient information. To resolve some of these issues we will in this work replace the process models with B-spline surrogate models. These surrogate models are accurate, smooth, fast to evaluate, and they offer gradients – all being favourable properties for optimization.

Using the B-spline surrogate models, we form a data reconciliation problem that we solve for a semi-realistic case with two subsea wells. A nice feature of the proposed method is that model errors, as well as measurement noise, are considered in the problem formulation. This allows for gross error detection to identify poorly calibrated models, which is a common issue in VFM systems; this is due to the lack of flow rate measurements for model calibration (Bieker et al., 2007).

4.2. Flow estimation

Let $\mathbf{y}$ be an n_y-vector of variables to be reconciled with the corresponding measurements $\bar{\mathbf{y}}$.[1] We denote the difference between the reconciled and measured values with $\mathbf{v}$, i.e. $\mathbf{v} = \mathbf{y} - \bar{\mathbf{y}}$. Furthermore, we denote with an n_x-vector $\mathbf{x}$ the unmeasured variables that we want to estimate. To estimate $\mathbf{x}$ we solve the following nonlinear

[1] Vectors are denoted with bold face $\mathbf{y}$ and vector elements with y_i. All measurements are denoted with a bar accent, e.g. $\bar{\mathbf{y}}$.

programming problem

$$\begin{aligned} \underset{\mathbf{x},\mathbf{y},\mathbf{v},\mathbf{w}}{\text{minimize}} \quad & ||\mathbf{v}||^2_{\mathbf{M}} + ||\mathbf{w}||^2_{\mathbf{N}} \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x},\mathbf{y}) = \mathbf{w} \\ & \mathbf{y} - \bar{\mathbf{y}} = \mathbf{v} \\ & \mathbf{x} \in \mathbf{X} \end{aligned} \tag{\mathbf{P}}$$

where $\mathbf{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \to \mathbb{R}^m$ are m maps between the reconciled (measured) variables y and the unmeasured variables $\mathbf{x}$. In general, $\mathbf{g}$ is a vector of nonlinear functions and, hence, $\mathbf{g}(\cdot) = \mathbf{w}$ describes a nonconvex constraint set. The variables $\mathbf{w} \in \mathbb{R}^m$ represent model errors; in the case of a perfect model $\mathbf{w} = \mathbf{0}$. The set $\mathbf{X}$ is a convex polytope which may include linear constraints on the estimated variables $\mathbf{x}$. Note that the measurements $\bar{\mathbf{y}}$ are not considered variables in $\mathbf{P}$.

The objective of $\mathbf{P}$ is a weighted least-squares quadratic function defined by the norms $||\mathbf{v}||^2_{\mathbf{M}} = \mathbf{v}^\mathsf{T}\mathbf{M}\mathbf{v}$ and $||\mathbf{w}||^2_{\mathbf{N}} = \mathbf{w}^\mathsf{T}\mathbf{N}\mathbf{w}$, representing penalties on measurement and modelling errors, respectively. The matrices $\mathbf{M}$ and $\mathbf{N}$ can be thought of as the inverse covariance matrices for the measurement noise and model errors. In this work we set $\mathbf{M} = \text{diag}(\boldsymbol{\mu})$ and $\mathbf{N} = \text{diag}(\boldsymbol{\nu})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are two vectors of non-negative weights, to obtain diagonal, positive definite matrices and a convex objective function.

Problem $\mathbf{P}$ is a steady-state data reconciliation problem. Next we describe how $\mathbf{P}$ may be configured to estimate the flow rates in a simple subsea production system with two wells. The sequential solution of this problem, incorporating new measurements as they become available, is often termed *virtual flow metering*.

4.2.1. Formulating a simple flow estimation problem

Here we present a configuration of $\mathbf{P}$ which can be applied to any two-well subsea template tied back through a single pipeline (see Fig. 4.1). Extensions to include more wells and/or more complex topologies are straightforward.

For a subsea production system the vector of measured variables is typically $\mathbf{y} = [\mathbf{p}^\mathsf{T}, \mathbf{t}^\mathsf{T}, \mathbf{u}^\mathsf{T}]^\mathsf{T}$, with measurements $\bar{\mathbf{y}} = [\bar{\mathbf{p}}^\mathsf{T}, \bar{\mathbf{t}}^\mathsf{T}, \bar{\mathbf{u}}^\mathsf{T}]^\mathsf{T}$, where $\mathbf{p}$ denotes pressures, $\mathbf{t}$ denotes temperatures, and $\mathbf{u}$ denotes choke openings. The unmeasured variables to be estimated are typically the flow rates, i.e. $\mathbf{x} = \mathbf{q}$, where $\mathbf{q}$ denotes the flow rates. The vector $\mathbf{g}$ may include pressure and temperature drop functions, as well as other relations between the variables. Below, we consider some commonly used pressure drop functions. For simplicity we assume perfect temperature and choke opening measurements and fix $\mathbf{t} = \bar{\mathbf{t}}$ and $\mathbf{u} = \bar{\mathbf{u}}$ in the formulation.

The *well performance* is usually described by the *inflow performance relationship* (IPR), which describes the inflow from the reservoir to the wellbore. The IPR, here denoted with f_i^{ipr}, relates the liquid rate q_i^{liq} to the flowing bottom hole pressure p_i^{bh}. It depends on factors such as rock properties (e.g. permeability), fluid properties, the well completion, *et cetera*. Denoting the IPR model error variable with w_i^{ipr}, we

model the IPR as follows:

$$w_i^{\mathsf{ipr}} = p_i^{\mathsf{bh}} - f_i^{\mathsf{ipr}}(q_i^{\mathsf{liq}}), \quad \forall\, i \in \{A, B\}. \tag{4.1}$$

The *vertical lift performance* (VLP) curve describes the relationship between the well flow and the pressure loss from the bottom hole to the wellhead, and depends on e.g. the well geometry and fluid properties. While a well can be modelled from the reservoir to the wellhead using the IPR and VLP curve, the two models can be combined to create a single *well performance curve* (WPC)

$$w_i^{\mathsf{wpc}} = p_i^{\mathsf{wh}} - f_i^{\mathsf{wpc}}(q_i^{\mathsf{liq}}), \quad \forall\, i \in \{A, B\}. \tag{4.2}$$

The VLP curve may be ambiguous with respect to flow rate due to gas lifting at low flow rates, therefore we prefer to use the WPC, which is usually more well-behaved.

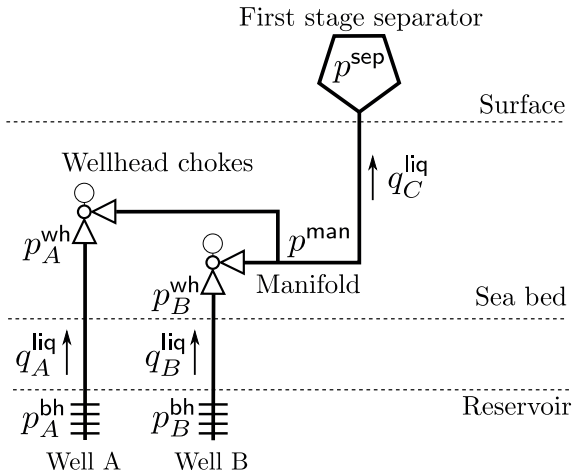


Figure 4.1.: Topology of production system.

Wellhead choke valves control the flow rates from each well. The flow rates through the choke valves depend on e.g. choke geometry and the upstream flow regime.

$$w_i^{\mathsf{chk}} = p^{\mathsf{man}} - f_i^{\mathsf{chk}}(q_i^{\mathsf{liq}}, p_i^{\mathsf{wh}}; \bar{t}_i^{\mathsf{wh}}, \bar{u}_i), \quad \forall\, i \in \{A, B\}. \tag{4.3}$$

The wellhead choke model used in this paper is a multiplier model, which is based on the simple valve equation together with a Morris multiphase multiplier and Chisholm slip correlation (see e.g. Schüller et al. (2003)).

The *flowline* is modelled using the OLGAS 3P multiphase flow correlation. The input variables to the correlation are upstream (manifold) pressure, liquid flow rate, gas-oil ratio (GOR) and water cut (WCT). The measured upstream temperature is considered a fixed parameter. The output is the downstream (separator) pressure:

$$w^{\mathsf{fl}} = p^{\mathsf{sep}} - f^{\mathsf{fl}}(p^{\mathsf{man}}, q_C^{\mathsf{liq}}, r_C^{\mathsf{gor}}, r_C^{\mathsf{wct}}; \bar{t}^{\mathsf{man}}). \tag{4.4}$$

For convenience, we collect all the model errors in a vector

$$\mathbf{w} = \left[w_A^{\mathsf{ipr}}, w_B^{\mathsf{ipr}}, w_A^{\mathsf{wpc}}, w_B^{\mathsf{wpc}}, w_A^{\mathsf{chk}}, w_B^{\mathsf{chk}}, w^{\mathsf{fl}}\right]^\top,$$

with corresponding weights $\boldsymbol{\nu}$. Similarly, we collect the measurement/reconciliation errors in a vector

$$\mathbf{v} = \left[v_A^{\mathsf{bh}}, v_B^{\mathsf{bh}}, v_A^{\mathsf{wh}}, v_B^{\mathsf{wh}}, v^{\mathsf{man}}, v^{\mathsf{sep}}\right]^\top,$$

with corresponding weights $\boldsymbol{\mu}$.

In addition to the pressure drop constraint functions in $\mathbf{g}$, we model interrelations between the unmeasured variables $\mathbf{x}$ with the constraint set $\mathbf{X}$. For example, we include mass balance constraints on the rate variables $\mathbf{q}$ in $\mathbf{X}$, e.g.

$$q_C^p = q_A^p + q_B^p, \quad \text{for } p \in \{\mathsf{oil}, \mathsf{gas}, \mathsf{wat}\}.$$

Other linear relations that we include in $\mathbf{X}$ are:

$$q_i^{\mathsf{liq}} = q_i^{\mathsf{oil}} + q_i^{\mathsf{wat}}, \quad q_i^{\mathsf{gas}} = r_i^{\mathsf{gor}} q_i^{\mathsf{oil}}, \quad q_i^{\mathsf{wat}} = r_i^{\mathsf{wct}} q_i^{\mathsf{liq}},$$

for $i = \{A, B\}$, where r_i^{gor} and r_i^{wct} are a constant GOR and WCT, respectively.

4.3. B-spline surrogate models

In practice, the nonlinear maps in $\mathbf{g}$, such as the pressure loss functions in the previous section, are given by some process simulator. Most commercially available process simulators are proprietary code and may be considered as "black-box calculators". A process simulator models the production network with complex, nonlinear functions that may be non-smooth in certain regions. Generally, no derivative information is made available and finite difference methods must be used when optimizing with gradient-based solvers, often resulting in a large number of evaluations. Furthermore, when coupling an optimization solver to a (black-box) process simulator, evaluation may be time consuming for several reasons: 1. the simulator may require convergence of the whole network model at each evaluation (even when perturbing a single component of the network), and 2. the IO-operations to transfer data between the solver and simulator may be time consuming. To solve the above problems we will replace the nonlinear maps $\mathbf{g}$ with B-spline approximations $\boldsymbol{\phi}$. The B-splines in $\boldsymbol{\phi}$ are referred to as B-spline surrogate models.

Note that the pressure drop functions in the previous section are on the form $g_i(\cdot) = y_i - f_i(\cdot) = w_i$. Thus, in the following we will approximate f_i (instead of g_i) by ϕ_i, i.e. $\phi_i \approx f_i$, and $g_i \approx y_i - \phi_i$.

4.3.1. B-splines

A B-spline is a piecewise polynomial function in the variable x, defined by a degree p, a vector of *knots* $\mathbf{t} \in \mathbb{R}^{n+p+1}$, and a vector of n coefficients $\mathbf{c} \in \mathbb{R}^n$ as follows:

$$\phi(x; p, \mathbf{t}) = \mathbf{c}^\mathsf{T} \mathbf{b}(x; p, \mathbf{t}). \tag{4.5}$$

$\mathbf{b}(x; p, \mathbf{t}) \in \mathbb{R}^n$ is a vector of n B-spline basis functions. The basis functions are overlapping, degree p, polynomial functions, as depicted in Fig. 4.2 for $n = 8$ and $p = 3$. The basis functions and their derivatives may be evaluated by the numerically stable and fast, recursive algorithms of De Boor (1972) and Cox (1972).

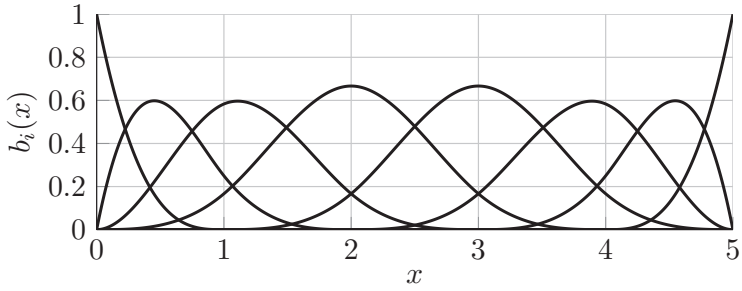


Figure 4.2.: B-spline basis functions for $p = 3$ and $n = 8$.

The B-spline in (4.5) generalizes to the multivariate case, where it is called the tensor product B-spline. Most properties of the univariate B-spline, such as a high degree of smoothness and local support, carry over to the multivariate case without any complications. For brevity we will discuss only univariate B-splines in the rest of this section. We note however that the discussion is valid also for tensor product B-splines. The interested reader is referred to the textbooks of Schumaker (2007) and Piegl and Tiller (1997) for an introduction to multivariate splines.

4.3.2. Cubic spline interpolation

Let any function $f : \mathbb{R} \to \mathbb{R}$, for example f_A^{wpc} in (4.2), be sampled on a regular (rectangular) grid to yield N data points $\{x_i, f(x_i)\}_{i=1}^N$. Several methods exist for constructing a B-spline that interpolates these N points. These methods vary in how the B-spline degree p and knots $\mathbf{t}$ are selected. The commonly preferred cubic spline ($p = 3$) can be obtain by using a *free end conditions* knot vector

$$\mathbf{t}_F = \{ \underbrace{x_1, \ldots, x_1}_{p+1 \text{ repetitions}}, x_3, \ldots, x_{m-2}, \underbrace{x_m, \ldots, x_m}_{p+1 \text{ repetitions}} \}.$$

To obtain the spline the following linear system is solved for the coefficients $\mathbf{c}$:

$$\underbrace{\left[\mathbf{b}(x_1)\ \mathbf{b}(x_2)\ \ldots\ \mathbf{b}(x_N)\right]^{\mathsf{T}}}_{\mathbf{B}} \mathbf{c} = \mathbf{f}, \tag{4.6}$$

where $\mathbf{f} = [f(x_i)]_{i=1}^N$ and $\mathbf{B} \in \mathbb{R}^{N \times n}$ is called the B-spline collocation matrix. Note that $\mathbf{b}(x) = \mathbf{b}(x; 3, \mathbf{t}_F)$ in (4.6).

One advantage with (cubic) spline interpolation is that it avoids the problem of Runge's phenomenon, in which oscillation occurs between the interpolation points

(as is evident in interpolation with high degree polynomials). The functions in $\mathbf{g}$ are often polynomial or near-polynomial and approximated by B-splines with little error. The authors' experience with approximating various pressure loss functions suggests that the approximation error typically lies in the order of $0.1 - 0.001\%$ (in fact, the error can be made arbitrarily small by increasing the sampling resolution). Arguably, the error between $\mathbf{g}$ and reality is orders of magnitude larger than this. To illustrate this with an example, let $\phi_N^{\mathsf{wpc}}(q^{\mathsf{liq}})$ be the B-spline approximation of the WPC $f^{\mathsf{wpc}}(q^{\mathsf{liq}})$ sampled in N points. Further, let $e_N(q^{\mathsf{liq}}) = 1 - \phi_N^{\mathsf{wpc}}(q^{\mathsf{liq}})/f^{\mathsf{wpc}}(q^{\mathsf{liq}})$ be the resulting relative approximation error. Approximation errors for $N = 50$, $N = 10$ and $N = 5$ are shown in Figure 4.3, while error measures are summarized in Table 4.1.

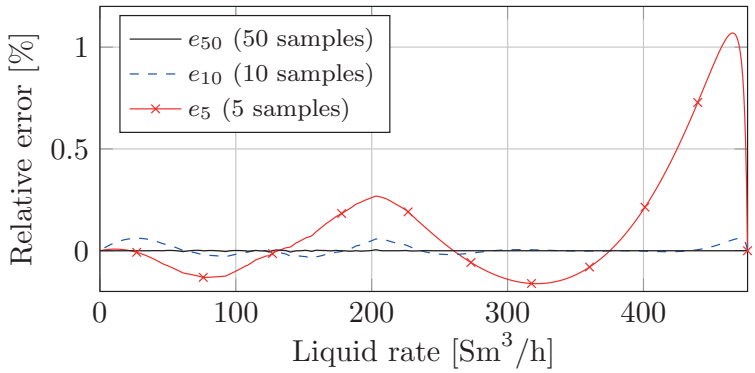


Figure 4.3.: B-spline approximation errors for a WPC.

Table 4.1.: Maximum and 2-norm errors.

N	$\|e_N\|_\infty$	$\|e_N\|_2$
5	$1.1 \cdot 10^{-2}$	$7.8 \cdot 10^{-2}$
10	$6.1 \cdot 10^{-4}$	$5.7 \cdot 10^{-3}$
50	$6.5 \cdot 10^{-5}$	$2.4 \cdot 10^{-4}$

The construction of a B-spline surrogate model is a two-step procedure: 1. sampling the simulator and 2. solving the linear system in (4.6) for the B-spline coefficients. This procedure can be run offline and the resulting B-splines stored in advance of optimizing $\mathbf{P}$.

4.4. Results and discussion

4.4.1. Reference OLGA simulation

To test the performance of the estimator, a production network model representative to Figure 4.1 was implemented in OLGA, which is considered the *de facto* industry standard for dynamic simulation of multiphase petroleum production systems (Bendiksen et al., 1991; Schlumberger, 2014b). A benchmarking simulation was run to obtain a set of noise-free measurements $\{\bar{\mathbf{y}}_k\}_{k=0}^{T}$ (pressures, temperatures and choke positions) and flow rates $\{\bar{\mathbf{q}}_k\}_{k=0}^{T}$, which were assumed to be unknown. Here, k are time indices (the simulation was run for 26 hours with a 10 second sampling interval). In the simulation, the choke valves were sequentially stepped up from 5 % to 60 % opening, as depicted in Figure 4.4. Default OLGA settings were used.

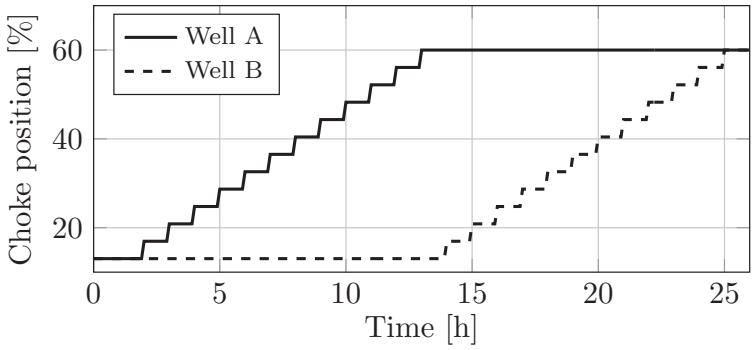


Figure 4.4.: Choke positions.

4.4.2. Obtaining the pressure drop models

To equip the estimator with the necessary pressure drop models, a model of the production network was implemented in Petroleum Experts' IPM software package (Petroleum Experts Ltd., 2014). IPRs, WPCs and the flowline model were sampled from the IPM module GAP, and approximated with cubic B-splines. For the chokes, we used a multiplier model based on the valve equation, which was also sampled and approximated with B-splines. Prior to sampling, the models were matched against multi-rate flow tests run in OLGA. The IPRs, WPCs and flowline model were matched using available tools in GAP, while the choke models were matched using a simple multiplication factor. For a large number of samples, it may take a few seconds to generate a B-spline, however, this single calculation is done *offline* and does not contribute to the time taken to solve $\mathbf{P}$.

4.4.3. Case 1 – Single model evaluation

We first present the estimation results obtained by evaluation each pressure drop model individually. This is equivalent to a nonredundant VFM method which uses a single pressure drop model for estimation. The resulting estimates are presented in Figures 4.5 (Well A), 4.6 (Well B) and 4.7 (Flowline). We note that the IPRs and WPCs tend to underestimate the flow rate slightly, while the choke models tend to overestimate the flow rate. Note how the choke model estimates degrade as the choke opens more, which is due to the increasing sensitivity of the flow rate with respect to pressure as the pressure drop across the choke decreases. The flowline rate estimate displays a large error compared to the well flow rate estimates, indicating that the flowline model is relatively poor.

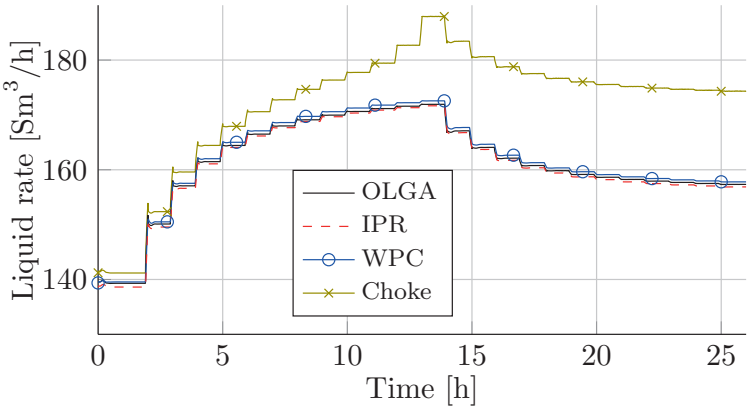


Figure 4.5.: Estimation by single model evaluation, well A.

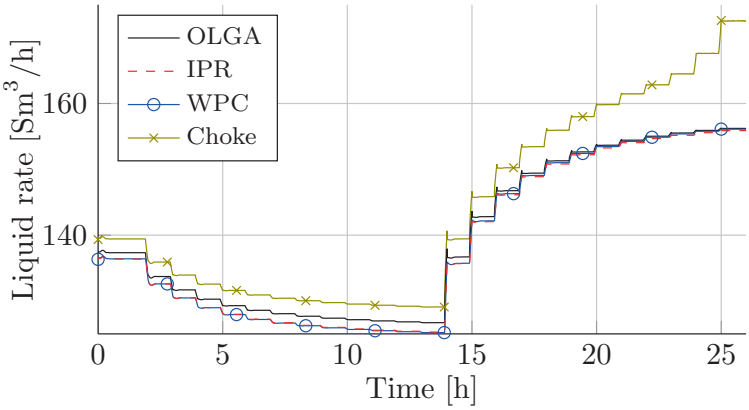


Figure 4.6.: Estimation by single model evaluation, well B.

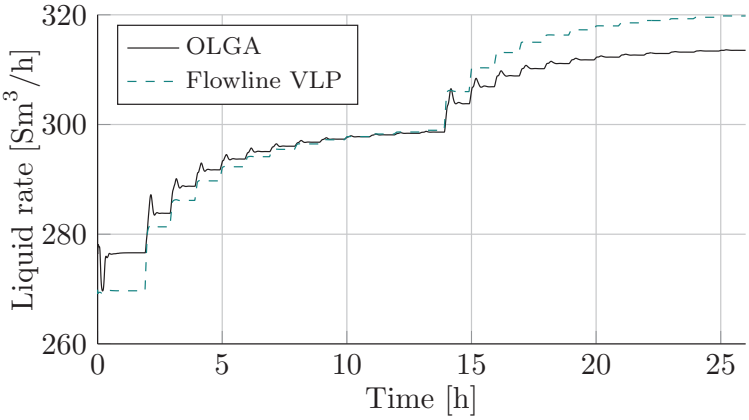


Figure 4.7.: Estimation by single model evaluation, flowline.

4.4.4. Case 2 – Uniform weights

Having made a qualitative assessment of the quality of each pressure drop model in Case 1, we now present the estimation results obtained by solving problem $\mathbf{P}$ with uniform weights, i.e. $\mathbf{N} = \mathbf{I}$. Since we have much higher confidence in the pressure measurements than the pressure drop models, $\mathbf{M}$ was configured with relatively large weights; $\mathbf{M} = 10^3 \cdot \mathbf{I}$. For each measurement $\bar{\mathbf{y}}_k$, problem $\mathbf{P}$ was configured as described in Sec. 4.2.1 and solved to local optimality to obtain the estimate $\mathbf{q}_k$ of the unmeasured flow rates $\bar{\mathbf{q}}_k$.

Qualitative gross error detection

The resulting values of the elements in $\mathbf{w}$ are shown in Figure 4.8. When qualitatively interpreting this figure, we note that: 1. Small values (i.e. close to zero) indicate that the pressure drop model agrees with the relevant reconciled pressures in the network, and 2. values which are close to each other indicate that the appropriate pressure drop models are in agreement with each other with respect to flow rates. In our case, the model error for the flowline VLP is relatively far from the remaining pressure drop models. This indicates that the rates predicted by the flowline VLP are not consistent with the other pressure drop models (nor with the reconciled pressures), and may be introducing unnecessary estimation errors. This is apparent from Fig. 4.7, however, in a real-life case, such a figure would not be available. We now proceed to adjusting the model error weights in an attempt to improve the flow rate estimates.

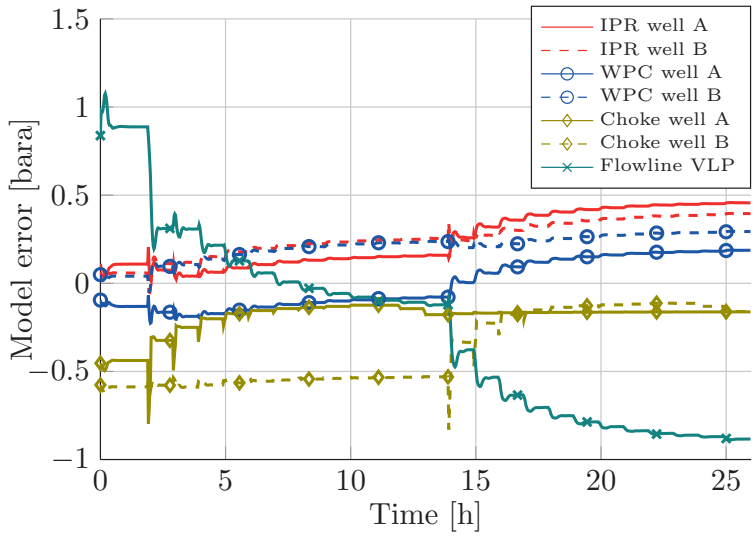


Figure 4.8.: Model errors with uniform weighting.

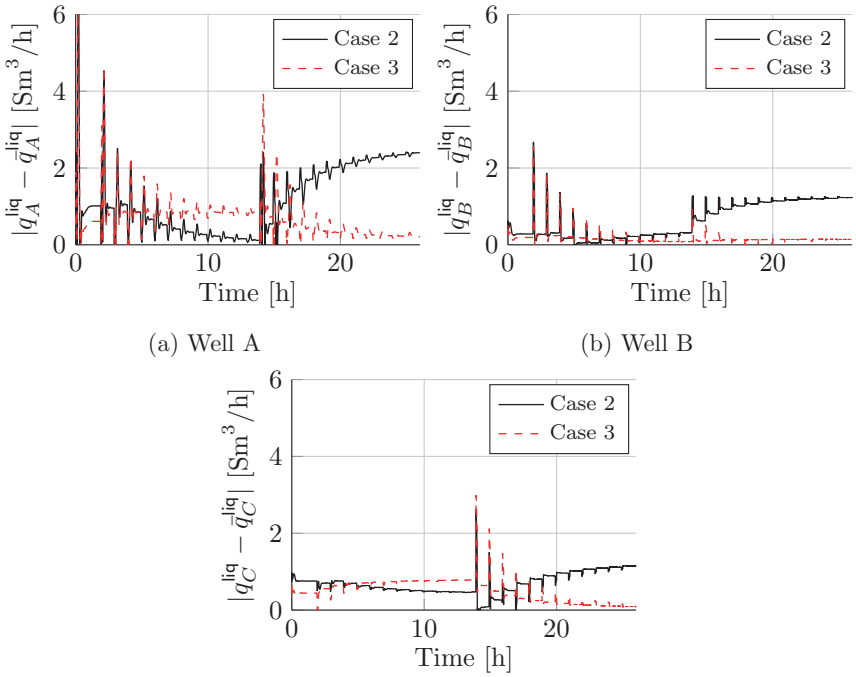
4.4.5. Case 3 – Heterogeneous weights

Finally, we present the estimation results obtained when we attempt to consider the observations made in Figure 4.8 and measured model uncertainty through the flow tests. Here, we select the weights $\boldsymbol{\nu}$ based on a normalized sum norm of errors between the measured pressures in the flow tests, and the corresponding pressures predicted by the matched models. The adjusted weights are shown in Table 4.2. Note the flowline weight ν^{fl} is selected relatively small.

Table 4.2.: Model error weighting $\boldsymbol{\nu}$ in Cases 2/3.

Case	ν_A^{ipr}	ν_B^{ipr}	ν_A^{wpc}	ν_B^{wpc}	ν_A^{chk}	ν_B^{chk}	ν^{fl}
2	1	1	1	1	1	1	1
3	1	0.79	0.25	0.19	0.41	0.37	0.05

Again, we configure and solve $\mathbf{P}$ for each measurement $\bar{\mathbf{y}}_k$. The resulting estimation errors in Cases 2 and 3 are shown in Figure 4.9. A clear reduction in estimation error is seen when our confidence in each model is taken into account through the weighting. In general, as seen in Table 4.3, the estimates in Case 3 are also better (or near as good) as the estimates produced by any single model. Equally important, we note that using a multi-model formulation increases *robustness* compared to a single-model approach; the resulting estimates become an "agreed consensus" between several models and are less prone to degradation in certain operating con-



(a) Well A

(b) Well B

(c) Flowline

Figure 4.9.: Absolute estimation errors for Case 2 and 3.

ditions (cf. a choke model subjected to a low differential pressure). Note that the spikes in estimation error are caused by unmodelled dynamic behaviour following the choke moves.

4.4.6. Solution times

The non-convex problems on the form in **P** were solved to (local) optimality by the nonlinear programming solver IPOPT (Wächter and Biegler, 2006) on a laptop computer with an Intel Core i7-3740QM CPU running at 2.7 GHz. The solution times are reported in Table 4.4. We notice that the solution times are below the 10 second real-time limit set by the sampling rate. In fact, as shown by the max-values, all problems were solved well within 10 seconds.

Problem **P** can be solved to global optimality by a solver that accepts spline constraints, such as CENSO (Grimstad and Sandnes, 2015). A considerable increase in computation time should then be expected, with the implication that the above real-time limit would most certainly be exceeded. A global solution could however be run in parallel to a local algorithm to identify if the local algorithm is stuck in a poor local solution.

Table 4.3.: Mean/max absolute errors (Sm^3/h).

	IPR	WPC	Chk.	FL	Case 2	Case 3
Mean						
Well A	0.42	0.53	10.8	-	0.64	0.17
Well B	0.94	0.88	4.39	-	0.69	0.50
Flowline	-	-	-	3.27	1.20	0.71
Max						
Well A	3.31	2.29	17.0	-	2.67	2.57
Well B	3.87	3.82	16.3	-	2.60	2.99
Flowline	-	-	-	8.81	6.35	7.08

Table 4.4.: Solution times (s).

	IPR	WPC	Chk.	FL	Case 2	Case 3
Mean	0.009	0.010	0.151	0.326	0.496	0.478
Max	0.096	0.093	1.396	2.104	2.472	3.032
% RT*	0.1	0.1	1.5	3.3	5.0	4.8

*Real Time

4.5. Concluding remarks

The proposed method for virtual flow metering was tested on a semi-realistic subsea production system with two wells. The successful test results are encouraging along several axes:

- the process models can be replaced with B-spline surrogate models without any significant loss of accuracy,
- a commercial NLP solver (IPOPT) can efficiently solve a series of data reconciliation problems **P** without any convergence problems (helped by the properties of the B-spline models),
- poorly calibrated models can be identified by analyzing the error variables in the problem formulation.

The authors hope to later improve the proposed method by automating the detection and de-weighting of poorly calibrated models, possibly by including gross error detection in **P**. Several simultaneous procedures for data reconciliation and gross error detection have been presented in the literature (Özyurt and Pike, 2004). These procedures are derived from robust statistics and solve the two problems as one nonlinear program (NLP).

Acknowledgements

We acknowledge the support of the Center for Integrated Operations in the Petroleum Industry at NTNU, Norway.

Chapter 5

On Why Model-Based Production Optimization is Difficult in the Upstream Industry

Grimstad, B., Almklov, P., Foss, B., and Gunnerud, V. (2014). On why model-based production optimization is difficult in the upstream industry. Published as a report in the IO center.

Summary

This paper presents an analysis on the use of model-based tools for production optimization in petroleum production. Its motivation is argued through a comparison with the downstream process industries where such tools are widely deployed. The empirical platform for the study includes interviews, secondments, experience with model-based tools in the downstream sector, and pilot testing of model-based tools in the upstream industry. In the paper we contextualize production optimization both in terms of technology and user groups before the main section on observations is presented. The discussion is grouped according to different aspects of data, technology and people, which reappear in a summarizing discussion.

5.1. Introduction

The petroleum industry has gone digital during recent years. Developments in sensors, data processing, and remote control technology have, together with improved video conferencing and data sharing facilities, inspired new strategies and operational models. In this paper we summarize some experiences from a part of this venture, the attempts to use the increased availability of production data, in particular real-time data, for purposes of production optimization. Even though there are success stories to be told, and promising research results that hold great potential, we believe it is useful to systematically address the difficulties related to production optimization. In this paper we summarize some findings and insights from our research in this respect.

In Norway, the digitization of the petroleum industry has been connected to the philosophy of Integrated Operations (IO) (Norwegian Oil Industry Association,

2005; Rosendahl and Hepsø, 2013; Besnard and Albrechtsen, 2013). Internationally, similar initiatives have been referred to as e-Field, Intelligent Field, Smart Field, and Field of the Future (Haavik, 2013; AbdulKarim et al., 2010). The authors of this paper are all affiliated with the IO Center at NTNU (IO Center, 2014), which has been sponsored by eight international oil companies and five suppliers. Its goal has been to develop new methods and tools for IO, which can be embedded in improved work processes in oil companies, and enhanced products and services from suppliers. The IO Center has conducted research on production optimization in tight cooperation with its industry sponsors since 2008, allowing researchers to observe and analyze the day-to-day operations of the oil companies. This has provided a unique opportunity to study work processes and hardware/software systems for production optimization.

The downstream industries such as refineries and chemical process plants consistently couple real-time data flows with online model-based optimization, both for automated control and for decision support for operating personnel (Bauer and Craig, 2008). The information and control hierarchy, which is needed to operate plants safely and efficiently, is often illustrated with a stack as in Figure 5.1. The base level of the stack is the actual process with its sensors and control handles. The sensors and control communicate with a basic control layer implemented in some state-of-the-art control system. This layer needs numerous inputs, in particular set points for the lower level controllers. The downstream industries typically supply these through an Advanced Process Control (APC) layer, which uses real-time data, models and optimization in fully automated control loops where Model Predictive Control has become the key technology (Qin and Badgwell, 2003). The APC structure cascades upwards to the top layer, where decisions are made and implemented by personnel, who rely on decisions support systems, rather than fully automated control loops. These support systems again rely extensively on the use real-time data, models and optimization, and the top layer is often denoted Real-Time Optimization (RTO). This layer is both used for plant wide optimization or for optimizing parts of the overall plant. It may be noted that a 5th layer focusing on planning and scheduling is omitted in Figure 5.1.

Currently, upstream production optimization decisions are dominated by heuristic approaches, even though they are supported by an increasing suite of tools for visualization and analyses. Thus, it is a fair statement that the upstream oil and gas industry lags far behind the downstream industries in technologies for and application of APC and RTO. Consequently, there is a potential for improved use of real-time data. IO Center research on production optimization has been driven by a desire to develop solutions for the RTO layer in the upstream industry rather than the APC layer. The justification is a belief that it is advantageous to introduce real-time model-based optimization in the RTO layer with humans in the loop, prior to the APC layer with fully automated control loops.

The IO Center has taken a holistic approach to the RTO layer challenge by simultaneously researching technology as well as work practices. Technology related research has focused on appropriate mathematical models for optimization

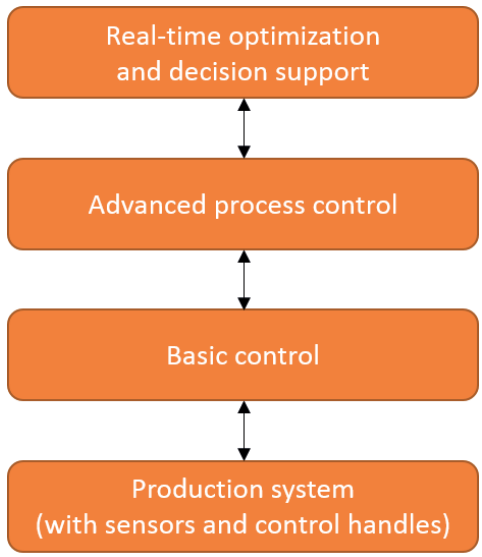


Figure 5.1.: Stack illustrating the different control levels.

and means for automatic calibration of such models. Examples of the former is spline-based proxy modeling (Grimstad et al., 2016) and of the latter is the SmartX method for well testing (Zenith et al., 2015). Further, efficient and robust optimization techniques have been developed and evaluated; one example is (Gunnerud and Foss, 2010). The advancements have been tested on a variety of industrial cases in collaboration with IO Center partners. An overview is provided in (Foss, 2012). Our research on operational work practices has particularly focused on sensor data interpretation and interdisciplinary decision making (Østerlie et al., 2012; Almklov et al., 2014). This paper seeks to integrate the insights from these research ventures and it is, as is further described in the method section, based on an extensive and prolonged interaction with the industry.

The availability and use of software for the RTO layer varies greatly between fields, companies and countries. Improving the understanding of this variation is one of the research gaps addressed in this paper. Rather than viewing this as a one-sided implementation issue, asking why people do not utilize new technologies, we seek explanations both in the human, organizational and technical domains. To achieve this goal we need an empirical platform, which is presented in the subsequent section. Subsequently, the context of production optimization will be discussed in more detail before the main findings are presented and justified. The paper ends with a discussion and some conclusions.

5.2. Methods and Data

The empirical basis for this work can be divided into five parts. First, the authors have experience from secondments that focused on production optimization in the operating and research parts of three oil companies and two suppliers. Second, one of the authors has experience with APC in the process industries through the development of MPC methodology, which subsequently was commercialized. Third, for the purpose of this paper we conducted interviews with seven informants. The outplacements and interviews come in addition to earlier interview series and observations at subsurface departments at three oil companies. Thus, the total empirical background involve five oil companies and two suppliers. These interviews primarily serve to support our discussion of the (interdisciplinary) operational context into which new tools and methods must be implemented, and the variation in production optimization challenges on different fields. Fourth, the interaction between the IO Center and its industry partners, in meetings discussing research challenges and implementation issues, provides highly relevant input for this paper. The 14 semi-annual IO Center Technical Committee meetings between production and optimization experts from partner companies and the IO center, and many bilateral workshops with individual companies, have been particularly useful as arenas for discussions on industry needs and implementation issues. Finally, a substantial number of scientific publications has provided us with insight from an academic perspective, in particular through conference presentations and review processes.

A limitation of the study is the fact that several oil companies, especially some of the large ones, possess internal and classified technologies for production optimization. Such information is not part of the empirical platform for this paper. Even though our data includes international companies and fields outside the Norwegian continental shelf (NCS), our paper is likely to be have a slight Norwegian bias as well as a focus on offshore production systems. Our findings still are, we believe, generic enough to have a transfer value to other parts of the world. Thus, this is no complete story on state-of-the-art model-based production optimization. Rather, our ambition is to document some examples and observations that contribute to explaining the hurdles for widespread and successful use. Hopefully, this will increase the understanding of the landscape into which model-based optimization software must be fitted. For the industry insider, in particular, this approach can provide a useful external perspective due to the authors' rather broad empirical platform. The paper may also contribute to an increased understanding of technology implementation issues in general and optimization technologies in particular.

5.3. The Context of Production Optimization

In this section we outline the context of production optimization in the technical and organizational domains. This description will serve as a platform on which we later will place and relate our findings. We will not paint a complete picture,

but rather outline the main characteristics of the typical environment in which (offshore) production optimization takes place. By viewing our findings in this context we seek to present new insight to their root causes.

5.3.1. The Production System and Production Optimization

The following description of the technical domain is related to the technology pyramid in Figure 5.2. The description starts with the *production system* at the bottom of the pyramid and continues upwards. Note that the bottom layer of Figure 5.2, the production system, is the same as the bottom layer of the stack in Figure 5.1.

The production system, as defined in this paper, comprises the wells and infrastructure necessary to transport the reservoir fluid to the surface processing facilities. Therefore, the production system is bounded by the reservoir upstream and the processing facilities downstream. A variety of external conditions are imposed at the boundaries of the production system. The well inflow from the reservoir may change in the short term due to intricate flow phenomena such as gas and water coning, and in the long term due to drainage and injection, whereas downstream, the processing facilities typically impose conditions such as gas and water production limits, gas-lift capacity restrictions, and separator pressure constraints.

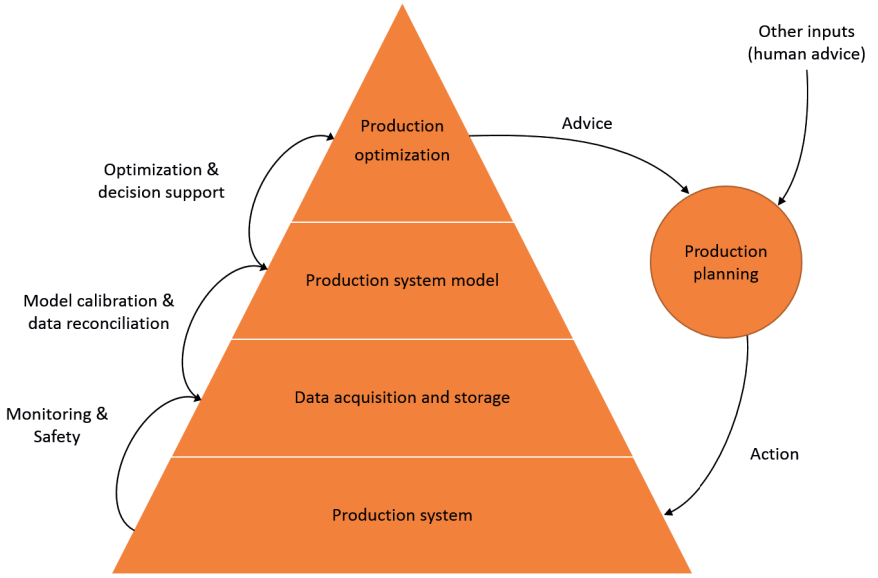


Figure 5.2.: Illustration of the top-to-bottom dependence on the technologies involved in production optimization. The pinnacle of the pyramid is production optimization for decision support.

In addition to these externally imposed boundary conditions the production system is subject to numerous operational constraints such as limits in well draw-down

pressure and choke valve settings, and constraints related to flow routing and flow assurance. Some constraints may be nontrivial to model. For example, a riser may have a fluid velocity constraint to avoid erosion of the inner tube coating due to sand production. In general the above constraints reflect the desire to operate optimally within safe and cost efficient limits. To complicate matters even more, operation may be constrained by maintenance, testing, and other disruptive work. Typical examples of this are well testing, which is performed regularly to monitor well performance and well integrity, and operations to remove deposits in wells and pipelines.

Data acquisition and storage is the collection of measurement data from the sensors installed in the production system, which subsequently is fed through the control system and stored in historian databases for later use. This data enables the operators and engineers to monitor and trend key variables in the production system, thus helping to ensure safe operation. Returning to Figure 5.2 data acquisition is placed on top of the production system layer.

The typical greenfield production system is well-instrumented with pressure and temperature sensors installed throughout the system. More advanced and expensive instruments such as subsea multi-phase flow (MPF) meters and downhole sensors are, however, also being installed more extensively than in older systems. This follows the modern mindset in the upstream industry that additional instrumentation may add more value than its self-cost (Gilman and Nordtvedt, 2014), where the added value is largely accounted for by improved operational awareness and safety. The improved operability does not come from the raw data alone—interpreting large amounts of interrelated data from a production system can be overwhelming, even for an engineer with intimate knowledge of the system—but rather from processing and visualization of the data.

Sensor data from the production system cannot be taken at face value. Sensors deteriorate over time and there are often interaction effects to be sorted out when interpreting them. In this interpretative work engineers draw on a multitude of information sources.[1] For example, sensor quality is often labeled by engineers (on a scale from faulty to good) based on the available information and interpretation. This implies that human interpretation is involved in the information processing upwards in the pyramid in Figure 5.2. Even measured production volumes, values that one may suppose are straight-forward to measure, often require human evaluation before they are passed on. Another example of situations where human judgement is involved in this process is when MPMs measure several wells on one template, and one needs to make inferences to evaluate the relative contribution of the individual wells. The fact that human evaluation is involved does not mean that it is a matter of guessing. It rather implies that the calculations are supported by some degree of human evaluation.

The value of real-time data may increase if it is combined with an appropriate

[1]This has been discussed in a theoretical studies by Østerlie et al. (2012) and Almklov et al. (2014), among others.

mathematical model of the production system, e.g. for virtual metering, data reconciliation, or prediction. This is illustrated as the *production system model* in the third layer in Figure 5.2. To capture important interactions in the production system, the model should represent the complete production system (confined according to the above definition of the production system), rather than a smaller part of this. A motivation for obtaining such a model of the production system is that it can serve more than one purpose if it is sufficiently extensive and accurate. It may for instance be used for real-time flow estimation, for flow assurance analyses or optimization case studies.

The production system model is an example of the modern, holistic modelling approach, where a multi-purpose model is sought and the modelling scope is extended to include crucial parts of the system (Stenhouse, 2006; Bakken et al., 2011; Crompton and Gilman, 2011). The various model components are maintained by their respective discipline engineers; e.g. well models are maintained by well engineers and reservoir fluid descriptions by the reservoir engineers. Thus, building and maintaining a production system model requires several competencies. As can be imagined, the technical and organizational challenges related to model maintenance are substantial since it requires extensive interaction across disciplines and sites in order to exchange relevant operational information. However, experience tells us that the predictive power of a multi-purpose production system model may be worth the modelling effort.

The purpose of model-based *production optimization* (PO) is to suggest operational settings that are optimal in some sense; e.g. that produce a targeted amount of oil, while minimizing cost and honoring relevant constraints. There are some model-based PO technologies available today. Bieker et al. (2007) gave a rather early overview of methods for model-based production optimization problem. They categorize the optimization methods based on traits such as the ability to handle routing decisions and on the use proxy models. It should be noted, however, that optimization solver technology has advanced significantly since 2007, especially within mixed-integer programming, and has thus changed the premises for some of the conclusions in the paper.

During the last decade, the industry has moved towards more integrated production optimization solutions, with the goal of coordinating all decision in one master problem. For example, the integrated problem may coordinate gas lift allocation, routing, and well balancing, while respecting flow assurance constraints. An integrated problem requires more input data than the individual sub-problems it contains; that is, it must include operational constraints from all the integrated production optimization sub-problems. As a result, the use of integrated solutions is more demanding since it requires more competent and alert users.

5.3.2. Production Planning and the Production Engineer

In the previous section we described the technological context of PO using Figure 5.2. The figure shows model-based PO as the pinnacle of the pyramid, resulting in

advice to the production planning process. This process is driven by the Production Engineers (PEs), who are responsible for the daily follow-up and production planning of an asset. They use their experience and knowledge of the system to find the best way to operate the field. This production strategy, however, needs to be aligned with the interests and concerns of other disciplines. In addition, the decisions are often performed in a dynamic and hectic environment with strict time constraints and often large uncertainties.[2] The work environment of the PEs is outlined in Figure 5.3. Using this figure we describe the organizational context of production optimization since this context is important to understand the decision making processes. Even though the groups identified in this figure are not exhaustive, they include key personnel. However, there is a variation in composition and relative importance of these groups between companies and within companies.

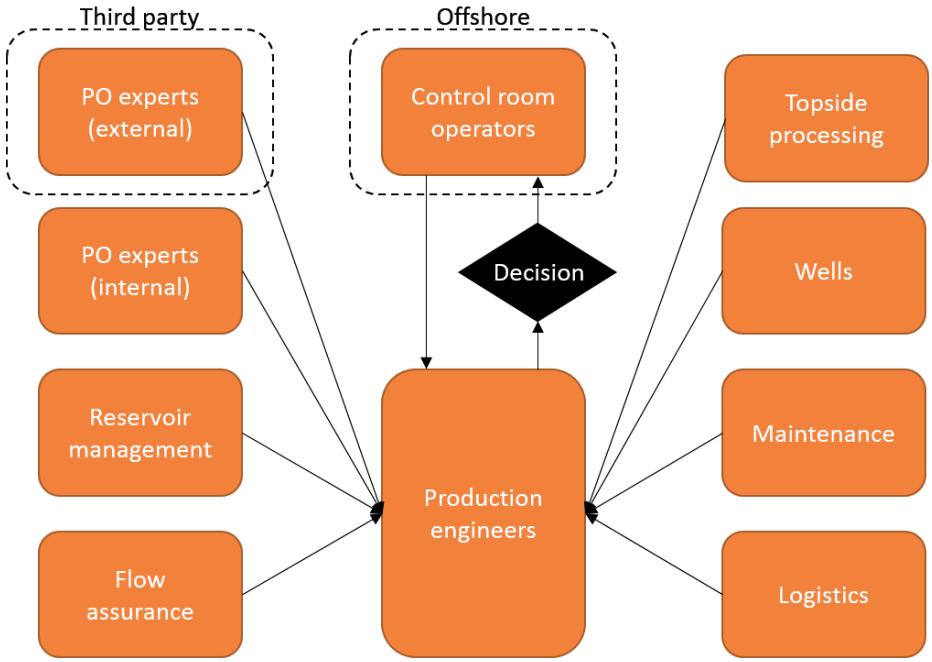


Figure 5.3.: Players that influence the production planning process.

For a typical offshore asset the PEs run an onshore operation center in teams of 2-4 persons. The daily operation normally begins with a morning meeting between the PEs and the operators located at the control room offshore. Together they share information about the current operation to prepare for imminent activities and to solve unresolved issues. To plan future production the PEs periodically (weekly to monthly) create a production and injection plan (PI plan). This plan, which is a

[2]The domains of geology and reservoir management, which influences PO decisions, are always hampered by significant uncertainty.

core activity for the PEs, requires extensive input from other groups such as reservoir management and topside processing. The engineers in these groups typically come from different disciplines and may have different incentives for changing the production plan. In particular, the PI plan should obey the production constraints from topside processing and the long-term injection plans from reservoir management. Even though some of these constraints are stable and predictable, others are less predictable and connected to specific events or temporary issues, like instabilities in the processing train or maintenance activities. The PI plan is discussed in daily meetings between the PEs and operators, and subsequently implemented by the operators. In the cases where model-based tools are used to develop the PI plan the production engineers normally also collaborate with specialists from the software supplier and company-internal specialist on modeling and simulation.

The value of a model-based production optimization tool is difficult to measure as it encompasses production gains, cost reduction, mitigation of risk, enhanced HSE, and possibly other non-economic values. To quantify its potential value, operations with and without model-based tools should be compared. This is problematic since operations and field properties change during the evaluation period. Therefore, it is important to assert the average increase or loss of value over a reasonable time span (Mochizuki et al., 2006). A common claim in the oil industry is that model-based production optimization has the potential to increase production value with 1-4%, especially for mature assets. It is quite hard to find open source references to support this claim. One example, however, is given in Teixeira et al. (2013) where the use of a model-based production optimization system increased oil production rate by 1.2%. Another example of added value by reducing costs and mitigating risks is less wear and tear on the equipment as a result of operating within system constraints to avoid erosion due to sand production (Castanier and Rausand, 2006).

5.4. Observations

A key motivation for this paper is to argue that state-of-the-art PO methods are generally not used as much as one would expect in the upstream industry, and to reflect on some causes for this. In the following we have grouped our observations from interacting with the industry as a set of themes. These are, by their very nature, interconnected, but we believe that this structure is a fair starting point for the discussion.

5.4.1. Instrumentation and Data Availability

A sufficient amount of informative real-time data is required for a continued evaluation and improvement of a production system model. Thus, model-based optimization software is indirectly dependent on having sufficient instrumentation and data availability.

During the last two decades we have seen major advancements in instrumentation and data availability. Integrations frameworks that collect real-time data and make it available to data analytic tools have been in use for some time now, and have proven to be an invaluable technology. Further, there has been a general increase in the number of installed downhole and subsea sensors per well. Today, the common subsea production system is well-instrumented with redundant pressure and temperature sensor pairs installed in the wellhead and wellbore. MPMs are also installed more frequently, but are still sparingly used due to high installation and maintenance costs. The growing trend in use of instrumentation is backed by advancements in technology such as fiber optic sensing (Kragas et al., 2001).

Increased use of instrumentation benefits greenfields, as well as mature fields when new wells are completed. However, many mature fields operate with a very low level of instrumentation, which usually is reflected in little to no modelling effort. Thus, for mature fields, where production optimization has the highest potential, lack of instrumentation is a major obstacle for the use of model-based tools. Model-based production optimization is considered to have a larger potential in mature fields than in greenfields (van der Linden and Busking, 2013; van der Linden, 2014). Reasons for this is that mature fields more often have complex and varying production bottle-necks, and flow assurance problems.

5.4.2. Uncertainty and Model Calibration

There are mainly two types of uncertainty of relevance to our discussion; measurement data uncertainty and model uncertainty. The latter is especially pronounced in well inflow models. This becomes important since well rates normally are estimated through infrequent well tests, thus rates have to be estimated between these temporal measuring points that are far apart. As a consequence, the challenge for PO is less related to measurement uncertainty than uncertainty in the models as such.

The starting point of a fruitful PO workflow, is accurate knowledge of the current operating strategy and operating point. The low temporal predictability of simple well inflow models (e.g. linear production index models), make it hard to assess how much each well is producing if the reservoir conditions change, or if the last well test is outdated. Moreover, temporal uncertainty is also related to scaling and wear of the choke valves. A move away the current operating point, suggested by a PO strategy, is subject to spatial uncertainty since the model may be less accurate at the new operating point due to un-modelled nonlinearities in the system. This uncertainty affects the production network model in particular, since re-routing of well streams and flow regime changes are particularly hard to model correctly. Infrequent well tests combined with low temporal predictability of the inflow models, and low spatial predictability of the production model are major hurdles for successful use of model-based PO.

As alluded to in the above discussion, the production system model must be calibrated on a regular basis due to the changing conditions in the reservoir and the

highly nonlinear behavior of multiphase flow. The model is a coarse approximation of the production system and has only local (spatial and temporal) prediction capabilities. To maintain a useful prediction capability for the upper layer in Figure 5.2, the model must be calibrated when the operating point has moved considerably or when a significant time has passed since the previous calibration. Calibration requires informative measurements of the system state. In practice, these are obtained by performing experiments on the system. For example, the performance of individual wells is measured by performing well tests.

Well testing is a disruptive operation that causes production losses and requires careful planning. Thus, there is a clear cost-benefit trade-off in the design and execution of well testing. Traditionally, the oil companies have kept the amount of well testing at a minimum. A consequence of this is that data for calibration purposes is scarce. To tune the tens to hundreds of parameters of a production system model from normal production data as well as data from campaigns one has to solve an underdetermined problem, which has many solutions, some of which give unrealistic parameters. To identify a realistic solution one therefore has to apply modelling and engineering knowledge. Even when considerable effort is invested in calibration, the resulting model will have limited prediction capabilities away from the snapshots.

Another challenge with model maintenance is that it requires modelling efforts across several disciplines, from both field engineers and optimization specialists, and possibly by personnel that do not see the immediate value of a well-maintained model. Moreover, this competence is usually limited resource in oil companies as well as service companies.

5.4.3. Disruptive Operational Events

There are two main reasons for changing the production strategy. These are varying reservoir conditions and disruptive events triggered by equipment failure and scheduled maintenance.

From the PE workflow perspective, the varying reservoir conditions, which normally are slowly changing, are fairly easy to handle. Further, the recalibration of inflow and production system models are usually done close to the current operating conditions, making it a less work demanding and complex task. Since the changes are small and the new optimal operating strategy is likely to be close to the current strategy, the production loss due to non-optimal operating conditions will be limited; thus, making the PO task less time critical.

However, disruptive events such as equipment failure or maintenance usually require larger changes to the production strategy. This again requires more complex and work intensive recalibration of the models and possibly also more time consuming optimization runs. Since the time period the new production strategy will be optimal often is shorter, e.g. until the equipment is fixed, the PE and operators might from a cost value perspective, be motivated by a PO strategy that fast returns reasonable solutions , rather than a time and work consuming close to perfect

solution.

There is a wide variety of operational events that may influence production. We have mentioned well tests. Another typical class of such disruptions are topside maintenance or sand control related. As an example, one informant mentions a field where deposition of asphalthenes in the flow lines requires regular cleanup operations ("pigging").

"It is a very regular activity. So on an average anywhere from two weeks to a month you have to do some type of maintenance on the subsea system, it all depends on the rate of deposition [..] and what we've found is that certain combinations of wells induce a certain frequency in which you have to perform maintenance, and those are the things that we are trying to collect more information on [..]."

The frequent occurrence of disruptive events that trigger the need for large changes to the production strategy, undermines the value of optimization efforts and is an important obstacle for today's best practice PO workflows.

5.4.4. Limitations in Software

Here we discuss the software limitations of the upper layers in Figure 5.2 in two parts; first we consider the limitations of the typical production system simulator, then the optimization software as such.

There is no doubt in the great value multi-phase process models have for studying and improving the operation of subsea production systems; see for instance Lunde et al. (2009). Virtual flow metering systems, especially, have been a successful technology. These systems have become a prerequisite for many of today's greenfields, even when MPMs are installed. The commercially available and in-house process simulators are improving every year; they become more accurate, faster and more complex. A key limitation, however, is their need to be manually calibrated with limited information available (as discussed in Section 5.4.2). Extra information is made available through campaigns such as well tests. Even with a shortage of information we believe that existing tools for model calibration can be improved to better support, or even automate, the current model calibration workflow. This may prevent human errors and allow for faster model adaption to new data.

Another common issue with process simulators is that they are packaged as *black-box* programs, that is, their inner workings are not revealed to the user to protect the intellectual property of the software vendor. As a direct result the machine-machine interface between the process simulator and optimization software is restrictive and may prohibit the use of efficient optimization solvers. As an example, it is not common for multi-phase flow simulators to offer gradient information, which is a prerequisite for most efficient solvers. Structural information about the network is another type of information that may aid the optimization software, but again this is normally unavailable through this interface.

Optimization software has advanced considerable during the last two decades. Today, there are several vendors that offer robust and fast optimization software that can handle nonlinearities and discrete decision variables, which often occur

in production optimization problems; e.g. as routing decisions. In several of the interviews we conducted we asked the following question: Which features of modern optimization software do you desire the most? The top choices were speed and integer handling to support routing decisions. We have not been able to identify any major limitations for these features in optimization software – on the contrary, state-of-the-art optimization software is both fast, reliable, and most certainly capable of handling discrete variables. In our experience, we find that the major limitations lie in the machine-machine interface of the process simulators since it prevents fast production optimization that can handle discrete decision variables rigorously.

5.4.5. Trust in Models

Trust is important for wider deployment of model-based decision support tools. We have seen several trust-related issues that negatively affect the reputation and lifespan of model-based PO. First of all, it is important that the engineers understand the output of the system, its limitations and uncertainty. As one of our interviewees pointed out: "it's not about the ease of using a product, it's about understanding the output from it." Of course, we believe both items are important. However, a problem with PO tools is that the "reasoning" behind the results may be difficult to comprehend, and thus, to trust the results that a tool offers. Distrust can, however, be mitigated through training courses by providing the users with a conceptual understanding of mathematical optimization in the context of production optimization. Second, it is hard to quantify benefits. This is fundamentally difficult since a field may be operated in only one way at a time, making direct comparison of operation with and without model-based PO difficult. Demonstration is, however, possible if the field is operated with and without model-based PO sequentially, as long as the state of the field is kept fairly constant.

According to our experience, production optimization software has a long way to go to achieve the necessary trust level of its users. Trust must be earned through the quality of the technology itself (the quality of the optimization as such, and the handling of uncertainty and information not included), support and management (e.g. demonstrating the value of small increases in production) and training (learning how to QA the recommendations from the software). Importantly, trust relies on a system for maintenance of the models. Thus, it is not only a matter of trusting the recommendation provided by the system, but also believing that maintaining the application is worthwhile.

Unfortunately, the self-energizing process where loss of trust in models occurs because of poor model maintenance is a frequently reported experience. When a poorly calibrated model predicts results that do not live up to the expectations of the users, interest is lost, and efforts in keeping the model up to date are reduced. This stereotypical process, illustrated in Figure 5.4 is a quite common phenomenon and can only be overcome by strong management support during implementation, deployment and production. In the success stories we know of, individual champions

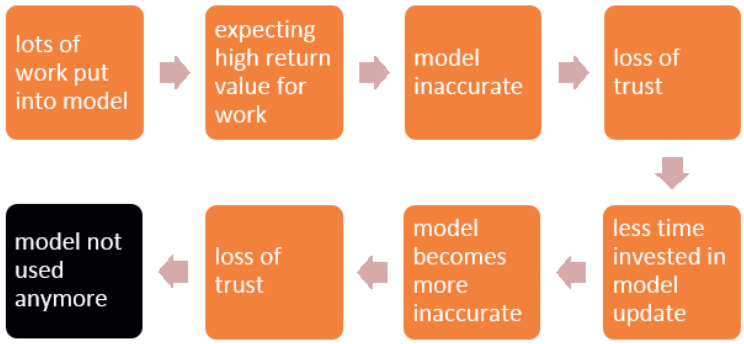


Figure 5.4.: The stereotypical cascade of events leading to loss of trust in a model.

with the attention of management have been crucial.[3]

5.4.6. Organization-wide Deployment of Model-based Tools

From an organizational point of view some degree of standardization of PO systems and practices between fields, and for some purposes between disciplines, is necessary. Even though it might be tempting and logical, it is not a trivial task to replace existing systems on specific sites and in specific disciplines. The literature on failed integration efforts of new information technology systems across disciplines and sites is extensive. Though standardization must be pursued, some level of tailoring is necessary to fit the specifics of each field. Key reasons for this are:

- The level of instrumentation of the field, hence the availability and quality of data, and the control options available vary, see Section 5.4.1.
- The specifics of each field such as geology, depth, hydrocarbon composition, technical lay-out of topside and subsurface systems, produces optimization challenges, and constraints differ considerably between fields.
- When deploying a tool, one must consider the fact that specific work practices co-develop with it. The expertise of the engineers is often tightly coupled to the tools and methods they work with. Almklov et al. (2014) show how key specific tools and systems over time co-evolve with work practices, which means that changing them might mean radical changes also to work practices.

Thus, even though it makes sense, from a knowledge management perspective, to standardize tools and practices within a company, the relevance and usefulness will vary and so also the friction involved in the standardization processes. Organization-wide deployment of model-based systems is perceived as difficult

[3]See (Stenhouse, 2008) for examples on the importance an application champion.

(Stenhouse et al., 2010) since models need specialization towards each individual asset. Generally, also outside petroleum production (see e.g. Ellingsen and Monteiro 2006), ICT- integration efforts across disciplines require much work, in particular to align terminology, categories, level of detail and specific needs found in one location with the need for standardization from an overall perspective. In sum, given the variety between different fields, organization-wide deployment of PO tools faces a challenge in terms of the need for standardization versus the need for tailoring.

5.4.7. Finding and Sustaining Integrated Competence

There is a distinct divide in competence between modelling and optimization specialists, and field engineers in most oil companies. Successful use of model-based tools for operational decision support requires both an understanding of the field as well as an understanding of how it translates into a production system model – with its advanced multiphase flow correlations and thermodynamics. In addition to the modelling competence, typically found among petroleum and chemical engineers, some basic knowledge of mathematical optimization is necessary when working with optimization tools. This competence is normally found among (process) control engineers. To obtain the combined, or integrated competence needed to run model-based production optimization, field engineers are typically supported by a centralized or external group of modelling and optimization experts.

Model maintenance, which we discussed in Section 5.4.2, is one aspect of model-based decision support in which field knowledge must be combined with modelling expertise. Consider the following anecdotal conversation overheard at an oil company by one of our interviewees: "'Look, the PVT model says <match>so it must be good.' No, you pushed the match button that is why it says <match>." Here, the user misunderstands the output from the product: the output says that the PVT has been matched, which does not necessarily mean that the match is good, i.e. that the PVT model will match reality. This must be verified by the engineer, as is clearly pointed out by the co-worker.

Cross-disciplinary competence is often needed in modeling tasks that interface closely with operational issues. This integrated competence is rarely found among field engineers or modeling and optimization specialists, and it is not commonly developed in courses and training within the oil companies. The integrated competence may develop over time by collaboration and information sharing between the different groups. However, as these may be geographically dispersed, there are significant obstacles to this specific competence.

5.4.8. Limited Sharing of Information

Information sharing across groups and sometimes company boundaries is normally necessary to fully benefit from model-based PO tools.

As discussed in Section 5.4.7, the PEs do not necessarily obtain the required modelling and optimization skills to maintain and use model-based PO tools. To support the PEs, in-house optimization specialists and/or external PO expertise are often engaged. Referring to Figure 5.3, the groups that we discuss are named PO experts (external), PO experts (internal), and Production Engineers. The external PO expert service is located at a vendor or service company, which typically has ownership of the PO software and a service contract with the E&P company. In-house optimization specialists are often located in centralized teams that provide organization-wide support to the company assets. Some configuration of these three groups make up the PO capability of an asset.

In our interviews with internal and external PO experts we spent a lot of time discussing the challenges related to information sharing. The core of these discussions was the following fact: for an external expert to provide a meaningful PO service, sufficient information must be transferred from the asset to the expert, possibly across company boundaries. Below we summarize the discussions on this matter.

As noted in Section 5.3, sensor data often needs human interpretation to be useful. They tell more to a PE with experience from the specific field than to an "outsider". This understanding of the data comes in part from an awareness of the operational situation. The outsider, in this case being the internal or external PO expert, will not understand this operational context in which the data makes sense, unless the PE has communicated this information. We argue that this is true also for internal PO experts that provide organization-wide support, since it is unrealistic for them to maintain knowledge of all asset operations. Consequently, the PO expert's understanding is based on assumptions of data and model quality, and even decision processes, that may be incorrect. This information gap may well lead to misguided PO advice.

When an external PO expert service is used to aid in the daily or weekly PO activity there is a need for communicating, across company boundaries, measurement data as well as operational information in regular batches. In return, the PO experts provide PO advice based on this information. There are several difficulties with this model for collaboration. Usually both parties have a shared software platform where the model-based PO software resides. With this platform they can acquire the same information (sensor and soft sensor data) about the production. However, other operational information, such as planned shutdowns, failing sensors, closed valves, injection of MEG, injection of water or gas and pigging operations, is usually not available. This contextual data constitutes important input to the PO analysis. There is, for instance, no point in disturbing the system by ramping up a well, which has a planned shut-in the next day. Likewise, if there has been an ongoing water injection campaign then some of the well performance models may be unrealistic and in need of updating. In this case there is little to no value in running a model-based PO tool based on the current models. It is not reasonable to expect that the PEs, who work in a hectic environment, can convey all relevant operational information, especially across company boundaries.

Another challenge with sharing of information across company boundaries is that production optimization is an activity that involves highly sensitive information for the E&P company, e.g. daily production rates. This information will necessarily be visible to anyone involved in a PO activity. Thus, a confidentiality agreement, which possibly limits the means for collaboration, is usually signed by all participants.

Based on the above discussion one may question if a production optimization strategy that relies on extensive information sharing across company boundaries is a sustainable solution. On a final note some of the challenges outlined above are also relevant beyond the operational context described above, for instance when a software vendor is engaged to develop, customize or maintain PO software for an E&P company.

5.4.9. KPIs and Incentives

As previously mentioned, the gains from model-based production optimization may be hard to quantify. Thus, it is difficult to define measurable KPIs that encourage efforts towards using and maintaining PO tools. We have earlier encountered examples of KPIs that define conflicting goals between the groups involved in Figure 5.3. One example appears in fields where an aggressive short-term production target and long-term recovery goal are conflicting, i.e. maximizing production rates in the short term will harm long term recovery. Some oil companies have bonus structures that reward the PEs if they hit yearly target rates – which may encourage a greedy, short-term production strategy.

KPIs for PO tools should communicate the desire to regularly improve the current operating condition. Finding ways to demonstrate and reward the improvements made by model-based tools is a key to achieving this. Measurable KPIs are important to motivate the continued use of model-based tools, by giving the users with incentives for maintaining and caring for the tools.

5.5. Discussion

The observed challenges presented in Section 5.4 act along three axes: data, technology, and people. Thus, this discussion is structured similarly.

Data. Data is a key for model validation and calibration, estimation and awareness, and production optimization. Poorly instrumented fields have a lack of quality data, and modeling and optimization may thus not be possible. Sharing data between people and companies is difficult for several reasons. We mentioned the interface between proprietary software and optimization systems, but also between different professions and companies, as challenging areas.

Technology. On a long-term horizon we see a trend towards faster and more accurate models for online use. The advancements in reservoir and multi-phase flow simulators come slowly since extensive research is required to increase our understanding of the intrinsic and difficult physics of multiphase flow. We believe

that better instrumentation and more informative data will be a game changer in modelling of future fields. It will allow for simpler models for production estimation, and produce more informative data for calibration of production optimization models. On a short-term horizon we believe that advancements can be made relatively quickly by improving process simulator interfaces to allow full utilization of the modern optimization solvers that are available at the commercial market today. This low-hanging fruit can be reached by better fitting two existing technologies, that is, by improving the interface between the process simulator and optimization software. Software for automatic model calibration is another technology advancement that may give value in the short-term. As discussed, model calibration is a prerequisite for model-based production optimization. It is an analytical activity, which should be performed whenever new calibration data is available. One obstacle for fully automating model calibration is that it may be difficult to build in the engineering knowledge that is applied to screen bad calibration settings. However, we do not believe that this obstacle is insuperable – operator and service companies in the upstream business have achieved goals far greater than this.

People. We have shown that model-based PO requires as composite set of skills and knowledge. Knowledge of the specific oil field and production system, and quite advanced modelling and optimization skills, need to be combined to implement and use such tools. This combination is sometimes found in multi-skilled individuals, but more often it is a product of extensive and continued collaboration between (internal and external) modeling experts and field engineers. This is organizationally challenging and requires management support. The integrated competence we identify as important, is thus both an individual quality and the outcome of successful collaboration. Model-based PO requires work and dedication, especially in the implementation phase, but also in terms of regular maintenance. This work may be challenging to align with KPIs for different departments and also to fit incentive structures for individual workers, as the "output" of a well-tuned PO tools are hard to quantify.

Task distribution and work processes vary somewhat between companies, and we have not gone in detail on work process organization as such. We have however identified several issues – i.e. trust, KPIs and the need for integrated competence – that should be addressed when developing and adjusting work processes regarding model-based optimization.

5.6. Conclusion

Referring back to the title of this paper we assert that the question is qualified: production optimization in the upstream industry *is* difficult. Numerous papers have documented failed attempts at implementing and achieving value from model-based production optimization, and we have provided several reasons that contribute to explaining this. In the introduction we compared real-time optimization and decision support in upstream and downstream systems. There are fundamental dif-

ferences related to instrumentation and data availability that makes it much more difficult to maintain models of upstream production systems. This has direct consequences for model-based production optimization. To answer on why model-based production optimization is difficult we have identified and discussed several challenges with data, technology, and people that the industry seems to agree on.

In spite of the challenges outlined in this paper, the authors are optimistic about the future role of model-based tools in the upstream business. We believe that the technology gap between upstream and downstream will shrink in terms of automation and reliance on model-based tools. This will give the oil companies opportunities for higher operational awareness, more optimal utilization of the reservoir and production system, and ultimately a safer operation.

Acknowledgements

This work was supported by the Center for Integrated Operations in the Petroleum Industry at NTNU in Norway (IO center).

Chapter 6

Concluding remarks

Model-based real-time optimization and monitoring for subsea production systems have been studied in this thesis. As argued in the introductory chapter, methods based on surrogate models are promising since they fit into the framework of mathematical programming with general algebraic (a priori known) constraint functions. They may hence utilize the full potential of available NLP or MINLP solvers. This is opposed to simulation-based optimization using black-box solvers, which allows for a simple problem formulation, but generally yields unpredictable results at high computational cost.

Several function approximation methods exist that may be applied to generate surrogate models replacing simulation-based constraint functions. In the introduction, a selection of surrogate models (categorized by approximation method) were presented and some of their properties related to accuracy, cost of evaluation, cost of construction, and smoothness, were listed. As argued for based on these properties and some experimentation with different surrogate models, the B-spline seemed a suitable candidate for representing simulation models. With nice properties such as local support, it is cheap to build and evaluate, and provides high accuracy and smoothness. It may also be used to interpolate or smooth data, the latter being useful in cases where sample data is noisy. Unfortunately, at the outset of this thesis it became clear that the previous research on optimization with B-splines was very limited. Additionally – or perhaps consequently – there were not any optimization solvers that could handle constraints with piecewise functions like the B-spline, at least not directly. Thus, a considerable effort had to be put into developing a rigorous theory for optimization with B-splines. This theory was subsequently implemented in the optimization framework CENSO, and benchmarked against other state-of-the-art solvers on a set of suitable problems. This was a difficult comparison since there were no support for B-splines in the other solvers. A compromise was made by using a test set consisting of polynomially constrained MINLP problems. The test set acted as a common ground for comparison since a B-spline may represent any polynomial. The work described above was published in the article by Grimstad and Sandnes (2015). It lay the foundation for performing real-time production optimization with B-spline surrogate models.

Trailing the development of CENSO, with some overlap, was the development of a mathematical framework for simulation-based production optimization. The framework was developed to simplify and ensure proper modelling of general pro-

duction optimization problems that rely on simulation-based models, surrogate models, or a mix of the two. The framework disaggregates the production system to a network of low-dimensional and computationally light (and robust) simulation units. Furthermore, the framework ensures proper handling of routing variables by excluding them from simulation-based constraints, and including them in linear constraints only. This increases the efficiency of bounds tightening techniques and may consequently lower the solution times. The disaggregation also allows the simulation units to be replaced with surrogate models, resulting in a generalized algebraic formulation suitable for a NLP or MINLP solver. Thus, the simulation and optimization is decoupled during the solution process, removing communication delays, evaluation costs, and convergence issues (for example due to hidden constraints). This decoupling of simulation and optimization was key to achieve solutions in real time – times measured in terms of minutes and seconds, not days and hours.

The developed mathematical framework can be used to model any multiphase flow network under mild assumptions. It combines mass, momentum, and energy conservation laws with sophisticated (simulation-based) relations for pressure and temperature drop. The inclusion of temperatures and enthalpies in the modelling helps to reduce the loss of accuracy introduced by surrogate models. As demonstrated in several cases, this loss of accuracy is negligible compared to the existing inaccuracies/uncertainties in the simulator model. The framework was implemented in C++ code using a graph data structure (see Appendix B). With minimal input from the user, the code generates a mathematical programming formulation representing the production optimization problem. The semi-automatic generation minimizes the probability for user error during modelling. The resulting problem formulation can be solved using CENSO.

Using B-spline surrogate models, CENSO was applied to solve production optimization cases for several real production systems. In these cases, a potential production increase in the range of 1-4% was demonstrated. The mathematical framework proved to be flexible in terms of modelling different operational constraints and network configurations. It also produced formulations with consistent behaviour across optimization solvers. This work was done in tight collaboration with the oil company BP, which facilitated and showed great interest in the development. The above application was documented in the paper by Grimstad et al. (2016).

The realistic cases discussed above showed that CENSO is capable of globally solving small to medium-sized production optimization problems in real time. These problems required CENSO to solve thousands of (local) subproblems. This shows that CENSO can be employed to locally solve large-scale production optimization problems. In general, global optimization scales exponentially with problem size – that is, the number of complicating variables – and it is not reasonable to expect a global solution for larger problems with today's technology. The fast solution times for subproblems may however improve current technologies by opening for: 1) larger problem-sizes and hence more accurate models, 2) consideration of

more routing options, 3) global optimization (possibly terminated early to honour time constraints), and 4) handling of uncertainties, for example by including chance constraints.

The use of B-spline surrogate models for simulation-based real-time monitoring was pursued in (Grimstad et al., 2015c). This paper was motivated by a desire to demonstrate the versatility and accuracy of B-spline surrogate models. As the paper concludes, the B-spline surrogate models are suitable also for real-time monitoring. The loss of accuracy caused by approximating each simulation unit with a B-spline surrogate model was negligible. Furthermore, the analytical derivatives of the B-spline contributed to a computational efficiency that allowed monitoring in real time.

There are many obstacles to achieving and maintaining a high-performance model-based tool for production optimization. A discussion based on industry experience and a series of interviews on the technological and organizational challenges was given in (Grimstad et al., 2014). As pointed out in this discussion, the upstream industry is lagging behind the downstream process industry in the application of RTO. The author hopes that the industry players are willing to focus their efforts on the issues identified in (Grimstad et al., 2014). By reducing these issues in a step-by-step fashion, the industry may finally be able to embrace RTO technology.

The work in this thesis has resulted in a large code base with tools facilitating simulation-based production optimization. The code code is written entirely in C++ and has been shared with the public; it is described in more detail in Appendix B. The author hopes that someone will use this software – with commercial or academic interests – for purposes that may enhance current RTO technology or continued research on optimization with surrogate models.

Several observations and demonstrations were made during the thesis work, a few of which are listed below:

- The B-spline offers great approximation flexibility and accuracy, low evaluation costs, analytical derivatives, and a high degree of smoothness. It is suitable for surrogate modelling in optimization.
- A mathematical programming framework offers great flexibility in modelling production optimization problems.
- Commercial optimization solvers offer low solution times as long as the problem formulation is good: i.e. is smooth and has derivatives. A good formulation gives consistency across solvers.
- There is a trade-off between model complexity and computation time. With sufficient model accuracy, small to medium-sized (daily) production optimization problems can be solved to global optimality in real time. It is currently not possible to globally solve (daily) production optimization problems with high model complexity (e.g. with many fluid components and many routing options) in real time.

- Poor model maintenance due to technological and organizational issues seems to be limiting the performance of current RTO applications in the upstream industry. There is a need for new infrastructure and work procedures that simplify and increase automation of model maintenance.

Before continuing with some suggestions for further research, the author would like to revisit the research objective posed in the introduction of this thesis. Has the work in question contributed towards an enhancement of current model-based decision tools for production optimization? The author proudly believes so. A new method for model-based real-time optimization has been developed, implemented, and successfully tested together with an industry partner. Several applications have been demonstrated. And an extensive code base has been documented and shared for the public's interest. As it usually is, the work has spawned more questions than it has answered, and in the final sections of this thesis the author would like to recommend some promising research paths that may be pursued. But caution is advised – new questions may spawn and head hairs may wither!

6.1. Recommendations for further research

There are several interesting directions in which further research on the topics in this thesis may be conducted. The recommendations given here will focus on research that may enhance current model-based RTO software. In particular, the following three directions will be discussed: comparing different surrogate models, handling of uncertainties, and integration of models. All of these are exciting avenues for further research that may yield significant contributions.

6.1.1. Comparison of surrogate models

In the introduction of this thesis a list of surrogate model types was given. While this work focused on B-spline surrogate models, there are several interesting candidates that may be just as suitable for production optimization. For example, stochastic process modelling, for example using Kriging, seems to be appropriate when incorporating noisy measurement data in the production optimization models. Such models have already been used to model expensive black-box functions in optimization (Jones et al., 1998). Furthermore, piecewise linear models have a long history as surrogate models in production optimization. A comparison of piecewise linear models was recently given by (Silva and Camponogara, 2014).

It would be interesting to compare existing nonlinear surrogate models, such as the Gaussian process models, RBF models, and splines. A comparison would map the strengths and weaknesses of the different surrogate models and provide a guidance for practitioners. Characteristics such as computational cost of construction and evaluation, scaling in terms of variables, accuracy, smoothness, availability derivatives and convex relaxations, ease of use and solver compatibility, and flexi-

bility in terms of interpolation or smoothing in case of noise, should be included in such a comparative study.

The nonlinearity of the above-mentioned models can be handled by a global optimization method, such as spatial branching. This is fundamentally different from the piecewise linear approach, which is handled by logic, for example by using integer variables or branching on special ordered sets. An unanswered question is whether piecewise linear models should be preferred over nonlinear approximations as surrogate models. Is integer branching better than spatial branching? The answer is possibly "Yes" today, since current MIP technology is more mature than current technology for spatial branching. The author would like to conjecture that, under the same conditions on accuracy, optimization with piecewise linear and nonlinear models, handled using branching on integer and continuous variables respectively, will prove to scale similarly in the limit, as the optimization technology matures.

6.1.2. Handling of uncertainties

A deliberate treatment of known uncertainties may enhance current RTO software for production optimization. In particular, it is of great interest to guard against control moves that are likely to violate operational constraints or yield less profitable operation. The former is crucial for the validity of control moves under operational considerations such as safety limits; the latter is important to build operator trust in model-based decision support tools. A model-based RTO software may suggest such poor control moves if uncertainties in the operational constraints and/or production system model are not treated systematically.

Uncertain operational constraints

Most operational constraints in daily production optimization can be considered to be uncertain. For the purpose of this discussion, operational constraints are classified as being measured or not measured, and as noisy or noise-free. A routing constraint may be regarded as a measured and noise-free constraint – there is no uncertainty related to the violation of such a constraint. However, most operational constraints are on pressures, temperatures, or rates, which are measured with significant noise. This noise should be accounted for by the RTO software. Some operational constraints are not directly measured; examples include constraints on in-situ velocity or amount of sand erosion. The violation of such constraints must be quantified by estimation, which naturally introduces model uncertainty. If the estimation is based on noisy measurements, the violation of the constraint is uncertain due to measurement noise and model uncertainty. As this discussion illustrates, there are several types of uncertainty related to operational constraints that each could be addressed differently. The *chance constraint* and the conditional value-at-risk are two available tools for a systematic handling of uncertainty in operational constraints (Hanssen et al., 2015). Modelling uncertain constraints using

the conditional value-at-risk seems promising, but there are still questions related to scalability as the problem size increases considerably with many scenarios.

Model uncertainty and model maintenance

In many cases the uncertainty in the production system model is more severe than that related to operational constraints. As discussed in Chapter 5 (Grimstad et al., 2014), model uncertainty can lower the value of model-based decision support software significantly. This challenging issue touches on one of the core difficulties of operating an upstream production system – that highly important decisions must be made with very little information at hand. There are several preemptive and reactive measures that may decrease model uncertainty. Some of these measures require technological advances as well as organizational adaptation – a change in operational philosophy.

First of all, model uncertainty is directly linked to lack of and/or poor use of available data in model calibration. Lack of data (suitable for model calibration) is due to infrequent well testing. Poor use of available historical production data may happen because of limitations in software for model calibration, difficulties in retrieving data, lack of time/resources for model calibration, or lack of competency, to mention some possible issues. Uncertainty degrades the predictive capabilities of the models and renders them useless in the worst case. The problem is in some cases mended somewhat by the impressive predictive capabilities of empirical models, even when improperly calibrated. However, also sophisticated models are subject to considerable uncertainty over time. Models should be periodically calibrated as the reservoir conditions change over time. Flow conditions in the production system – if subject to scaling, wax deposition, or other flow assurance issues – may also change over time.

The most obvious way to reduce model uncertainty is to increase the number of well tests. This may, however, not be possible due to the high costs of well testing. This has caused researchers to look at alternative well-testing methods that attempt to extract more information during well testing (Goh et al., 2007) or during normal production (Zenith et al., 2015). Additional calibration points can sometimes be extracted from historical data using statistical methods and used for model calibration (Melbø et al., 2003). Another preemptive strategy is to detect uncertainty in models using statistical methods. Uncertainty may then be indicated to the users, who may initiate a calibration. One such method is discussed by Skibeli (2015). Keeping the users (production engineers) informed of possible uncertainties can save their time by helping to pinpoint model uncertainties. Ultimately, the indication of uncertainties may increase operational awareness.

A technological improvement that may better the situation for model-based tools is automation in the model calibration process. This may decrease the resources put into model calibration and increase calibration frequency. Full automation is, however, difficult since the expertise of engineers is required to validate the model parameters resulting from calibration.

Finally, a last avenue for research is the development of algorithms for production optimization under model uncertainty. When optimizing with models that lack predictive capability the search should be limited to a neighbourhood of the current operating point. This limitation can for example be specified using a trust region (Conn et al., 2009). The following steps outline a possible algorithm for optimization under uncertainty: 1) do a locally constrained optimization, 2) if it is optimal to move, take a small step away from the current operating point in the direction indicated by the optimization, 3) adjust the operating point to be feasible, in a fashion similar to a line-search, 4) if possible, update the model by performing a model calibration, 5) repeat the process from 1. This pseudo code has many similarities with an inexact sequential quadratic programming algorithm. Also, it has many similarities to the (less systematic) workflow of production engineers and operators. As mentioned above, to handle uncertainty in models may require more than technological advancements – it may require a change in operational philosophy.

6.1.3. Integrating models

The last recommendation for further research is related to integration of models for the reservoir, the production system, and the topside facilities. The motivation for performing such an integration is that it may expand the scope of production optimization both spatially and temporally. The integration may give solutions that are more optimal than those returned by siloed approaches that consider a single model and a reduced scope. It has already been demonstrated that short-term optimization may have a negative effect on long-time solutions. However, due to the gradually increasing uncertainty of predictions into the future, a trade-off between short-term and long-term solution must be found. The possible trade-offs lies on a Pareto front. As in economics, the trade-off illustrates risk versus reward. There are currently no systematic ways of asserting where on the Pareto front one should lie. Hence, practitioners of reservoir management stay on the (long-term) end, and practitioners of daily production optimization stay on the other (short-term) end of the Pareto curve.

Integration of models is a daunting task for several reasons. Firstly, it is difficult to couple and simulate models with different time scales. It is not trivial, for example, to define synchronization points when using variable step simulation. Secondly, integration may increase the simulation time considerably and render the integrated model useless for optimization purposes. Thus, the use of surrogate models, such as the B-splines presented in this thesis, is advised. Thirdly, integration of models is such a demanding task that it requires several persons and competencies. And lastly, the models to be integrated may be implemented in different software that are incompatible or has complicating interfaces. Another type of difficulty arises when integrating models that use different fluid compositions and accompanying thermodynamics. In spite of these challenges, several integration attempts have been documented in the literature (Queipo et al., 2003; Barber et al., 2007;

Rahmawati et al., 2010, 2012).

A motivation for integrating models can be found in the history of production optimization: In the beginning, optimizations were performed on isolated parts of the system due to limitations in computational power and solver capabilities. For example, gas lift was allocated to one well at the time, following some prioritization, generally leading to suboptimal solutions. Today, several problems are solved in one optimization problem formulation. Gas lift allocation, with constraints on total available lift gas, is but one problem addressed by such a formulation.

Appendix A

Notes related to optimization with splines

In this appendix some unpublished notes related to optimization with B-splines are collected.

For brevity the following derivations are performed for a univariate B-spline. Remark that the derivations extend to the multivariate case with few adjustments. The derivations are concerned with the B-spline constraint

$$f(x) = \sum_{j=0}^{n+p} c_j B_{j,p}(x) = y, \tag{A.1}$$

where $B_{j,p}$ is a p-th degree B-spline basis function defined by a $(p+1)$-regular knot vector. We denote the domain of f with $X = [x^L, x^U]$. The image of f is constrained to the box $Y = [y^L, y^U]$.

A.1. Bounds tightening with B-spline constraints

Utilizing the properties of nonnegativity $B_{j,p}(x) \geq 0 \ \forall j, p$ and partition of unity $\sum_j B_{j,p}(x) = 1$ for all $x \in X$, we may reduce the domain of f, i.e. the bounds on x, by investigating the control points alone. Below we illustrate the approach with an example for the constraint $f(x) = y = 0$. f is plotted in Figure A.1.

First, we attempt to tighten the lower bound x, that is, we want to see if x^L can be increased. Moving from left to right we consider one knot span at the time. The first non-empty knot span is $[t_1, t_2] = [0, 1]$ in which basis functions B_0 and B_1 are nonzero. The control points related to these basis functions are P_0 and P_1; and the related coefficients are $c_0 = 1$ and $c_1 = 1$, which both are strictly positive. Consequently, $f(x) > 0$ and the constraint $f(x) = 0$ cannot be fulfilled in this knot span. Thus, we can set $x^L = 1$. An alternative way of looking at the procedure follows from the fact that the B-spline lies in the convex hull of the control points related to the knot span, and if the convex hull does not intercept the zero line the constraint $f(x) = 0$ cannot be fulfilled.

We continue with the next knot span $[t_2, t_3] = [1, 2]$ and investigate the related control points P_1 and P_2. Related to P_2 is the coefficient $c_2 = 0$. Thus, the B-spline f may attain a value of zero (actually it can only achieve that value at the boundary $x = 2$). Similarly, if $c_2 < 0$ we could not be guaranteed that $f(x) > 0$ for all $x \in [1, 2]$. Thus, we terminate the bounds tightening of the lower bound x^L.

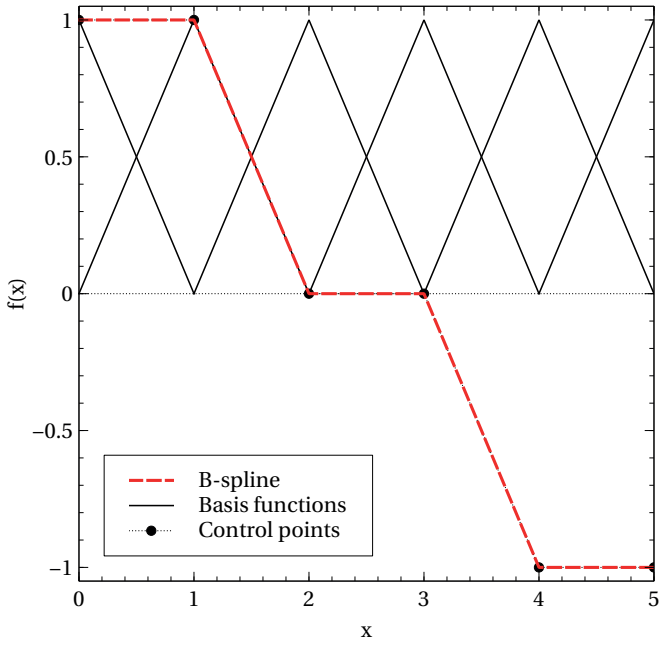


Figure A.1.: A linear ($p = 1$) B-spline function with knot vector $\mathbf{t} = \{0, 0, 1, 2, 3, 4, 5, 5\}$ and control points $\mathsf{P}_0 = \{0, 1\}$, $\mathsf{P}_1 = \{1, 1\}$, $\mathsf{P}_2 = \{2, 0\}$, $\mathsf{P}_4 = \{3, 0\}$, $\mathsf{P}_5 = \{4, -1\}$, and $\mathsf{P}_6 = \{5, -1\}$.

The procedure for the upper bound x^U is similar, but considers knot spans from right to left. The reader may confirm that the upper bound can be tightened to $x^U = 4$. By applying the procedure to the upper and lower bound a relative tightening of $100\left(1 - \frac{(4-1)}{(5-0)}\right) = 40\%$ is achieved.

A.2. Piecewise convex hull relaxation of B-spline constraints

As shown here, a strengthened convex hull relaxation for the B-spline constraint in (A.1) can be obtained by piecewise relaxation. The piecewise relaxation follows naturally from the properties of the B-spline and is, as will be shown, tighter than the convex hull relaxation in Lemma 2.1. Using binary variables, the piecewise relaxation is expressed as a MIP. The approach is similar to that of the piecewise McCormick relaxation (Kolodziej et al., 2013).

In the following, we utilize that the B-spline in (A.1) has local support

$$B_{j,p}(x) = 0 \text{ for } x \notin [t_j, t_{j+p+1}],$$

and a partition of unity

$$\sum_{j=i-p}^{i} B_{j,p}(x) = 1 \ \forall x \in t_i, t_{i+1},$$

to device a strengthened piecewise convex hull relaxation for constraints $f(x) = y$.

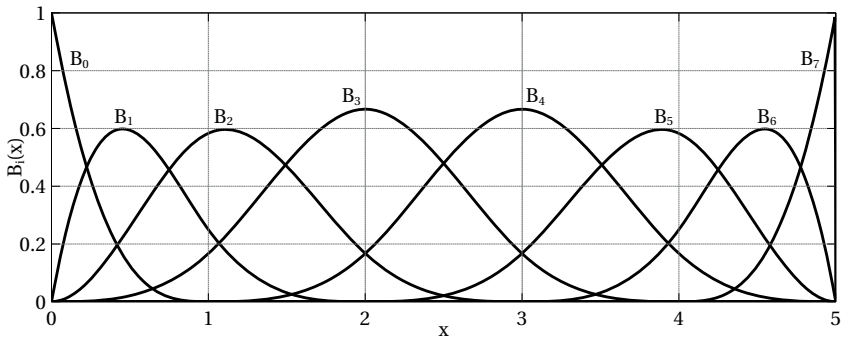


Figure A.2.: Cubic basis functions for knot vector $\mathbf{t} = [0, 0, 0, 0, 1, 2, 3, 4, 5, 5, 5, 5]$.

For a $(p+1)$-regular knot sequence the $p+1$ first knots are identical; the same is true for the $p+1$ last knots. In the example in Figure A.2, the first $p+1$ knots are $t_0 = \ldots = t_3 = 0$. Now, consider the first non-empty knot span $x \in [t_p, t_{p+1}] = [t_3, t_4] = [0, 1]$. As illustrated in Figure A.2, the spline has support from four basis functions in this interval, namely $\{B_0, B_1, B_2, B_3\}$; all other basis functions are zero. Thus, we can write the B-spline as

$$f(x) = \sum_{j=0}^{p} c_j B_{j,p}(x) \text{ for } x \in [0, 1].$$

In general we can split the B-spline into n segments $i = 0, \ldots, n-1$ as

$$f(x) = \sum_{j=i}^{i+p} c_j B_{j,p}(x) \text{ for } x \in [t_{i+p}, t_{i+p+1}].$$

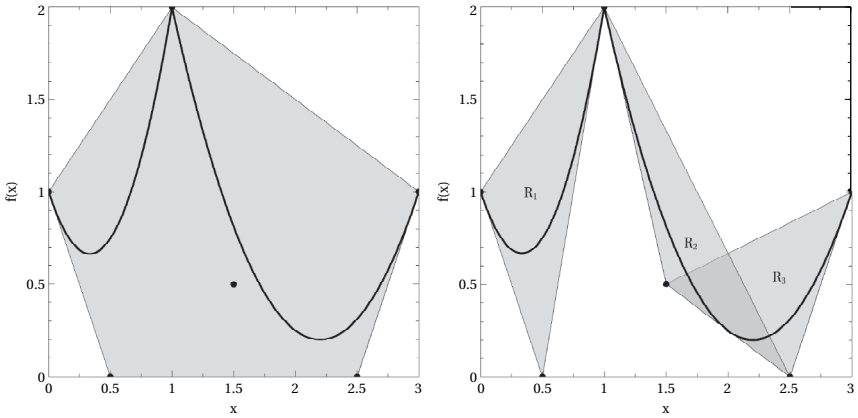
In each segment, the B-spline is a convex combination of its control points. That is, for each segment i we have that $(x, y) \in \text{conv}\{P_i, \ldots, P_{i+p}\}$ is a convex relaxation of $f(x) = y$ for $x \in [t_{i+p}, t_{i+p+1}]$. Using disjunctions we may express a convex relaxation for $f(x) = y$ for the interval $[t_0, t_{n+p}]$, i.e. the whole domain, as

$$\bigvee_{i=0}^{n-1} \left[\ (x, y) \in \mathcal{R}_i \ \right], \tag{A.2}$$

where

$$\mathcal{R}_i = \{(x, y) : x \in [t_{i+p}, t_{i+p+1}],\ (x, y) \in \operatorname{conv}\{P_i, \ldots, P_{i+p}\}\}\,. \tag{A.3}$$

The piecewise relaxation in (A.2) can also be written: $(x, y) \in \bigcup_i \mathcal{R}_i$. The disjunction can be modelled using binary variables as described in (Grossmann and Lee, 2003), resulting in a MIP formulation.



(a) (b)

Figure A.3.: The convex hull (a) and piecewise convex hull (b) relaxation for a quadratic B-spline ($p = 2$) with knots $t = [0, 0, 0, 1, 1, 2, 3, 3, 3]$ and coefficients $c = [1, 0, 2, 0.5, 0, 1]$.

Lemma A.1 (Piecewise convex hull relaxation)**.** *The piecewise convex relaxation in* (A.2) *is tighter than the convex relaxation in Lemma 2.1.*

Proof. For any segment $[t_{i+p}, t_{i+p+1}]$ the convex hull relaxation from Lemma 2.1 is the set

$$\mathcal{R} = \{(x, y) : x \in [t_{i+p}, t_{i+p+1}],\ (x, y) \in \operatorname{conv}\{P_0, \ldots, P_{n-1}\}\}\,. \tag{A.4}$$

Clearly, since the convex hull of a set of points is equal to or larger than the convex hull of a subset of the same points we have that $\mathcal{R}_i \subseteq \mathcal{R}$ on segment i. Since the piecewise convex hull relaxation is at least as tight as $\mathcal{R}$ on any segment, the overall relaxation must be at least as tight as $\mathcal{R}$. That is, $\bigcup_i \mathcal{R}_i \subseteq \mathcal{R}$, which concludes the proof. The consequence of the proof is illustrated in Figure A.3. □

A.3. Outline of a computational complexity analysis for Algorithm 3

In this section we provide an outline of a computational complexity analysis for Algorithm 3 in Chapter 2. Algorithm 3 is a spatial branch-and-bound algorithm, in which integer variables and complicating continuous variables are branched upon.

The analysis, as outlined below, may suggest some practical limits on the problem sizes one can expect to solve with Algorithm 3. It may also illuminate the trade-off between knot refinement and spatial branching. Knot refinement serves to tighten the convex hull and promotes fathoming of nodes, consequently reducing the tree size. It does, however, increase the size of the lower bounding problems. Spatial branching, on the other hand, is computationally cheap, but increases the tree size. Computational experience shows that a balanced combination of knot refinement and spatial branching is likely to give the lowest solution times on average.

We begin by investigating the *while* loop in Algorithm 3. For simplicity, we assume that the problem does not contain integer variables (the analysis may easily be extended to include the combinatorial aspect of branching on integer variables). Furthermore, we disregard the *optional* computations (i.e. the bounds tightening, knot refinement and upper bounding). The *while* loop then contains two required and computationally expensive operations: first a convex lower bounding problem must be solved, then spatial branching is performed.

The convex lower bounding problem $\mathbf{R}$ can be solved in polynomial time by an interior-point algorithm (Karmarkar, 1984). The convex hull relaxation used in Algorithm 3 produces LP relaxations. LP problems have a computational complexity of $O(n^c L^2)$, where n is the number of variables, c is a constant (e.g. $c = 3.5$ for the interior-point algorithm of Karmarkar 1984, and $c = 6$ for the ellipsoid algorithm), and L is the number of bits in the input (Bland et al., 1981). The bound can be interpreted as $O(nL)$ steps, of $O(n^{2.5})$ arithmetic operations, with $O(L)$ precision. We note here that L will be a function of the number of basis functions in the B-spline constraints. It will depend in particular on the number of control points used in the convex relaxations. The number of control points grows exponentially with the dimension d of a given B-spline constraint. This growth must be included in a proper analysis of the overall computational complexity of Algorithm 3.

Next, the computational complexity of Algorithm 1 must be investigated. This algorithm is used in the spatial branching to subdivide B-splines. In the first step of the algorithm, d knot insertion matrices $\{\mathsf{A}_i\}_{i=1}^{d}$ are computed. We assume that all matrices are of size $\mathsf{A} \in \mathbb{R}^{n \times \tilde{n}}$, with $n \leq \tilde{n}$ (here n and $\tilde{n}$ denote the number of univariate basis functions before and after knot insertion). The number of operations required by the Oslo 1 algorithm to compute each matrix is in the order of $\mathcal{O}(p^2\tilde{n})$ according to Lyche et al. (1985).[1] Thus, step one of Algorithm 1 requires $\mathcal{O}(dp^2\tilde{n})$, where d is the number of variables (dimension) of the tensor product B-spline. The second step of the algorithm is a sequence of Kronecker products. Since $n \leq \tilde{n}$, we simplify the computation of the upper bound by assuming that the matrices are square $\tilde{n} \times \tilde{n}$ (since $n \leq \tilde{n}$). Then, the Kronecker products require the following number of operations:

$$\prod_{i=1}^{d} \tilde{n}^2 = \tilde{n}^{2d}$$

[1] An improved Oslo 1 algorithm exists that requires fewer operations than the original algorithm.

The total number of operations for Algorithm 1 is thus $\mathcal{O}(\tilde{n}^{2d} + dp^2\tilde{n})$. In practice, the number of operations will be significantly lower if matrix sparsity is exploited. Still, this bound indicates that it may be computationally intractable to apply Algorithm 1 to B-splines in, say, $d = 10$ variables or more.

To summarize, the computational complexity of the *while* loop in Algorithm 3 is $\mathcal{O}(n^{3.5}L^2 + \tilde{n}^{2d} + dp^2\tilde{n})$, where L is a function of the number of control points $\tilde{n}^d$ in the B-spline constraints. The computational complexity of Algorithm 3 is bounded by the number of explored nodes times this bound. It is nontrivial to compute the number of nodes that must be explored to ensure termination. The bound in Equation (2.11) is however a good starting point. Finally, we note that knot refinement will increase $\tilde{n}$ and consequently increase the cost of each iteration. However, it will at the same time reduce the number of nodes in the tree according to the bound in Equation (2.11).

Appendix B

Software

This appendix gives an overview of the software developed during the course of this thesis work to facilitate production optimization using surrogate models. The software has been used to solve the cases in (Grimstad and Sandnes, 2015; Grimstad et al., 2015c, 2016; Foss et al., 2015), which include several realistic production optimization cases provided by BP Exploration Operating Company Ltd.

B.1. SPLINTER

SPLINTER is a C++ library for multivariate function approximation (Grimstad et al., 2015b). The first draft of the code was developed during the spring of 2012 by Anders Sandnes and Bjarne Grimstad. Bjarne Grimstad continued the development until the code was released publicly in October 2014. Since then, several people in the open-source community have contributed to the code. Today, SPLINTER is being managed by Anders Wenhaug and Bjarne Grimstad.

SPLINTER's most important features are listed here: a data structure for storing and sorting data samples, a tensor product B-spline implementation for function approximation by smoothing or interpolating data samples, an implementation of ordinary least squares for polynomial fitting, and an implementation of radial basis function interpolation with various functional forms.

The code is shared under the permissive (weak copyleft) Mozilla Public License 2.0.

B.2. CENSO

A C++ framework for global optimization with spline constraints, described in (Grimstad and Sandnes, 2015; Grimstad et al., 2016). CENSO stands for Convex ENvelopes for Spline Optimization (Grimstad et al., 2015a).

The framework facilitates modelling of optimization problems, and in particular modelling of problems with piecewise constraint functions. Spline functions are represented and manipulated using the SPLINTER library. It includes a spatial branch-and-bound solver specialized for spline constraints. It also has an interface to the following solvers: IPOPT (Wächter and Biegler, 2006), BONMIN (Bonami et al., 2008), and GUROBI (Gurobi Optimization, Inc., 2014).

The code was recently shared with the public under the permissive (weak copyleft) Mozilla Public License 2.0.

B.3. Graph Problem Builder

The Graph Problem Builder is a C++ code for building production optimization problems formulated as in (Grimstad et al., 2016) using a graph data structure. The code requires the following, minimal information about the problem: network topology, fluid characteristics, operational and other custom constraints, and surrogate models (e.g. splines built using SPLINTER) for the nonlinear pressure and temperature drops in the system. Mass balances, momentum balances, and energy balances, as well as routing logic, is automatically included in the final optimization problem.

The maturity of this code is not at the same level as that of SPLINTER and CENSO. Nevertheless, it was successfully used to accelerate the modelling of the production optimization cases in (Grimstad et al., 2016). The code has not been released publicly and the time of writing.

Bibliography

Aamo, O. M., Eikrem, G. O., Siahaan, H. B., and Foss, B. A. (2005). Observer design for multiphase flow in vertical pipes with gas-lift - Theory and experiments. *Journal of Process Control*, 15(3):247–257.

ABB (2015). OptimizeIT Enhanced Oil Production Suite. `http://www.abb.com`. Accessed: 2015-05-30.

AbdulKarim, A., Al-Dhubaib, T., Elrafie, E., and Alamoudi, M. O. (2010). Overview of Saudi Aramco's intelligent field program. In *Intelligent Energy Conference and Exhibition*, Utrecht, The Netherlands.

Adjiman, C., Androulakis, I., and Floudas, C. (1997). Global optimization of MINLP problems in process synthesis and design. *Computers & Chemical Engineering*, 21(Supplement):S445–S450.

Adjiman, C. S., Androulakis, I. P., and Floudas, C. A. (1998a). A global optimization method, aBB, for general twice-differentiabe constrained NLPs - II. Implementation and computational results. *Computers & Chemical Engineering*, 22(9):1159–1179.

Adjiman, C. S., Androulakis, I. P., and Floudas, C. A. (2000). Global optimization of mixed-integer nonlinear problems. *AIChE Journal*, 46(9):1769–1797.

Adjiman, C. S., Dallwig, S., Floudas, C. A., and Neumaier, A. (1998b). A global optimization method, aBB, for general twice-differentiable constrained NLPs-I. Theoretical advances. *Computers and Chemical Engineering*, 22(9):1137–1158.

Ahmed, S. J., Recham, R., Nozari, A., Bughio, S., Schulze-riegert, R., and Ben, R. (2013). Uncertainty Quantification Workflow for Mature Oil Fields: Combining Experimental Design Techniques and Different Response Surface Models. In *SPE Middle East Oil and Gas Show and Conference*.

Ahmed, T. (2006). *Reservoir Engineering Handbook*. Gulf Professional Publishing, 3 edition.

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network flows: theory, algorithms and applications*, volume 1. Prentice Hall.

Akeze, A. C., Sikandar, A. S., and Laforce, T. (2009). Uncertainty Evaluation in Field Development and Export Planning. In *SPE EUROPEC/EAGE Annual Conference and Exhibition*, Amsterdam, The Netherlands.

Akl, S. G. and Toussaint, G. T. (1978). A fast convex hull algorithm. *Information Processing Letters*, 7(5):219–222.

Allen, S. (2011). 10 Years of Sub Arctic Subsea Projects - Stepping Stones for Arctic Development. In *Proceedings of OTC Arctic Technology Conference*.

Almklov, P. G., Østerlie, T., and Haavik, T. K. (2014). Situated with Infrastructures: Interactivity and Entanglement in Sensor Data Interpretation. *Journal of the Association for Information Systems*, 15(5):263–286.

Alperovits, E. and Shamir, U. (1977). Design of optimal water distribution systems. *Water Resources Research*, 13(6):885–900.

Aronofsky, J. S. (1983). Optimization Methods in Oil and Gas Development. *Society of Petroleum Engineers*.

Audet, C. and Dennis, J. E. (2006). Mesh Adaptive Direct Search Algorithms for Constrained Optimization. *SIAM Journal on Optimization*, 17(1):188–217.

Ausen, H. (2012). A Study in MINLP-class Optimization Problems for Simulated Petroleum Production. Master's thesis, Norwegian University of Science and Technology.

Aziz, K. and Settari, A. (1979). *Petroleum Reservoir Simulation*. Applied Science Publishers Ltd.

Baker Hughes (2015). Neuraflow Virtual Flow Meter. `http://www.bakerhughes.com`. Accessed: 2015-05-30.

Bakken, A., Grimstad, B., and Larsen, M. (2011). Life of field tool for optimal subsea design, condition monitoring, virtual metering, and flow assurance advice using a common field model. *Society for Underwater Technology*.

Barber, A., Shippen, M. E., Barua, S., Velázquez, J. C., Hernández, A. M. G., Klumpen, H. E., Moitra, S. K., Morales, F. L., Sauvé, S. R. B., Sagli, J. R., and Weber, M. (2007). Optimizing Production from Reservoir to Process Plant. *Oilfield Review*, 19(1):18–29.

Bauer, M. and Craig, I. K. (2008). Economic assessment of advanced process control–a survey and framework. *Journal of Process Control*, 18(1):2–18.

Beggs, D. and Brill, J. (1973). A Study of Two-Phase Flow in Inclined Pipes. *Journal of Petroleum Technology*, 25(5):607–617.

Bellout, M. C., Echeverría Ciaurri, D., Durlofsky, L. J., Foss, B., and Kleppe, J. (2012). Joint optimization of oil well placement and controls. *Computational Geosciences*, 16(4):1061–1079.

Belotti, P., Cafieri, S., Lee, J., and Liberti, L. (2010). Feasibility-based bounds tightening via fixed points. In *4th Annual International Conference on Combinatorial Optimization and Applications, COCOA 2010*, pages 65–76, Kailua-Kona, HI, USA. Springer Berlin Heidelberg.

Belotti, P., Lee, J., Liberti, L., Margot, F., and Wächter, A. (2009). Branching and bounds tighteningtechniques for non-convex MINLP. *Optimization Methods and Software*, 24(4-5):597–634.

Belsim (2015). VALI: Virtual flow metering and sensors monitoring solution. `http://www.belsim.com/upstream`. Accessed: 2015-05-30.

Bendiksen, K., Maines, D., Moe, R., and Nuland, S. (1991). The Dynamic Two-Fluid Model OLGA: Theory and Application. *SPE Production Engineering*, 6(2):171–180.

Bergamini, M. L., Aguirre, P., and Grossmann, I. (2005). Logic-based outer approximation for globally optimal synthesis of process networks. *Computers & Chemical Engineering*, 29(9):1914–1933.

Bernt, T. (2004). Subsea Facilities. In *Offshore Technology Conference*, Houston, Texas, U.S.A.

Besnard, D. and Albrechtsen, E. (2013). *Oil and Gas, Technology and Humans: Assessing the Human Factors of Technological Change*. Ashgate Publishing, Ltd.

Bieker, H. P., Slupphaug, O., and Johansen, T. A. (2007). Real-Time Production Optimization of Oil and Gas Production Systems: A Technology Survey. *SPE Production & Operations*, 22(4):382–391.

Bland, R. G., Goldfarb, D., and Todd, M. J. (1981). The Ellipsoid Method: A survey. *Operations Research*, 29(6):1039–1091.

Boehm, W. (1980). Inserting new knots into B-spline curves. *Computer-Aided Design*, 12(4):199–201.

Bohannon, J. (1970). A Linear Programming Model for Optimum Development of Multi-Reservoir Pipeline Systems. *Journal of Petroleum Technology*, 22(11):1429–1436.

Bonami, P., Biegler, L. T., Conn, A. R., Cornuéjols, G., Grossmann, I. E., Laird, C. D., Lee, J., Lodi, A., Margot, F., Sawaya, N., and Wächter, A. (2008). An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, 1 edition.

BP (2015a). ISIS: Integrated Surveillance Information System. `http://www.bp.com`. Accessed: 2015-05-30.

BP (2015b). MBOS: Model-Based Operational Support. `http://www.bp.com`. Accessed: 2015-05-30.

Burer, S. and Letchford, A. N. (2012). Non-convex mixed-integer nonlinear programming: A survey. *Surveys in Operations Research and Management Science*, 17(2):97–106.

Campos, M. C. M. M., Teixeira, A. F., Von Meien, O. F., Simões, S., Santos, W., Pimenta, A., and Stender, A. (2013). Advanced control systems for offshore production platforms. In *Offshore Technology Conference*, Rio de Janeiro, Brazil.

Carrizosa, E., Hansen, P., and Messine, F. (2004). Improving interval analysis bounds by translations. *Journal of Global Optimization*, 29(2):157–172.

Carroll, J. A. and Horne, R. N. (1992). Multivariate Optimization of Production System. *Journal of Petroleum Technology*, 44(7):782–789.

Castanier, B. and Rausand, M. (2006). Maintenance optimization for subsea oil pipelines. *International journal of pressure vessels and piping*, 83(4):236–243.

Codas, A. and Camponogara, E. (2012). Mixed-integer linear optimization for optimal lift-gas allocation with well-separator routing. *European Journal of Operational Research*, 217(1):222–231.

Codas, A., Campos, S., Camponogara, E., Gunnerud, V., and Sunjerga, S. (2012). Integrated production optimization of oil fields with pressure and routing constraints: The Urucu field. *Computers & Chemical Engineering*, 46:178–189.

Cohen, E., Lyche, T., and Riesenfeld, R. (1980). Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics. *Computer Graphics and Image Processing*, 14(2):87–111.

Cohen, E. and Schumaker, L. L. (1985). Rates of convergence of control polygons. *Computer Aided Geometric Design*, 2(1-3):229–235.

Conn, A. R., Scheinberg, K., and Vicente, L. N. (2009). *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics.

Cordeau, J. F., Toth, P., and Vigo, D. (2009). A Survey of Optimization Models for Train Routing and Scheduling. *Transportation Science*, 43(4):407–407.

Cox, M. G. (1972). The numerical evaluation of b-splines. *IMA Journal of Applied Mathematics*, 10(2):134–149.

Cozad, A., Sahinidis, N. V., and Miller, D. C. (2014). Learning surrogate models for simulation-based optimization. *AIChE Journal*, 60(6):2211–2227.

Crompton, J. and Gilman, H. (2011). The Future of Integrated Operations. *SPE Economics & Management*, 3(1):45–51.

Croxton, K. L., Gendron, B., and Magnati, T. L. (2003). A Comparison of Mixed-Integer Programming Models for Nonconvex Piecewise Linear Cost Minimization Problems. *Management Science*, 49(9):1268–1273.

Curry, H. B. and Schoenberg, I. J. (1966). On Pólya frequency functions IV: the fundamental spline functions and their limits. *Journal d'Analyse Mathématique*, 17(1):71–107.

Dantzig, G. B. (1987). Origins of the Simplex Method. In *A History of Scientific Computing*, page 13. ACM.

De Boor, C. (1972). On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50–62.

Denney, D. (2008). Autonomous Systems Improve Production. *Journal of Petroleum Technology*, 60(5):66–71.

Denney, D. (2012). Intelligent-Field Management: Monitoring and Optimization of the Greater Ekofisk Area. *Journal of Petroleum Technology*, 64(9):80–83.

Dias, R., Garcia, N. L., and Zambom, A. Z. (2010). A penalized nonparametric method for nonlinear constrained optimization based on noisy data. *Computational Optimization and Applications*, 45(3):521–541.

Digabel, S. L. and Wild, S. M. (2015). A Taxonomy of Constraints in Simulation-Based Optimization. *Available on Optimization Online*.

DNB Markets (2015). Economic outlook. `https://www.dnb.no`. Accessed: 2015-05-15.

Eie, R. (2015). Deepwater Arctic Subsea Separation & Storage Systems. In *OTC Arctic Technology Conference*.

Eilers, P. H. and Marx, B. D. (1996). Flexible Smoothing with B-splines and Penalties. *Statistical Science*, 11(2):89–102.

Ellingsen, G. and Monteiro, E. (2006). Seamless integration: standardisation across multiple local settings. *Computer Supported Cooperative Work (CSCW)*, 15(5-6):443–466.

Emerson (2015). Roxar Fieldwatch. `http://www.emersonprocess.com`. Accessed: 2015-05-30.

Floudas, C. A. and Pardalos, P. M. (1990). *A Collection of Test Problems for Constrained Global Optimization Algorithms*. Springer-Verlag.

Floudas, C. A., Pardalos, P. M., and Adjiman, C. S. (1999). *Handbook of Test Problems in Local and Global Optimization*, volume 33. Springer Science.

FMC Technologies (2015). FlowManager. `http://www.fmctechnologies.com`. Accessed: 2015-05-30.

Foot, J., Webster, M., Trueman, D., Yusti, G., and Grose, T. (2006). ISIS—A Real-Time Information Pipeline. In *SPE Intelligent Energy Conference and Exhibition*, Amsterdam, The Netherlands.

Foss, B. (2012). Process control in conventional oil and gas fields-Challenges and opportunities. *Control Engineering Practice*, 20(10):1058–1064.

Foss, B., Grimstad, B., and Gunnerud, V. (2015). Production optimization – facilitated by divide and conquer strategies. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Florianópolis, Brazil.

Fujii, H. and Horne, R. (1995). Multivariate Optimization of Networked Production Systems. *SPE Production & Facilities*, 10(3):165–171.

GAMS Development Corporation (2013). General Algebraic Modeling System (GAMS) Release 24.2.1. `http://www.gams.com/`. Washington, DC, USA.

Garloff, J., Jansson, C., and Smith, A. P. (2003). Lower bound functions for polynomials. *Journal of Computational and Applied Mathematics*, 157(1):207–225.

Garloff, J. and Smith, A. P. (2001). Investigation of a subdivision based algorithm for solving systems of polynomial equations. *Nonlinear Analysis: Theory, Methods & Applications*, 47(1):167–178.

Gatzke, E. P., Tolsma, J. E., and Barton, P. I. (2002). Construction of convex relaxations using automated code generation techniques. *Optimization and Engineering*, 3(3):305–326.

Gilman, H. and Nordtvedt, J.-e. (2014). Intelligent Energy: The Past, the Present, and the Future. *SPE Economics & Management*, 6(4):185–190.

Goh, K.-C., Overschee, P. V., and Briers, J. (2007). Production Surveillance and Optimization with Data Driven Models. In *International Petroleum Technology Conference*, Dubai, U.A.E.

Grimstad, B. (2012a). Global optimization. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2012b). Optimization of a simulated well cluster using surrogate models. Presentation. The 8th International Conference on Integrated Operations in the Petroleum Industry, Trondheim, Norway.

Grimstad, B. (2012c). Production optimization. Presentation. IBM, Oslo, Norway.

Grimstad, B. (2012d). Short-term production optimization. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2012e). Trustworthy production optimization. Stand. IO conference, Trondheim, Norway.

Grimstad, B. (2013a). BP/IOC joint production optimization project kick-off meeting. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2013b). BP/IOC joint project on production optimization. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2013c). Field optimization. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2013d). Global production optimization – algorithmic development. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2013e). Short-term production optimization. Presentation. BP and IO center workshop on short-term production optimization, Sunbury, UK.

Grimstad, B. (2014a). A nonlinear, adaptive observer for gas-lift wells operating under slowly varying reservoir pressure. Presentation. The 19th IFAC World Congress, Cape Town, South Africa.

Grimstad, B. (2014b). Production optimization using spline surrogate models. Presentation. IOC Technical Committee Meeting, Trondheim, Norway.

Grimstad, B. (2014c). Production optimization with surrogate models. Presentation. FMC Technologies, Asker, Norway.

Grimstad, B. (2014d). Production optimization with surrogate models. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B. (2014e). Short-term simulation-based production optimization. Presentation. FMC Technologies, Asker, Norway.

Grimstad, B. (2014f). Short-term simulation-based production optimization. Presentation. IOC Technical Committee Meeting, Trondheim, Norway.

Grimstad, B. (2014g). Short-term simulation-based production optimization. Presentation. BP, Imperial College, NTNU joint workshop at Imperial College, London, UK.

Grimstad, B. (2015a). Black-box optimization with spline surrogate models. Seminar and lecture. Santa Catarina Federal University, Florianópolis, Brazil.

Grimstad, B. (2015b). Daily production optimization. Presentation. IOC and BP workshop on daily production optimization, Rio de Janeiro, Brazil.

Grimstad, B. (2015c). Production optimization using surrogate models. Presentation. BP Exploration Operating Company Limited, Sunbury, UK.

Grimstad, B., Almklov, P., Foss, B., and Gunnerud, V. (2014). On why model-based production optimization is difficult in the upstream industry. Published as a report in the IO center.

Grimstad, B., Ausen, H., Lervik, V., Gunnerud, V., and Ljungquist, D. (2012). Optimization of a simulated well cluster using surrogate models. In *1st IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Trondheim, Norway.

Grimstad, B. et al. (2015a). CENSO: a framework for global optimization of nonconvex, possibly spline-constrained, MINLP problems. `https://github.com/bgrimstad/censo`. Accessed: 2015-05-16.

Grimstad, B. et al. (2015b). SPLINTER: a library for multivariate function approximation. `https://github.com/bgrimstad/splinter`. Accessed: 2015-05-16.

Grimstad, B. and Foss, B. (2014). A nonlinear, adaptive observer for gas-lift wells operating under slowly varying reservoir pressure. In *World Congress*, volume 19, pages 2824–2829, Cape Town, South Africa.

Grimstad, B., Foss, B., Heddle, R., and Woodman, M. (2016). Global optimization of multiphase flow networks using spline surrogate models. *Computers & Chemical Engineering*, 84:237 – 254.

Grimstad, B., Robertson, P., and Foss, B. (2015c). Virtual flow metering using B-spline surrogate models. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Florianópolis, Brazil.

Grimstad, B. and Sandnes, A. (2015). Global optimization with spline constraints: a new branch-and-bound method based on B-splines. *Journal of Global Optimization*.

Grossmann, I. E. (2002). Review of Nonlinear Mixed-Integer and Disjunctive Programming Techniques. *Optimization and Engineering*, 3(3):227–252.

Grossmann, I. E. and Lee, S. (2003). Generalized convex disjunctive programming: Nonlinear convex hull relaxation. *Computational Optimization and Applications*, 26(1):83–100.

Gruehagen, H. and Lim, D. (2009). Subsea Separation and Boosting – An Overview of Ongoing Projects. In *SPE Asia Pacific Oil and Gas Conference & Exhibition*, Jakarta, Indonesia.

Grynning, A., Larsen, S. V., and Skaale, I. (2009). Tyrihans Raw Seawater Injection. In *Offshore Technology Conference*, Houston, Texas, USA.

Guennebaud, G., Jacob, B., et al. (2010). Eigen v3. `http://eigen.tuxfamily.org`. Accessed: 2012-01-01.

Gunnerud, V. and Foss, B. (2010). Oil production optimization-A piecewise linear model, solved with two decomposition strategies. *Computers and Chemical Engineering*, 34(11):1803–1812.

Guo, B., Lyons, W. C., and Ali Ghalambor (2007). *Petroleum Production Engineering: A Computer-Assisted Approach*. Gulf Professional Publishing, Elsevier.

Gurobi Optimization, Inc. (2014). Gurobi Optimizer Reference Manual. `http://www.gurobi.com`. Accessed: 2014-10-01.

Güyagüler, B. and Horne, R. (2004). Uncertainty Assessment of Well-Placement Optimization. *SPE Reservoir Evaluation & Engineering*, 7(1).

Haavik, T. K. (2013). *New tools, old tasks: Safety implications of new technologies and work processes for integrated operations in the petroleum industry*. Ashgate Publishing, Ltd.

Håheim, S. and Gaillard, X. (2009). A Simplified Subsea Separation and Pumping System. In *SPE Annual Technical Conference and Exhibition*, New Orleans, Louisiana, USA.

Hansen, E. (2003). *Global Optimization Using Interval Analysis*. Marcel Dekker Inc., 2 edition.

Hanssen, K., Foss, B., and Teixeira, A. F. (2015). Production optimization under uncertainty with constraint handling. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, Florianópolis, Brazil.

Hawkins, D. M. (2004). The Problem of Overfitting. *Journal of Chemical Information and Computer Sciences*, 44(1):1–12.

Heinkenschloss, M. and Vicente, L. N. (2002). Analysis of Inexact Trust-Region SQP Algorithms. *SIAM Journal on Optimization*, 12(2):283–302.

Henrion, D., Lasserre, J.-b., and Löfberg, J. (2009). Gloptipoly 3: moments, optimization and semidefinite programming. *Optimization Methods and Software*, 24(4-5):761–779.

Holmås, K. and Løvli, A. (2011). FlowManager Dynamic: A multiphase flow simulator for online surveillance, optimization and prediction of subsea oil and gas production. In *15th International Conference on Multiphase Production Technology*, Cannes, France.

Holmås, K., Lunde, G. G., Setyadi, G., Angelo, P., and Rudrum, G. (2013). Ormen Lange Flow Assurance System (FAS) - Online Flow Assurance Monitoring and Advice. In *Offshore Technology Conference*, Rio de Janeiro, Brazil.

Hooker, J. (2000). *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. John Wiley & Sons, Inc.

Horst, R. and Tuy, H. (1996). *Global Optimization: Deterministic Approaches*. Springer, 3 edition.

IO Center (2014). `http://iocenter.no`. Trondheim, Norway.

Jahanshahi, E. and Grimstad, B. (2014). Energy balance of three-phase mixing processes with standard conditions information. Technical report, IO center and Norwegian University of Science and Technology.

Jahanshahi, E., Grimstad, B., and Foss, B. (2015). Spline fluid models for optimization. Submitted for publication.

Jansen, J. D. (2011). Adjoint-based optimization of multi-phase flow through porous media - A review. *Computers and Fluids*, 46(1):40–51.

Jansen, J. D., Douma, S. D., Brouwer, D. R., den Hof, P. M. J. V., Bosgra, O. H., and Heemink, A. W. (2009). Closed Loop Reservoir Management. In *SPE Reservoir Simulation Symposium*, Woodlands, Texas, USA.

Jones, D. R. (2001). A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization*, 21(4):345–383.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

JPT (2008). Holistic production optimization achieved one workflow at a time. *Journal of Petroleum Technology*, 60(4):30–34.

Karmarkar, N. (1984). A New Polynomial-Time Algorithm for Linear Programming. In *STOC '84 Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311.

KBC (2015). FEESA Maximus Thermal Hydraulic Integrated Production Modelling Software. `http://www.kbcat.com/feesa-maximus`. Accessed: 2015-08-07.

Keha, A. B., de Farias, I. R., and Nemhauser, G. L. (2006). A Branch-and-Cut Algorithm Without Binary Variables for Nonconvex Piecewise Linear Optimization. *Operations Research*, 54(5):847–858.

Kesavan, P. and Barton, P. I. (2000). Generalized branch-and-cut framework for mixed-integer nonlinear optimization problems. *Computers and Chemical Engineering*, 24(2-7):1361–1366.

Knudsen, B. R., Grossmann, I. E., Foss, B., and Conn, A. R. (2014). Lagrangian relaxation based decomposition for well scheduling in shale-gas systems. *Computers and Chemical Engineering*, 63:234–249.

Kolda, T. G., Lewis, R. M., and Torczon, V. (2003). Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods. *SIAM Review*, 45(3):385–482.

Kolodziej, S., Castro, P. M., and Grossmann, I. E. (2013). Global optimization of bilinear programs with a multiparametric disaggregation technique. *Journal of Global Optimization*, 57(4):1039–1063.

Kongsberg Oil & Gas Technologies (2015). LedaFlow. `http://www.kongsberg.com/ledaflow`. Accessed: 2015-05-30.

Kosmidis, V. D., Perkins, J. D., and Pistikopoulos, E. N. (2004). Optimization of Well Oil Rate Allocations in Petroleum Fields. *Industrial & Engineering Chemistry Research*, 43(14):3513–3527.

Kosmidis, V. D., Perkins, J. D., and Pistikopoulos, E. N. (2005). A mixed integer optimization formulation for the well scheduling problem on petroleum fields. *Computers & Chemical Engineering*, 29(7):1523–1541.

Koziel, S. and Michalewicz, Z. (1999). Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary computation*, 7(1):19–44.

Kragas, T. K., Williams, B. A., and Myers, G. A. (2001). The optic oil field: deployment and application of permanent in-well fiber optic sensing systems for production and reservoir monitoring. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers.

Lasserre, J. B. (2001). Global Optimization with Polynomials and the Problem of Moments. *SIAM Journal on Optimization*, 11(3):796–817.

Lebbah, Y., Michel, C., and Rueher, M. (2007). An efficient and safe framework for solving optimization problems. *Journal of Computational and Applied Mathematics*, 199(2):372–377.

Lee, A. S. and Aronofsky, J. S. (1958). A Linear Programming Model for Scheduling Crude Oil Production. *Journal of Petroleum Technology*, 10(7):51–54.

Lerlertpakdee, P., Jafarpour, B., and Gildin, E. (2014). Efficient Production Optimization With Flow-Network Models. *SPE Journal*, 19(06):1083–1095.

Li, H.-L. and Chang, C.-T. (1998). An approximate approach of global optimization for polynomial programming problems. *European Journal of Operational Research*, 107(3):625–632.

Liberti, L. and Pantelides, C. C. (2003). Convex envelopes of monomials of odd degree. *Journal of Global Optimization*, 25(2):157–168.

Liberti, L. and Pantelides, C. C. (2006). An exact reformulation algorithm for large nonconvex NLPs involving bilinear terms. *Journal of Global Optimization*, 36(2):161–189.

Lima, F., Storstenvik, A., and Nyborg, K. (2011). Subsea Compression : A Game Changer. In *Offshore Technology Conference*, Rio de Janeiro, Brazil.

Lin, Y. and Schrage, L. (2009). The global solver in the LINDO API. *Optimization Methods and Software*, 24(4-5):657–668.

Liu, D. C. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528.

Liuzzi, G., Lucidi, S., and Rinaldi, F. (2015). Derivative-Free Methods for Mixed-Integer Constrained Optimization Problems. *Journal of Optimization Theory and Applications*, 164(3):933–965.

Lizon, C., D'Ambrosio, C., Liberti, L., Le Ravalec, M., and Sinoquet, D. (2014). A mixed-integer nonlinear optimization approach for well placement and geometry. In *ECMOR XIV-14th European conference on the mathematics of oil recovery*.

Locatelli, M. and Schoen, F. (2014). On convex envelopes for bivariate functions over polytopes. *Mathematical Programming*, 144(1-2):65–91.

Luathep, P., Sumalee, A., Lam, W. H., Li, Z.-C., and Lo, H. K. (2011). Global optimization method for mixed transportation network design problem: A mixed-integer linear programming approach. *Transportation Research Part B: Methodological*, 45(5):808–827.

Lunde, G. G., Vannes, K., Mcclimans, O. T., Burns, C., and Wittmeyer, K. (2009). Advanced Flow Assurance System for the Ormen Lange Subsea Gas Development. In *Offshore Technology Conference*, Houston, Texas, U.S.A. Offshore Technology Conference.

Lyche, T., Cohen, E., and Mø rken, K. (1985). Knot line refinement algorithms for tensor product B-spline surfaces. *Computer Aided Geometric Design*, 2(1-3):133–139.

Mandl, C. E. (1980). Evaluation and Optimization of Urban Public Transportation Networks. *European Journal of Operational Research*, 5(6):396–404.

Martin, A., Möller, M., and Moritz, S. (2006). Mixed integer models for the stationary case of gas network optimization. *Mathematical Programming*, 105(2-3):563–582.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part I - Convex underestimating problems. *Mathematical Programming*, 10(1):147–175.

McDonald, D. B., Grantham, W. J., Tabor, W. L., and Murphy, M. J. (2007). Global and local optimization using radial basis function response surface models. *Applied Mathematical Modelling*, 31(10):2095–2110.

McKie, C. J. N., Rojas, E. a., Quintero, N. M., Chacon Fonseca, J. R., and Perozo, N. J. (2001). Economic Benefits From Automated Optimization of High Pressure Gas Usage in an Oil Production System. In *SPE Production and Operations Symposium*, Oklahoma City, Oklahoma.

Meeraus, A. (2013). GLOBALLib. `http://www.gamsworld.org/global/globallib.htm`. Accessed: 2013-10-01.

Melbø, H., Morud, S. A., Bringedal, B., van der Geest, R., and Stenersen, K. (2003). Software That Enables Flow Metering of Well Rates With Long Tiebacks and With Limited or Inaccurate Instrumentation. In *Offshore Technology Conference*, Houston, Texas, U.S.A.

Messine, F. (2004). Deterministic global optimization using interval constraint propagation techniques. *RAIRO - Operations Research*, 38(4):277–293.

Meyer, C. A. and Floudas, C. A. (2005a). Convex envelopes for edge-concave functions. *Mathematical Programming*, 103(2):207–224.

Meyer, C. A. and Floudas, C. A. (2005b). Convex underestimation of twice continuously differentiable functions by piecewise quadratic perturbation: Spline aBB underestimators. *Journal of Global Optimization*, 32(2):221–258.

Meyer, C. A., Floudas, C. A., and Neumaier, A. (2002). Global Optimization with Nonfactorable Constraints. *Industrial and Engineering Chemistry Reseach*, 41(25):6413–6424.

Misener, R. and Floudas, C. A. (2010). Piecewise-linear approximations of multidimensional functions. *Journal of Optimization Theory and Applications*, 145(1):120–147.

Misener, R., Gounaris, C. E., and Floudas, C. A. (2009). Global optimization of gas lifting operations: A comparative study of piecewise linear formulations. *Industrial and Engineering Chemistry Research*, 48(13):6098–6104.

Misener, R., Thompson, J. P., and Floudas, C. A. (2011). APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. *Computers and Chemical Engineering*, 35(5):876–892.

Mjaavatten, A., Aasheim, R., Saelid, S., and Groenning, O. (2008). A Model for Gas Coning and Rate-Dependent Gas/Oil Ratio in an Oil-Rim Reservoir. *SPE Reservoir Evaluation & Engineering*, 11(5).

Mochizuki, S., Saputelli, L., Kabir, C. S., Cramer, R., Lochmann, M., Reese, R., Harms, L., Sisk, C., Hite, J., and Escorcia, A. (2006). Real-Time Optimization: Classification and Assessment. *SPE Production & Operations*, 21(4):455–466.

Narasimhan, S. and Jordache, C. (1999). *Data Reconciliation and Gross Error Detection*. Gulf Professional Publishing.

Natali, J. M. and Pinto, J. M. (2009). Piecewise polynomial interpolations and approximations of one-dimensional functions through mixed integer linear programming. *Optimization Methods and Software*, 24(4-5):783–803.

Nataraj, P. S. V. and Arounassalame, M. (2007). A new subdivision algorithm for the Bernstein polynomial approach to global optimization. *International Journal of Automation and Computing*, 4(4):342–352.

Nataraj, P. S. V. and Arounassalame, M. (2011). Constrained global optimization of multivariate polynomials using Bernstein branch and prune algorithm. *Journal of Global Optimization*, 49:185–212.

Nedregaard, T. (2003). *Produksjon av olje og gass*. Vett og Viten AS, 2 edition.

Nelder, B. J. A. and Mead, R. (1965). A simplex method for function minimization. *The computer journal*, 7(4):308–313.

Nemhauser, G. L. and Wolsey, L. A. (1999). *Integer and Combinatorial Optimization*. John Wiley & Sons, Inc., 1 edition.

Nesterov, Y. and Nemirovskii, A. (1994). *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM.

Nicotra, G., Godi, A., Cominelli, A., and Christie, M. (2005). Production Data and Uncertainty Quantification: A Real Case Study. In *Proceedings of SPE Reservoir Simulation Symposium*, Houston, Texas, U.S.A.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer.

Norsk Oljemuseum (2015). `http://www.norskolje.museum.no`. Accessed: 2015-05-15.

Norwegian Oil Industry Association (2005). Integrated work processes: future work processes on the Norwegian continental shelf.

Norwegian Petroleum Directorate (2015). `http://npd.no`. Accessed: 2015-05-15.

Oberwinkler, C., Mayfield, D., Dixon, D., and Holland, J. (2006). Automated Production Surveillance. *SPE Projects Facilities & Construction*, 1(2):1–8.

Oberwinkler, C. and Stundner, M. (2005). From Real-Time Data to Production Optimization. *SPE Production & Facilities*, 20(03):229–239.

Orlowski, R. T. C., Euphemio, M. L. L., Castro, F. G., Andrade, C. A., Guedes, F. M. F., Silva, L. C. T., Pestana, R. G., de Cerqueira, G. C., Lourenco, I., Pivari, A., Witka, A., Folhadella, H., Pacheco, L., Kronemberger, S., and Vilela, J. (2012). Marlim 3 Phase Subsea Separation System – Challenges and Solutions for the Subsea Separation Station to Cope with Process Requirements. In *Offshore Technology Conference*, Houston, Texas, U.S.A.

Østerlie, T., Almklov, P. G., and Hepsø, V. (2012). Dual materiality and knowing in petroleum production. *Information and Organization*, 22(2):85–105.

Overschee, P. V. and Moor, B. D. (1996). *Subspace identification for linear systems: Theory - Implementation - Applications*. Kluwer Academic Publishers.

Ozdogan, U. and Horne, R. (2006). Optimization of Well Placement Under Time-Dependent Uncertainty. *SPE Reservoir Evaluation & Engineering*, 9(2):135–145.

Özyurt, D. B. and Pike, R. W. (2004). Theory and practice of simultaneous data reconciliation and gross error detection for chemical processes. *Computers & Chemical Engineering*, 28(3):381–402.

Padberg, M. and Rinaldi, G. (1991). A Branch-and-Cut Algorithm for the Resolution of Large-Scale Symmetric Traveling Salesman Problems. *SIAM Review*, 33(1):60–100.

Park, S. (2012). Approximate branch-and-bound global optimization using B-spline hypervolumes. *Advances in Engineering Software*, 45(1):11–20.

Pereira, F. H., Grassi, F., and Nabeta, S. I. (2015). A Surrogate-Based Two-Level Genetic Algorithm Optimization Through Wavelet Transform. *IEEE Trans. On Magnetics*, 51(3).

Petroleum Experts Ltd. (2014). Integrated Production Modelling software (IPM). `http://www.petex.com/products/`. Accessed: 2014-10-29.

Piegl, L. A. and Tiller, W. (1997). *The NURBS Book*. Monographs in Visual Communications. Springer Berlin Heidelberg, 2 edition.

Pina, A. C. D. and Jacob, B. P. (2013). Applying Wavelet Neural Networks As Surrogate Models in the Design of Moored Floating Systems for Offshore. In *22nd International Congress on Mechanical Engineering (COBEM 2013)*, pages 7034–7045, Ribeirão Preto, SP, Brazil.

Pinter, J. D. (1997). LGO—a program system for continuous and Lipschitz global optimization. In *Developments in Global Optimization*, pages 183–197. Springer.

Pochet, Y. and Wolsey, L. A. (2006). *Production planning by mixed integer programming*. Springer-Verlag New York, 1 edition.

Poulisse, H., van Overschee, P., Briers, J., Moncur, C., and Goh, K.-C. (2006). Continuous Well Production Flow Monitoring and Surveillance. In *Intelligent Energy Conference and Exhibition*, Amsterdam, The Netherlands.

Prautzsch, H. and Kobbelt, L. (1994). Convergence of subdivision and degree elevation. *Advances in Computational Mathematics*, 2:143–154.

Qin, S. J. and Badgwell, T. A. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764.

Queipo, N. V., Zerpa, L. E., Goicochea, J. V., Verde, A. J., Pintos, S. a., and Zambrano, A. (2003). A model for the integrated optimization of oil production systems. *Engineering with Computers*, 19(2-3):130–141.

Raghunathan, A. U. (2013). Global Optimization of Nonlinear Network Design. *SIAM Journal on Optimization*, 23(1):268–295.

Rahmawati, S. D., Whitson, C. H., Foss, B., and Kuntadi, A. (2010). Multi-Field Asset Integrated Optimization Benchmark. In *SPE EUROPEC/EAGE Annual Conference and Exhibition*, Barcelona, Spain.

Rahmawati, S. D., Whitson, C. H., Foss, B., and Kuntadi, A. (2012). Integrated field operation and optimization. *Journal of Petroleum Science and Engineering*, 81:161–170.

Rashid, K., Bailey, W., and Couët, B. (2012). A survey of methods for gas-lift optimization. *Modelling and Simulation in Engineering*, 2012.

Regis, R. G. (2014). Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. *Engineering Optimization*, 46(2):218–243.

Regis, R. G. and Shoemaker, C. A. (2005). Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31(1):153–171.

Reif, U. (2000). Best bounds on the approximation of polynomials and splines by their control structure. *Computer Aided Geometric Design*, 17(6):579–589.

Richardson, S., Keron, H., Jatmiko, W., Booker, J., Osugo, L., and Donachie, J. (2004). Real-time data management boosts production, reservoir knowledge. *World Oil*, 225(2).

Rios, L. M. and Nikolaos V. Sahinidis (2013). Derivative-free optimization: A review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293.

Roald, P.-K., Aas, T., Smith, G. G., Soosaipillai, C., Alfstad, D. E., and Bressand, J.-Y. (2013). Condition Performance Monitoring for Subsea : Experience and Value documentation from the Gjøa Field. In *Offshore Technology Conference*, Houston, Texas, U.S.A.

Robertson, P. M. (2014). Dynamic estimation for controlling a subsea production system. Master's thesis, Norwegian University of Science and Technology.

Rosendahl, T. and Hepsø, V. (2013). *Integrated Operations in the Oil and Gas Industry: Sustainability and Capability Development*. IGI Global.

Ryoo, H. S. and Sahinidis, N. V. (1996). A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8(2):107–138.

Sahinidis, N. V. (2003). Global Optimization and Constraint Satisfaction. In *First International Workshop on Global Constraint Optimization and Constraint Satisfaction, COCOS 2002*, pages 1–16, Valbonne-Sophia Antipolis, France.

Sandnes, A. (2013). Solving a Network Flow Decision Problem with Sampled Nonlinearities. Master's thesis, Norwegian University of Science and Technology.

Sasena, M. J. (2002). *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations*. PhD thesis, University of Michigan.

Sasena, M. J., Papalambros, P., and Goovaerts, P. (2002). Exploration of Metamodeling Sampling Criteria for Constrained Global Optimization. *Engineering Optimization*, 34(3):263–278.

Schlumberger (2014a). ECLIPSE Industry Reference Reservoir Simulator. `http://www.software.slb.com/products/foundation/Pages/eclipse.aspx`. Accessed: 2014-10-29.

Schlumberger (2014b). OLGA Dynamic Multiphase Flow Simulator. `http://www.software.slb.com/products/foundation/Pages/olga.aspx`. Accessed: 2014-10-29.

Schlumberger (2014c). PIPESIM Steady-State Multiphase Flow Simulator. `http://www.software.slb.com/products/foundation/Pages/pipesim.aspx`. Accessed: 2014-10-29.

Schönberg, I. J. (1946). Contributions to the problem of approximation of equidistant data by analytic functions. *Quart. Appl. Math*, 4(2):45–99.

Schüller, R. B., Solbakken, T., and Selmer-Olsen, S. (2003). Evaluation of Multiphase Flow Rate Models for Chokes Under Subcritical Oil / Gas / Water Flow Conditions. *SPE Production & Facilities*, 18(3):170–181.

Schumaker, L. L. (2007). *Spline Functions: Basic Theory*. Cambridge University Press, 3 edition.

Sengul, M. A. and Bekkousha, M. A. B. (2002). Applied Production Optimization: I-Field. In *SPE Annual Technical Conference and Exhibition*, San Antonio, Texas, USA.

Sharma, S. (2013). Mixed-Integer Nonlinear Programming Heuristics Applied to a Shale Gas Production Optimization Problem. Master's thesis, Norwegian University of Science and Technology.

Sharma, S., Knudsen, B., and Grimstad, B. (2015). Towards an objective feasibility pump for convex MINLPs. *Computational Optimization and Applications*.

Shell (2015). FieldWare Production Universe. `http://www.shell.com`. Accessed: 2015-05-30.

Sherali, H. D. and Tuncbilek, C. H. (1995). A reformulation-convexification approach for solving nonconvex quadratic programming problems. *Journal of Global Optimization*, 7(1):1–31.

Silva, T. L. and Camponogara, E. (2014). A computational analysis of multidimensional piecewise-linear models with applications to oil production optimization. *European Journal of Operational Research*, 232(3):630–642.

Skibeli, H. (2015). Automatic detection of poorly calibrated models in state estimation applied to oil and gas production systems. Master's thesis, Norwegian University of Science and Technology.

Skogestad, S. and Postlethwaite, I. (2005). *Multivariable Feedback Control*. John Wiley & Sons, Ltd, 2 edition.

Smith, A. P. (2009). Fast construction of constant bound functions for sparse polynomials. *Journal of Global Optimization*, 43(2-3):445–458.

Smith, E. M. and Pantelides, C. C. (1997). Global optimisation of nonconvex MINLPs. *Computers & Chemical Engineering*, 21(Supplement):S791–S796.

Smith, E. M. B. and Pantelides, C. C. (1999). A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex MINLPs. *Computers & Chemical Engineering*, 23(4-5):457–478.

Solheim, A., Prichard, M., and Newcombe, R. (1989). The Tommeliten Subsea Project. In *Offshore Europe*, Aberdeen, United Kingdom.

Standards Norway (2015). `http://standard.no`. Accessed: 2015-05-16.

Statoil (2015). `http://statoil.no`. Accessed: 2015-05-15.

Stenhouse, B. (2006). Learnings on Sustainable Model-Based Optimisation — The Valhall Optimiser Field Trial. In *Intelligent Energy Conference and Exhibition*, Amsterdam, The Netherlands.

Stenhouse, B. (2008). Modelling and Optimisation in BP Exploration and Production: Case Studies and Learnings. In *Proceedings of Intelligent Energy Conference and Exhibition*.

Stenhouse, B., Woodman, M., and Griffiths, P. (2010). Model Based Operational Support - Adding Assurance to Operational Decision Making. In *SPE Intelligent Energy*, Utrecht, The Netherlands.

Sturm, J. F. (1999). Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653.

Sullivan, J. (1982). A Computer Model for Planning the Development of an Offshore Gas Field. *Journal of Petroleum Technology*, 34(7).

Tarhan, B., Grossmann, I. E., and Goel, V. (2009). Stochastic Programming Approach for the Planning of Offshore Oil or Gas Field Infrastructure under Decision-Dependent Uncertainty. *Industrial & Engineering Chemistry Research*, 48(6):3078–3097.

Tawarmalani, M. and Sahinidis, N. V. (2002). *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Springer Science.

Tawarmalani, M. and Sahinidis, N. V. (2004). Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99(3):563–591.

Tawarmalani, M. and Sahinidis, N. V. (2005). A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2):225–249.

Teixeira, A. F., Barreto, F. P., Rosa, V. R., Arraes, F. F., and Stender, A. S. (2013). Model Based Production Optimization Applied to Offshore Fields. In *Offshore Technology Conference*, Rio de Janeiro, Brazil.

Thakur, G. (1996). What Is Reservoir Management? *Journal of Petroleum Technology*, 48(6):520–525.

Vaidyanathan, R. and El-Halwagi, M. (1996). Global optimization of nonconvex MINLP's by interval analysis. In *Global Optimization in Engineering Design*, pages 175–193. Springer US.

van der Linden, R. J. P. (2014). Optimizing production from mature fields. IO Conference, Trondheim, Norway.

van der Linden, R. J. P. and Busking, T. E. (2013). Real-Time Optimization of a maturing North Sea gas asset with Production Constraints. In *SPE Middle East Intelligent Energy Conference and Exhibition*, Dubai, U.A.E. Society of Petroleum Engineers.

Venkataraman, P. (2009). *Applied optimization with MATLAB programming*. John Wiley & Sons.

Vielma, J. P., Ahmed, S., and Nemhauser, G. (2010). Mixed-Integer Models for Nonseparable Piecewise-Linear Optimization: Unifying Framework and Extensions. *Operations Research*, 58(2):303–315.

Vielma, J. P. and Nemhauser, G. L. (2011). Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming*, 128(1-2):49–72.

Vu, V. K., Fantoft, R., Shaw, C. K., and Gruehagen, H. (2009). Comparison of subsea separation systems. In *Offshore Technology Conference*, Houston, Texas, U.S.A.

Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57.

Waki, H., Kim, S., Kojima, M., and Muramatsu, M. (2006). Sums of Squares and Semidefinite Program Relaxations for Polynomial Optimization Problems with Structured Sparsity. *SIAM Journal on Optimization*, 17(1):218–242.

Wang, P. (2003). *Development and applications of production optimization techniques for petroleum fields*. PhD thesis, Stanford University.

Wang, P., Litvak, M., and Aziz, K. (2002). Optimization of Production Operations in Petroleum Fields. In *SPE Annual Technical Conference and Exhibition*, San Antonio, Texas, USA.

Weatherford (2015). Weatherford Field Office. `http://www.weatherford.com/products-services/production/software`. Accessed: 2015-05-30.

Westerlund, T., Pettersson, F., and Grossmann, I. (1994). Optimization of pump configurations as a MINLP problem. *Computers & Chemical Engineering*, 18(9):845–858.

Wild, S. M. and Shoemaker, C. (2011). Global convergence of radial basis function trust region derivative-free algorithms. *SIAM Journal on Optimization*, 21:761–781.

Willersrud, A., Imsland, L., Hauger, S. O., and Kittilsen, P. (2013). Short-term production optimization of offshore oil and gas production using nonlinear model predictive control. *Journal of Process Control*, 23(2):215–223.

Witting, F. (2006). Snohvit and Beyond - Setting New Standards For Subsea to Shore Developments. In *SPE Russian Oil and Gas Technical Conference and Exhibition*, Moscow, Russia.

Wood Group (2015). Virtuoso. `http://www.multiphase.com/software-solutions`. Accessed: 2015-08-07.

Zamora, J. M. and Grossmann, I. E. (1999). A branch and contract algorithm for problems with concave univariate, bilinear and linear fractional terms. *Journal of Global Optimization*, 14(3):217–249.

Zenith, F., Foss, B., Tjønnås, J., and Hasan, A. (2015). Well Testing by Sinusoidal Stimulation. *SPE Reservoir Evaluation & Engineering*.