Doctoral theses at NTNU, 2016:128

Lars Hellemo

# Managing Uncertainty in Design and Operation of Natural Gas Infrastructure

NTNU
Norwegian University of Science and Technology
Faculty of Social Sciences and Technology Management
Department of Industrial Economics and Technology Management

Doctoral Thesis

ISBN 978-82-326-1594-0 (printed version)
ISBN 978-82-326-1595-7 (electronic version)
ISSN 1503-8181

Lars Hellemo
Doctoral theses at NTNU, 2016:128

NTNU
Norwegian University of Science and Technology

NTNU
Norwegian University of Science and Technology
NTNU

Lars Hellemo

# Managing Uncertainty in Design and Operation of Natural Gas Infrastructure

Thesis for the degree of Philosophiae Doctor

Trondheim, June 2016

Norwegian University of Science and Technology
Faculty of Social Sciences and Technology Management
Department of Industrial Economics and Technology Management

NTNU
Norwegian University of Science and Technology

**NTNU**
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Social Sciences and Technology Management
Department of Industrial Economics and Technology Management



ISBN 978-82-326-1594-0 (printed version)
ISBN 978-82-326-1595-7 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2016:128

Printed by Skipnes Kommunikasjon as

# Acknowledgements

This thesis is the result of my work for the degree of Philosophiae Doctor at the Department of Industrial Economics and Technology Management at the Norwegian University of Science and Technology (NTNU).

First, I would like to thank my supervisor Professor Asgeir Tomasgard for giving me the opportunity to do a PhD at NTNU and for giving me the freedom to explore new subjects and take risks, thanks for many interesting discussions, some times bordering on the philosophical.

I would like to thank professor Paul I. Barton for hosting me at his Process Systems Engineering (PSE) Lab at MIT for seven months in 2009, giving me an opportunity to work on algorithms for global optimization. Staying there was a unique experience and I had the pleasure of getting to know several labmates, in particular Xiang Li, Patricio Ramirez, Matt Stuber and Thomas A. Adams II. Thanks to Ana Cardoso and Francisco Pereira for taking us out of Boston for outings, for inviting us to the Azores and for being great friends.

Special thanks to professor Stein-Erik Fleten for encouraging me to take on a PhD, thanks to Arne Stokka and Frode Rømo for letting me take time off from SINTEF and for letting me persue my academic interests, without cutting the chord completely. Thanks to my colleagues at SINTEF for maintaining a connection with the real world, and for silly Friday lunch discussions. Thanks to Marte Fodstad, Kjetil Midthun and Adrian Werner for many interesting and fun discussions, it has been a pleasure to work and write with you all.

Finally, I am grateful to my family for being there for me when I needed you. Thanks to Osvald for coming with me to the US, and for always asking why, like a true scientist. Thanks to Agnes for cheering me up with your bright smile and making sure daddy also gets a taste of the food. Thank you Line for not letting me get stuck in a local minimum and for going around the world with me. I look forward to taking on new projects with you all!

Lars Hellemo

Trondheim, August, 2013

# Contents

# Chapter 1

# Introduction

This thesis consists of papers describing several optimization models for decision support primarily motivated by the needs of natural gas producers. Through developing models for investments and operations in production, transport and processing of natural gas, we seek to improve understanding of the issues related to the design of these infrastructures on a system level, taking both long term uncertainty and short term variability into account. We also extend the literature on stochastic programming with decision-dependent uncertainty. Decision-dependent uncertainty is relevant in many problems related to exploitation of natural resources.

The first part of this thesis consists of this introduction, which provides context and motivation for the works presented, and demonstrates the contributions from each paper and how they relate to existing work in literature. The introduction ends with a presentation of each paper and specifies my contribution to each paper.

The second part of this thesis consists of the six papers. The first paper, *Optimizing the Norwegian Natural Gas Production and Transport*, introduces a flow optimization model for natural gas transport that has been developed over several years in close cooperation with industry partners. It demonstrates how useful mathematical programming can be as a decision support tool, and also the importance of close dialogue with analysts.

The second paper, *Natural Gas Infrastructure Design with an Operational Perspective* introduces a deterministic investment model for capacity expansion of natural gas infrastructure: development of new fields, investments in transport capacity and processing facilities. The model maximizes net present values from investments, operational costs and revenue from selling gas at market nodes.

Paper three, *Multi-Stage Stochastic Programming for Natural Gas Infrastructure Design with a Production Perspective* presents an extension of paper two to a multi-stage stochastic programming model. Investments in infrastructure for petroleum exploitation involve high capital expenses, and uncertainty in e.g. future prices or demand will have a large impact on the profitability of different prospects. To incorporate both operational variability and long term uncertainty, a novel multi-horizon scenario tree is introduced.

In paper four, *Discretizations of Natural Gas Pooling Problems*, we investigate different ways of discretizing the pooling problem that arises from multi-commodity flow optimization problems with pooling and quality constraints and introduce a new discretization scheme that shows great promise.

Paper five, *A Generalized Global Optimization Formulation of the Pooling Problem with Processing Facilities*, compares different global optimization formulations of the pooling problem, and introduces a more general formulation that also facilitates the extension to include a simplified model of processing facilities. We also compare performance of the continuous global optimization formulation and discretized versions of this formulation.

In the final paper, *Stochastic Programming with Decision-Dependent Probabilities*, we present a taxonomy of stochastic programming problems with decision-dependent uncertainty, and introduce several novel models where the scenario probabilities are manipulated by decision variables, either through distortions or by setting the probability distribution parameters directly.

## 1.1 The Natural Gas Industry

The search for petroleum on the Norwegian Continental Shelf (NCS) began in the 1960s, and production started in the early 1970s. Ever since, the petroleum industry has been an important industry in Norway, and it is the largest sector by far in terms of value creation.

In the early years, oil was the dominating product from the NCS, but natural gas has gained importance over the years, and the industry has been subject to great change following the introduction of the European Union Gas Directives. Natural gas export from Norway accounts for almost 20% of the natural gas consumption in Europe.

The challenges of natural gas production change over time as early developments age. New discoveries often have different characteristics from existing fields, and may introduce quality challenges. New areas are opened for exploration that may need different solutions from fields already in production, and may or may not be connected with existing infrastructure. Markets can also change rapidly as was seen when large volumes from shale gas production in the United Stated changed the dynamics of world energy markets.

From Norway, the exported volume of oil is expected to go down, while natural gas production is expected to increase further in the decade to come. The Norwegian Ministry of Petroleum and Energy and The Norwegian Petroleum Directorate [2012]

In addition to market uncertainty and resource uncertainty, the industry is also subject to regulations of emissions, in particular of $CO_2$, as awareness of the challenges related to climate change increases.

## 1.2 Methodology

The method applied in the decision support models described in this thesis is mathematical programming. Problems are studied in terms of mathematical models where a real function is sought minimized (maximized) over an allowed set of real or integer values. In particular, the subclasses of mathematical programming applied in this thesis are Linear Programming (LP), Mixed Integer Linear Programming (MILP), Global Optimization (GO) and Stochastic Programming (SP).

### Linear Programming (LP)

Linear Programming is one of the great success stories of Operations Research. Many problems can be formulated as the minimization (maximization) of a linear objective function over a convex polyhedron defined by linear side constraints.

The work on linear programming started with military planning problems in the 1940s and 1950s and the development of the simplex method for solving LPs, see Dantzig [1949], Wood and Dantzig [1949]. While the simplex algorithm is not guaranteed to converge in polynomial time, with later refinements it has proven extremely useful and is used both to solve large LPs and to solve relaxations of more computationally demanding problems.

A general LP can be formulated as follows:

$$\min c^\top x \tag{1.1}$$

$$\text{s.t.} Ax \leq b \tag{1.2}$$

$$x \in \mathbb{R}_+^{n_x} \tag{1.3}$$

where coefficients $c$ indicate the costs in the objective function, and the matrix $A$ gives the coefficients of the linear constraints on the decision variables $x$. All constraints must satisfy the right hand side parameters $b$.

The first polynomial time algorithm for LPs was proven by Khachiyan [1980], see also Grötschel et al. [1981], but this method has been of little practical use, rather variations of the simplex method and interior point methods following the work of Karmarkar [1984] are widely applied in industry applications.

### Mixed Integer Linear Programming (MILP)

MILP is an extension of Linear Programming where some or all decision variables are allowed to take only discrete values, giving Mixed Integer Linear Programs or Integer Programs, respectively. This opens up for the possibility to model a much wider range of decision problems, of which many are of a combinatorial nature.

By definition, Mixed Integer and Integer Linear Programs are non-convex. In practice, many MILP predominantly consider binary variables, or 0-1 variables.

A general Mixed Integer Linear Program can be formulated as follows:

$$\min c^\top x + d^\top y \tag{1.4}$$

$$\text{s.t.} Ax + By \leq b \tag{1.5}$$

$$x \in \mathbb{R}_+^{n_x} \tag{1.6}$$

$$y \in \mathbb{Z}_+^{n_y} \tag{1.7}$$

where coefficients $c$ denote the cost in the objective function incurring from continuous variables $x$, $d$ denote cost coefficients incurred from discrete variables $y$. $A$ gives the coefficients for linear constraints in $x$ and $B$ coefficients for constraints in $y$. All constraints must satisfy right hand side parameters $b$.

To solve MILP, branch and bound methods and cutting planes techniques are widely applied, for text books on Integer and Mixed Integer Linear Programming, see e.g. Wolsey [1998], Nemhauser and Wolsey [1988].

While MILP is NP hard, there are several high performance commercial solvers for MILP available, and this model class is widely applied to solving optimization problems in industry. See Bixby and Rothberg [2007] for a fascinating tale of the remarkable progress in MILP solvers over the last decades.

## Global Optimization (GO)

Many relationships in nature are non-linear, and modelling them as a mathematical program may give a model of a more general form where the objective function or the feasible set is non-convex.

Such problems can often be approximated through MILP using piece-wise linear approximations, or solved as they are using other techniques. One often applied technique is to apply convex programming techniques (local solver) that will give a local optimal solution, e.g. using a gradient method. However, such methods may give sub-optimal solutions for non-convex programs. Methods where optimality within a given threshold $\epsilon$ can be proven are called global optimization techniques.

For global optimization, LP, MILP and local solvers are typically used to solve relaxed versions of the problem for some sub domain until convergence can be proven through a branch and bound scheme. While not as large problems can be solved as for LP and MILP, great improvements have been achieved over the last years, and commercial global optimization solvers are available.

A general non-linear mathematical program (NLP) can be formulated as follows:

$$\min_x f(x) \tag{1.8}$$
$$\text{s.t.} g(x) \leq 0 \tag{1.9}$$
$$x \in X \in \mathbb{R}^n \tag{1.10}$$

where $f : X \to \mathbb{R}$, $g_i : X \to \mathbb{R}$, $i = 1 \ldots m$ are continuous. If at least one of these functions is non-convex, the NLP is a non-convex program, and global optimization techniques are necessary to guarantee global optimal solutions. Similar to MILP, integrality constraints on some of the decision variables may be added for Mixed Integer Non-Linear Programs (MINLP). For a text book on Global Optimization, see e.g. Tawarmalani and Sahinidis [2002].

## Stochastic Programming with Recourse (SP)

Many practical decision problems are characterized by uncertainty. A commonly occurring problem in long term planning of production and transport capacity, for example, is the uncertainty in future prices and demand for commodities. The model class of stochastic programming problems with recourse applies to this kind of problems when the distribution of the stochastic parameters is known.

Recourse problems have two or more stages, where the stages represent discrete points in time before and after the true value of some or all stochastic parameters become known. Some decisions have to be made in earlier stages; other decisions (recourse) can be made after the realization of the stochastic parameter is known.

A general formulation for a stochastic two stage problem with recourse can be as follows (Birge and Loveaux [1997]):

$$\min c^\top x + E_\xi Q(x, \xi) \tag{1.11}$$
$$\text{s.t} Ax = b \tag{1.12}$$
$$x \in \mathbb{R}_+^{n_x} \tag{1.13}$$

where $Q(x, \xi) = \min q^\top y | Wy = h - Tx, y \in \mathbb{R}_+^{n_y}$, $\xi$ is the vector formed by the components of $q^\top, h^\top$ and $T$ and $E_\xi$ is the mathematical expectation with respect to $\xi$.

Specialized decomposition approaches are commonly applied to stochastic programming problems, although many problems can be successfully solved using a direct translation to the deterministic equivalent of the problem, which can be solved by a general purpose solver.

The deterministic equivalent of the SP given above can be formulated as follows:

$$\min c^\top x + \sum_{s \in \mathcal{S}} p_s d_s^\top y_s \tag{1.14}$$

$$\text{s.t.} Ax = b \tag{1.15}$$

$$T_s x + W_s y_s = h_s, \forall s \in \mathcal{S} \tag{1.16}$$

$$x \in R_+^{n_x}, y_s \in R_+ \forall s \in \mathcal{S} \tag{1.17}$$

Work on Stochastic (linear) Programming goes back to the 1950s, see Dantzig [1955], Beale [1955]. For text books on Stochastic Programming, see Birge and Loveaux [1997], Kall and Wallace [1994]. Higle [2005] gives a short tutorial on Stochastic Programming, focusing on recourse problems.

Most results in stochastic programming literature are on LP and MILP, but SP can also be formulated as NLP or non-convex NLP. A common premise for SP is that uncertainty is exogenous, i.e. that the distribution and dynamics of the stochastic parameters is independent of decision variables. SP can be generalized to include such dependencies in stochastic programming problems with decision-dependent uncertainty, as described by e.g. Jonsbråten et al. [1998], and we contribute to the extension of this generalized SP in paper six by describing stochastic programming problems with decision-dependent probabilities.

## 1.3 Operational Models for Natural Gas

The petroleum industry has a long history of applications of optimization models in decision support, and the literature is vast. In each part of the value chain, complex problems need to be solved and trade offs made. From the production planning and well management, through processing facilities, transport through pipelines or by ships and on a system level transport capacity booking and contracts management.

In the following we will concentrate on the literature concerning production and transport of natural gas through pipelines, as that has been the focal point of the work in this thesis.

In the short term, the planning problem for a system of coordinated production sites with common transport and processing facilities can be seen as a production portfolio problem and network flow maximization problem with side constraints. Several of these side constraints show system effects, meaning that a change in one part of the system may affect capacity in remote parts of the system.

## Pressure

One property that is a source to system effects is the flow pressure relationship for gas flowing through pipelines. This steady-state relationship was first described by Weymouth [1912], later refined in the Panhandle equations. $Q$ denotes the flow through the pipeline, $K$ is a constant derived from the properties of the pipeline, and $p_{in}$ and $p_{out}$ are the pressure at the inlet and outlet, respectively.

$$Q = K\sqrt{p_{in}^2 - p_{out}^2} \tag{1.18}$$

The system effect stems from the fact that a node in the network will have one pressure, but several pipelines can meet in this point. Raising or lowering the pressure in one node may influence the effective capacity of all those pipelines, and Midthun et al. [2009] shows that it is difficult, if not impossible, to determine appropriate static capacities in a natural gas network.

Including the Weymouth flow pressure relationship as an equality constraint gives a non-convex problem, however in some systems, such as the transport system at the NCS, the upper capacity constraints are usually the main concern, and relaxing the constraint to an upper bound on flow allows for a convex model that can be linearized with reasonable accuracy. Tomasgard et al. [2007] and Rømo et al. [2009] present a linearization of the Weymouth equation which enables the analysis of large networks and stochastic problems. Selot et al. [2008] present an operational model for production and routing planning in the natural gas value chain. The authors combine a detailed infrastructure model with a complex contractual model but do not include a market for natural gas. The infrastructure model comprises nonlinear equations relating pressure and flow in wells and pipelines, multi-commodity flows and contractual agreements (delivery pressure and quality of the gas).

In paper one, Rømo et al. [2009], we demonstrate the inclusion of linearization of pressure constraints and discretization of quality constraints in an optimization tool that is in regular use in the natural gas industry. The users from industry give testimony to the estimated monetary value of including such a tool in their analyses.

## Quality

While natural gas as a commodity is often thought of as a perfectly substitutable commodity in the market place, natural gas from different sources exhibit a wide range of qualities given by the exact composition of gases in the gas mixture. As gas is transported through the network, it can be blended with gas from other sources and processed in processing facilities to remove heavy components and contaminants. The final product must meet certain specifications, for safety reasons, to avoid damage on equipment such as corrosion, and to meet market specifications in terms of e.g. energy content and substitutability between different sources. See Gas Processors Suppliers Association [2012] for typical quality specifications and UK Government [1996] for quality constraints for delivery to the UK.

In a large network with a portfolio of sources where the gas may be blended in several locations, the possibility to blend gas to avoid unnecessary processing becomes relevant. However simple to formulate, this gives a difficult to solve non-convex problems due to the bilinear terms that need to be introduced. In the global optimization literature, such problems are referred to as pooling problems, and they can be traced back to Haverly [1978], who presented a classic problem with two sources, one pool, and two sinks and showed that pooling problems are computationally hard with many local optima. Haverly [1979, 1980]. Visweswaran and Floudas [1990] were the first to solve pooling problems using a global optimization algorithm to the pooling problem, that is, an algorithm guaranteeing to find the global optimal solution. They solved three problems posed by Haverly by solving a series of primal and relaxed dual problems. Foulds et al. [1992] were the first to apply McCormick underestimators and branch and bound (BB) methods to the pooling problem.

Two different, but equivalent formulations are widespread. The original formulation of Haverly [1978] where flow and quality are modeled is often called the P formulation. Ben-Tal et al. [1994] introduced a different formulation based on the flow of individual components and proportions to enter the pool. This formulation is often referred to as the Q formulation. Kocis and Grossmann [1989] also considered a formulation based on flows and fractions going out from splitters. Quesada and Grossmann [1995] introduced a new formulation with extra constraints combining formulations P and Q, and this formulation is called PQ. Tawarmalani and Sahinidis [2002] discuss different global optimization formulations and prove that PQ gives tighter bounds than formulations P and Q. For a nice overview of recent advances in pooling models, see Misener and Floudas [2009].

Another approach to pooling problems is to take advantage of the great progress in MILP solvers and solve discretized pooling problems. Ulstein [2000], Ulstein et al. [2007], Tomasgard et al. [2007], and Hellemo and Werner [2015] applied discretization approaches to large pipeline networks for natural gas with pooling on multiple levels. Hellemo et al. [2012a,b] applied discretization to pooling in network design problems. In comparing global optimization techniques and discretization Haugland [2010], Gupte [2012], Gupte et al. [2012], find that a discretization approach to pooling problems may yield better performance than a global optimization approach, and discretization techniques have been integrated in several global optimization algorithms for pooling problems, see e.g. Castro and Teles [2013] and Kolodziej et al. [2013] for a recent example.

The original pooling problem has been generalized in several ways, both by extending to a network structure which allows interconnected pools. Audet et al. [2004] and Alfaki and Haugland [2013] present a generalization of the PQ formulation for a general network structure.

In paper four, Hellemo and Werner [2015], we introduce a new discretization scheme to take advantage of the fact that the optimal solution may be close to the solution of a single component auxiliary problem, and show that for a number of real world problems this is advantageous. In paper five, Hellemo and Tomasgard [2015] we integrate discrete models at NTNU/SINTEF and continuous models from global optimization community, and add the superclass that includes more general structure and easy model for extraction of some components. We test these formulations on a set of standard test problems from literature and add some test cases based on the network on the NCS.

## Processing

Natural gas that does not satisfy market or terminal requirements need to be processed, in order to remove contaminants such as $H_2S$ and $CO_2$, water and heavy hydrocarbons. Processing facilities are often large on-shore facilities comprising a range of chemical processes and utilizing a range of techniques to separate components to create separate products. However, simpler processing may also take place off shore on production platforms or even in sub-sea installations.

Diaz et al. [1997] present a model for configuration of natural gas processing plants based on turbo expansion, using a MINLP. Feed gas mixtures with varying content of contaminants are analyzed to evaluate the plant design and operational consequences. Grossmann et al. [1999] gives an overview of optimization models for facility and process design for chemical processing in general, and show that they often yield non-linear programming models. A recent example of an optimization model for a gas processing plant considering uncertainty in feed flow rate and composition modelled with the use of chance constraints is Mesfin and Shuhaimi [2010].

In paper 5, Hellemo and Tomasgard [2015] we suggest a generalization of pooling problems that include simplified processing facilities at a system level, similar to Ulstein [2000], Ulstein et al. [2007] and Rømo et al. [2009], with the possibility to specify the removal of each gas component of the gas mixture and the possibility to give bounds on the absolute volumes removed of each component. This is a simplification of the workings of the processing facility, and we do not consider the specific characteristics of the different processes involved. More detailed models of large processing facilities are possible using the basic modelling components e.g. for each process train. We also demonstrate how composite quality constraints can be added, and how this is facilitated by modelling the component flows directly.

### Booking

Following the implementation of the EU Gas Directives EU Commission [1998, 2003], the pipeline transport system at the NCS is managed by and independent system operator, Gassco. The models presented in this thesis do not consider the details of the booking regime for capacity.

Some recent papers that do consider booking problems are e.g., Kalashnikov et al. [2010] who model the transport of gas by a transport company and a pipeline operating company as a stochastic bi-level problem, and Fodstad et al. [2015], where the authors consider the introduction of interruptible transport capacity to the NCS transport system.

### LNG Transport

Natural gas may alternatively be transported over long distances as Liquefied Natural Gas (LNG). We do not consider this option explicitly in any of the models in this thesis; however, it would be possible to include a very stylized approximation of an LNG transport by modelling it as a pipeline with fixed capacity. For some recent papers on LNG transport, see e.g. Fodstad et al. [2010], Rakke et al. [2011], Stålhane et al. [2012]

## 1.4 Strategic Models for Natural Gas

### Deterministic Models

There exist a number of deterministic investment models, and early overviews can be found in Sullivan [1988] and Haugland et al. [1988]. Nygreen et al. [1998] present a multi-period MIP model used by the Norwegian Petroleum Directorate. The model is employed for the investment planning of fields in the North Sea which contain a mixture of oil and gas. In van den Heever and Grossmann [2001], a model for design and planning of offshore field infrastructure projects is presented. The model is a multi-period mixed-integer nonlinear programming model (MINLP) and incorporates complex fiscal rules such as tariff, tax and royalty calculations. The net present value of projects is discussed in the light of these fiscal rules. There are also some models which incorporate uncertainty. Jørnsten [1992] presents an integer model for sequencing offshore oil and gas fields, where the objective is to maximize total economic benefit.

In paper two, Hellemo et al. [2012b], we present a deterministic strategic model for planning development of infrastructure for natural gas production, transport and processing. This model allows the inclusion of pressure constraints and quality constraints, similar to what has been used for operational models, in a strategic planning model.

## Stochastic Models

Haugen [1996] develops a stochastic dynamic programming model to analyze a supplier's problem of scheduling fields and pipelines in order to be able to meet contractual agreements. The uncertainty in this model is in the resources (production profiles). In Jonsbråten [1998], a stochastic MIP model for optimal development of an oil field is presented. The objective of the model is to maximize the expected net present value of the oil field given uncertain future oil prices. Goel and Grossmann [2004] present a stochastic MIP model for the planning of offshore gas field developments. The expected net present value is maximized under uncertainty in reserves. A multi-stage investment model with decision-dependent uncertainty is presented by Tarhan et al. [2009] along with a branch-and-bound solution method for non-convex mixed-integer nonlinear sub problems. They take into account nonlinear reservoir models and gradually revealed uncertainties concerning initial flow rate, recoverable volume and other characteristics of the reservoirs.

Meyer and Floudas [2006] generalized the pooling problem to include network design, leading to combinatorial pooling problems. Li et al. [2011a] consider a stochastic pooling problem with network design and operations and present a global optimization decomposition algorithm for solving such problems in Li et al. [2011b].

## Game Theoretical Models

When several different stakeholders are involved in the production, transport, processing and sales of natural gas in the same system, it opens up for strategic behavior. This kind of behavior, where each participant anticipates the others' reactions before deciding what to do, is commonly modelled using game theory and typically use some kind of equilibrium models, often formulated as complementarity problems. See Gabriel and Smeers [2006] for an overview of complementarity problems in the natural gas industry.

We do not consider the gaming situation between different actors in the natural gas value chain, but consider what could be achieved by fully coordinated actions. While equilibrium models are very interesting, they pose other challenges in terms of interpretation of results and the premises for the analysis, for example regarding the availability of full information for all players. We are aware that with less coordinated companies with different ownership shares in different parts of the network, a fully coordinated solution can not be expected; still we believe it is useful to optimize the entire system as a benchmark for maximization of value creation.

## 1.5 Integrating Operational and Strategic Models

In making strategic models for processes which exhibit large short term variability, the modeler is often presented with a dilemma. Including too much detail in describing short term effects will make the model computationally intractable. However, representing variable parameter values by expected values or maximum values, for instance, will typically give under dimensioning or over dimensioning, respectively. It will not correctly represent the trade off between gains from extra capacity to serve periods of high demand, versus the extra cost of this extra capacity.

In order to ensure that short-term fluctuations and peak demand situations can be accounted for in the system design, we include a representation of operational flexibility in a strategic investment model. Not much work can be found which focuses on both aspects at the same time. Schütz et al. [2009] include short-term variations in a strategic model for the Norwegian meat industry. De Jonghe et al. [2011] use an equilibrium model to study generation expansion. They integrate the short-term demand response in their strategic model and discuss the effects on flexibility of the generation capacity. The approach, however, considers only a one-period static model. Sönmez et al. [2011] analyze technology choice in LNG transport and discuss the impact of using a stochastic model for LNG throughput. They show that operational flexibility is important in order to cope with short-term variations and that is has a significant impact on profitability.

We suggest a multi-horizon tree in paper 3, Hellemo et al. [2012a] to incorporate short term variability in stochastic strategic models. By including sub trees representing operational time periods for each strategic node in the scenario tree, we are able to take short term variability into account while keeping the scenario tree relatively small. Singh et al. [2009] describe a multi-stage capacity-planning problem for an electricity distribution network which also allows stochastic operational sub models. They solve their model using Dantzig-Wolfe decomposition and variable splitting. The concept of multi-horizon scenario trees is developed further in Kaut et al. [2013].

## 1.6 Decision-Dependent Uncertainty

A standard assumption in stochastic programming is that the uncertainty is independent of the decisions made in the model, that is both in terms of the probability distribution of the stochastic parameters and the dynamics of the uncertainty, e.g. when uncertainty is resolved.

This can be a limitation when considering decision problems concerning the design of or upgrade of existing infrastructure for petroleum exploitation. One of the main sources of uncertainty in development of new reservoirs is the size and characteristics of each new reservoir. Although great progress has been made within seismic analysis, there still remains uncertainty to the exact qualities of each reservoir until actual drilling takes place. Drilling offshore wells is expensive, and the decisions on how, where and when to drill requires careful analysis and planning. The crucial point is that the uncertainty is not resolved until the well has been drilled. Similarly, as infrastructure grows older, the probability of unwanted events increases, and by improving, refurbishing or replacing ageing infrastructure, this probability can be reduced.

These are examples of situations where it would be useful to relax this assumption and allow the model to change the characteristics of the uncertainty as seen by the model, given some decisions, in other words to allow decision-dependent uncertainty in the model. The literature on such models is sparse. To our knowledge, the first attempt to make such a model was Pflug [1990] who solved the model finding the stochastic quasigradient. Pflug suggests possible applications of his simulation based approach to the optimal design of a communications network, optimal allocation of resources in a queuing system, optimal size of warehouses and optimal design of power plants. A generalization of stochastic programming that includes problems where the uncertainty depends on decision variables was introduced by Jonsbråten [1998], Jonsbråten et al. [1998]. Goel and Grossmann [2006] suggested problems with decision-dependent uncertainty are of one of two types: problems with decision-dependent probabilities are of type 1, and problems with decision-dependent information revelation are of type 2.

Dupačová [2006] provided another overview over such problems, She identifies two fundamental classes of problems with endogenous uncertainty. One where the probability distribution is known and the decisions influence the parameters and one where some decision will cause the probability distribution to be chosen between a finite set of probability distributions. The first attempt to explicitly model the relationship between the probability measure and the decision variable was made by Ahmed [2000]. He formulates single stage stochastic programs that are applied on network design, server selection and p-choice facility location.

In the final paper, Hellemo et al. [2015], we expand previous taxonomies presented for stochastic programming problems with decision-dependent uncertainty, and we present several two stage recourse models where probability distribution can be manipulated through distortions, by directly changing the parameters of the distribution for an exact pdf or for an approximation.

## 1.7 Summary of papers

In this section I will give a short presentation of each paper and my contributions to each paper.

### Paper I — Optimizing the Norwegian Natural Gas Production and Transport, page 27

*Authors: Frode Rømo, Asgeir Tomasgard, Lars Hellemo, Marte Fodstad, Bjørgulf Haukelidsæter Eidesen, and Birger Pedersen*

Reprinted by permission, Interfaces Vol. 39, No. 1, January–February 2009, pp 46–56, Copyright (2009), the Institute for Operations Research and the Management Sciences, 5521 Research Park Drive, Suite 200, Catonsville, Maryland 21228 USA.

This paper presents a model for network flow optimization for natural gas pipeline transmission grids, which is in daily use for planning purposes on the NCS. We present the deterministic MILP model for network flow maximization with linearization of the Weymouth equation as an upper bound and discretization of multi-component flow fractions to find an approximate solution to the pooling problem. The project is a testament to the great importance of close cooperation with industry partners in developing operations research models for use in industry. The industry partners estimate the financial gain stemming from having access to such a model to be in the order of USD 2 billion over the time that the model has been in use. The work presented in this paper was selected as a finalist in the 2008 INFORMS Franz Edelman competition.

The modelling and implementation has been performed at SINTEF, with some contributions from researchers at NTNU. I have participated in refinement of the model and interface, and taken main responsibility for writing and submitting the paper.

## Paper II — Natural Gas Infrastructure Design with an Operational Perspective, page 41

*Authors: Lars Hellemo, Kjetil Midthun, Asgeir Tomasgard, and Adrian Werner*

Reprinted by permission, Energy Procedia 26 (2012) 67–73

This paper presents a model for natural gas infrastructure investments and capacity expansions. We consider both existing infrastructure and potential expansions, and the model is formulated as a deterministic MILP. We include a linearization of the Weymouth equation for flow pressure relationship and include the possibility to add quality constraints through a discretization of the pooling problem. We also discuss a relevant investment case from the NCS.

The modelling is joint work with colleagues at SINTEF and NTNU. I have contributed to the implementation and performed numerical experiments. I have contributed in equal part in discussions and in writing the paper.

## Paper III — Multi-Stage Stochastic Programming for Natural Gas Infrastructure Design with a Production Perspective, page 51

*Authors: Lars Hellemo, Kjetil Midthun, Asgeir Tomasgard, and Adrian Werner*

Published in Ziemba, W. T., Wallace, S. W., Gassman, H. I. (Eds.), Stochastic programming — Applications in finance, energy, planning and logistics. Vol. 4 of World Scientific series in finance. World Scientific, 2012, pp. 259–288, Copyright (2012) World Scientific

In paper three we present a multi-stage stochastic model for natural gas investments and capacity expansions. We consider candidate projects and existing infrastructure together when optimizing the operations of the available infrastructure during each time period. Several uncertain parameters can be included, both upstream and downstream, e.g. reservoir size and quality composition, market demand and prices. We also include analysis of the effects on the expected production assurance from the solutions chosen by the model.

In order to analyze both long term uncertainty and short term variability we introduce a novel scenario tree structure intended to reduce dimensionality and improve computational tractability. The multi-horizon tree structure allows branching into separate sub trees for each strategic node independently of further branching of the long term uncertainty.

The modelling is joint work with colleagues at SINTEF and NTNU. I have contributed to the implementation and performed numerical experiments. I have contributed in equal part in discussions and in writing the paper.

## Paper IV — Discretizations of Natural Gas Pooling Problems, page 83

*Authors: Lars Hellemo and Adrian Werner*

Submitted to an international, peer-reviewed journal

In the fourth paper we evaluate several discretization schemes for the non-convex pooling problem. We introduce a novel discretization scheme, where information from an auxiliary LP relaxation is used to determine the discretization. We also introduce post processing problem, a set of linear equations solved as an LP to determine the quality flows in problem instances where the quality constraints are non-binding. Whether the constraints are binding is not known in advance, and we propose a solution scheme where the computationally cheap post processing is performed first, resorting to more elaborate discretization schemes depending on problem properties. We show that the proposed scheme works well on a set of real world problem instances motivated by natural gas transport on the Norwegian Continental Shelf.

This work is joint work with my colleague at SINTEF. I have contributed in equal parts in discussions and implementation. I have taken main responsibility for writing the paper, and I have performed the numerical experiments.

## Paper V — A Generalized Global Optimization Formulation of the Pooling Problem with Processing Facilities and Composite Quality Constraints, page 117

*Authors: Lars Hellemo and Asgeir Tomasgard*

Accepted for publication in TOP

In this paper we present a new continuous formulation of the pooling problem which is more general than the normal pooling problem. Our formulation allows several levels of pools, also when intermediate pools are not directly connected to sources or sinks.

By using a multi-commodity flow formulation, it is easy to extend the standard pooling model to include simple processing facilities that can alter the flow composition and composite quality constraints.

We compare the performance of our formulation with other formulations, using test cases from literature. We also introduce new test cases based on real word problems stemming from the Norwegian Continental Shelf. We also investigate the extra computational effort required to solve problems with composite quality constraints.

This is joint work with my supervisor at NTNU. I have been the main author and have performed all implementation and numerical experiments.

## Paper VI — Stochastic Programming with Decision-Dependent Probabilities, page 173

*Authors: Lars Hellemo, Paul I. Barton, and Asgeir Tomasgard*

Submitted to an international, peer-reviewed journal

In the final paper we present a taxonomy of stochastic programming problems with decision-dependent uncertainty, which expands on previous taxonomies of such problems.

We also introduce two kinds of two-stage recourse problems with decision-dependent probabilities where the probability distributions are manipulated directly as a continuous function of some first stage decision variables. The manipulation is either performed as an affine distortion of some pre existing discrete probability distribution (models 1 and 2), or as the direct manipulation of some parameters of a continuous distribution (models 3 and 4). The four models are: Distortion of subsets of scenarios and corresponding correction of remaining scenarios, convex combination of probabilities, Kumaraswamy distribution and an approximation to the Normal distribution with change of mean through change of variables.

We include test instances of all four problem types with different number of discrete scenarios and test the computational implications of each problem type through numerical experiments. The decomposition approach we implemented was outperformed by a commercial global optimization solver with selective branching.

This is joint work with my supervisor at NTNU and my co-supervisor at MIT. I have been the main author and have contributed in equal part in discussions. I also performed the implementation and numerical experiments.

# Bibliography

S. Ahmed. *Strategic Planning under Uncertainty – Stochastic Integer Programming Approaches*. PhD thesis, Graduate College, University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2000.

M. Alfaki and D. Haugland. A multi-commodity flow formulation for the generalized pooling problem. *Journal of Global Optimization*, 56:917–937, 2013.

C. Audet, J. Brimberg, P. Hansen, S. Le Digabel, and N. Mladenović. Pooling problem: Alternate formulations and solution methods. *Management Sscience*, pages 761–776, 2004.

E. M. L. Beale. On minimizing a convex function subject to linear inequalities. *Journal of the Royal Statistical Society, Series B*, 17(2):173–184, 1955.

A. Ben-Tal, G. Eiger, and V. Gershovitz. Global minimization by reducing the duality gap. *Mathematical Programming*, 63(1):193–212, 1994.

J. R. Birge and F. V. Loveaux. *Introduction to Stochastic Programming*. Springer-Verlag, New York, 1997.

R. Bixby and E. Rothberg. Progress in computational mixed integer programming–a look back from the other side of the tipping point. *Annals of Operations Research*, 149(1):37–41, 2007.

P. M. Castro and J. P. Teles. Comparison of global optimization algorithms for the design of water-using networks. *Computers & Chemical Engineering*, 52: 249–261, 2013.

G. B. Dantzig. Programming of interdependent activities: II mathematical model. *Econometrica, Journal of the Econometric Society*, pages 200–211, 1949.

G. B. Dantzig. Linear programming under uncertainty. *Management Science*, 1: 197–206, 1955.

C. De Jonghe, B. Hobbs, and R. Belmans. Integrating short-term demand response into long-term investment planning. Cambridge working papers in economics 1132, Faculty of Economics, University of Cambridge, 2011.

M. Diaz, A. Serrani, J. Bandoni, and E. Brignole. Automatic design and optimization of natural gas plants. *Industrial & engineering chemistry research*, 36 (7):2715–2724, 1997.

J. Dupačová. Optimization under exogenous and endogenous uncertainty. In L. Lukáš, editor, *Proc. of MME06*, pages 131 – 136. University of West Bohemia in Pilsen, 2006.

EU Commission. Directive 98/30/EC of the European Parliament and of the Council of 22 June 1998 concerning common rules for the internal market in natural gas, 1998.

EU Commission. Directive 2003/55/EC of the European Parliament and of the Council of 26 June 2003 concerning common rules for the internal market in natural gas and repealing Directive 98/30/EC, 2003.

M. Fodstad, K. T. Uggen, F. Rømo, A.-G. Lium, G. Stremersch, and S. Hecq. LNGScheduler: a rich model for coordinating vessel routing, inventories and trade in the liquefied natural gas supply chain. *Journal of Energy Markets*, 3 (1):1–34, 2010.

M. Fodstad, K. T. Midthun, and A. Tomasgard. Adding flexibility in a natural gas transportation network using interruptible transportation services. *European Journal of Operational Research*, 243(2):647–657, 2015.

L. R. Foulds, D. Haugland, and K. Jørnsten. A bilinear approach to the pooling problem. *Optimization*, 24:165–180, 1992.

S. Gabriel and Y. Smeers. *Complementarity problems in restructured natural gas markets*. Springer, 2006.

Gas Processors Suppliers Association. *Engineering Data Book*. Tulsa, 12th edition, 2012.

V. Goel and I. Grossmann. A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. *Computers and Chemical Engineering*, 28(8):1409–1429, 2004.

V. Goel and I. Grossmann. A class of stochastic programs with decision dependent uncertainty. *Mathematical Programming*, 108(2):355–394, 2006.

I. E. Grossmann, J. A. Caballero, and H. Yeomans. Mathematical programming approaches to the synthesis of chemical process systems. *Korean Journal of Chemical Engineering*, 16(4):407–426, 1999.

M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

A. Gupte. Mixed integer bilinear programming with applications to the pooling problem, 2012.

A. Gupte, S. Ahmed, S. S. Dey, and M. S. Cheon. Pooling problem, 2012. URL `www.optimization-online.org/DB_FILE/2012/10/3658.pdf`.

K. Haugen. A Stochastic Dynamic Programming model for scheduling of offshore petroleum fields with resource uncertainty. *European Journal of Operational Research*, 88(1):88–100, 1996.

D. Haugland. An overview of models and solution methods for pooling problems. *Energy, Natural Resources and Environmental Economics*, pages 459–469, 2010.

D. Haugland, Å. Hallefjord, and H. Asheim. Models for petroleum field exploitation. *European Journal of Operational Reseach*, 37:58–72, 1988.

C. Haverly. Studies of the behavior of recursion for the pooling problem. *ACM SIGMAP Bulletin*, 25:19–28, 1978.

C. Haverly. Behavior of recursion model—more studies. *ACM SIGMAP Bulletin*, (26):22–28, 1979.

C. Haverly. Recursion model behavior: more studies. *ACM SIGMAP Bulletin*, (28):39–41, 1980.

L. Hellemo and A. Tomasgard. A Generalized Global Optimization Pooling Formulation With Processing Facilities and Compoisite Quality Constraints. 2015.

L. Hellemo and A. Werner. Discretizations of natural gas pooling problems. In Review, 2015.

L. Hellemo, K. Midthun, A. Tomasgard, and A. Werner. Multi-stage stochastic programming for natural gas infrastructure design with a production perspective. In W. T. Ziemba, S. W. Wallace, and H. I. Gassman, editors, *Stochastic programming – Applications in finance, energy, planning and logistics*, volume 4 of *World Scientific series in finance*, pages 259–288. World Scientific, 2012, 2012a.

L. Hellemo, K. Midthun, A. Tomasgard, and A. Werner. Natural Gas Infrastructure Design with an Operational Perspective. *Energy Procedia*, 26:67–73, 2012b.

L. Hellemo, A. Tomasgard, and P. I. Barton. Stochastic Programming with Decision-Dependent Probabilities. 2015.

J. L. Higle. Stochastic programming: optimization when uncertainty matters. *Cole Smith J (ed) Tutorials in operations research*, pages 30–53, 2005.

T. Jonsbråten. Oil field optimization under price uncertainty. *Journal of the Operational Research Society*, 49(8):811–818, 1998.

T. Jonsbråten, R. Wets, and D. Woodruff. A class of stochastic programs with decision dependent random elements. *Annals of Operations Research*, 82:83–106, 1998.

K. O. Jørnsten. Sequencing offshore oil and gas fields under uncertainty. *European Journal of Operational Research*, 58:191–201, 1992.

V. Kalashnikov, G. Pérez-Valdés, A. Tomasgard, and N. Kalashnykova. Natural gas cash-out problem: Bilevel stochastic optimization approach. *European Journal of Operational Research*, 206(1):18–33, 2010.

P. Kall and S. W. Wallace. *Stochastic Programming.* John Wiley & Sons, Chichester, 1994.

N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

M. Kaut, K. T. Midthun, A. S. Werner, A. Tomasgard, L. Hellemo, and M. Fodstad. Multi-horizon stochastic programming. *Computational Management Science*, pages 1–15, 2013.

L. G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

G. Kocis and I. Grossmann. A modelling and decomposition strategy for the MINLP optimization of process flowsheets. *Computers & Chemical Engineering*, 13(7):797–819, 1989.

S. Kolodziej, P. M. Castro, and I. E. Grossmann. Global optimization of bilinear programs with a multiparametric disaggregation technique. *Journal of Global Optimization*, 57(4):1039–1063, 2013.

X. Li, E. Armagan, A. Tomasgard, and P. I. Barton. Stochastic pooling problem for natural gas production network design and operation under uncertainty. *AIChE Journal*, 57:2120–2135, 2011a.

X. Li, A. Tomasgard, and P. Barton. Decomposition strategy for the stochastic pooling problem. *Journal of Global Optimization*, 54:765–790, 2011b.

G. Mesfin and M. Shuhaimi. A chance constrained approach for a gas processing plant with uncertain feed conditions. *Computers & chemical engineering*, 34 (8):1256–1267, 2010.

C. Meyer and C. Floudas. Global optimization of a combinatorially complex generalized pooling problem. *AIChE journal*, 52(3):1027–1037, 2006.

K. Midthun, M. Bjørndal, and A. Tomasgard. Modeling optimal economic dispatch and system effects in natural gas networks. *Energy Journal*, 30:155–180, 2009.

R. Misener and C. Floudas. Advances for the pooling problem: Modeling, global optimization, and computational studies. *Applied and Computational Mathematics*, 8(1):3–22, 2009.

G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, New York, 1988.

B. Nygreen, M. Christiansen, K. Haugen, T. Bjørkvoll, and Ø. Kristiansen. Modelling Norwegian petroleum production and transportation. *Annals of Operations Research*, 82:251–267, 1998.

G. C. Pflug. On-line optimization of simulated markovian processes. *Mathematics of Operations Research*, 15(3):381–395, August 1990.

I. Quesada and I. Grossmann. Global optimization of bilinear process networks with multicomponent flows. *Computers & Chemical Engineering*, 19(12):1219–1242, 1995.

J. G. Rakke, M. Stålhane, C. R. Moe, M. Christiansen, H. Andersson, K. Fagerholt, and I. Norstad. A rolling horizon heuristic for creating a liquefied natural gas annual delivery program. *Transportation Research Part C: Emerging Technologies*, 19(5):896–911, 2011.

F. Rømo, A. Tomasgard, L. Hellemo, M. Fodstad, B. Eidesen, and B. Pedersen. Optimizing the Norwegian natural gas production and transport. *Interfaces*, 39(1):46–56, 2009.

P. Schütz, A. Tomasgard, and S. Ahmed. Supply chain design under uncertainty using sample average approximation and dual decomposition. *European Journal of Operational Research*, pages 409–419, 2009.

A. Selot, L. Kuok, M. Robinson, T. Mason, and P. Barton. A short-term operational planning model for natural gas production systems. *AIChE Journal*, 54 (2):495–515, 2008.

K. J. Singh, A. B. Philpott, and R. K. Wood. Dantzig-wolfe decomposition for solving multistage stochastic capacity-planning problems. *Operations Research*, 57(5):1271–1286, 2009.

E. Sönmez, S. Kekre, A. Scheller-Wolf, and N. Secomandi. Strategic analysis of technology and capacity investments in the liquefied natural gas industry. Technical report, Carnegie Mellon Tepper School of Business, 2011.

M. Stålhane, J. G. Rakke, C. R. Moe, H. Andersson, M. Christiansen, and K. Fagerholt. A construction and improvement heuristic for a liquefied natural gas inventory routing problem. *Computers & Industrial Engineering*, 62(1): 245–255, 2012.

J. Sullivan. The application of mathematical programming methods to oil and gas field development planning. *Mathematical Programming*, 42:189–200, 1988.

B. Tarhan, I. Grossmann, and V. Goel. 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, 2009.

M. Tawarmalani and N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming.* Kluwer Academic Publishers, Norwell, MA, USA, 2002.

The Norwegian Ministry of Petroleum and Energy and The Norwegian Petroleum Directorate. Facts 2012 — The Norwegian petroleum sector. http://www.npd.no/en/Publications/Facts/Facts-2012/, 2012.

A. Tomasgard, F. Rømo, M. Fodstad, and K. Midthun. Optimization models for the natural gas value chain. In G. Hasle, K.-A. Lie, and E. Quak, editors, *Geometric Modelling, Numerical Simulation and Optimization*, pages 521–558. Springer Verlag, 2007.

UK Government. Gas safety (management) regulations 1996, 1996. UK Statutory Instruments.

N. L. Ulstein. Short Term Planning of Gas Production (In Norwegian), 2000. Masters thesis, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology.

N. L. Ulstein, B. Nygreen, and J. R. Sagli. Tactical planning of offshore petroleum production. *European Journal of Operational Research*, 176(1):550–564, January 2007.

S. A. van den Heever and I. E. Grossmann. A Lagrangean decomposition heuristic for the design and planning of offshore hydrocarbon field infrastructure with complex economic objectives. *Industrial & Engineering Chemistry Research*, 40:2857–2875, 2001.

V. Visweswaran and C. Floudas. A global optimization algorithm (GOP) for certain classes of nonconvex NLPs — II. Application of theory and test problems. *Computers & chemical engineering*, 14(12):1419–1434, 1990.

T. Weymouth. Problems in natural gas engineering. *Transactions of the ASME*, 34:185–234, 1912.

L. A. Wolsey. *Integer Programming.* Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, New York, 1998.

M. K. Wood and G. B. Dantzig. Programming of interdependent activities: I general discussion. *Econometrica, Journal of the Econometric Society*, pages 193–199, 1949.

# Paper I

Frode Rømo, Asgeir Tomasgard, Lars Hellemo, Marte Fodstad,
Bjørgulf Haukelidsæter Eidesen, and Birger Pedersen:

# Optimizing the Norwegian Natural Gas Production and Transport

Is not included due to copyright

# Paper II

Lars Hellemo, Kjetil Midthun, Asgeir Tomasgard, and Adrian Werner:

# Natural Gas Infrastructure Design with an Operational Perspective

*Reprinted by permission, Energy Procedia 26 (2012) 67–73*

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Energy Procedia 26 (2012) 67 – 73

Energy Procedia

2nd Trondheim Gas Technology Conference

# Natural Gas Infrastructure Design with an Operational Perspective

Lars Hellemo[a], Kjetil Midthun[b], Asgeir Tomasgard[a], Adrian Werner[b]

[a]*Norwegian University of Science and Technology, Department for Industrial Economics and Technology Management, Alfred Getz veg 3, NO–7491 Trondheim, NORWAY*
[b]*SINTEF Technology and Society, Department for Applied Economics and Operations Research, NO–7465 Trondheim, NORWAY*

**Abstract**

We present an investment analysis tool for natural gas infrastructure development. The model takes a system perspective and considers all existing infrastructure as well as the potential expansions. We formulate the investment problem as a deterministic mixed-integer linear program. We extend existing infrastructure analysis models within natural gas by adding pressure flow relationships and by modeling the gas quality in the transportation system. In this paper we present the motivation for the model functionality as well as the main components in the model. We also discuss a relevant investment case from the Norwegian Continental Shelf to exemplify typical model sizes as well as solution times.



*Keywords:* natural gas, investment analysis, decision support

## 1. Introduction

Investment analysis of petroleum infrastructure has been widely studied. One major motivation for these studies is the large costs associated with both production facilities and transport facilities related to, in particular, offshore production. The natural gas value chain also has some special characteristics that make careful analysis and good decision support tools very valuable. In our study we have used the Norwegian Continental Shelf as the motivating case. The transport network on the Norwegian Continental Shelf is the largest subsea gas transport network in the world. It consists of approximately 7800 km of pipelines with large diameters that are operated at high pressure levels. The production facilities have mixed characteristics in terms of some flexible production fields and some fields that are primarily oil fields where the gas production must be maintained in order to keep the oil production level high. The fields also produce gas with different quality. We model flow through pipelines as a multi-commodity flow where the commodities are different gas components, such as methane, butane, propane, $CO_2$, and $H_2S$. It is important to keep track of the gas quality since there are specifications in the market nodes that must be met, with respect to both the energy content of deliveries in terms of gross calorific value (GCV) and the maximum content of $CO_2$. Gas that does not meet the specifications can be either processed or blended with gas from fields with better quality. Deciding between these two options is one of the important choices that must be




doi:10.1016/j.egypro.2012.06.011

made when designing new infrastructure. The investment costs will normally be lower when quality issues can be solved by blending rather than extra processing capacity. However, it also leads to a less flexible system that may reduce the security of supply and total throughput in the system. Reducing production on fields with high quality gas implies also a lower production on fields with gas with lower quality because they depend on blending with gas of high quality.

The main contribution of our work is a unified framework for analyzing investment decisions while taking into account both physical properties of the network and the dynamics of short-term planning. Important physical properties that influence the optimal design of the infrastructure are system effects and quality requirements in the market nodes. We use a system perspective where we consider the value chain for natural gas from production fields to delivery in market nodes. We perform a portfolio optimization where all existing infrastructure as well as all potential projects or extensions is included in the same analysis. With this approach we are able to take into account how the candidate extensions will influence the operation of the rest of the network and vice versa.

A tool that has been used by both authorities and companies that invest in natural gas infrastructure is presented in Nygreen et al. [1]. We build on the investment formulation from this model, and extend it by more details on the operations of the network (such as pressure flow modeling and gas quality). This way we can analyze projects such as branch-offs and the trade-off between processing plants and blending of natural gas from different fields. The basis for our modeling of the natural gas transport is given in Rømo et al. [2] and Midthun [3]. A natural gas value chain is also analyzed in Ulstein et al. [4], while Westphalen [5] discusses models both for natural gas and for electricity. Sullivan [6] and Haugland et al. [7] both review mathematical programming models for investment analysis in the petroleum sector. A multi-period, non-linear mixed-integer programming model for offshore field infrastructure is presented in van den Heever and Grossmann [8]. De Jonghe et al. [9] provide an example for equilibrium models analyzing investment planning while Sönmez et al. [10] present an investment analysis model for the LNG value chain which also incorporates uncertainty.

The mathematical model which is the basis for our investment analysis tool can be formulated as a deterministic mixed-integer linear program (MILP). In this paper, we focus on basic model functionality which we will discuss in the following section. For a detailed mathematical formulation we refer to Werner et al. [11]. In Section 3, we present a realistic case study from the Norwegian Continental shelf and focus on how the model size and solution time depend on several model features. We conclude the paper in Section 4.

## 2. The mathematical model

In the mathematical model, we can distinguish between modeling design changes in the network and the operation of the network itself. Both aspects are covered in the same model, but we discern investment time periods and operational time periods: the decisions made in the investment time periods give the operating conditions for the operational time periods (in terms of network design and network components), while the operational time periods determine the cash flow and security of supply in the network. This time structure is illustrated in Figure 1. The distinction between investment periods and operational periods allows for a detailed representation of the operational aspects of the network analysis. At the same time we can consider a sufficiently long time horizon for the investments while limiting the number of binary variables associated with the investment decisions.

In the following, we present the main elements of the mathematical model, before we elaborate on two key aspects of the model, system effects and gas quality.

### 2.1. Model formulation

The objective of the model is maximization of net present value. For the investment projects we include costs of installing new infrastructure as well as costs of removing old infrastructure. In addition, there are operating costs related to maintaining and operating the infrastructure and the production on the different fields. Revenue is calculated in the operational time periods, and results from sales in contracts and on spot markets. The resulting cash flows are discounted with a given required rate of return.

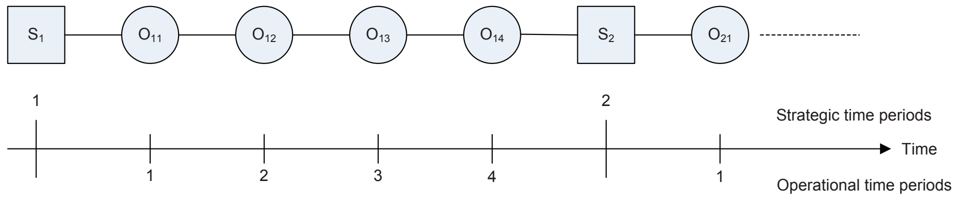



Fig. 1. The time structure used in our model. The square nodes represent investment periods where decisions regarding infrastructure development are made, whilst the circular nodes represent operational periods where the operation of the network is determined and revenues and costs are found.

*Investment decisions.* The investment decisions in the model are whether or not to invest in proposed projects such as pipelines, fields, processing plants, landing points, and junction nodes and are, therefore, modeled using binary variables. A further set of decisions concerns the potential shut-down of projects (with a cost for their removal). Also existing infrastructure is modeled in this way. All projects have some common characteristics, such as a time window for investment (earliest and latest start-up time), construction time (delay between investment decision and time of operation), and costs. In addition, there are characteristics that are unique for the different projects. For the fields, for instance, we specify reservoir levels, production rate levels and composition of the gas in the reservoirs. In order to represent different field development possibilities at one site, we include several investment projects, each describing one development option. In such a case, we also add constraints ensuring that only one of the projects can be chosen. Generally, constraints can also be used to tie projects together in groups such that if one of the projects in the group is invested in, then the rest of the projects in the group must also be invested in (or only one project from the group can be invested in).

*Operational decisions.* In the operational time periods we include constraints for production limits in the fields, mass balances in the network, gas quality, market demand, and relationships between flow and pressure. Parameters such as demand and prices often show a seasonal effect in addition to large daily variations. These variations can influence optimal network design choices. An investment analysis using average values over long time periods neglects all short-term variations. This is similar to using a deterministic model (where the uncertainty is averaged) rather than a stochastic model, and can lead to some of the same effects. Averaging values ignores their variability, and the network may end up being inflexible and unable both to handle negative events and to take advantage of positive events such as very high prices or high demand. By considering a fine time resolution for the operational time periods we avoid this challenge and can represent different operating conditions for the natural gas value chain.

### *2.2. System effects*

In a natural gas network it is normally not possible to predefine capacities in the pipelines since the capacity of one pipeline will depend on how the surrounding pipelines are operated (for a discussion, see Midthun et al. [12]). The extent of such system effects can be illustrated with a simple investment example (see Figure 2). The original network, consisting of a production field *A* and a market node *B*, has a transport capacity of 51.3 $MSm^3/d$. There are two possible points, “1” and “2”, that can be used to connect market node *C* to this network. Even if all pipeline characteristics (except of length) are identical between the two pipelines, the two connection points will still give two very different networks: If we fix the flow between *A* and *C* to 10 units and then maximize the flow between nodes *A* and *B*, choosing point “1” gives a flow of 47.5 $MSm^3/d$ while point “2” results in a flow of 44.1 $MSm^3/d$.

The capacity in a given pipeline depends, at any point in time, on the pressure into the pipeline, the pressure out of the pipeline and the pipeline characteristics (such as length, diameter, or friction). The relationship between flow, pressure levels, and design parameters can be expressed by the Weymouth equation:

$$f_{ij} = K_{ij}^W \sqrt{p_i^2 - p_j^2}, \quad i,\ j \in \mathcal{N} \tag{1}$$

where $f_{ij}$ is the flow between network nodes $i$ and $j$ when the pressures in these nodes are $p_i$ and $p_j$, respectively. The other pipeline characteristics are aggregated in the constant $K_{ij}^W$. The value of the constant $K_{ij}^W$ is based on the theoretical value found from the characteristics of the pipeline (see, for instance, [13]) and is, when needed, adjusted according to historical observations and tests. This equation can be linearized by using a first-order Taylor expansion around $L$ fixed input and outlet pressure points $PI_{il}$ and $PO_{jl}$ for the pipeline between nodes $i$ and $j$:

$$f_{ijo} \leq K_{ij}^W \left( \frac{PI_{il}}{\sqrt{PI_{il}^2 - PO_{jl}^2}} p_i - \frac{PO_{jl}}{\sqrt{PI_{il}^2 - PO_{jl}^2}} p_j \right), \quad i, j \in \mathcal{N},\ l \in 1..L \tag{2}$$

This linearization allows us to analyze large networks and many operational periods. For details regarding the linearization we refer to Rømo et al. [2]. In our presentation of the pressure-flow relationship we have assumed that our network is a directed graph such that the flow direction is pre-specified. The network may also contain some bi-directional pipelines. These are treated in our model as two separate pipelines with different flow directions. Then, a binary variable indicates which of the pipelines is operated in any given time period. If we were to include the network downstream of the market nodes, we would also need to consider network cycles.

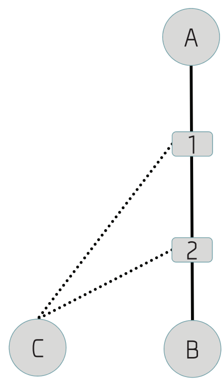



Fig. 2. A simple investment problem where a new market node $C$ shall be connected to an existing pipeline going from the field node $A$ to the market node $B$. There are two potential connection points, “1” and “2”.

### 2.3. Gas quality

Each production field produces gas with different quality (different composition). In the market nodes there are requirements on the gas quality, such as GCV and $CO_2$ content. In order to meet these requirements, one may process gas in on-shore terminals or blend gas from different sources such that the resulting gas blend meets the specifications.

Modeling the gas quality leads to the pooling problem (see, for instance, Haverly [14]), which represents a computationally hard problem. It arises when gas from different sources is mixed in a junction node and then transported in two or more connected pipelines. In this case, one must make sure that the quality of the natural gas is identical in all pipelines that exit the junction node.

To illustrate the pooling problem, we refer to Figure 3 and Equation (3). For ease of presentation, our example comprises just two components, $c_1$ and $c_2$, but the model is not limited to a specific number of gas components. For each of the pipelines exiting node $i$, the composition of the natural gas in the flows ($f$) must be the same (although the volumes may differ, of course). This can be formulated with the following set of equations:

$$\frac{f_{ij}^{c_1}}{f_{ik}^{c_1}} = \frac{f_{ij}^{c_2}}{f_{ik}^{c_2}}, \quad i \in \mathcal{N}, \ j,k \in O(i), \tag{3}$$

where the set $O(i)$ denotes the set of nodes connected to a pipeline exiting from node $i$. Each equation results in a bilinear expression, and this non-linearity is approximated through discretization using predefined split options.

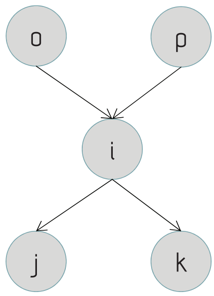



Fig. 3. A small network to illustrate the pooling problem. The gas composition in the pipeline from $i$ to $j$ must be the same as in the pipeline from $i$ to $k$.

## 3. Case study

To illustrate the use of the model we use a typical realistic investment case on the Norwegian Continental Shelf. The investment projects comprise new branch-offs, pipelines to connect to other existing infrastructure, and extra compression capacity. Figure 4 shows the network infrastructure with the existing elements in black color and potential investment projects in gray color.

In this investment case we consider in total 201 projects of which 177 are already existing and 24 are candidate projects that can be invested in. These 201 projects comprise 117 pipelines, 7 production nodes, 12 market nodes, and 65 junction nodes. Of the 65 junction nodes, there are 40 split nodes – that is, 40 network elements that increase the model complexity due to pooling. This network gives a quite accurate representation of the network on the Norwegian Continental Shelf although some aggregation and simplifications have been made. We have tested our model on this network with a time horizon of 11 years (11 investment periods) and one operational time period associated with each investment period. We show results for the model with and without constraints on $CO_2$ content in the gas that is delivered in the market nodes. The sizes of the corresponding MILP problems and their solution times using XPressMP 7.2 are given in Table 1. The dramatic increase in solution time is related to the complexity of the pooling problem as discussed above. When the analysis is performed without constraints on the quality in the market nodes (or intermediate nodes), we first solve the model as a uniform flow and then calculate the resulting quality in all network elements in a post-processing step, based on the gas composition in the fields and the flow pattern given by the model.

## 4. Conclusions

We presented an investment analysis model for natural gas infrastructure. The model finds the optimal network design given existing infrastructure and potential extension projects. We keep a system perspective

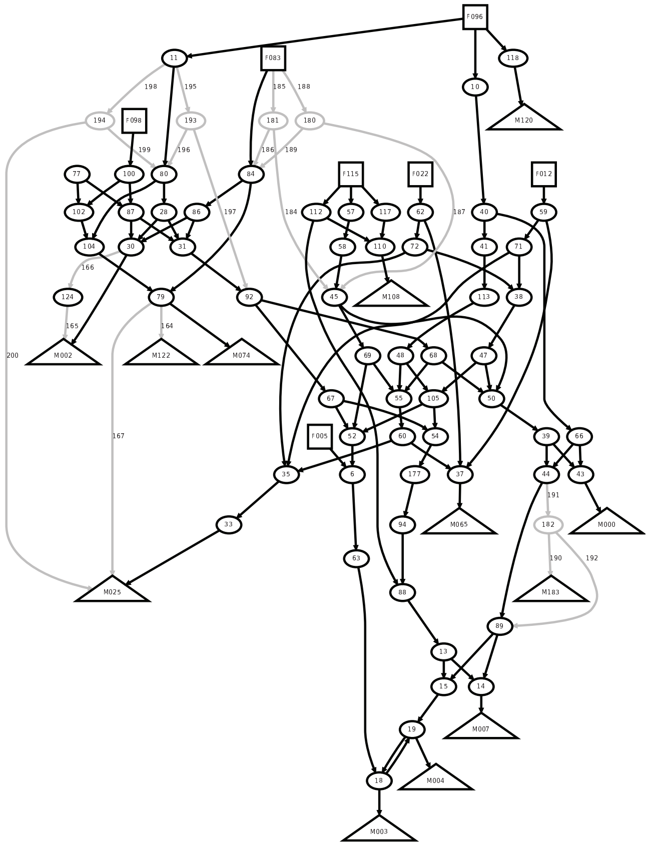



Fig. 4. A typical investment case on the Norwegian Continental Shelf. The squares are production fields, the triangles are markets while the ovals represent junction nodes. The arcs that link the nodes are pipelines.

where all projects are considered simultaneously, and where the operational analysis reflects the real physical situation in the whole network. We also illustrate the importance of including aspects such as system effects and modeling of gas quality for an investment analysis. The short-term variations in demand, prices and other parameters may influence the optimal design in terms of both security of supply and profitability. We handle these short-term variations by associating several operational time periods with each investment period to evaluate operational feasibility and profitability.

To address uncertainty in the investment planning we are also working on a stochastic model that can handle both long-term and short-term uncertainty. When the uncertainty is included in the analysis, the model size increases rapidly, requiring stronger assumptions for the analysis. As such, the two models will complement each other: The model presented in this paper can be used for a detailed analysis of the operations on the network, while the stochastic model will be well suited to analyze the impact of uncertainty – at the expense of operational detail.

Table 1. Numerical results from solving a realistic investment case.

| | No quality constraints | Limited $CO_2$ delivery |
|---|---|---|
| Rows | 101,054 | 327,467 |
| Columns | 15,065 | 145,052 |
| Binary variables | 9,553 | 17,473 |
| Gap | 0.01% | 0.5% |
| Solution time [s] | 22 | 5,886 |

## References


[1] B. Nygreen, M. Christiansen, K. Haugen, T. Bjørkvoll, and Ø. Kristiansen. Modelling Norwegian petroleum production and transportation. *Annals of Operations Research*, 82:251–267, 1998.

[2] F. Rømo, A. Tomasgard, L. Hellemo, and M. Fodstad. Optimizing the Norwegian natural gas production and transport. *Interfaces*, 39(1):46–56, 2009.

[3] K. T. Midthun. *Optimization models for liberalized natural gas markets*. PhD thesis, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim (Norway), 2007. Theses at NTNU, 2007:205.

[4] N. L. Ulstein, B. Nygreen, and J. R. Sagli. Tactical planning of offshore petroleum production. *European Journal of Operational Research*, 176(1):550–564, January 2007.

[5] M. Westphalen. *Anwendungen der Stochastischen Optimierung im Stromhandel und Gastransport*. PhD thesis, University Duisburg-Essen (Germany), 2004.

[6] J. Sullivan. The application of mathematical programming methods to oil and gas field development planning. *Mathematical Programming*, 42:189–200, 1988.

[7] D. Haugland, Å. Hallefjord, and H. Asheim. Models for petroleum field exploitation. *European Journal of Operational Reseach*, 37:58–72, 1988.

[8] S. A. van den Heever and I. E. Grossmann. A Lagrangean decomposition heuristic for the design and planning of offshore hydrocarbon field infrastructure with complex economic objectives. *Industrial & Engineering Chemistry Research*, 40:2857–2875, 2001.

[9] C. De Jonghe, B.F. Hobbs, and R. Belmans. Integrating short-term demand response into long-term investment planning. Technical report, Faculty of Economics, University of Cambridge, 2011.

[10] E. Sönmez, S. Kekre, A. Scheller-Wolf, and N. Secomandi. Strategic analysis of technology and capacity investments in the liquefied natural gas industry. Technical report, Carnegie Mellon Tepper School of Business, 2011.

[11] A. Werner, L. Hellemo, and K. Midthun. Ramona. Multi-stage stochastic programming in natural gas networks: Model specification and documentation for prototype implementation. Technical Report A21866, SINTEF, Trondheim (Norway), January 2012.

[12] K.T. Midthun, M. Bjørndal, and A. Tomasgard. Modeling optimal economic dispatch and system effects in natural gas networks. *Energy Journal*, 30:155–180, 2009.

[13] Gas Processors Suppliers Association. *Engineering Data Book*. Tulsa, 11th edition, 1998.

[14] C.A. Haverly. Studies of the behavior of recursion for the pooling problem. *ACM SIGMAP Bulletin*, 25:19–28, 1978.

# Paper III

Lars Hellemo, Kjetil Midthun, Asgeir Tomasgard, and Adrian Werner:

# Multi-Stage Stochastic Programming for Natural Gas Infrastructure Design with a Production Perspective

*Published in Ziemba, W. T., Wallace, S. W., Gassman, H. I. (Eds.), Stochastic programming — Applications in finance, energy, planning and logistics. Vol. 4 of World Scientific series in finance. World Scientific, 2012, pp. 259–288,* 

# Chapter 4

# Multi-Stage Stochastic Programming for Natural Gas Infrastructure Design with a Production Perspective


**Abstract:**

We present a multi-stage stochastic model that analyzes investments in natural gas fields and infrastructure. New projects are evaluated together with existing infrastructure and planned expansions. Several uncertain factors both upstream and downstream such as reservoir volumes, the composition of the gas in new reservoirs, market demand and price levels can influence the optimal decisions. The model focuses also on the impact of the sequencing of field developments and new infrastructure on the expected security of supply. In order to analyze all these aspects in one model, we propose a novel approach to scenario trees, combining long-term and short-term uncertainty. Dimensionality and solution times of realistic investment cases from the Norwegian Continental Shelf are discussed.


## 4.1 Introduction

The pipeline transport system on the Norwegian Continental Shelf (NCS) is the world's largest subsea gas transport system with 7800 km of pipelines. When new projects such as platforms, pipelines, compressors or processing facilities are considered, they should work well with existing and future infrastructure rather than be evaluated isolated from the total system. Current models for natural gas infrastructure development are often deterministic, where the distribution of uncertain parameters is replaced by mean values. There is, however, uncertainty in demand, prices, unplanned events, and gas quality and volumes from fields yet to be developed. Such uncertain parameters can be categorized with respect to the time horizon over which they vary. Some of the parameters vary from day to day, such as prices and demand in the markets, nominations in the long-term contracts, and unplanned events in the network. These variations will affect both the profitability of the operations in the natural gas value chain and the availability of network resources. Unplanned events can drastically reduce capacity in parts of the network and therefore cause large problems for the security of supply in the system. Examples of such unplanned events are production stops on fields, compressor failures, or reduced capacities of processing plants. By considering only average values for these parameters, important details may be lost through aggregation. For example, bottlenecks on the operational level during peak demand can be completely disguised when using only average demand values. On the long-term horizon there are also several uncertain parameters such as gas volumes in undeveloped reservoirs, the quality of the gas in such reservoirs, discoveries of new fields as well as long-term changes in price and demand levels.

The pipeline network exhibits system effects in the sense that the pressure and flow in one part of the network may influence the capacity in other parts of the network. To find an efficient routing of gas from production fields to markets we have to take into account relationships between pressure and flow and blending of gas with different quality. Consequently, these considerations are important both for evaluating how the proposed solutions work with existing infrastructure and for choosing solutions that are flexible enough to be reused for subsequent developments.

The infrastructure investment model we present here is part of the Ramona project[1] on production assurance and security of supply. We have developed tools for infrastructure design that value production assurance in addition to traditional profit or cost objectives. With the development of new gas field with low quality gas, this task has become more challenging as the quality issues require more processing capacity or blending with good quality gas to reach target specifications on gas quality in addition to volume requirements. With a sufficient number of scenarios we are able to use risk measures such as Conditional Value at Risk (CVaR) to ensure a high security of supply in the system.

Our main contribution is the development of a decision support system for infrastructure development that is very flexible with respect to the level of detail describing the physical processes in the natural gas value chain. The system is based on a multi-stage stochastic mixed-integer linear programming (MIP) model. This model can incorporate aspects such as pressure–flow relationship, multi-commodity flow, line pack, processing plants, and storages. At the same time we present a novel approach to analyzing both short-term and long-term uncertainty in the same model. We introduce a new scenario tree structure that combines strategic periods with operational periods, where the operational periods are representative for a given period. The structure allows a large number of such operational periods without causing more than a linear increase in the size of the scenario tree. We also present results from a realistic investment case on the NCS.

**Paper structure** We present an overview of the relevant literature in the following section. Uncertainty that needs to be incorporated in our stochastic MIP

[1]The Research Council of Norway, project number 175967

model is discussed in section 4.3. In section 4.4, we describe the main elements of the mathematical model formulation. Finally, we present a case study with preliminary computational results in section 4.7. Section 4.8 concludes the paper. The notation will be introduced when needed in the mathematical formulation, while we refer to section 4.A for a full overview.

## 4.2 Literature review

In the literature, we find several examples of models that analyze different aspects of offshore petroleum investments. This is not surprising given the large risks and costs associated with such projects.

**Strategic models** There exist a number of deterministic investment models, and early overviews can be found in Sullivan [1988] and Haugland et al. [1988]. Nygreen et al. [1998] present a multi-period MIP model used by the Norwegian Petroleum Directorate. The model is employed for the investment planning of fields in the North Sea which contain a mixture of oil and gas. In van den Heever and Grossmann [2001], a model for design and planning of offshore field infrastructure projects is presented. The model is a multi-period mixed-integer nonlinear programming model (MINLP) and incorporates complex fiscal rules such as tariff, tax and royalty calculations. The net present value of projects is discussed in the light of these fiscal rules. There are also some models which incorporate uncertainty. Jørnsten [1992] presents an integer model for sequencing offshore oil and gas fields, where the objective is to maximize total economic benefit. Haugen [1996] develops a stochastic dynamic programming model to analyze a supplier's problem of scheduling fields and pipelines in order to be able to meet contractual agreements. The uncertainty in this model is in the resources (production profiles). In Jonsbråten [1998], a stochastic MIP model for optimal development of an oil field is presented. The objective of the model is to maximize the expected net present value of the oil field given uncertain future oil prices. Goel and Grossmann [2004] present a stochastic MIP model for the planning of offshore gas field developments. The expected net present value is maximized under uncertainty in reserves. A multi-stage investment model with decision dependent uncertainty is presented by Tarhan et al. [2009] along with a branch-and-bound solution method for non-convex mixed-integer nonlinear sub problems. They take into account nonlinear reservoir models and gradually revealed uncertainties concerning initial flow rate, recoverable volume and other characteristics of the reservoirs.

**Operational models** We use a value chain approach where we consider elements in the natural gas value chain from production field to market and optimize the resulting system. This approach has become even more valuable and important after the liberalization process, which meant an increase in flexibility for the participants in the value chain. In Ulstein et al. [2007], planning of offshore petroleum production is studied on a tactical level. The model has a value chain approach where production plans, network routing, processing of natural gas and sales in the markets are considered. In addition, multi-commodity flows and quality restrictions in the markets are taken into account. Pressure constraints in the network are, however, not included. The nonlinear splitting for chemical processing is linearized with binary variables. The resulting model is a mixed-integer programming model. Selot et al. [2008] presents an operational model for production and routing planning in the natural gas value chain. The authors combine a detailed infrastructure model with a complex contractual model but do not include a market for natural gas. The infrastructure model comprises nonlinear equations relating pressure and flow in wells and pipelines, multi-commodity flows and contractual agreements (delivery pressure and quality of the gas). The contractual model is based on a set of logical conditions for production sharing and customer requirements. The combined model is a mixed-integer nonlinear programming model. Li et al. [2011] present a global optimization approach to a stochastic pooling problem. They solve the integrated design and operations problem of industrial networks such as natural gas networks where the quality is stochastic.

The transportation of natural gas is one of the key elements when studying the natural gas industry and is paramount for analyzing the design of the network. Tomasgard et al. [2007] and Rømo et al. [2009] present a linearization of the Weymouth equation which enables the analysis of large networks and stochastic problems. There is a large number of publications with a technical approach to gas transportation. The models are detailed and accurate in their description of the physics of gas transportation, such as transient flow and interaction with compressors. A discussion of transient flows is given in Kelling et al. [2000], while the homepage of the Pipeline Simulation Interest Group (www.psig.org) gives a comprehensive overview on modeling, simulation and optimization of natural gas flows. Midthun et al. [2009] shows that it is difficult, if not impossible, to determine appropriate static capacities in a natural gas network. The system effects are discussed and a framework for economic analysis in natural gas networks is provided. Problems related to transport booking are studied by, e.g., Kalashnikov et al. [2010] who model the transport of gas by a transport company and a pipeline operating company as a stochastic bi-level problem.

**Strategic models with operational variability** In order to ensure that short-term fluctuations and peak demand situations can be accounted for in the system design, we include a representation of operational flexibility in a strategic investment model. Not much work can be found which focuses on both aspects at the same time. Schütz et al. [2009] include short-term variations in a strategic model for the Norwegian meat industry. De Jonghe et al. [2011] use an equilibrium model to study generation expansion. They integrate the short-term demand response in their strategic model and discuss the effects on flexibility of the generation capacity. The approach, however, considers only a one-period static model. Sönmez et al. [2013] analyze technology choice in LNG transport and discuss the impact of using a stochastic model for LNG throughput. They show that operational flexibility is important in order to cope with short-term variations and that is has a significant impact on profitability.

## 4.3 Uncertainty

### Uncertainty in the context of our model

The investment problem we address contains uncertainty both on the long-term and the short-term horizons. In the *long term*, there is considerable uncertainty regarding future energy prices and demand. It is driven by the uncertainty about the future mix of energy sources, technology shifts, and imports from other geographical regions. Other important aspects are the gas composition and reservoir levels of new fields yet to start production. The uncertainty in gas composition poses challenges related to keeping the delivered gas quality within specifications, while the volume uncertainty will influence optimal pipeline designs. Also the gas quality specifications may not be perfectly known, for example, the maximum allowed content of contaminants such as $CO_2$ might be lowered in the future.

In the *short term* there is uncertainty in the prices in each market. These prices have a seasonal profile, in addition to high variability and large spikes. Also the demand in the long-term contracts is uncertain in the short term as the customers have the flexibility to nominate volumes within certain levels. In addition, parts of the production or transport capacity may be reduced during an unforeseen *event*. Such events can last much shorter than an operational period but may block network elements completely. We model this by stating how much of the network element's capacity is available over the operational period.

It is important to deal with the uncertainty in an appropriate way to ensure a robust and flexible infrastructure design allowing profitable operations under various, also adverse, conditions. For the investment analysis we need an optimization horizon of several years, typically between 20 and 50. The operational analysis, on the other hand, requires a much finer time resolution, such as days or hours. Combining the long time horizon, the fine time resolution and the uncertainty in both the long and the short term into a common paradigm, the size of a traditional scenario tree will explode. It is, therefore, worthwhile to investigate alternative approaches. One way of handling the effects of the short-term uncertainty is to iteratively run the investment model and a short-term model that incorporates the operational aspects (see, for instance, Myklebust [2010]). This approach poses, however, large challenges with respect to coordination and convergence of the solution. We propose a different approach that allows us to analyze long time horizons consisting of several years while still taking into account the effects of operational decisions and uncertainty in the same model. The following section describes our approach in more detail.

## The multi-horizon scenario tree

We have developed a new scenario tree structure to represent both short-term and long-term uncertainty in a unified framework. To the best of our knowledge, such a framework has not been presented before. The approach includes key aspects of operational variability and uncertainty while keeping the size of the scenario tree manageable. Our approach is based on multi-horizon scenario trees, where the long-term and short-term horizons are handled differently but in a unified framework. The strategic nodes span long time periods such as years, while the operational nodes typically represent days. All decisions related to investments and network design is made in the strategic nodes, while all decisions related to operating the network are made in the operational nodes. This distinction between strategic and operational tree nodes in a multi-horizon tree is stronger than in a traditional scenario tree.

The operational nodes represent sub-trees in the scenario tree. In these sub-trees, the leaf nodes are always operational nodes while the root node will be an investment node. At each investment node we can then add several sub-trees. This way we can test the infrastructure in the connected investment node on an operational level. This connection is illustrated in Figure 4.1, where the strategic node is given as a square while the operational nodes are oval. The figure also shows that we distinguish between two different types of operational scenarios. The first type is included in the objective function of our model and provides us with the costs and revenues from operating the network. The benefits from adding these nodes are the greater detail we can add to the calculation of the profits with a finer time resolution as well as the possibility of adding several scenarios (using the expected value in the objective function). In addition, we have a second type of operational nodes (on the far right in Figure 4.1) that are used to find the security of supply (or production assurance) in the network. These operational nodes represent extreme scenarios that include events with negative impact on operations that may occur with a small probability as well as scenarios representing the normal operations with an associated high probability. These scenarios are included to enforce the satisfaction of production assurance targets given the occurrence of such events. The results from these extreme scenarios do not contribute to the objective function in terms of revenue or operational expenses. Rather, production assurance targets may be enforced through hard constraints or violations penalized in the objective function. This way it is possible to influence the infrastructure solutions proposed by the model to be more robust against failures, also in situations that may not directly reduce overall profitability by much due to the short duration of the events.

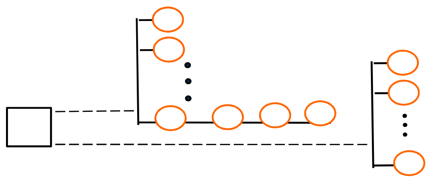


**Figure 4.1:** Coupling of strategic nodes (square) and the two different groups of operational subtrees (ovals).

The branching in our scenario tree is performed on the strategic nodes only, as illustrated in Figure 4.2. That is, the only connection between the operational nodes and the next strategic node is through the expected profits from operations (in addition to the expected production which will change the reservoir levels). This way, we can incorporate operational details and short-term uncertainty allowing us to thoroughly test design decisions, and at the same time use traditional scenarios expressing the long-term uncertainty, thus analyzing long time horizons without an explosion in problem size. This corresponds to a contingent scenario analysis for the operational problem for each strategic node and is, in general, a relaxation from the real information structure which would include branching in the main tree also at the operational level. Such a relaxation allows us to take into account the operational variability to some degree without suffering from the immense growth of the scenario tree needed for a model with frequent branching in a short-term horizon included in a strategic model. Observe that, if both the long-term uncertainty is independent of previous short-term realizations and the strategic decisions are independent of previous short-term decisions, the scenario tree structure actually reflects the exact information structure.

We denote the strategic scenarios of the tree by $s \in \mathcal{S}$ and the operational scenarios by $o \in \mathcal{S}^{Op}$ with the corresponding root node $R(o) \in \mathcal{S}$. The set $\mathcal{S}_{ts}^{OpSub}$ denotes all operational scenarios in the sub-tree associated with a strategic node in time period $t \in \mathcal{T}$ and scenario $s \in \mathcal{S}$. The operational (leaf) nodes $o \in \mathcal{S}^{Op}$

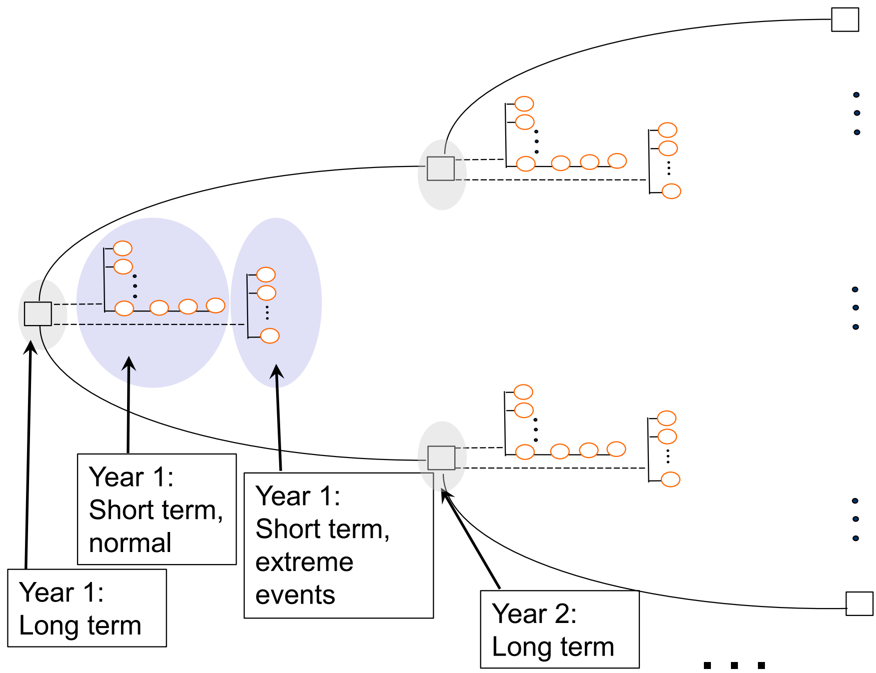



**Figure 4.2:** Multi-horizon scenario tree combining long- and short-term uncertainty for investment and operational decision nodes. The first group of operational time periods denotes the representative normal days, while the second group represents events.

have no descendant nodes in $\mathcal{S}$, but all have a root node $R(o) \in \mathcal{S}$. We sometimes refer to $o$ as the operational scenario contingent on $R(o)$. To be able to explore effects of time bindings (for example, when studying line pack or storages), we may consider groups $g \in \{1, ..., G\}$ consisting of several consecutive operational nodes. As discussed earlier in this section, we can distinguish two disjoint subsets: The set $\mathcal{G}_t^{norm} \subseteq \{1, ..., G\}$ defines all such groups representing normal operations in strategic period $t$ while the set $\mathcal{G}_t^{extreme} \subseteq \{1, ..., G\}$ denotes all groups of nodes representing extreme situations in that period. Note that such an aggregation into groups of potentially varying length requires a scaling factor $\gamma_g$ to include operational terms into the objective function in the right order of magnitude.

Note that, although not illustrated above, the approach can easily accommodate also more complex tree structures such as multistage operational sub-trees, branching at operational nodes within sub-trees. This is conceivable in particular for the sub-trees representing normal operating conditions.

## 4.4 Model structure

### Time structure, investment and operational decisions

We assume that investment decisions are made at the beginning of an investment period $t \in \mathcal{T}$ while operational decisions are made at the operational time periods $d \in \mathcal{D}_t$ associated with the investment period $t$. Obviously, the sets of investment and operational periods are disjoint. In the remainder of this paper, we denote investment time periods often as "years" and operational time periods as "days". This is done purely for reasons of notational clarity. However, an investment time period may also cover, for example, several calendar years and varying time granularity can easily be taken into account.

### Modularity

We employ a two-layered model design: A fully working *core model* comprises basic formulations of all necessary aspects while a number of separate *extension modules* are designed to improve functionality and/or level of detail and to allow for more sophisticated modeling of selected areas.

This modular approach enhances the flexibility and generality of the mathematical model because it allows conducting various analyses with different focus without compromising the underlying model and its implementation. It enables also a stepwise extension of the model as new functionality can be added as required. The level of detailed can be chosen in a trade off between richness in modeling and computational complexity.

The flexibility in modeling will in some cases lead to a less tight formulation than what might be possible for a more specialized model, and we rely on the effectiveness of the presolve step of available commercial MILP solvers.

#### Core model

In the *core model*, the objective is to maximize net present value including cash flow from operations and penalties for insufficient production assurance. Main constraints included in the core model are the following:

- Investment decision modeling
- Demand satisfaction in the market nodes
- Mass balances in all network nodes

- Flow capacity limits in pipelines
- Production capacity limits
- Reservoir modeling

The main decisions concern the single project investments (network elements and alternatives: nodes and pipelines) and the flow through each of the elements invested in.

### Extension modules

At present, the model contains the following modules:
**Production assurance modeling.** We penalize deviations from production assurance requirements in the objective function. The extension adds two sets of constraints. The first set ensures that the probability of days with non-satisfactory production assurance is not too large. The second set limits the expected value of such underperformance by applying risk measures such as Conditional Value at Risk (Rockafellar and Uryasev [2000]).
**Pressure dynamics.** This module allows to model pipelines with pressure dynamics. It expands on the core functionality of maximum flow constraints. We include upper and lower limits on pressures in each node and in pipelines, relationships between pressures into and out of a node as well as in ingoing and outgoing pipelines (Weymouth equation), and contracted pressure in market (customer) nodes. These features are described in section 4.6.
**Reservoir modeling.** The core model uses a simple tank implementation for each reservoir where a certain maximum volume is given which is exhausted when the accumulated production has reached this level. This is replaced in the more advanced module by an approach similar to a production profile varying over time. The single production volumes are limited with respect to how much has been produced so far. Details on this module are given in section 4.6.
**Multi-component flow modeling.** The extension module introduces variables and constraints for modeling heterogeneous flows, see section 4.6. In most cases, this module will considerably increase the computational complexity.
**Relations between projects** This module introduces constraints which specify that, from a given group of projects, at most one alternative (or a certain minimum/maximum number) can be started. Constraints may ensure the startup or shutdown of a certain number of projects from one group before the startup/ shutdown of projects from another group or vice versa. Another relation controls the replacement of one (system of) project(s) by another. Mutually exclusive (systems of) projects may also be specified.
**Processing nodes** The module adds the option to extract some components at

dedicated processing plants in the network and to sell them in separate markets.

As the needs of the analysts develop, additional modules may be included, for example modules for line pack, storage or bidirectional pipelines.

## 4.5 Core model

### Objective function

The main objective in the model is to maximize the expected net present value $NPV$, arising from infrastructure investments and their operation, taken over the values in the single long-term scenarios $s \in \mathcal{S}$. The parameter $Prob_{ts}$ denotes the absolute probability of the strategic tree node at time period $t \in \mathcal{T}$ in scenario $s \in \mathcal{S}$. Included in the revenue, $Rev_{ts}$, are both potential salvage value from obsolete infrastructure and income from gas sales while costs $Cost_{ts}$ comprise both investment costs and operational expenses at each time period $t \in \mathcal{T}$.

$$NPV = \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{T}} Prob_{ts} \delta_t \left( Rev_{ts} - Cost_{ts} \right) \tag{4.1}$$

The single objective function terms are discussed in more detail in section 4.5 below.

Cash flows are discounted in each investment period $t \in \mathcal{T}$ at a discount rate $\sigma$, using the discount factor $\delta_t$,

$$\delta_t = \frac{1}{(1+\sigma)^t}$$

### Investment constraints

All investment decisions are characterized by certain common properties such as startup, shutdown, capacities (possibly varying over time), and costs. We model these properties by means of a common class, denoted as *projects*, describing the common structure of all network elements such as fields, pipelines, or nodes. For the considered gas transport network, projects comprise pipelines, production nodes (fields), transport nodes (junctions, processing plants, etc.) and consumption nodes (markets). Projects can be either existing or new projects. Summarizing, a project $p \in \mathcal{P}$ is any option to invest in a network element under scenario $s$ which, at time $t$, is characterized by the *binary* variables $start_{pts}$, $stop_{pts}$, and $prd_{pts}$ defining the project's status: start, stop, and production, respectively.

The sets $\mathcal{I}(p)$ and $\mathcal{O}(p)$ denote the projects which are located directly upstream and downstream of a project $p$. For network nodes $p$, these sets $\mathcal{I}(p)$, $\mathcal{O}(p)$ mean the pipelines leading to and from $p$. For pipelines $p$, they denote the input and outlet nodes.

The startup date of *existing* projects $p \in \mathcal{P}^{ex}$ is set to the start of the optimization horizon, $start_{p1s} = 1$. Each *new* project $p \in \mathcal{P}^{New}$ can start up at most once during the optimization horizon and has a certain time window $Start_p^e, ..., Start_p^l \in \mathcal{T}$ for startup,

$$\sum_{t=Start_p^e...Start_p^l} start_{pts} \leq 1$$

Each project can be shut down at most once during the optimization horizon,

$$\sum_{t=Start_p^e}^{T} stop_{pts} \leq 1$$

Production can only take place if a project has started up and has not been shut down yet. If there is a delay $\Delta_p$ between the time when deciding to start a *new* project $p \in \mathcal{P}^{New}$ and the time it is available for production, startup decisions after $t - \Delta_p$ do not need to be taken into account (but shut-down decisions should).

$$\forall t \in \{Start_p^e + \Delta_p...T\} : \; prd_{pts} \leq \sum_{\tau=Start_p^e}^{t-\Delta_p} start_{p\tau s} - \sum_{\tau=Start_p^e}^{t} stop_{p\tau s}$$

To avoid an undue postponement of a project shutdown to reduce the net present value of shutdown costs, the following inequality enforces the immediate stop of a project after production halts.

$$\forall t \in \{Start_p^e, ..., T\} \; : \quad prd_{pt-1s} - prd_{pts} \leq stop_{pts}$$

The binary variable $stprd_{pt\tau s}$ indicates whether a project $p$ which started at $\tau \in \mathcal{T}$ is producing at $t \in \mathcal{T}$.

$$prd_{pts} = \sum_{\substack{\tau \in \mathcal{T} \\ \tau \leq t-\Delta_p}} stprd_{pt\tau s}$$

$$\forall \tau \in \{Start_p^e, ..., t - \Delta_p\} : \; stprd_{pt\tau s} \leq start_{p\tau s}$$

**Time series of parameters – profiles** All projects can be described by parameters that change over time. For example, production capacity may be low at the beginning due to necessary adjustments, high during peak production and low again at the end when reservoirs are exhausted. We call these series of values "profiles" and distinguish two types: Floating profiles depend on the start date of a project and are relative to this start date. They are denoted by $\Pi_{float}$. An example is a capacity profile. Fixed profiles refer to calendar dates and are independent of project start. They are denoted by $\Pi_{fix}$. Examples are market demand or prices. See Appendix 4.A for a complete list of profiles used in the model.

## Operational constraints

**Mass balances** We have modeled the flow *through* a network element in a unified way although it may have slightly different meanings for different network elements in real life. For example, the flow through production nodes $p \in \mathcal{P}^{Prod}$ is the total amount produced in that node in the considered period, the flow through consumption nodes $p \in \mathcal{P}^{Cons}$ is the total amount sold in that node in that period.

The following constraints ensure the mass balances in the network for each project $p$ at each operational period $d \in \mathcal{D}_t$, $t \in \mathcal{T}$ and each operational scenario $o \in \mathcal{S}^{Op}$. For production nodes $p \in \mathcal{P}^{Prod}$, the amount produced must equal the outflow into the network (4.2a), the amount sold in a market node $p \in \mathcal{P}^{Cons}$ must equal the inflow from the network (4.2b) and the flow into a transport node $p \in \mathcal{P}^{Transp}$ must equal the flow out of it (4.2c). For pipelines $p \in \mathcal{P}^{Pipe}$, no explicit equations are necessary as they have exactly one direct upstream and downstream project. Note that the flow through a transport node $p \in \mathcal{P}^{Transp}$ is defined only implicitly through the mass balance (4.2c). It is specified in equation (4.2d) in order to state, e.g., the lower and upper flow bounds (4.3).

$$flow_{ptdo} = \sum_{j \in \mathcal{O}(p)} flow_{jtdo} \tag{4.2a}$$

$$flow_{ptdo} = \sum_{j \in \mathcal{I}(p)} flow_{jtdo} \tag{4.2b}$$

$$\sum_{i \in \mathcal{I}(p)} flow_{itdo} = \sum_{j \in \mathcal{O}(p)} flow_{jtdo} \tag{4.2c}$$

$$flow_{ptdo} = \sum_{j \in \mathcal{I}(p)} flow_{jtdo} = \sum_{j \in \mathcal{O}(p)} flow_{jtdo} \tag{4.2d}$$

**Flow bounds** Capacity bounds on the flow through a project depend on this project's age and are given through floating profiles. These profiles indicate the physically available capacity and are defined on a strategic level. For projects

$p \in \mathcal{P}$ with startup decision made at $\tau$, flow in the operational periods $d \in \mathcal{D}_t$ is bounded by the value of the floating profiles at a position relative to the project start, taking into account possible delays $\Delta_p$. In addition, a stochastic parameter $CapPerc_{ptdo}$ denotes the percentage of that capacity which is actually available in the operational period $d \in \mathcal{D}_t$ under scenario $o \in \mathcal{S}^{Op}$. This allows us to model unforeseen events affecting network capacity, and the input parameters' consistency between reduced upper and lower bounds must be ensured.

$$flow_{ptdo} \leq CapPerc_{ptdo} \sum_{\substack{\tau \in \mathcal{T} \\ \tau \leq t}} \Pi^{MaxFlow}_{pt-\tau-\Delta_p+1} \cdot stprd_{pt\tau s} \tag{4.3a}$$

$$flow_{ptdo} \geq \sum_{\substack{\tau \in \mathcal{T} \\ \tau \leq t}} \Pi^{MinFlow}_{pt-\tau-\Delta_p+1} \cdot stprd_{pt\tau s} \tag{4.3b}$$

The amount sold at a consumption node $p \in \mathcal{P}^{Cons}$ cannot exceed the contracted volume in this node during that time period and scenario. In the presence of production assurance constraints, the sales volume bound $Vol_{ptdo}$ is interpreted as contracted volume, in the absence of such constraints as possible sales in spot markets.

$$flow_{ptdo} \leq Vol_{ptdo}$$

**Reservoir constraints** In the core model, reservoir volumes are modeled through a simple tank implementation: the parameter $ResMax_p$ states an upper bound on the totally available volume in the reservoir $p \in \mathcal{P}^{Prod}$ for operational scenarios $o \in \mathcal{S}^{Op}$.

$$\sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}^{norm}_t} \gamma_g \sum_{d=d^0_g}^{d^L_g} flow_{ptdo} \leq ResMax_p \tag{4.4}$$

Additionally, there is a given yearly production rate on the field which should not be exceeded in any year $t \in \mathcal{T}$,

$$\sum_{g \in \mathcal{G}^{norm}_t} \gamma_g \sum_{d=d^0_g}^{d^L_g} flow_{ptdo} \leq ResProdRate_{pt}$$

## Objective function terms

### Investment costs

**Costs** Projects have costs that, like production characteristics, depend on the age of the project while other costs depend on calendar dates. It is therefore reasonable to operate with both floating and fixed profiles.

Costs incurred by investment decisions are startup, fix investment and removal costs,

$$Cost_{ts}^{Inv} = \sum_{p \in \mathcal{P}} \left( Cost_{pts}^{invst} + Cost_{pts}^{invfix} + Cost_{pts}^{invrem} \right)$$

Startup costs may depend on the startup point of time, for example it may be cheaper to start an extension project soon than to wait until the additional capacity is needed.

$$Cost_{pts}^{invst} = \Pi_{pt}^{invst} \cdot start_{pts}$$

Fixed investment costs may occur at investment periods during the project's lifetime and depend on the age of the project,

$$Cost_{pts}^{invfix} = \sum_{\tau=1}^{t} \Pi_{pt-\tau+1}^{invfixC} \cdot stprd_{pt\tau s}$$

Removal costs are incurred by the stop of a project $p$ and may depend on both the age of the project and the point of time, i.e. they have a fixed and a floating component.

$$Cost_{pt}^{invrem} = \Pi_{fix,pt}^{invremC} \cdot stop_{pts} + \sum_{\tau=1}^{t} \Pi_{float,pt-\tau+1}^{invremC} \cdot stop_{p\tau s}$$

In addition to costs associated with stopping we allow a positive salvage value where applicable.

### Operational expenditure and revenue

**Costs** Total operational costs during investment period $t \in \mathcal{T}$ under scenario $s \in \mathcal{S}$ are composed of operational expenditure, fixed and variable costs arising for projects $p \in \mathcal{P}$ in all operational periods (days) $d \in \mathcal{D}_t$ in this investment period. These costs are representative costs, as only a few operational periods are considered, and they are scaled up to a yearly basis.

$$Cost^{Op}_{ts} = \sum_{p \in \mathcal{P}} \Bigg( Cost^{opex}_{pts} + \sum_{o \in \mathcal{S}^{OpSub}_{ts}} \sum_{g \in \mathcal{G}^{norm}_{t}} \gamma_g \sum_{d=d^0_g}^{d^L_g} Prob_{do} \Big( Cost^{opfix}_{ptdo} + Cost^{opvar}_{ptdo} \Big) \Bigg)$$

Operational expenditure (opex) is independent of the production volume or flow but can vary over the lifetime of the project and depends, hence, on the project's age. It arises only when the project is actually active (i.e. projects which have started up but are not producing yet do not generate opex).

$$Cost^{opex}_{pts} = \sum_{\tau=1}^{t} \Pi^{opex}_{pt-\tau-\Delta_p+1} \cdot stprd_{pt\tau s}$$

Fixed operational costs depend neither on the age of a project nor on the flow but on the operational date $d \in \mathcal{D}_t$ and the associated investment period $t$,

$$Cost^{opfix}_{ptds} = \Pi^{opfix}_{ptd} \cdot prd_{pts}$$

Variable operational costs depend on the flow through that project and are independent of the project age; they may also include costs of production factors or similar,

$$Cost^{opvar}_{ptds} = \Pi^{opvar}_{ptd} \cdot flow_{ptdo}$$

**Revenue** Revenue from operations comes from sales at market nodes at operational periods $d \in \mathcal{D}_t$ under scenarios $o \in \mathcal{S}^{OpSub}_{ts}$ illustrating normal operating conditions. This revenue is considered representative revenue, and the scaling factor $\gamma_g$ is applied. For the core model we assume there is no sale of by-products at any node $p \in \mathcal{P}^{Prod} \cup \mathcal{P}^{Transp} \cup \mathcal{P}^{Cons}$. Sales price profiles at the single market nodes $p \in \mathcal{P}^{Cons}$ depend on the point of time and may be stochastic,

$$Rev^{Op}_{ts} = \sum_{o \in \mathcal{S}^{OpSub}_{ts}} \sum_{g \in \mathcal{G}^{norm}_{t}} \gamma_g \sum_{d=d^0_g}^{d^L_g} Prob_{do} \sum_{p \in \P^{Cons}} Price^{Spot}_{ptdo} \cdot flow_{ptdo}$$

**Production assurance** Production assurance is studied in terms of *deliverability*, referring to deliveries at market nodes in each operational period $d \in \mathcal{D}_t$ under scenario $o \in \mathcal{S}_{ts}^{OpSub}$ and period $d \in \mathcal{D}_t$ in strategic scenario $s \in \mathcal{S}$ and period $t \in \mathcal{T}$. It is defined as *(contracted volumes - deviations)/contracted volumes* where *deviations* means the difference between contracted and delivered volumes. The measure is aggregated over all markets $p \in \mathcal{P}^{Cons}$,

$$Del_{tdo} = \frac{\sum\limits_{p \in \mathcal{P}^{Cons}} flow_{ptdo}}{\sum\limits_{p \in \mathcal{P}^{Cons}} Vol_{ptdo}}$$

For the core model, we achieve sufficient production assurance by penalizing negative deviations from the target production assurance $PATarget_t$. The actual production assurance is realized through the investment and all operational decisions in a given investment period $t \in \mathcal{T}$, scenario $s \in \mathcal{S}$,

$$Dev_{ks} = \max \left\{ 0, PATarget_t - \sum_{o \in \mathcal{S}_{ts}^{OpSub}} \sum_{d=d_g^0}^{d_g^L} Prob_{do} \frac{Del_{tdo}}{d_g^L - d_g^0 + 1} \right\}$$

We then include penalty terms

$$\delta_t Pen_t \sum_{g \in \mathcal{G}_t^{extreme} \cup \mathcal{G}_t^{norm}} Dev_{ks}$$

for each time period $t \in \mathcal{T}$ and scenario $s \in \mathcal{S}$ into the objective function (4.1).

## 4.6 Selected extension modules

As discussed in section 4.4, extension modules provide more sophisticated modeling than the core model and can be switched on as required for various analyses. In the following, we describe some relevant modules in more detail, in particular, advanced pressure modeling, reservoir modeling, and multi-component flow / quality modeling.

### Pressure dynamics

For this module it is convenient to consider a set $\mathcal{P}^{Nodes}$ comprising all network elements other than pipelines. The current model does not include a detailed compressor model. Instead we allow pressure increase within capacity bounds for projects representing a compressor.

Consider a pipeline $p \in \mathcal{P}^{Pipe}$ connecting network nodes $i$ and $j$ which have the respective pressures $pres_{itd}$ and $pres_{jtd}$. Simple flow constraints as in the core model are used for short pipelines where the pressure drop is not important for the flow calculation,

$$pres_{ptdo}^{out} = pres_{ptdo}^{in}$$

For all other pipelines, a linearization of the Weymouth equation (Gas Processors Suppliers Association [1998]) gives an upper bound on the flow $flow_{ptdo}$ coming from $i \in \mathcal{I}(p)$ and going out through $j \in \mathcal{O}(p)$ at each linearization breakpoint $b \in \mathcal{B} = \{1, \ldots, B\}$.

$$flow_{ptdo} \leq K_p^W \frac{P_{ib}}{\sqrt{P_{ib}^2 - P_{jb}^2}} pres_{itdo} - K_p^W \frac{P_{jb}}{\sqrt{P_{ib}^2 - P_{jb}^2}} pres_{jtdo}$$

The parameter $K_p^W$ denotes the pipeline specific Weymouth constant while $P_{ib}$ and $P_{jb}$ are approximations of the pressures at the in- and outlet points of pipeline $p$ at the linearization breakpoint $b \in \mathcal{B} = \{1, ..., B\}$.
Additionally, the pressure $pres_{itdo}$ in the node $i$ needs to be within some minimum and maximum bounds,

$$\underline{P_i} \leq pres_{itdo} \leq \overline{P_i}$$

Such bounds are defined, e.g., through design parameters of the network and contractual agreements and may also have a time index, $\underline{P_{it}}$, $\overline{P_{it}}$, $t \in \mathcal{T}$. See Tomasgard et al. [2007] for further details.

## Reservoir modeling

While the basic reservoir modeling in constraint (4.4) only restricts the *total* or aggregated volume produced on a field, the advanced reservoir constraints set upper bounds on the production volumes based on how much has already been produced from that field. Observe that these constraints do not apply for groups $g \in \mathcal{G}_t^{extreme}$ of operational periods testing the viability of decisions during more extreme scenarios.
The production rate at each time period $t$ should be at most a certain percentage of the volume still available in the reservoir. This describes a floating profile of yearly production rates. Over time there may be several such percentages, resulting in a complex profile structure. A procedure is employed to derive a set

of $M$ linear constraints for projects $p \in \mathcal{P}^{Prod}$,

$$\sum_{g \in \mathcal{G}_t^{norm}} \gamma_g \sum_{d=d_g^0}^{d_g^L} \mathit{flow}_{ptdo} \leq Prod_{pm}^{Abs} + Prod_{pm}^{Rel} \cdot \mathit{flow}_{pts}^{A}$$

These constraints describe the production profile as a convex set where the production capacity at any time $t$ depends on the accumulated production $\mathit{flow}_{pts}^{A}$ until $t$ which is given by

$$\mathit{flow}_{pts}^{A} = \sum_{\tau=1}^{t} \sum_{g \in \mathcal{G}_t^{norm}} \gamma_g \sum_{o \in \mathcal{S}_{ts}^{OpSub}} Prob_{do} \sum_{d=d_g^0}^{d_g^L} \mathit{flow}_{ptdo}$$

The accumulated production must not exceed the total volume to be produced from the reservoir,

$$\mathit{flow}_{pts}^{A} \leq MaxProd_p$$

The absolute and relative coefficients $Prod_{pm}^{Abs}$ and $Prod_{pm}^{Rel}$ $(m = 1, ..., M)$ and the limits $MaxProd_p$ are found based on user specified yearly production rates $ProdRate_{pt}$.

## Quality / Multi-component flow modeling

The fractions of the component flows going out of a split node $p \in \mathcal{P}^{Transp}$ into the different downstream pipelines $p_1, p_2 \in \mathcal{O}(p)$ must be equal for all components $c_1,\ c_2 \in \mathcal{C}$ in the flow:

$$\frac{\mathit{flow}_{p_1 c_1}^{C}}{\mathit{flow}_{p_2 c_1}^{C}} = \frac{\mathit{flow}_{p_1 c_2}^{C}}{\mathit{flow}_{p_2 c_2}^{C}}$$

This results in a bilinear expression which is approximated through a linearization using pre-defined split options $\alpha_{p_1 h}$, $h = 1, ..., H$, to find the share of the flow through a pipeline $p_1$. These pre-defined split options can be found based on, for example, the split percentages from a single-component run of the model or a fixed set of split options (Tomasgard et al. [2007]).

The following constraints define the flow $\mathit{flow}_{p_1 ctdo}^{C}$ of component $c \in \mathcal{C}$ through this pipeline at day $d \in \mathcal{D}_t$, year $t \in \mathcal{T}$, operational scenario $o \in \mathcal{S}_{ts}^{OpSub}$:

$$\begin{aligned} \forall\ h \in \{1, ..., H\}:\ \mathit{flow}^S_{p_1chtdo} &\leq \alpha_{p_1htdo} \cdot \mathit{flow}^C_{pctdo} \\ \mathit{flow}^C_{p_1ctdo} &= \sum_{h\in\{1,...,H\}} \mathit{flow}^S_{p_1chtdo} \\ \sum_{h\in\{1,...,H\}} \mathit{flow}^S_{p_1chtdo} &\leq UB_{p_1tdo} \cdot \lambda_{p_1htdo} \\ \alpha_{p_1htdo} \cdot \mathit{flow}^C_{pctdo} &\leq \mathit{flow}^S_{p_1ctdo} + UB_{p_1tdo} \cdot (1 - \lambda_{p_1htdo}) \end{aligned}$$

With an upper bound

$$UB_{p_1tdo} = CapPerc_{p_1tdo} \sum_{\substack{\tau\in\mathcal{T} \\ \tau\leq t}} \Pi^{MaxFlow}_{p_1t-\tau-\Delta_{p_1}+1}$$

on the capacity of pipeline $p_1$, the auxiliary binary decision variables $\lambda_{p_1htdo}$ for the choice of split options $h \in \{1, ..., H\}$ and the continuous variable $\mathit{flow}^S_{p_1chtdo}$ denoting the component flow of component $c \in \mathcal{C}$ through $p_1$ if option $h$ would be chosen. Note that, for each split node $p$, these constraints need to be defined only for all but one downstream pipelines $\mathcal{O}(p)$.

The volume of each component $c \in \mathcal{C}$ in the flow from a production node $p \in \mathcal{P}^{Prod}$ is given through

$$\mathit{flow}^C_{pctdo} = \mathit{flow}_{ptdo} \cdot CompProd_{pctdo}$$

$CompProd_{pctdo}$ defines the ratio of component $c$ contained in the flow produced at $p$.

The aggregated flow through the other nodes is the sum of the component flows,

$$\mathit{flow}_{ptdo} = \sum_{c\in\mathcal{C}} \mathit{flow}^C_{pctdo}$$

Component-wise mass balances for all $c \in \mathcal{C}$ replace the aggregated mass balances

(4.2a)–(4.2c):

$$\forall p \in \mathcal{P}^{Prod}: \quad \mathit{flow}^{C}_{pctdo} = \sum_{p_1 \in \mathcal{O}(p)} \mathit{flow}^{C}_{p_1ctdo}$$
$$\forall p \in \mathcal{P}^{Cons}: \quad \mathit{flow}^{C}_{pctdo} = \sum_{p_1 \in \mathcal{I}(p)} \mathit{flow}^{C}_{p_1ctdo}$$
$$\forall p \in \mathcal{P}^{Transp}: \quad \sum_{p_1 \in \mathcal{I}(p)} \mathit{flow}^{C}_{p_1ctdo} = \sum_{p_1 \in \mathcal{O}(p)} \mathit{flow}^{C}_{p_1ctdo}$$

These constraints connect the multi-component module to all other model functionality. Additional constraints may reflect quality requirements at market nodes.

## 4.7 Case study

We present a case study based on a realistic set of investment opportunities from the NCS. The actual investment projects are confidential, and we altered the names and the geographical locations of the network nodes. The data used in our analysis is partly real and partly synthetic. Most of the upstream data is real while the market data (prices and demand) is synthetic, but based on an analysis of historical data. We consider uncertainty both in volumes in the reservoirs for not yet developed fields and in the quality of the gas.

The analyzed network structure is shown in Figure 4.3. The case includes existing fields, markets and pipelines connecting them. Candidate new investment projects comprise branch-offs and compressors within the main transport network and the development of new fields and pipelines connecting them to the existing network. The existing infrastructure is shown in black, while candidate projects are shown in light gray color. Production fields are indicated by squares, consumption nodes by triangles and nodes within the network by circles. Arcs in the network indicate the pipelines in the system while arrows show the direction of flow.

As can be seen from the figure, the investment possibilities include relatively isolated new parts of the network as well as extensions and changes to the existing network. The system perspective taken in our analysis is very important when there are close links between the existing system and the new investment possibilities. Even though the links between the new parts of the system and the existing ones may seem weak on a first inspection, the consequences of ignoring them may be large due to network effects on pressure or quality. For example, projects 180 and 181 (just above the center of the figure) are parts of two alternative branch-offs that would make the existing link between projects 83 and 84

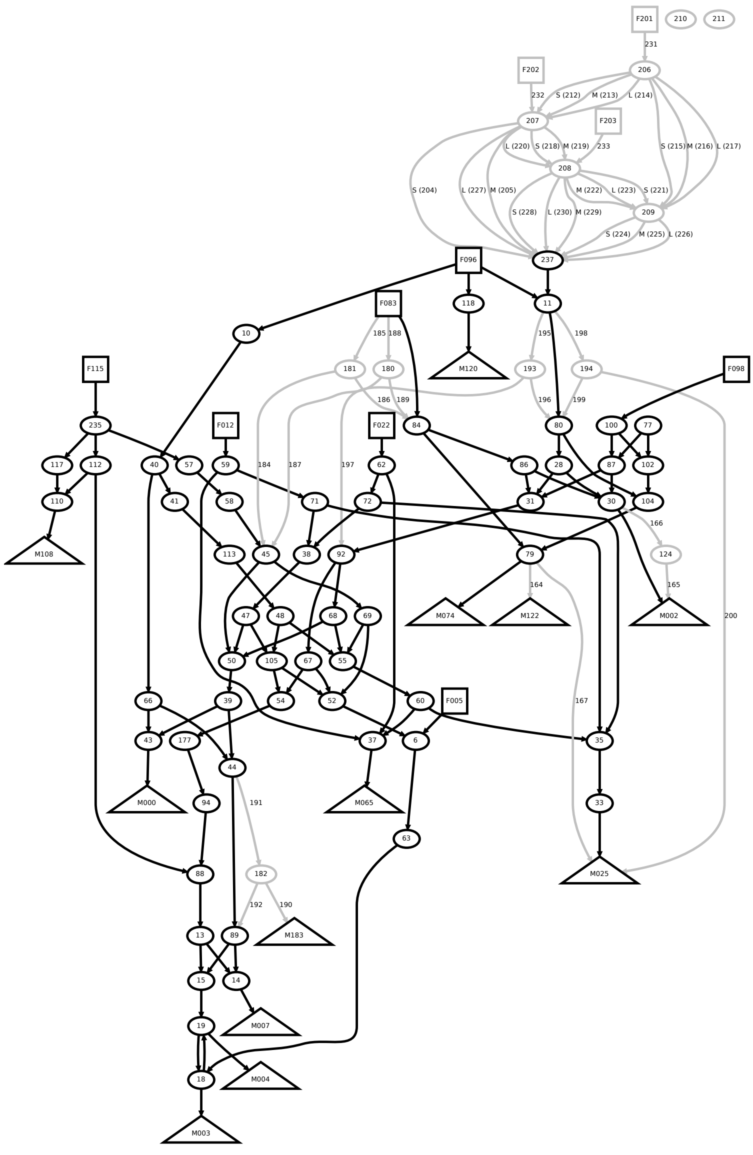



**Figure 4.3:** The network structure considered in the test instance.

obsolete. The branch-off is located at different places in the two cases, resulting in different pressure parameters. Examples of new fields are projects 201 through 203, while projects 204, 227, and 205 represent examples of pipeline alternatives with small, medium and large capacity, respectively, and corresponding costs.

Our test instance is a two-stage stochastic program with 11 years time horizon. The stochastic parameters are price and demand at the market nodes as well as the volumes and quality available from new fields. There are 240 projects in total, of which 182 are already existing infrastructure, and 58 represent investment opportunities. The two-stage model with 27 scenarios generates a model instance with 27.390.199 rows, 372.832 columns and 38.082.317 nonzero variables. Gurobi 4.6.0 reduces this to 599.988 rows, 110.872 columns and 2.094.249 nonzero variables during the presolve step. This presolved model instance has 20.441 continuous and 90.431 binary variables. It is solved to optimality in approximately 3 days on a six-core AMD Opteron processor 2431 with 24Gb memory. The computer was running Linux 2.6.18 (Rocks 5.3).

## 4.8 Conclusions

We have developed a multi-stage stochastic optimization model for analyzing natural gas infrastructure investment decisions in an integrated approach. New investment and shut-down options, including their proper timing, are studied in the light of the existing network structure, thus taking into account system effects as well as future production and development plans. Evaluating production assurance alongside common objectives such as profit maximization leads to a more robust and flexible infrastructure design. The inclusion of aspects such as pressure–flow relationships, multi-commodity flows or line-pack allows assessing the impact of the investment decisions on daily operations. We present a novel approach to analyze both long- and short-term uncertainty in a unified framework. For this purpose we introduce a new scenario tree structure, allowing a large number of operational periods without causing the typical explosion in size often observed with more traditional structures. We also discuss an application of the modeling framework to a realistic investment case on the Norwegian Continental Shelf.

## Acknowledgments

The model presented in this paper has been developed as part of the Ramona project (The Research Council of Norway, project number 175967) on production assurance and security of supply. We thank our industrial partners Statoil and Gassco for many inspiring discussions and support as well as providing the initial

layout of the case study.

## Bibliography

C. De Jonghe, B. Hobbs, and R. Belmans. Integrating short-term demand response into long-term investment planning. Cambridge working papers in economics 1132, Faculty of Economics, University of Cambridge, 2011.

Gas Processors Suppliers Association. *Engineering Data Book*. Tulsa, 11th edition, 1998.

V. Goel and I. Grossmann. A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. *Computers and Chemical Engineering*, 28(8):1409–1429, 2004.

K. Haugen. A Stochastic Dynamic Programming model for scheduling of offshore petroleum fields with resource uncertainty. *European Journal of Operational Research*, 88(1):88–100, 1996.

D. Haugland, Å. Hallefjord, and H. Asheim. Models for petroleum field exploitation. *European Journal of Operational Reseach*, 37:58–72, 1988.

T. Jonsbråten. Oil field optimization under price uncertainty. *Journal of the Operational Research Society*, 49(8):811–818, 1998.

K. O. Jørnsten. Sequencing offshore oil and gas fields under uncertainty. *European Journal of Operational Research*, 58:191–201, 1992.

V. Kalashnikov, G. Pérez-Valdés, A. Tomasgard, and N. Kalashnykova. Natural gas cash-out problem: Bilevel stochastic optimization approach. *European Journal of Operational Research*, 206(1):18–33, 2010.

C. Kelling, K. Reith, and E. Sekirnjak. A practical approach to transient optimization for gas networks. Technical report, Pipeline Simulation Interest Group (PSIG), 2000.

X. Li, A. Tomasgard, and P. Barton. Decomposition strategy for the stochastic pooling problem. *Journal of Global Optimization*, 54:765–790, 2011.

K. Midthun, M. Bjørndal, and A. Tomasgard. Modeling optimal economic dispatch and system effects in natural gas networks. *Energy Journal*, 30:155–180, 2009.

J. Myklebust. *Techno-economic modelling of value chains based on natural gas - with consideration of CO2 emissions.* PhD thesis, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, 2010.

B. Nygreen, M. Christiansen, K. Haugen, T. Bjørkvoll, and Ø. Kristiansen. Modelling Norwegian petroleum production and transportation. *Annals of Operations Research*, 82:251–267, 1998.

R. Rockafellar and S. Uryasev. Optimization of Conditional Value-at-Risk. *Journal of Risk*, 2:21–42, 2000.

F. Rømo, A. Tomasgard, L. Hellemo, M. Fodstad, B. Eidesen, and B. Pedersen. Optimizing the Norwegian natural gas production and transport. *Interfaces*, 39(1):46–56, 2009.

P. Schütz, A. Tomasgard, and S. Ahmed. Supply chain design under uncertainty using sample average approximation and dual decomposition. *European Journal of Operational Research*, pages 409–419, 2009.

A. Selot, L. Kuok, M. Robinson, T. Mason, and P. Barton. A short-term operational planning model for natural gas production systems. *AIChE Journal*, 54 (2):495–515, 2008.

E. Sönmez, S. Kekre, A. Scheller-Wolf, and N. Secomandi. Strategic analysis of technology and capacity investments in the liquefied natural gas industry. *European Journal of Operational Research*, 226(1):100–114, 2013.

J. Sullivan. The application of mathematical programming methods to oil and gas field development planning. *Mathematical Programming*, 42:189–200, 1988.

B. Tarhan, I. Grossmann, and V. Goel. 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, 2009.

A. Tomasgard, F. Rømo, M. Fodstad, and K. Midthun. Optimization models for the natural gas value chain. In G. Hasle, K.-A. Lie, and E. Quak, editors, *Geometric Modelling, Numerical Simulation and Optimization.* Springer-Verlag, Berlin, 2007.

N. L. Ulstein, B. Nygreen, and J. R. Sagli. Tactical planning of offshore petroleum production. *European Journal of Operational Research*, 176(1):550–564, January 2007.

S. A. van den Heever and I. E. Grossmann. A Lagrangean decomposition heuristic for the design and planning of offshore hydrocarbon field infrastructure with complex economic objectives. *Industrial & Engineering Chemistry Research*, 40:2857–2875, 2001.

# 4.A Notation

## Sets and indices

| Notation | Description |
|---|---|
| $b \in \mathcal{B} = \{1, ..., B\}$ | break points for Weymouth linearization |
| $c \in \mathcal{C} = \{1, ..., C\}$ | components in the flow |
| $d \in \mathcal{D}_t = \{1, ..., D_t\}$ | operational periods associated with investment period $t$ |
| $d_g^0$ | first operational period in group $g$ |
| $d_g^L$ | last operational period in group $g$ |
| $h \in \{1, ..., H\}$ | split options for multi-component flow |
| $\mathcal{I}(p) \subset \mathcal{P}$ | direct upstream ("input") projects of $p$ |
| $g \in \{1, ..., G\}$ | groups of consecutive operational periods |
| $\mathcal{G}_t^{norm} \subseteq \{1, ..., G\}$ | sets of operational periods representing normal operation |
| $\mathcal{G}_t^{extreme} \subseteq \{1, ..., G\}$ | sets of operational periods representing extreme situations |
| $n_0 \in \mathcal{S}$ | root node in the scenario tree |
| $\mathcal{O}(p) \subset \mathcal{P}$ | direct downstream ("output") projects of $p$ |
| $i,\ j,\ p \in \mathcal{P}$ | all projects |
| $\mathcal{P}^{Cons}$ | consumption projects (markets etc.) |
| $\mathcal{P}^{ex}$ | existing projects |
| $\mathcal{P}^{new}$ | new projects |
| $\mathcal{P}^{Nodes}$ | network nodes (all projects except pipelines) |
| $\mathcal{P}^{Pipe}$ | pipeline projects |
| $\mathcal{P}^{Prod}$ | production projects (fields etc.) |
| $\mathcal{P}^{Transp}$ | transport nodes |
| $s \in \mathcal{S} = \{1, ..., S\}$ | investment scenarios |
| $o \in \mathcal{S}^{Op}$ | all operational scenarios |
| $o \in \mathcal{S}_{ts}^{OpSub}$ | all operational scenarios associated with strategic tree node determined through year $t$, investment scenario $s$ |
| $t \in \mathcal{T} = \{1, ..., T\}$ | investment time periods |

## Decision variables

| Notation | Description |
|---|---|
| strategic level: | |
| $prd_{pts} \in \{0,1\}$ | project $p$ producing at $t$ under scenario $s$ |
| $start_{pts} \in \{0,1\}$ | project $p$ started at $t \in \mathcal{T}$ under scenario $s$ |
| $stop_{pts} \in \{0,1\}$ | $p$ stopped/removed at $t$ under scenario $s$ |
| $stprd_{pt\tau s} \in \{0,1\}$ | project $p$ started at $\tau$ and producing at $t$ |
| operational level: | |
| $flow_{ptdo}$ | flow through project $p$ |
| $flow^{A}_{pts}$ | accumulated flow through production node $p$ until (including) period $t$ |
| $flow^{C}_{pcto}$ | volume flow of component $c$ through project $p$ |
| $flow^{S}_{pchtdo}$ | volume flow of component $c$ through pipeline $p$ if split option $h$ is chosen |
| $\lambda_{phtdo} \in \{0,1\}$ | split option $h$ chosen for flow |
| $pres_{ptdo}$ | pressure in project $p$ (in- and output pressures modeled through pressures in in- and output projects) |

## Variables / functions

| Notation | Description |
|---|---|
| $Cost_{ts}$ | all costs in period $t \in \mathcal{T}$, scenario $s \in \mathcal{S}$ |
| $Cost^{Inv}_{ts}$ | investment related costs |
| $Cost^{invfix}_{pts}$ | fixed investment costs project $p$ |
| $Cost^{invrem}_{pts}$ | stop / removal costs project $p$ |
| $Cost^{invst}_{pts}$ | startup/investment costs project $p$ |
| $Cost^{Op}_{ts}$ | expected operations related costs |
| $Cost^{opex}_{pts}$ | operational expenditure of project $p$ |
| $Cost^{opfix}_{ptdo}$ | fixed operational costs of project $p$ |
| $Cost^{opvar}_{ptdo}$ | variable operational costs of project $p$ |
| $Del_{tdo}$ | production assurance measure: deliverability |
| $Rev_{ts}$ | all revenue in period $t \in \mathcal{T}$, scenario $s \in \mathcal{S}$ |
| $Rev^{Op}_{ts}$ | expected operations related revenue |

## Parameters / constants

| Notation | Description |
|---|---|
| $CapPerc_{ptdo}$ | percentage of flow capacity available for usage |
| $CompProd_{pctdo}$ | fraction of component $c$ in flow from production node $p$ |
| $\delta_t$ | discount factor in (investment) period $t \in \mathcal{T}$ |
| $\Delta_p$ | time between decision to invest and availability for production |
| $\gamma_g$ | factor to scale operational cash flow / volumes to yearly basis |
| $K_p^W$ | Weymouth constant for pipeline $p$ |
| $P_{ib}$ | approximation of pressure at in-/outlet point $i$ of a pipeline, breakpoint $b$ |
| $\underline{P}_i,\ \overline{P}_i$ | lower / upper bounds for pressure in node $i$ |
| $PATarget_t$ | performance target for production assurance |
| $Pen_t$ | penalty for insufficient production assurance |
| $Price^{Spot}_{ptdo}$ | sales price at market node / project $p$ |
| $ResMax_p$ | total volume to be produced from reservoir $p$ |
| $Prod^{Abs}_{pm}$ | absolute coefficient for linearization of reservoir profile |
| $Prod^{Rel}_{pm}$ | relative coefficient for linearization of reservoir profile |
| $Prob_{do}$ | probability of scenario tree node representing operational period $d \in \mathcal{D}_t$, scenario $o \in \mathcal{S}^{Op}$ |
| $\mathbf{R}(n) \in \mathcal{S}$ | root node of operational node $o \in \mathcal{S}^{Op}$ |
| $\sigma$ | discount rate |
| $Start^e_p$ | earliest startup date for (new) project $p$ |
| $Start^l_p$ | latest startup date for (new) project $p$ |
| $UB_{ptdo}$ | upper bound on capacity of pipeline $p$ |
| $Vol_{ptdo}$ | contracted delivery volume in market node $p \in \mathcal{P}^{Cons}$ |

## Profiles

Note that all strategic profiles have "yearly" (investment time unit) resolution while operational profiles have a "daily" resolution. This holds also for the floating operational profiles based on the age of the project although this age is expressed in investment time units.

| Notation | Description |
|---|---|
| Investment level, fixed (time dependent) – at $t \in \mathcal{T}$ for project $p \in \mathcal{P}$: | |
| $\Pi^{invremC}_{fix,pt}$ | removal costs (per year) |

| Notation | Description |
|---|---|
| $\Pi^{invst}_{pt}$ | startup/investment costs (per year) |
| Investment level, floating (age dependent) – at $t \in \mathcal{T}$ for project $p \in \mathcal{P}$ started at $\tau \in \mathcal{T}$: | |
| $\Pi^{invfixC}_{pt-\tau+1}$ | fixed investment costs |
| $\Pi^{invremC}_{float,pt-\tau+1}$ | removal costs |
| Operational level, fixed (time dependent) – at $d \in D_t,\ t \in \mathcal{T}$ for project $p \in \mathcal{P}$: | |
| $\Pi^{opfix}_{ptd}$ | constant operational costs (per day) |
| $\Pi^{opvar}_{ptd}$ | variable operational costs (per unit of flow) |
| Operational level, floating (age dependent) – at $t \in \mathcal{T}$ for project $p \in \mathcal{P}$ started at $\tau \in \mathcal{T}$: | |
| $\Pi^{MaxFlow}_{pt-\tau+1}$ | upper bound on daily flow |
| $\Pi^{MinFlow}_{pt-\tau+1}$ | lower bound on daily flow |
| $\Pi^{opex}_{pt-\tau+1}$ | operational expenditure (per year – averaged to daily levels) |

# Paper IV

Lars Hellemo and Adrian Werner:

# Discretizations of Natural Gas Pooling Problems

*Submitted to an international, peer-reviewed journal*

# Chapter 5

# Discretizations of Natural Gas Pooling Problems

**Abstract:**

Multi-component flow optimization problems in natural gas transport networks with pooling opportunities are large-scale non-convex problems. Such problems may be discretized and solved as mixed-integer linear programming (MILP) problems. Uniform discretization with a relatively small number of discretization points makes such approximations prone to converge to sub-optimal solutions.
We propose discretization schemes D1 to D3 and show that, by selecting discretization points based on information from the solution of an auxiliary linear problem, we improve the time to find good solutions considerably. To safeguard against cutting off optimal solutions, we propose verification by warm starting a problem with much finer discretization.
We show for a number of industry cases that the proposed discretization scheme D3 does not cut off the optimal solution. Rather, it provides better or optimal solutions much faster than with uniformly distributed discretization points (D1 and D2). Some instances can even be solved using auxiliary linear programs, and by exploiting the different problem properties, large numbers of natural gas flow problems with quality constraints can be solved more efficiently.

## 5.1 Introduction

Solving problems involving network flows is an important part of planning and operating the natural gas transport system from producing fields to market nodes. Network flow problems involving a single commodity can be solved very efficiently. However, natural gas is not homogeneous, which complicates the computations considerably. Depending on the geological conditions of the reservoir, the gas may have distinct characteristics in terms of composition, and therefore energy content, as well as the amount of contaminants such as $CO_2$ and $H_2S$. The composition of gas may affect how much can be transported, how much it must be treated in processing facilities and whether it can be sold in a market with defined quality requirements. When gas of different quality is mixed during transportation in the network, the mathematical properties of the problems used to solve network flow problems change. This is because one must keep track of the composition of the resulting flows. Such problems are often referred to as pooling problems, and can be traced back to Haverly [1978], who presented a classic problem with two sources, one pool, and two sinks and showed that pooling problems are computationally hard with many local optima. This is due to the non-convexity inherent in determining volumes of source gas, the resulting quality, and the volumes sent to each sink. These considerations have to be kept consistent and give bilinear terms in the optimization problem. If we ignore the quality issues and assume the gas to be homogeneous, the problem is reduced to a linear program (LP).

Haverly [1979, 1980] followed up on the original pooling formulation, and several studies were performed using successive linear programming (SLP), including, for instance, Baker and Lasdon [1985] up to more recent work by Frimannslund and Haugland [2009] who applied SLP with parallelization. SLP is an efficient algorithm, but is not guaranteed to converge to a global optimum.

Visweswaran and Floudas [1990] were the first to solve pooling problems using a global optimization algorithm to the pooling problem, that is, an algorithm guaranteeing to find the global optimal solution. They solved three problems posed by Haverly by solving a series of primal and relaxed dual problems. Foulds et al. [1992] were the first to apply McCormick underestimators and branch and bound (BB) methods to the pooling problem. Lodwick [1992] developed preprocessing techniques that would find implicit bounds. Androulakis et al. [1995] applied their $\alpha$BB approach to the pooling problem and Adhya et al. [1999] developed a global optimization technique, with bounds based on Lagrangian relaxation in combination with branch and bound methods. Audet et al. [2000] developed a branch and cut algorithm that was also applied to pooling problems.

Pooling problems can be formulated in two different, but equivalent ways. The original formulation of Haverly [1978] where flow and quality are modeled is often called the P formulation. Ben-Tal et al. [1994] introduced a different formulation based on the flow of individual components and proportions to enter the pool. This formulation is often referred to as the Q formulation. Kocis and Grossmann [1989] also considered a formulation based on flows and fractions going out from splitters. Quesada and Grossmann [1995] introduced a new formulation with extra constraints combining formulations P and Q, and this formulation is called PQ. Tawarmalani and Sahinidis [2002] discuss different global optimization formulations and prove that PQ gives tighter bounds than formulations P and Q.

Lee and Grossmann [2003] developed a two-level branch and bound algorithm and applied it to pooling problems in water management. Gounaris et al. [2009] showed that bounds and convergence time can be improved by applying piecewise linear relaxations. Discussing different approaches to solving pooling problems in a recent survey, Misener and Floudas [2009] give a good overview of the development during the early 2000s. Selot et al. [2008] applied global optimization methods to similar natural gas operational problems with additional disjunctive constraints for contracts regulating production allowances.

As an alternative to global optimization techniques, the pooling problem may be discretized and solved using a MILP solver. Ulstein [2000], Ulstein et al. [2007], Tomasgard et al. [2007], and Rømo et al. [2009] applied discretization approaches to large pipeline networks for natural gas with pooling on multiple levels. They used commercial mixed-integer linear programming solvers, and did not investigate the effects of the discretization on the quality of the solutions obtained. However, Alfaki and Haugland [2011] find that a discretization approach to pooling problems may perform better than a global optimization approach. Our work builds on the approach of Ulstein et al. [2007], Tomasgard et al. [2007], Rømo et al. [2009].

Recently, discretization as a part of global optimization algorithms has gained popularity. For example, Pham et al. [2009] discretized the quality attribute and reduced the problem dimensionality by limiting the number of possible combinations of pool qualities. Misener et al. [2011] used a piecewise linear relaxation of the bilinear function with a logarithmic number of binary variables. Gupte et al. [2012] compared the results of various discretization approaches with the global optimization solver BARON. Faria and Bagajewicz [2012] used a MILP as lower bounding problem and added an interval elimination technique before partitioning using a traditional branch and bound approach. Castro and Teles [2013] and Kolodziej et al. [2013] used an iteratively finer discretization by a logarithmic number of binary variables. They also compared piece-wise McCormick relaxations and multiparametric disaggregation.

Several attempts have been undertaken to generalize pooling problems in terms of network structure allowing multiple levels of interconnected pools. They include Audet et al. [2004] and Alfaki and Haugland [2013] who present a generalization of the PQ formulation for a general network structure. Meyer and Floudas [2006] generalized the problem to include network design, leading to combinatorial pooling problems. Li et al. [2011a] consider a stochastic pooling problem with network design and operations and present a global optimization decomposition algorithm for solving such problems in Li et al. [2011b].

The methods discussed in this paper are implemented as part of a large-scale capacity expansion model described in detail in Hellemo et al. [2012a]. We seek to maximize profit from selling natural gas on markets and assume that the price in each market is independent of the actual quality delivered, as long as this quality stays within predefined quality bands. In this paper, we focus purely on aspects related to routing multi-component gas flows and pooling. The natural gas transportation networks can be complex, with several levels of consecutive pools on the path between sources and sinks. Our problem shares many characteristics with the classic pooling problem, most notably the non-convexity that follows from blending gas from sources with different composition in pools before transporting it to the network sink. Problems may be formulated as single-period flow problems with pooling or as multi-period investment problems with many integer variables and an embedded pooling problem for each operational time period. Problems may also be formulated as stochastic programs, further increasing the dimensionality and, hence, the number of embedded pooling problems and integer variables. As a unified framework for approaching these problems, discretization appears to be a natural way to integrate pooling with potentially large-scale integer programming problems since state-of-the-art MILP solvers are known to be very efficient. While the problems solved in Li et al. [2011b] are certainly large-scale non-linear problems, the number of binary variables is low compared with investment models such as the one described in Hellemo et al. [2012a,b].

We present a discretization approach that is very promising in terms of finding good solutions to large-scale pooling problems faster than previous discretization approaches. This will enable us to perform more detailed scenario analysis or Monte Carlo simulations, which can be very useful to analyze potential infrastructure developments, or to perform security of supply analyses, reliability test or flow capacity analyses. It will also enable us to solve more detailed investment problems in a reasonable time frame.

We continue this paper in Section 5.2 with a discussion of the specifics of pooling problems and why they are computationally hard. We present two linear auxiliary problems that we use in our algorithm in Section 5.3 and detail three discretization approaches in Section 5.4. Our suggestions for algorithms combining auxiliary problems and discretization schemes follow in Section 5.5. We discuss computational results from test cases based on real-world problems in Section 5.6 before we conclude in Section 7.7. The full nomenclature for the mathematical formulations is given in Section 5.A.

## 5.2 Formulating the pooling problem

We present a general formulation of the pooling problem in the sense that we allow pools on multiple levels. The pooling problems may also be combinatorial as they may be part of investment models as described in Hellemo et al. [2012a,b]. Our formulation resembles the formulation of Kocis and Grossmann [1989] and Ruiz and Grossmann [2011] in modeling the fraction of flow moving to each downstream leg of a pool explicitly. In contrast to most global optimization approaches discussed above, we use a reformulation with discretization to solve pooling problems within an MILP framework. While our approach does not guarantee convergence of upper and lower global bounds, using finer discretization will limit the extent that optimal solution can be cut off. As the distance between discretization points approach the solver tolerance, the solutions potentially cut off will become correspondingly closer to a feasible solution of the discretized problem.

We consider the flow of natural gas to consist of $C$ components. The gas composition can be described by molar percentages for each gas component. We denote the flow of a component $c$ from a node $i$ to the next node $j$ by $f_{ij}^c$. The flow of a component in one pipeline out from a split node has to be equal to the inflow of that component into the node less the flow in the other pipelines out from this node:

$$f_{nj}^c = \sum_{i \in \mathcal{I}(n)} f_{in}^c - \sum_{k \in \mathcal{O}(n) \setminus j} f_{nk}^c, \quad c \in \mathcal{C}, n \in \mathcal{B}, j \in \mathcal{O}(n) \tag{5.1}$$

We only allow positive flows:

$$f_{nj}^c \geq 0 \;\forall n \in \mathcal{B}, j \in \mathcal{O}(n), c \in \mathcal{C} \tag{5.2}$$

The following constraint defines the volume of each component in the flow from a production node

$$f_{nj}^c = f_{nj} Q_n^c, \;\forall n \in \mathcal{P}, j \in \mathcal{O}(n),\; c \in \mathcal{C} \tag{5.3}$$

where the parameter $Q_n^c$ defines the percentage of component $c$ contained in the flow produced at production node $n \in \mathcal{P}$.

The aggregated flow $f_{nj}$ through a network element is the sum of the component flows,

$$f_{nj} = \sum_{c \in \mathcal{C}} f_{nj}^c, \forall n \in \mathcal{N} \setminus \mathcal{P}, j \in \mathcal{O}(n). \tag{5.4}$$

For simplicity, let us consider only two pipelines out from a split node $n \in \mathcal{B}$ going to nodes $j_1$ and $j_2$. We denote the first component in the set $\mathcal{C}$ of components by $c_1$. We assume full blending and that the ratio of the volume split between the pipelines to $j_1$ and $j_2$ must be equal for all components,

$$\frac{f_{nj_1}^{c_1}}{f_{nj_2}^{c_1}} = \frac{f_{nj_1}^{c}}{f_{nj_2}^{c}}, \forall n \in \mathcal{B}, j_1, j_2 \in \mathcal{O}(n), c \in \mathcal{C} \setminus c_1. \tag{5.5}$$

This quality relationship can be reformulated in the following way, yielding many bilinear terms:

$$f^{c_1}_{nj_1} f^{c}_{nj_2} = f^{c_1}_{nj_2} f^{c}_{nj_1}, \forall n \in \mathcal{B}, j_1, j_2 \in \mathcal{O}(n), c \in \mathcal{C} \setminus c_1. \tag{5.6}$$

At market nodes $m \in \mathcal{M}$, there are quality requirements. For example, the energy content in terms of Gross Calorific Value (GCV) must be within the bounds $\underline{GCV_m}$ and $\overline{GCV_m}$ defined for each market,

$$\underline{GCV_m} \leq \sum_{n \in \mathcal{I}(m)} \sum_{c \in \mathcal{C}} GCV^c f^c_{nm} \leq \overline{GCV_m}, \forall m \in \mathcal{M}. \tag{5.7}$$

Also, the percentage of $CO_2$ must be below the maximum $CO_2$ content allowed at each market, $\overline{CO_{2m}}$ $m \in \mathcal{M}$:

$$\sum_{n \in \mathcal{I}(m)} f^{CO_2}_{nm} \leq \overline{CO_{2m}} \sum_{n \in \mathcal{I}(m)} \sum_{c \in \mathcal{C}} f^c_{nm}, \forall m \in \mathcal{M} \tag{5.8}$$

## Special cases

While generally the pooling problem is computationally hard, there are some important special cases that require much less computational effort (compare Table 5.3 to tables 5.4 and 5.5).

1. The optimal flow pattern is “similar” to the flow pattern of the single-component flow solution: In this case we may find the global optimum in the neighborhood of this solution.

2. The quality constraints are all non-binding: In this case the optimal flow pattern is equal to the solution from a single-component flow relaxation. Because the single-component solution can be found as a linear combination of the multi-component flows, it is easy to find the single-component solution corresponding to a multi-component solution and vice versa. Suppose that the optimal multi-component solution is better than the optimal single-component solution. Then we could easily find the equivalent single-component flow – which means the original single-component solution could not have been optimal. An analogous argument applies when assuming that the optimal single-component solution is better than the optimal multi-component solution. In contrast, if the constraints were binding and the single-component solution is better than the multi-component solution, then it would be infeasible for the multi-component problem.

We will take advantage of these observations in the subsequent sections.

## 5.3 Auxiliary models

During the solution procedure, we use two auxiliary problems that are simplifications of the model presented in Section 5.4: A model assuming homogeneous flow (A1, referred to as pre-processing) and a model where the total flow volumes are fixed and just the component flows need to be determined (A2, referred to as post-processing).

We will later use information from the solution of A1 as part of the solution procedure for the discretized model. For some easy problems as identified in Section 5.2, solving A1 (pre-processing) followed by A2 (post-processing) will be sufficient to find the global optimum.

### Homogeneous flow (A1)

For the homogeneous-flow auxiliary problem, we assume that all flows consist of only one component. This means that the optimization problem is no longer a pooling problem, but can be formulated as an LP and solved very efficiently. This is analogous to the first step in the procedure used by Rømo et al. [2009], see Figure 5.2a.

We maximize the total revenue from deliveries to market nodes:

$$\max \sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{I}(m)} \Pi_m f_{nm} \tag{5.9}$$

subject to the following constraints:
Flow from production node $p$ must not exceed the production capacity of that node, $P_p$.

$$\sum_{n \in \mathcal{O}(p)} f_{pn} \leq P_p, \forall p \in \mathcal{P} \tag{5.10}$$

Flow to market nodes $m$ must not exceed demand $D_m$ in that market:

$$\sum_{n \in \mathcal{I}(m)} f_{nm} \leq D_m, \forall m \in \mathcal{M} \tag{5.11}$$

Mass balances must hold for all nodes except production and market nodes.

$$\sum_{i \in \mathcal{I}(n)} f_{in} = \sum_{o \in \mathcal{O}(n)} f_{no}, \forall n \in \mathcal{N} \setminus (\mathcal{P} \cup \mathcal{M}) \tag{5.12}$$

### Fixed flows and fractions (A2)

After solving a homogeneous-flow problem A1, a post-processing routine or a simplified model can be run where all flows $F_{ij}$ and split fractions $\Phi_{jk}$ are fixed while the component flows $f_{ij}^c$ need to be determined. The resulting problem is a system of linear equations that may be solved as a linear program with an arbitrarily chosen objective function, e.g.:

$$\max \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{M}} \sum_{i\in\mathcal{I}(j)} \Pi_j f_{ij}^c \tag{5.13}$$

subject to:

$$f_{ij}^c = Q_i^c F_{ij}, \; \forall c \in \mathcal{C}, i \in \mathcal{P}, j \in \mathcal{O}(i) \tag{5.14}$$

$$f_{jk}^c = \Phi_{jk} \sum_{i\in\mathcal{I}(j)} f_{ij}^c, \; \forall c \in \mathcal{C}, j \in \mathcal{N} \setminus \mathcal{M}, k \in \mathcal{O}(j) \tag{5.15}$$

$$\sum_{c\in\mathcal{C}} f_{ij}^c = F_{ij}, \forall j \in \mathcal{N} \setminus \mathcal{P}, i \in \mathcal{I}(j) \tag{5.16}$$

The flow $f_{ij}^c$ of component $c$ from a producing field $i$ is defined by the quality $Q_i^c$ of the gas produced at that field (Equation (5.14)). Each component flow from a split node into a downstream leg must equal the inflow to this node times the split fraction $\Phi_{jk}$ into that leg (Equation (5.15)). Finally, for all pipelines, the total flow of all components must equal the pre-calculated total flow volume for that pipeline (Equation (5.16)).

If the problem has quality constraints, these constraints can be checked for violations. In the case that some quality constraint is violated, one of the discretizations of the full pooling problem must be solved.

## 5.4 Discretizations

In this paper, we discuss three approaches to discretization. The following formulations include the parts of an optimization model that are relevant to the discussion of the discretization of the pooling problem. The complete model also includes flow/pressure constraints, processing nodes, which alter the gas composition by extracting parts of some components, and economic aspects such as production and transportation costs or demand satisfaction. A full model specification is provided in Hellemo et al. [2012a].

In the following, we discuss the basic discretization formulation with uniformly distributed split options (D1), a binary split formulation (D2), and discretization with concentrated split options (D3).

## Uniform discretization (D1)

A straight-forward approach to discretizing the bilinear terms in Equation (5.6) is to define a set of allowed split fractions and use binary variables to select a suitable split fraction. This reduces accuracy as splits are only allowed to take predefined values. In selecting the number of binary variables, and thus the accuracy of the approximation, a balance has to be struck between precision and computational tractability.

For each split node $n$, we use a set of binary variables $\lambda_{nj}^{z}$ where $z = 1, 2, \ldots, Z$, each representing the choice of a split fraction for the share of natural gas going to node $j_1$. The set $\mathcal{B}$ consists of all split nodes in the network.

We also define a new variable $e_{nj}^{cz}$ representing the flow from $n$ to $j$ of component $c$ if $\lambda_{nj}^{z} = 1$. The flow $f_{nj}^{c}$ of component $c$ from $n$ to $j$ equals the sum of partial flows $e_{nj}^{cz}$ from $n$ to $j$ (of which all except one equal zero):

$$f_{nj}^{c} = \sum_{z=1}^{Z} e_{nj}^{cz}, \ \forall\ n \in \mathcal{B}, j \in \mathcal{O}(n). \tag{5.17}$$

$E_{nj}$ is the maximum possible flow through the pipeline between $n$ and $j$, and $e_{nj}^{cz}$ the partial flow associated with breakpoint $z$. For each $\lambda_{nj}^{z}$ we define a constant $\alpha_{nj,z}$ giving the fraction associated with $z$. For the discretization approach D1 we let the split fractions $\alpha_{nj,z}$ be uniformly distributed. For example, with $Z = 5$, $\alpha_{nj,z} \in \{0, 0.25, 0.5, 0.75, 1.0\}, \forall n \in \mathcal{B}, j \in \mathcal{O}(n)$. The $e_{nj}^{cz}$ variables are constrained by the capacity of the split node if $\lambda_{nj}^{z}$ is active.

$$\sum_{c \in \mathcal{C}} e_{nj}^{cz} \leq \alpha_{nj,z} E_{nj} \lambda_{nj}^{z}, \tag{5.18a}$$

$$0 \leq e_{nj}^{cz} \leq \lambda_{nj}^{z} \alpha_{nj,z} E_{nj}, \tag{5.18b}$$

$$\alpha_{nj,z} = \frac{z-1}{Z-1}, \tag{5.18c}$$

$$\forall n \in \mathcal{B}, j \in \mathcal{O}(n), \tag{5.18d}$$

$$z \in \{1, 2, \ldots, Z\}, \lambda_{nj}^{z} \in \{0, 1\} \tag{5.18e}$$

Equation (6.18b) is not strictly necessary, but improves the numerical properties of the solution.

Each partial flow of a component flow equals the split fraction times the total inflow of this component:

$$e_{nj}^{cz} \leq \alpha_{nj,z} \sum_{i \in \mathcal{I}(n)} f_{in}^{c}, \forall n \in \mathcal{B}, j \in \mathcal{O}(n), z \in \{1, 2, \ldots, Z\} \tag{5.19}$$

$$\begin{gathered}\alpha_{nj,z} \sum_{i \in \mathcal{I}(n)} f_{in}^{c} - e_{nj}^{cz} \leq \alpha_{nj,z} E_{nj} (1 - \lambda_{nj}^{z}), \\ \forall n \in \mathcal{B}, j \in \mathcal{O}(n), c \in \mathcal{C}, z \in \{1, 2, \ldots, Z\}\end{gathered} \tag{5.20}$$

We make sure only one $\lambda_{nj}^{z}$ is positive for each node (this may be specified as an SOS1 over $z \in \{1, 2, \ldots, Z\}$).

$$\sum_{z=1}^{Z} \lambda_{nj}^{z} = 1, \ \forall \ n \in \mathcal{B}, j \in \mathcal{O}(n). \tag{5.21}$$

## Binary formulation (D2)

This formulation takes inspiration from binary numbers. The component flows $f_{nj}^{c}$ are considered linear combinations of the partial component flows $e_{nj}^{cz}$, and the potential split options $\alpha_{nj,z}$ are combined to find the actual split fraction. For example, with $Z = 3$, we get $\alpha_{nj,z} \in \{0.5, 0.25, 0.125\}$ and can represent the fraction 0.625 by setting $\lambda^1 = 1, \lambda^2 = 0$ and $\lambda^3 = 1$. By using $Z$ binary variables for each split point in the network, we achieve a discretization of $\frac{1}{2^{(Z-1)}}$ possible split points. For example, with a resolution of $Z = 11$, we get discrete steps of $\frac{1}{1024}$ or about 0.001. A similar resolution using the uniform discretization approach as above would require $2^{Z-1} + 1$ (1025) binary variables per split node.

The formulation of D2 is similar to D1 with some important exceptions: We do not include a constraint as in Equation (6.21), but allow all $\lambda_{nj}^{z}$ to take values one or zero. Also the definition of $\alpha_{nj,z}$ is different, compare Equation (6.18) to Equation (6.22). $E_{nj}$ is the maximum possible flow from $n$ to $j$ and $e_{nj}^{cz}$ the partial flow associated with breakpoint $z$ as defined by Equation (6.18) where Equation (5.18c) is replaced by Equation (6.22).

$$\alpha_{nj,z} = \begin{cases} \frac{1}{2^z}, z \in \{1, 2, ..., Z-1\} \\ \frac{1}{2^{z-1}}, z = Z \end{cases} \tag{5.22}$$

Each binary fraction of a component flow equals the split fraction times the total inflow of this component as formulated in Equations (6.19) to (6.20). The flow of each component through a pipeline equals the sum of all binary parts of the flow of this component through the pipeline as defined in Equation (6.17), where in this case some or even all may be non-zero.

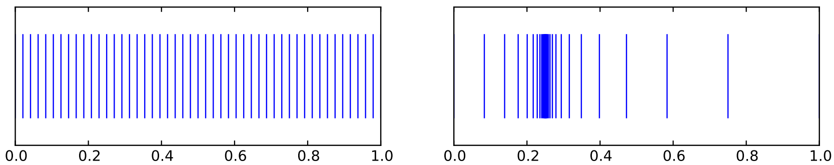



**Figure 5.1:** Distribution of 50 split points. Uniform split (D1) on the left, split points concentrated around $\alpha = 0.25$ (D3) on the right.

This formulation also has a uniform distribution of potential split options, but has the attractive feature that it allows a much finer grained discretization with the same number of binary variables as formulation D1. The disadvantage of this approach is that the small size of the coefficients for the least significant contributions can lead to numerical problems as $Z$ increases.

## Concentrated discretization (D3)

While the approaches presented above assumed a uniform distribution of potential split points, this may not be necessary. If there is reason to believe that an optimal solution may deviate little from an already known split, the split points could be placed with higher density around this predefined split fraction.

The discretization approach D3 is based on the same formulation as D1, Equations (6.17) to (6.21). The split fractions $\alpha_{nj,z}$ are computed according to the procedure presented below. This procedure finds a set of $Z$ split options on the interval $[0,1]$ which are distributed more densely around the fraction $\alpha_{nj,S}$ coming from solving the single-commodity model (A1). We calculate one set for each pipeline out from a split node except one pipeline. For this remaining pipeline, the component flows are determined by the node inflows less the flows into the former pipelines.

The calculation of split points is based on a geometric series with a factor less than one, giving split points that are progressively closer. This method to calculate split points is given in algorithm 1. A visualization of the distribution of split points can be seen in Figure 5.1. The parameter $q > 1$ controls the density of the points around $\alpha_{nj,S}$; higher values of $q$ contract the split options closer to $\alpha_{nj,S}$, lower values will spread the options more evenly. Additionally, the split options will be distributed over the $[0,1]$ interval proportionally to the size of $\alpha_{nj,S}$, i.e., for $\alpha_{nj,S}$ close to 1, most of the split points will be smaller than $\alpha_{nj,S}$. The number of split points used for values smaller than $\alpha_{nj,S}$ is $Z_L$ and the number of points larger than $\alpha_{nj,S}$ is $Z_U$. We also include $\alpha_{nj,S}$ as a potential split point.

If the distance between consecutive points contracts too fast, a gap will occur between the original split fraction $\alpha_{nj,S}$ and the split point closest to it. To avoid this situation, we require that this distance shall not be larger than for evenly distributed points:

$$\frac{\alpha_{nj,S}}{q^{Z_L-1}} \leq \frac{\alpha_{nj,S}}{Z_L} \tag{5.23}$$

which can be reformulated to $q \geq \sqrt[Z_L-1]{Z_L}$. If this condition does not hold, we adjust the value of $q$ by $q_0 \sqrt[Z_L-1]{Z_L}$ where $q_0 \geq 1$ is a coefficient to boost the effect of the adjustment. A similar reasoning applies to the distance between the split option $\alpha_{nj,S} = \alpha_{nj,Z_L+1}$ and the adjacent option $\alpha_{nj,Z_L+2}$. The original or adjusted factors form thus the factors $q_L$ and $q_U$ for calculating the split points smaller and larger than $\alpha_{nj,S}$, respectively.

## Redundant constraints

Several authors have noted that including redundant constraints in the problem formulation may give improved performance. In addition to Equation (6.18b), we have investigated two sets of redundant constraints: adding the McCormick convex envelopes of the bilinear terms (McCormick [1976]), and adding an explicit split fraction for all pipelines downstream of a split node (where the first is implicit in the above formulation) together with a constraint that all split fractions downstream of a split node must sum to one. The former gives better bounds, and the formulation is shown below, while the latter led to nearly doubling the number of binary variables in our cases, and this decreases performance significantly.

To define the McCormick constraints, we use auxiliary variables $\alpha_{nj}$:

$$\alpha_{nj} = \sum_{z=1}^{Z} \alpha_{nj,z} \lambda_{nj}^{z} \;\forall\; n \in \mathcal{B}, j \in \mathcal{O}(n). \tag{5.24}$$

Note that the lower bounds $\underline{\alpha_{nj}}$ and upper bounds $\overline{\alpha_{nj}}$ on the split fractions are 0 and 1, respectively, $\forall n \in \mathcal{B}, j \in \mathcal{O}(n)$. $\underline{f_{nj}}$ and $\overline{f_{nj}}$ denote the lower and upper bounds on flow variables. We define the constraints below for all pipelines where $\alpha_{nj}$ is explicitly defined. For pipelines where the split $\alpha_{nj}$ is implicit, we use corresponding constraints where $\alpha_{nj}$ is replaced by $1 - \sum_{k \in \mathcal{O}(n)} \alpha_{nk}$.

$$f_{jk} \leq f_{nj}\underline{\alpha_{nj}} + \overline{f_{nj}}\alpha_{nj} - \overline{f_{nj}}\underline{\alpha_{nj}} \tag{5.25}$$

$$f_{jk} \leq f_{nj}\overline{\alpha_{nj}} + \underline{f_{nj}}\alpha_{nj} - \underline{f_{nj}}\overline{\alpha_{nj}} \tag{5.26}$$

$$f_{jk} \geq f_{nj}\overline{\alpha_{nj}} + \overline{f_{nj}}\alpha_{nj} - \overline{f_{nj}}\,\overline{\alpha_{nj}} \tag{5.27}$$

$$f_{jk} \geq f_{nj}\underline{\alpha_{nj}} + \underline{f_{nj}}\alpha_{nj} - \underline{f_{nj}}\,\underline{\alpha_{nj}} \tag{5.28}$$

**Input:** $q \geq 0, q_0 \geq 1, \alpha_{nj,S} \in [0,1], Z \in \mathbb{N}$
$Z_L \leftarrow \max\{1, \lceil \alpha_{nj,S} Z \rceil\}$ ▷ Initialize
$Z_U \leftarrow Z - Z_L - 1.$
**if** $q \geq \sqrt[Z_L-1]{Z_L}$ **then** ▷ Check convergence lower part
    $q_L \leftarrow q$
**else**
    $q_L \leftarrow q_0 \sqrt[Z_L-1]{Z_L}$
**end if**
**if** $q \geq \sqrt[Z_U-1]{Z_U}$ **then** ▷ Check convergence upper part
    $q_U \leftarrow q$
**else**
    $q_U \leftarrow q_0 \sqrt[Z_U-1]{Z_U}$
**end if**
**for all** $z \in \{1, \dots, Z\}$ **do** ▷ Calculate split fractions lower part
    **if** $z \in \{1, ..., Z_L\}$ **then**
        $\alpha_{nj,z} \leftarrow \alpha_{nj}^S - \frac{\alpha_{nj,S}}{q_L^{z-1}}$
    **else if** $z = Z_L + 1$ **then**
        $\alpha_{nj,z} \leftarrow \alpha_{nj,S}$
    **else** ▷ Calculate split fractions upper part
        $\alpha_{nj,z} \leftarrow \alpha_{nj,S} + \frac{1-\alpha_{nj,S}}{q_U^{Z-z}}$
    **end if**
**end for**

**Algorithm 1:** Calculate split points

## 5.5 Algorithms

The discretization schemes D1–D3 and auxiliary problems A1–A2 may be combined in different ways to solve pooling problems. In this section, we discuss some ways to combine the faster but potentially inaccurate formulations with more accurate but computationally more demanding formulations.

## Hybrid split algorithm

A three step methodology to improve the calculation speed for flow problems with 12 components is described in Rømo et al. [2009] with pre-processing as in A1, followed by a coarse discretization with SOS1 sets and discretization D2 depending on the problem properties, see Figure 5.2a. We propose an improved algorithm for multi-component flow calculation based on model properties. The algorithm makes sure that the faster calculation (A1+A2) is applied whenever the problem falls into the class of well behaved problems, moving on to the more complex models (D2+D3) only when necessary, see Figure 5.2b.

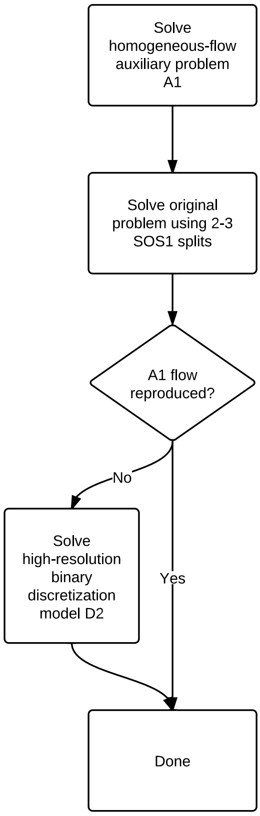
## Warm start

We get a global upper bound on the flow from the objective value of the solution from A1 or from the objective value of the root relaxation of either discretization. The solution of the discretized problem may give a flow that is lower than this upper bound. If the gap between the solution and the upper bound is greater than some threshold, we may try to verify the solution with a finer discretization scheme. We may also wish to verify the solution for other reasons. In that case, a model instance with fine-grained discretization using D2 may be warm started from a D3 solution.

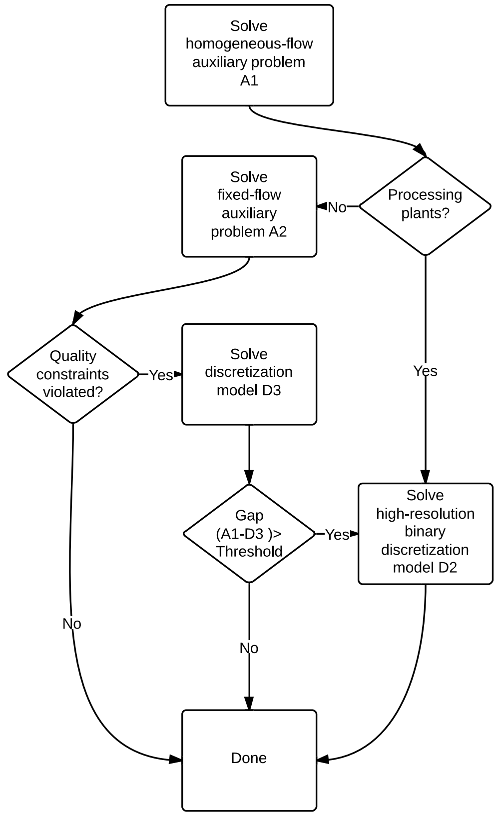
```
Solve the homogeneous-flow auxiliary problem A1
Save the split fractions
Z ← 10
Solve the concentrated discretization D3
if Gap < GapLim or Time > TimeLim then
    Z ← 25
    Load problem D2
    Load solution from D3
    Solve problem D2
end if
```

**Algorithm 2:** Warm start

This method will give a good estimate of the upper bound on the flow maximization in the warm start of D2 due to the fine discretization, while exploiting the reduced problem size in D3. Depending on the particular problem instance, this can, in the best case, give quick convergence to a global optimum, or, in the worst case, detect that the solution found by solving D3 is far from a proven global optimum.




**(a)** Flow chart for the procedure used in Rømo et al. [2009].




**(b)** Flow chart showing how different flow models are invoked depending on input data and problem properties.

**Figure 5.2:** Flow charts

### General approach

To summarize, we exploit the observations from Section 5.2 and apply the simplest possible approach for a given problem instance. If there are no quality constraints, then pre-processing A1 followed by the post-processing A2 is sufficient. For problems where the optimal solution is close to the solution of A1, D3 will give good results, and for more difficult problems, D2 is more robust.

Even for problems where there are quality constraints, the post-processing is so much cheaper in terms of computation time than running a full-scale pooling model, that it can be worthwhile to solve the simple model and check whether the constraints are satisfied before running the full scale model. With the abundance of parallel computing due to multi-core processors, several methods can be started simultaneously. For example, the computations of A2 and D3 can be started simultaneously and D3 aborted if solving A2 turns out to be sufficient. For serial execution, we propose the general solution strategy shown in the flow chart in Figure 5.2b and in algorithm 3.

```
Solve homogeneous-flow auxiliary problem A1
Solve fixed-flow auxiliary problem A2
if not quality constraints violated then
    return solution
else
    Solve concentrated discretization problem D3
    if bound deviation < threshold then
        return solution
    else
        Solve fine-grained binary discretization problem D2
        return solution
    end if
end if
```

**Algorithm 3:** Solve pooling problem

## 5.6 Computational results

In many analysis situations, getting a good solution fast is as important as having the solver prove that the program was solved to optimality. In this section, we investigate the behavior of the discussed methods by means of five test cases. We evaluate the quality of the solution after 120 seconds for single period analyses (cases 1-3) and one hour for large multi-period investment models (cases 4-5). We also evaluate the time to reach a certain gap between the best solution found and the pre-computed estimate of the global upper bound, i.e. to reach a sufficiently good solution.

The true global upper bound on the objective function value is given by the homogeneous-flow problem. All our discretization schemes are approximations that are not guaranteed to give an appropriate bound. However, to estimate the upper bound, we use the best solution found by a very fine grained discretization (algorithm 2) after several days of computation. We use the gap between the solution found within the set time limit and this estimate of the global upper bound to evaluate the quality of the found solutions: As the discretization steps approach the precision of the solver software, the solution of the discretization will converge to the continuous solution. With 50 binary variables, the theoretical precision approaches that of IEEE double (15 significant digits), and thus this solution will be of high quality. Only considering the bounds on the coarse discrete approximation can be misleading, as the global optimum may have been cut off.

## Test cases

The five test cases are all based on real industry cases from the Norwegian Continental Shelf. Norwegian gas export covers close to 20% of European Union natural gas consumption, and the export is expected to increase by 30-50% over the next decade. With a few exceptions, this gas is transported through a subsea pipeline network with pipelines going as deep as 1000 meters below sea level. Gas under high pressure, often 150 bar and above, is transported over large distances with single pipelines being several hundred kilometers long. The entire system constitutes over 8000 km of pipelines, the largest subsea gas transport network in the world (The Norwegian Ministry of Petroleum and Energy and The Norwegian Petroleum Directorate [2014]).

Cases 1 through 3 are converted from industry cases specified through the model presented in Rømo et al. [2009]. Cases 4 and 5 are investment problems similar to the one presented in Hellemo et al. [2012b]. The size of the network in each test case is given in Table 5.1, the size of each problem instance in terms of variables and constraints in Table 5.2.

## Results

All reported computation times are elapsed wall clock time in seconds using Gurobi 5.6.3 on a six-core AMD Opteron processor 2431 with 24Gb memory running Linux 2.6.18 (Rocks 5.3). We have used default settings with the exception of setting the time limit and increasing the priority of numerical correctness over speed.

The solution times for the linear auxiliary problems are shown in Table 5.3. All these problems are solved in a matter of seconds, which makes the computation time negligible compared with the solution time for the full problems. Thus, solving the fixed-flow auxiliary problem A2 and checking for quality constraints is computationally so cheap that we suggest to check first whether this is already sufficient for finding the optimum of the original problem.

As may be expected, the coarser discretizations converge quicker than the finer discretizations, and D1 and D3 with a relative small number of split candidates often converge before reaching the time limit. On the other hand, the coarse discretization cuts off solutions, while the finer discretizations give better upper bounds, but will often take longer to find a good solution. For our test cases, formulation D3 seems to be less prone to cutting off solutions, and we use it in combination with formulation D2 to verify the solution quality.

Having a relatively low number of binary variables for each split node reduces the search space, and, hence, improves the speed of convergence of the discretized problem. Apparently, the concentrated split points in D3 include often the relevant potential split fractions, giving a better solution using D3 in most of our test cases. To illustrate the differences in computation time to find solutions, we include some plots of the development of discretization upper and lower bounds and compare with the best solution available. Because the discretization with the lowest number of binary variables tended to give best results with respect to computation time, the figures showing the bounds over time all show the results using 10 binary variables for each split node. All plots show discretization scheme D1 on top, D2 in the middle, and D3 at the bottom. Figure 5.3a shows the development of discretization upper and lower bounds as well as the best available solution for case 1 during 120 seconds. Discretization scheme D1 with 10 binary variables for each split node gives an approximated problem that converges fast, but is cutting off the optimal solution, thus converging to a suboptimal solution. Discretization scheme D3 gives a problem that converges very fast until reaching a gap of a few percent. For the larger infrastructure development models in cases 4 and 5, we plot the developments of upper and lower bounds during one hour.

The quality of the obtained solutions can be compared with the precomputed solutions as shown in tables 5.4 and 5.5 in the Gap column. Note that these results do not include good, but not optimal, solutions found early, as shown in the plots mentioned above.

We also note that while precision can be improved infinitely in theory, the solver precision is limiting the actual achievable precision. This limit is around 20 binary variables per split node using formulation D2, which gives a precision in the order of $10^{-6}$. For the experiments with warm start, we have used 10 binary variables for formulation D3 and 25 binary variables for formulation D2. The time limit for D3 was set to one third of the total time, and the time limit for D2 was set to the total time limit less the time spent solving D3. As can be seen in Figure 5.4, we are able to combine the solution of formulation D3 with the bounds from formulation D2. Evidently, these upper bounds are looser, but do not exclude potential solutions, within the precision of the discretization D2.

For our test cases, the warm start gives slightly inferior solutions compared to using D3 alone. This is to be expected as there is less time allocated to finding a solution using D3. The computation of the upper bound starts from scratch using the fine-resolution D2 formulation that will guarantee within the resolution of the discretization that no solution has been cut off.

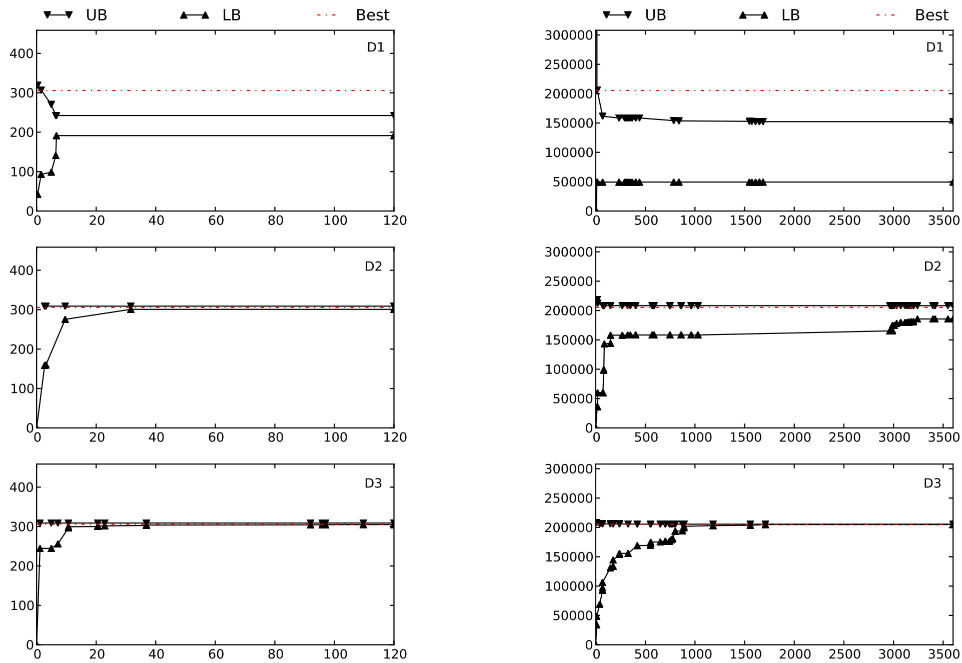
We believe our approach will very useful in settings where a large number of similar cases are to be analyzed, for example, for Monte Carlo simulations. Of a set of problem instances, some will be solved using auxiliary problem A1 and A2 in combination and solving the MILP can be avoided altogether. Some problem instances will be solved successfully using the warm start combination of D3 and D2 within the time limit, while some instances may require longer computational time. By freeing up computational capacity from some easier problems, this capacity may be allocated to the harder problem instances. The distribution of problem properties among the problem instances in a set of problems depends on the input data and is not generally known in advance. We suggest detecting easier problem instances and exploiting that they may be solved by a simpler and faster model.

## 5.7 Conclusions

We have presented several approaches to solving the pooling problem for multi-component gas flows by discretizing the split fractions for all split nodes. Some problems can be solved by an efficient post-processing procedure. Other problems may be solved efficiently if an appropriate discretization scheme is chosen. The problem properties may be known in advance, e.g., if there are no quality constraints, or can be determined dynamically as suggested in algorithm 3.

From our case studies, we see that a straightforward discretization (D1) easily converges to a suboptimal solution. The discretization regime D3 with the discretization points concentrated around the split fraction solutions from the homogeneous-flow auxiliary problem A1 seems less prone to this problem. Our computational experiments show that this approach yields good results. There is, however, no guarantee that it does not converge to a suboptimal solution when using a low number of discretization points. Therefore, we suggest that a new problem is solved with a much finer discretization, and that the solution from the coarse discretized problem is used to warm start this more accurate problem.

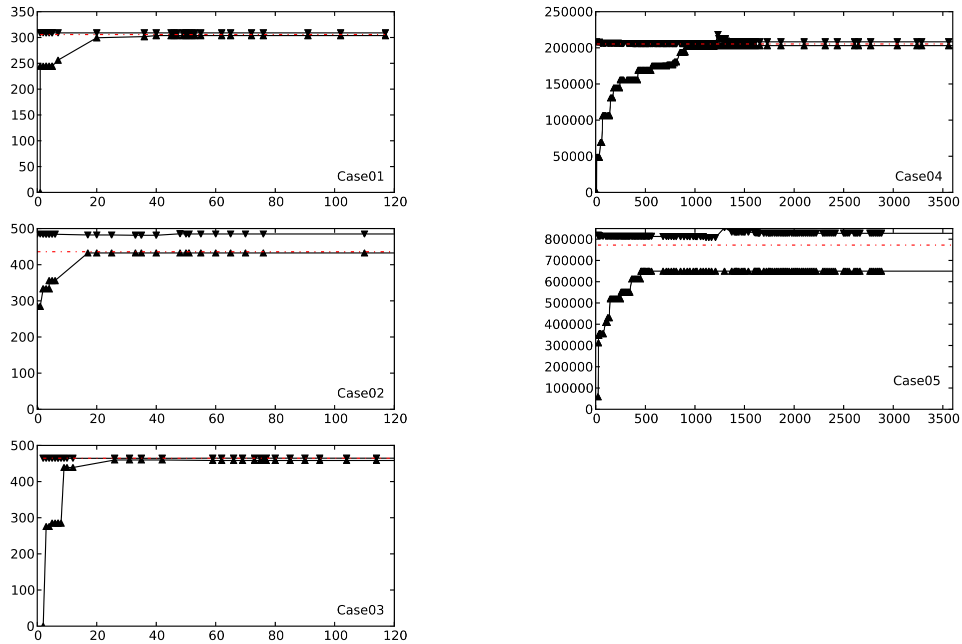
We demonstrate that, with our new approach, we are able to find good solutions with quality guarantees for problems that were previously practically unsolvable.




**(a)** Case 1, over 120 seconds.

**(b)** Case 4, over 3600 seconds.

**Figure 5.3:** Convergence of discretization lower and upper bounds and best available solution in cases 1 and 4 with 10 binary variables per split node.




**(a)** Warm start of cases 1,2 and 3, time limit 120 seconds.

**(b)** Warm start of cases 4 and 5, time limit 3600 seconds.

**Figure 5.4:** Results from warm start using first D3 with 10 binary variables per split, followed by D2 using 25 binary variables per split.

This means that more detailed quality models may be used for scenario analysis and Monte Carlo simulations to analyze infrastructure developments or to perform security of supply analyses, reliability tests and flow capacity analyses. The approach will also enable us to solve more detailed investment problems.

For future research, replacing the warm start of the high-resolution discrete model with a continuous global optimization model would be interesting, as it would often provide a good initial solution for the global optimization solver. Utilizing split fractions from the solution of a previous full discretized model (D2 or D3) for a successive similar problem could prove beneficial when solving many similar optimization problems as part of a Monte Carlo simulation. Our approach also appears to be well suited for integration in the algorithm with successively finer discretization proposed by Castro and Teles [2013] and Kolodziej et al. [2013]. Also, in our case the feasible region is limited by the flow pressure constraints, and the significance of this effect on the pooling solutions would be interesting to investigate further.

## Acknowledgements

This research was funded by The Research Council of Norway, project numbers 175967 and 176089. We would like to thank Matthias Nowak for originally suggesting the binary split formulation.

## Bibliography

Adhya, N., Tawarmalani, M., Sahinidis, N., 1999. A Lagrangian approach to the pooling problem. Industrial & Engineering Chemistry Research 38 (5), 1956–1972.

Alfaki, M., Haugland, D., 2011. Comparison of discrete and continuous models for the pooling problem. Models and Solution Methods for the Pooling Problem.

Alfaki, M., Haugland, D., 2013. A multi-commodity flow formulation for the generalized pooling problem. Journal of Global Optimization 56, 917–937.

Androulakis, I., Maranas, C., Floudas, C., 1995. $\alpha$bb: A global optimization method for general constrained nonconvex problems. Journal of Global Optimization 7 (4), 337–363.

Audet, C., Brimberg, J., Hansen, P., Le Digabel, S., Mladenović, N., 2004. Pooling problem: Alternate formulations and solution methods. Management Sscience, 761–776.

Audet, C., Hansen, P., Jaumard, B., Savard, G., 2000. A branch and cut algorithm for nonconvex quadratically constrained quadratic programming. Mathematical Programming 87 (1), 131–152.

Baker, T., Lasdon, L., 1985. Successive linear programming at Exxon. Management Science, 264–274.

Ben-Tal, A., Eiger, G., Gershovitz, V., 1994. Global minimization by reducing the duality gap. Mathematical Programming 63 (1), 193–212.

Castro, P. M., Teles, J. P., 2013. Comparison of global optimization algorithms for the design of water-using networks. Computers & Chemical Engineering 52, 249–261.

Faria, D. C., Bagajewicz, M. J., 2012. A new approach for global optimization of a class of MINLP problems with applications to water management and pooling problems. AIChE Journal 58 (8), 2320–2335.

Foulds, L. R., Haugland, D., Jørnsten, K., 1992. A bilinear approach to the pooling problem. Optimization 24, 165–180.

Frimannslund, L., Haugland, D., 2009. Parallel solution of the pooling problem with application to the cell broadband engine architecture. In: International Conference on Computers & Industrial Engineering, 2009. CIE 2009. IEEE, pp. 354–359.

Gounaris, C., Misener, R., Floudas, C., 2009. Computational comparison of piecewise-linear relaxations for pooling problems. Industrial & Engineering Chemistry Research 48 (12), 5742–5766.

Gupte, A., Ahmed, S., Dey, S. S., Cheon, M. S., 2012. Pooling problem.
URL `www.optimization-online.org/DB_FILE/2012/10/3658.pdf`

Haverly, C., 1978. Studies of the behavior of recursion for the pooling problem. ACM SIGMAP Bulletin 25, 19–28.

Haverly, C., 1979. Behavior of recursion model—more studies. ACM SIGMAP Bulletin (26), 22–28.

Haverly, C., 1980. Recursion model behavior: more studies. ACM SIGMAP Bulletin (28), 39–41.

Hellemo, L., Midthun, K., Tomasgard, A., Werner, A., 2012a. Multi-stage stochastic programming for natural gas infrastructure design with a production perspective. In: Ziemba, W. T., Wallace, S. W., Gassman, H. I. (Eds.), Stochastic programming – Applications in finance, energy, planning and logistics. Vol. 4 of World Scientific series in finance. World Scientific, 2012, pp. 259–288.

Hellemo, L., Midthun, K., Tomasgard, A., Werner, A., 2012b. Natural Gas Infrastructure Design with an Operational Perspective. Energy Procedia 26, 67–73.

Kocis, G., Grossmann, I., 1989. A modelling and decomposition strategy for the MINLP optimization of process flowsheets. Computers & Chemical Engineering 13 (7), 797–819.

Kolodziej, S., Castro, P. M., Grossmann, I. E., 2013. Global optimization of bilinear programs with a multiparametric disaggregation technique. Journal of Global Optimization 57 (4), 1039–1063.

Lee, S., Grossmann, I., 2003. Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks. Computers & Chemical Engineering 27 (11), 1557–1575.

Li, X., Armagan, E., Tomasgard, A., Barton, P. I., 2011a. Stochastic pooling problem for natural gas production network design and operation under uncertainty. AIChE Journal 57, 2120–2135.

Li, X., Tomasgard, A., Barton, P., 2011b. Decomposition strategy for the stochastic pooling problem. Journal of Global Optimization 54, 765–790.

Lodwick, W., 1992. Preprocessing nonlinear functional constraints with applications to the pooling problem. ORSA Journal on Computing 4 (2), 119–131.

McCormick, G. P., 1976. Computability of global solutions to factorable nonconvex programs: Part I – Convex underestimating problems. Mathematical programming 10 (1), 147–175.

Meyer, C., Floudas, C., 2006. Global optimization of a combinatorially complex generalized pooling problem. AIChE journal 52 (3), 1027–1037.

Misener, R., Floudas, C., 2009. Advances for the pooling problem: Modeling, global optimization, and computational studies. Applied and Computational Mathematics 8 (1), 3–22.

Misener, R., Thompson, J. P., Floudas, C. A., 2011. Apogee: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. Computers & Chemical Engineering 35 (5), 876–892.

Pham, V., Laird, C., El-Halwagi, M., 2009. Convex hull discretization approach to the global optimization of pooling problems. Industrial & Engineering Chemistry Research 48 (4), 1973–1979.

Quesada, I., Grossmann, I., 1995. Global optimization of bilinear process networks with multicomponent flows. Computers & Chemical Engineering 19 (12), 1219–1242.

Rømo, F., Tomasgard, A., Hellemo, L., Fodstad, M., Eidesen, B., Pedersen, B., 2009. Optimizing the Norwegian natural gas production and transport. Interfaces 39 (1), 46–56.

Ruiz, J. P., Grossmann, I. E., 2011. Using redundancy to strengthen the relaxation for the global optimization of MINLP problems. Computers & Chemical Engineering 35 (12), 2729–2740.

Selot, A., Kuok, L., Robinson, M., Mason, T., Barton, P., 2008. A short-term operational planning model for natural gas production systems. AIChE Journal 54 (2), 495–515.

Tawarmalani, M., Sahinidis, N. V., 2002. Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming. Kluwer Academic Publishers, Norwell, MA, USA.

The Norwegian Ministry of Petroleum and Energy and The Norwegian Petroleum Directorate, 2014. Facts 2014 — The Norwegian petroleum sector. URL `http://www.npd.no/en/Publications/Facts/Facts-2014/`.

Tomasgard, A., Rømo, F., Fodstad, M., Midthun, K., 2007. Optimization models for the natural gas value chain. In: Hasle, G., Lie, K.-A., Quak, E. (Eds.), Geometric Modelling, Numerical Simulation and Optimization. Springer Verlag, pp. 521–558.

Ulstein, N. L., 2000. Short Term Planning of Gas Production (In Norwegian). Masters thesis, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology.

Ulstein, N. L., Nygreen, B., Sagli, J. R., January 2007. Tactical planning of offshore petroleum production. European Journal of Operational Research 176 (1), 550–564.

Visweswaran, V., Floudas, C., 1990. A global optimization algorithm (GOP) for certain classes of nonconvex NLPs — II. Application of theory and test problems. Computers & Chemical Engineering 14 (12), 1419–1434.

## 5.A Nomenclature

### Sets

| | |
|---|---|
| $\mathcal{B}$ | Nodes where gas flows are split into two or more pipelines. $\mathcal{B} \subset \mathcal{N}$ |
| $\mathcal{C}$ | Components defining the chemical content of the natural gas. |
| $\mathcal{I}(n)$ | Nodes with pipelines going into node $n \in \mathcal{N}, \mathcal{I} \subset \mathcal{N}$. |
| $\mathcal{M}$ | Market nodes (sinks) in the network, $\mathcal{M} \subset \mathcal{N}$. |
| $\mathcal{N}$ | All nodes in the network. |
| $\mathcal{O}(n)$ | Nodes with pipelines going out from node $n \in \mathcal{N}, \mathcal{O}(n) \subset \mathcal{N}$. |
| $\mathcal{P}$ | Production nodes (sources) in the network, $\mathcal{P} \subset \mathcal{N}$. |
| $\mathcal{Z}$ | Split percentages used to discretize potential split fractions in the network. |

### Parameters

| | |
|---|---|
| $\alpha_{nj,S}$ | Split fraction calculated from solution of A1, $n \in \mathcal{N}, j \in \mathcal{O}(n)$ |
| $\alpha_{nj,z}$ | Split fraction for $z \in \{1, 2, \ldots, Z\}$, $n \in \mathcal{N}, j \in \mathcal{O}(n)$ |
| $\overline{CO_{2_m}}$ | Upper bound on $CO_2$ content in gas delivered to market $m \in \mathcal{M}$ |
| $D_m$ | Demand from market $m \in \mathcal{M}$ |
| $E_{nj}$ | Maximum flow from node $n \in \mathcal{N}$ to $j \in \mathcal{O}(n)$ |
| $F_{nj}$ | Pre-computed flow from node $n \in \mathcal{N}$ to node $j \in \mathcal{O}(n)$. |
| $\Phi_{ij}$ | Pre-computed fraction of flow going from node $i \in \mathcal{N}$ to node $j \in \mathcal{O}(i)$. |
| $\overline{GCV_m}$ | Upper bound on GCV of gas delivered to market $m \in \mathcal{M}$ |
| $\underline{GCV_m}$ | Lower bound on GCV of gas delivered to market $m \in \mathcal{M}$ |
| $\overline{GCV^c}$ | Energy content of component $c \in \mathcal{C}$ |
| $P_p$ | Production capacity of node $p \in \mathcal{P}$ |
| $\Pi_m$ | Price in market $m \in \mathcal{M}$ |
| $Q_i^c$ | Given fraction of component $c \in \mathcal{C}$ delivered from node $i \in \mathcal{P}$. |
| $Z$ | Number of binary variables to represent split options |
| $Z_U$ | Number of split options for values greater than $\alpha_{nj,S}$ |
| $Z_L$ | Number of split options for values smaller than $\alpha_{nj,S}$ |

### Variables

| | |
|---|---|
| $e_{ij}^{cz}$ | Partial flow of component $c \in \mathcal{C}$ from $i$ to $j$ for $z \in \{1, 2, \ldots, Z\}, i \in \mathcal{N}, j \in \mathcal{O}(i)$ |
| $f_{ij}^c$ | Component flow for component $c \in \mathcal{C}$ from node $i \in \mathcal{N}$ to node $j \in \mathcal{O}(i)$. |
| $f_{ij}$ | Total flow from node $i \in \mathcal{N}$ to node $j \in \mathcal{O}(i)$. |
| $\lambda_{nj}^z$ | Possible split option for node $n \in \mathcal{B}$ where $z \in \{1, 2, \ldots, Z\}, j \in \mathcal{O}(n)$ |

**Table 5.1:** Number of network elements in each test case. Natural gas inserted into the network in production nodes (sources) is transported through pipelines that are connected by transport nodes. Some transport nodes are split nodes, others simple junctions, and others again have processing capabilities. The gas finally reaches markets (sinks).

| Case | Elements | Production | Pipelines | Markets | Split Nodes |
|---|---|---|---|---|---|
| 1 | 181 | 7 | 99 | 11 | 33 |
| 2 | 181 | 10 | 97 | 8 | 36 |
| 3 | 697 | 10 | 368 | 11 | 116 |
| 4 | 201 | 7 | 117 | 12 | 39 |
| 5 | 201 | 7 | 117 | 12 | 39 |

## 5.B Test instances

The size of each network is given in Table 5.1. The resulting sizes of the MILPs for each instance are given in Table 5.2.

**Table 5.2:** Problem sizes for each test instance. The table shows case number (Case), binary variables per split node (BVs), discretization scheme (D), number of constraints (NumConstrs), number of variables (NumVars) and number of integer (binary) variables (NumIntVars).

| Case | BVs | D | NumConstrs | NumVars | NumIntVars |
|---|---|---|---|---|---|
| 1 | 10 | 1 | 18,920 | 7,795 | 1,064 |
| 1 | 10 | 2 | 18,920 | 7,795 | 1,064 |
| 1 | 10 | 3 | 18,954 | 7,795 | 1,064 |
| 1 | 20 | 1 | 32,700 | 12,215 | 1,404 |
| 1 | 20 | 2 | 32,700 | 12,215 | 1,404 |
| 1 | 20 | 3 | 32,734 | 12,215 | 1,404 |
| 1 | 50 | 1 | 74,040 | 25,475 | 2,424 |
| 1 | 50 | 2 | 74,040 | 25,475 | 2,424 |
| 1 | 50 | 3 | 74,074 | 25,475 | 2,424 |
| 2 | 10 | 1 | 18,489 | 7,460 | 1,028 |
| 2 | 10 | 2 | 18,489 | 7,460 | 1,028 |

**Table 5.2:** Problem sizes for each test instance. The table shows case number (Case), binary variables per split node (BVs), discretization scheme (D), number of constraints (NumConstrs), number of variables (NumVars) and number of integer (binary) variables (NumIntVars).

| Case | BVs | D | NumConstrs | NumVars | NumIntVars |
|---|---|---|---|---|---|
| 2 | 10 | 3 | 18,521 | 7,460 | 1,028 |
| 2 | 20 | 1 | 31,889 | 11,620 | 1,348 |
| 2 | 20 | 2 | 31,889 | 11,620 | 1,348 |
| 2 | 20 | 3 | 31,921 | 11,620 | 1,348 |
| 2 | 50 | 1 | 72,089 | 24,100 | 2,308 |
| 2 | 50 | 2 | 72,089 | 24,100 | 2,308 |
| 2 | 50 | 3 | 72,121 | 24,100 | 2,308 |
| 3 | 10 | 1 | 59,737 | 28,305 | 3,938 |
| 3 | 10 | 2 | 59,737 | 28,305 | 3,938 |
| 3 | 10 | 3 | 59,852 | 28,305 | 3,938 |
| 3 | 20 | 1 | $1 \cdot 10^5$ | 43,255 | 5,088 |
| 3 | 20 | 2 | $1 \cdot 10^5$ | 43,255 | 5,088 |
| 3 | 20 | 3 | $1 \cdot 10^5$ | 43,255 | 5,088 |
| 3 | 50 | 1 | $2.21 \cdot 10^5$ | 88,105 | 8,538 |
| 3 | 50 | 2 | $2.21 \cdot 10^5$ | 88,105 | 8,538 |
| 3 | 50 | 3 | $2.21 \cdot 10^5$ | 88,105 | 8,538 |
| 4 | 10 | 1 | $3.33 \cdot 10^5$ | $1.06 \cdot 10^5$ | 14,503 |
| 4 | 10 | 2 | $3.33 \cdot 10^5$ | $1.06 \cdot 10^5$ | 14,503 |
| 4 | 10 | 3 | $3.33 \cdot 10^5$ | $1.06 \cdot 10^5$ | 14,503 |
| 4 | 20 | 1 | $5.33 \cdot 10^5$ | $1.71 \cdot 10^5$ | 19,453 |
| 4 | 20 | 2 | $5.33 \cdot 10^5$ | $1.71 \cdot 10^5$ | 19,453 |
| 4 | 20 | 3 | $5.34 \cdot 10^5$ | $1.71 \cdot 10^5$ | 19,453 |
| 4 | 50 | 1 | $1.13 \cdot 10^6$ | $3.64 \cdot 10^5$ | 34,303 |
| 4 | 50 | 2 | $1.13 \cdot 10^6$ | $3.64 \cdot 10^5$ | 34,303 |
| 4 | 50 | 3 | $1.13 \cdot 10^6$ | $3.64 \cdot 10^5$ | 34,303 |
| 5 | 10 | 1 | $5.83 \cdot 10^5$ | $2.01 \cdot 10^5$ | 19,453 |
| 5 | 10 | 2 | $5.83 \cdot 10^5$ | $2.01 \cdot 10^5$ | 19,453 |
| 5 | 10 | 3 | $5.84 \cdot 10^5$ | $2.01 \cdot 10^5$ | 19,453 |
| 5 | 20 | 1 | $9.84 \cdot 10^5$ | $3.3 \cdot 10^5$ | 29,353 |
| 5 | 20 | 2 | $9.84 \cdot 10^5$ | $3.3 \cdot 10^5$ | 29,353 |
| 5 | 20 | 3 | $9.85 \cdot 10^5$ | $3.3 \cdot 10^5$ | 29,353 |

**Table 5.2:** Problem sizes for each test instance. The table shows case number (Case), binary variables per split node (BVs), discretization scheme (D), number of constraints (NumConstrs), number of variables (NumVars) and number of integer (binary) variables (NumIntVars).

| Case | BVs | D | NumConstrs | NumVars | NumIntVars |
|---|---|---|---|---|---|
| 5 | 50 | 1 | $2.19 \cdot 10^6$ | $7.16 \cdot 10^5$ | 59,053 |
| 5 | 50 | 2 | $2.19 \cdot 10^6$ | $7.16 \cdot 10^5$ | 59,053 |
| 5 | 50 | 3 | $2.19 \cdot 10^6$ | $7.16 \cdot 10^5$ | 59,053 |

## 5.C Additional results

**Table 5.3:** Time to solve the linear auxiliary problems for each test instance in seconds.

| Case | Homogeneous Flow A1 | Fixed Total Flow A2 |
|---|---|---|
| 1 | $9.62 \cdot 10^{-3}$ | $3.8 \cdot 10^{-3}$ |
| 2 | $1.2 \cdot 10^{-2}$ | $4.1 \cdot 10^{-3}$ |
| 3 | $3.24 \cdot 10^{-2}$ | $-1$ |
| 4 | 7.36 | 0.12 |
| 5 | 1.63 | 0.27 |

**Table 5.4:** Results after 120 seconds. The table shows case number (Case), number of binary variables per split node (BVs), discretization scheme (D), time in seconds (T), lower bound (LB), upper bound (UB), best available solution (Optimal), optimality gap betwen LB and UB (D-Gap) and the gap between LB and Optimal (Gap).

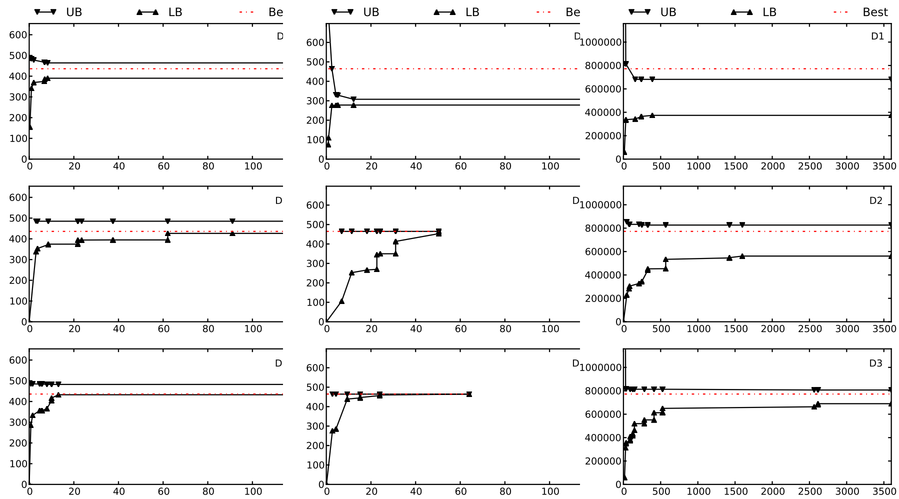
| Case | BVs | D | T[s] | LB | UB | Optimal | D-Gap[%] | Gap[%] |
|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 1 | 6.7 | $1.9 \cdot 10^2$ | 242.3 | 305.7 | $2.7 \cdot 10^1$ | $6 \cdot 10^1$ |
| 1 | 10 | 2 | 31.6 | $3.0 \cdot 10^2$ | 308.9 | 305.7 | $2.7 \cdot 10^0$ | $1.6 \cdot 10^0$ |
| 1 | 10 | 3 | 109.7 | $3.0 \cdot 10^2$ | 308.9 | 305.7 | $1.3 \cdot 10^0$ | $2.6 \cdot 10^{-1}$ |

**Table 5.4:** Results after 120 seconds. The table shows case number (Case), number of binary variables per split node (BVs), discretization scheme (D), time in seconds (T), lower bound (LB), upper bound (UB), best available solution (Optimal), optimality gap betwen LB and UB (D-Gap) and the gap between LB and Optimal (Gap).

| Case | BVs | D | T[s] | LB | UB | Optimal | D-Gap[%] | Gap[%] |
|---|---|---|---|---|---|---|---|---|
| 1 | 20 | 1 | 17.9 | $2.3 \cdot 10^2$ | 248.3 | 305.7 | $6.2 \cdot 10^0$ | $3.1 \cdot 10^1$ |
| 1 | 20 | 2 | 129.2 | $3.0 \cdot 10^2$ | 308.9 | 305.7 | $4.1 \cdot 10^0$ | $3 \cdot 10^0$ |
| 1 | 20 | 3 | 66.9 | $3.0 \cdot 10^2$ | 308.9 | 305.7 | $2.1 \cdot 10^0$ | $9.9 \cdot 10^{-1}$ |
| 1 | 50 | 1 | 17.6 | $2.2 \cdot 10^2$ | 283.1 | 305.7 | $2.8 \cdot 10^1$ | $3.8 \cdot 10^1$ |
| 1 | 50 | 2 | 167.2 | $3.0 \cdot 10^2$ | 308.9 | 305.7 | $2.6 \cdot 10^0$ | $1.5 \cdot 10^0$ |
| 1 | 50 | 3 | 120.4 | $3.0 \cdot 10^2$ | 308.9 | 305.7 | $1.8 \cdot 10^0$ | $6.9 \cdot 10^{-1}$ |
| 2 | 10 | 1 | 8.2 | $3.9 \cdot 10^2$ | 464.0 | 435.9 | $1.9 \cdot 10^1$ | $1.2 \cdot 10^1$ |
| 2 | 10 | 2 | 91.1 | $4.3 \cdot 10^2$ | 485.0 | 435.9 | $1.4 \cdot 10^1$ | $2.3 \cdot 10^0$ |
| 2 | 10 | 3 | 13.1 | $4.3 \cdot 10^2$ | 482.3 | 435.9 | $1.2 \cdot 10^1$ | $7.9 \cdot 10^{-1}$ |
| 2 | 20 | 1 | 23 | $3.9 \cdot 10^2$ | 466.0 | 435.9 | $1.9 \cdot 10^1$ | $1.2 \cdot 10^1$ |
| 2 | 20 | 2 | 150.2 | $4.2 \cdot 10^2$ | 485.0 | 435.9 | $1.5 \cdot 10^1$ | $3.8 \cdot 10^0$ |
| 2 | 20 | 3 | 79.7 | $4.3 \cdot 10^2$ | 482.3 | 435.9 | $1.2 \cdot 10^1$ | $7.9 \cdot 10^{-1}$ |
| 2 | 50 | 1 | 69.5 | $3.9 \cdot 10^2$ | 432.7 | 435.9 | $1.1 \cdot 10^1$ | $1.2 \cdot 10^1$ |
| 2 | 50 | 2 | 70.5 | $2.9 \cdot 10^2$ | 485.0 | 435.9 | $6.8 \cdot 10^1$ | $5.1 \cdot 10^1$ |
| 2 | 50 | 3 | 111.1 | $4.0 \cdot 10^2$ | 485.5 | 435.9 | $2.1 \cdot 10^1$ | $8.4 \cdot 10^0$ |
| 3 | 10 | 1 | 12.2 | $2.8 \cdot 10^2$ | 307.6 | 464.7 | $1.1 \cdot 10^1$ | $6.7 \cdot 10^1$ |
| 3 | 10 | 2 | 50.4 | $4.6 \cdot 10^2$ | 464.7 | 464.7 | $2 \cdot 10^{-4}$ | $-5.6 \cdot 10^{-5}$ |
| 3 | 10 | 3 | 64 | $4.6 \cdot 10^2$ | 464.3 | 464.7 | $4.1 \cdot 10^{-4}$ | $7.7 \cdot 10^{-2}$ |
| 3 | 20 | 1 | 43.3 | $2.8 \cdot 10^2$ | 307.6 | 464.7 | $1.1 \cdot 10^1$ | $6.7 \cdot 10^1$ |
| 3 | 20 | 2 | 116.1 | $4.6 \cdot 10^2$ | 464.7 | 464.7 | $1.7 \cdot 10^{-4}$ | $-8.8 \cdot 10^{-5}$ |
| 3 | 20 | 3 | 37.2 | $4.6 \cdot 10^2$ | 464.7 | 464.7 | $9.4 \cdot 10^{-4}$ | $6.8 \cdot 10^{-4}$ |
| 3 | 50 | 1 | 77.3 | $2.8 \cdot 10^2$ | 324.3 | 464.7 | $1.7 \cdot 10^1$ | $6.7 \cdot 10^1$ |
| 3 | 50 | 2 | 134.4 | $3.4 \cdot 10^2$ | 464.7 | 464.7 | $3.8 \cdot 10^1$ | $3.8 \cdot 10^1$ |
| 3 | 50 | 3 | 149.4 | $4.6 \cdot 10^2$ | 464.7 | 464.7 | $1 \cdot 10^{-1}$ | $1 \cdot 10^{-1}$ |

**Table 5.5:** Results after 3600 seconds

| Case | BVs | D | T[s] | LB | UB | Optimal | D-Gap[%] | Gap[%] |
|---|---|---|---|---|---|---|---|---|
| 4 | 10 | 1 | 1,685.5 | $4.9 \cdot 10^4$ | $1.5 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^2$ | $3.2 \cdot 10^2$ |
| 4 | 10 | 2 | 3,548.3 | $1.9 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^5$ | $1.2 \cdot 10^1$ | $1.1 \cdot 10^1$ |
| 4 | 10 | 3 | 1,709.1 | $2.0 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^5$ | $3.3 \cdot 10^{-1}$ | $2.4 \cdot 10^{-1}$ |
| 4 | 20 | 1 | 3,046 | $4.9 \cdot 10^4$ | $1.5 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^2$ | $3.2 \cdot 10^2$ |
| 4 | 20 | 2 | 3,609.9 | $1.3 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^5$ | $5.6 \cdot 10^1$ | $5.3 \cdot 10^1$ |
| 4 | 20 | 3 | 609.9 | $1.8 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^5$ | $1.8 \cdot 10^1$ | $1.7 \cdot 10^1$ |
| 4 | 50 | 1 | 554 | $4.9 \cdot 10^4$ | $1.6 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.3 \cdot 10^2$ | $3.2 \cdot 10^2$ |
| 4 | 50 | 2 | 1,005.4 | $1.2 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^5$ | $7.7 \cdot 10^1$ | $7.5 \cdot 10^1$ |
| 4 | 50 | 3 | 1,663.8 | $1.8 \cdot 10^5$ | $2.1 \cdot 10^5$ | $2.1 \cdot 10^5$ | $1.3 \cdot 10^1$ | $1.2 \cdot 10^1$ |
| 5 | 10 | 1 | 388.4 | $3.7 \cdot 10^5$ | $6.8 \cdot 10^5$ | $7.7 \cdot 10^5$ | $8.2 \cdot 10^1$ | $1.1 \cdot 10^2$ |
| 5 | 10 | 2 | 1,597.8 | $5.6 \cdot 10^5$ | $8.3 \cdot 10^5$ | $7.7 \cdot 10^5$ | $4.7 \cdot 10^1$ | $3.8 \cdot 10^1$ |
| 5 | 10 | 3 | 2,614.2 | $6.9 \cdot 10^5$ | $8.1 \cdot 10^5$ | $7.7 \cdot 10^5$ | $1.7 \cdot 10^1$ | $1.2 \cdot 10^1$ |
| 5 | 20 | 1 | 1,159.5 | $3.6 \cdot 10^5$ | $6.7 \cdot 10^5$ | $7.7 \cdot 10^5$ | $8.6 \cdot 10^1$ | $1.1 \cdot 10^2$ |
| 5 | 20 | 2 | 3,724.5 | $5.5 \cdot 10^5$ | $8.3 \cdot 10^5$ | $7.7 \cdot 10^5$ | $5 \cdot 10^1$ | $4 \cdot 10^1$ |
| 5 | 20 | 3 | 1,544.9 | $5.8 \cdot 10^5$ | $8.2 \cdot 10^5$ | $7.7 \cdot 10^5$ | $4.2 \cdot 10^1$ | $3.4 \cdot 10^1$ |
| 5 | 50 | 1 | 3,202.5 | $3.6 \cdot 10^5$ | $6.7 \cdot 10^5$ | $7.7 \cdot 10^5$ | $8.6 \cdot 10^1$ | $1.1 \cdot 10^2$ |
| 5 | 50 | 2 | 2,660.7 | $4.6 \cdot 10^5$ | $8.3 \cdot 10^5$ | $7.7 \cdot 10^5$ | $8.1 \cdot 10^1$ | $6.9 \cdot 10^1$ |
| 5 | 50 | 3 | 2,981.4 | $4.3 \cdot 10^5$ | $8.2 \cdot 10^5$ | $7.7 \cdot 10^5$ | $9 \cdot 10^1$ | $7.9 \cdot 10^1$ |




**(a)** Case 2, over 120 seconds. **(b)** Case 3, over 120 seconds. **(c)** Case 5, over 3600 seconds.

**Figure 5.5:** Convergence of discretization lower and upper bounds and best available solution in cases 2, 3, and 5 with 10 binary variables per split node.

# Paper V

Lars Hellemo and Asgeir Tomasgard:

# A Generalized Global Optimization Formulation of the Pooling Problem with Processing Facilities and Composite Quality Constraints

*Accepted for publication in TOP*

# Chapter 6

# A Generalized Global Optimization Formulation of the Pooling Problem with Processing Facilities and Composite Quality Constraints

**Abstract:**

We present a generalized formulation of the pooling problem. Our formulation is different from the standard formulations in explicitly modelling component flows. Modelling the physical components directly, allows easy inclusion of processing facilities that may alter the flow composition. It also allows adding composite quality constraints that can not be added directly as quality parameters as they do not blend linearly.
We provide new test instances motivated by natural gas transport problems at the Norwegian Continental Shelf and give computational results. We show examples of nonlinear composite constraints on quality attributes and give computational results on the effect of adding such constraints to the new set of test instances.
The increased flexibility of our formulation comes at the cost of less tight constraints and a performance penalty on existing test cases in literature. The advantage is that it can solve the more general test cases described above. We use GAMS implementations of the various formulations and solve the problems with BARON. We compare the performance of our bilinear formulation in BARON with discretizations solved as a Mixed Integer Linear Program (MILP) using CPLEX. The discretized versions do not generally perform as well as the continuous model, however they have better worst case behavior on the new test cases.

## 6.1 Introduction

Pooling problems arise whenever flows with different quality are blended in a network, and then subsequently sent in different directions where there are bounds on the allowed quality of the mixture. This is generally the situation when natural gas from fields with distinct composition is transported through a network with blending possibilities, such as the offshore transport network on the Norwegian Continental Shelf (NCS). At some point in the network, there will typically be processing facilities that remove some proportion of the gas entering the facility. The processed gas reenters the network downstream to be transported to the terminals.

We present a general formulation of the pooling problem, allowing for multiple levels of pools and alteration of flow composition at processing nodes. By modelling the physical components directly, the model is easily extended to include processing facilities that may alter the flow composition. It also allows adding composite quality constraints that could not be added directly as quality parameters as they do not blend linearly.

Formulating the quality as physical flow components is equivalent to using more general quality parameters, and this formulation may also be used for traditional pooling problems after a linear transformation of the quality attributes. This formulation is not as tight as state-of-the-art pooling formulations, however. We provide new test instances motivated by natural gas transport problems at the Norwegian Continental Shelf and give computational results. By modelling the flow of the physical gas components we may add constraints on quality attributes. Using such quality attribute parameters directly would break the linear blending assumption of standard pooling problems. We show examples of such constraints and give computational results on the effect of adding such constraints to the set of test instances.

Similar problems have previously been solved using Mixed Integer Linear Progams (MILP), and we compare our continuous formulation with two discretized reformulations. We also compare the performance of our formulation using available test instances from the literature.

We give a short overview of the literature concerning pooling problems in Section 6.2. The mathematical formulations are given in Section 6.3. Our problem instances are described in Section 6.4, and numerical results are presented in Section 6.5 before we conclude in Section 7.7.

## 6.2 Literature Review

Pooling problems can be traced back to Haverly [1978], who presented a classic problem with two sources, one pool, and two sinks and showed that pooling problems are computationally hard with many local optima. Haverly [1979, 1980] followed up on the original pooling formulation, and several studies were performed using successive linear programming (SLP), including, for instance, Baker and Lasdon [1985] up to more recent work by Frimannslund and Haugland [2009] who applied SLP with parallelization. SLP is an efficient algorithm, but is not guaranteed to converge to a global optimum.

Foulds et al [1992] were the first to apply a global optimization algorithm to the pooling problem, using McCormick underestimators and branch and bound

(BB). Lodwick [1992] developed pre-processing techniques that would find implicit bounds for pooling problems. Androulakis et al [1995] applied their $\alpha$BB technique to the pooling problem and Adhya et al [1999] developed a global optimization technique, with bounds based on Lagrangian relaxation in combination with branch and bound methods. Audet et al [2000] developed a branch and cut algorithm that was also applied to pooling problems.

Most efforts in solving pooling problems have been applying global optimization techniques, working on tighter formulations and solution algorithms. Two different, but equivalent formulations are widespread. The original formulation of Haverly [1978] where flow and quality are modeled is often called the P formulation. Ben-Tal et al [1994] introduced a different formulation based on the flow of individual components and proportions to enter the pool. This formulation is often referred to as the Q formulation. Kocis and Grossmann [1989] also considered a formulation based on flows and fractions going out from splitters. Quesada and Grossmann [1995] introduced a new formulation with extra constraints combining formulations P and Q, and this formulation is called PQ. Tawarmalani and Sahinidis [2002] discuss different global optimization formulations and prove that PQ gives tighter bounds than formulations P and Q.

Lee and Grossmann [2003] developed a two-level branch and bound algorithm and applied it to pooling problems in water management. Gounaris et al [2009] showed that bounds and convergence time can be improved by applying piece-wise linear relaxations. Discussing different approaches to solving pooling problems in a recent survey, Misener and Floudas [2009] give a good overview of the development during the early 2000s. Selot et al [2008] applied global optimization to natural gas operational problems with additional disjunctive constraints for contracts regulating production allowances. Recent advances in specialized algorithms for solving pooling problems include piece-wise MILP relaxations (Misener et al [2011]) and multiparametric disaggregation (Teles et al [2012], Castro and Teles [2013]), the latter motivated by water treatment problems.

Another approach to pooling problems is to take advantage of the great progress in MILP solvers and solve discretized pooling problems. Ulstein [2000], Ulstein et al [2007], Tomasgard et al [2007], and Rømo et al [2009], Hellemo and Werner [2014] applied discretization approaches to large pipeline networks for natural gas with pooling on multiple levels. Hellemo et al [2012a,b] applied discretization to pooling in network design problems. In comparing global optimization techniques and discretization Haugland [2010], Gupte [2012], Gupte et al [2012], Dey and Gupte [2015], find that a discretization approach to pooling problems may yield better performance than a global optimization approach.

The original pooling problems has been generalized both by extending to a network structure which allows interconnected pools and by including network design. Audet et al [2004] and Alfaki and Haugland [2013a] present a generalization of the PQ formulation for a general network structure. Meyer and Floudas [2006] generalized the problem to include network design, leading to combinatorial pooling problems. Li et al [2011a] consider a stochastic pooling problem with network design and operations. They present a global optimization decomposition algorithm for solving such problems in Li et al [2011b].

The formulation in this paper is another generalization of the pooling problem. Our formulation allows several layers of interconnected pools. The pools need not be directly connected to sources or sinks. This network structure is not represented in the set of test instances published by Alfaki and Haugland [2013a] or anywhere else to the best of our knowledge. We also present a pooling model where a simple model of a processing facility is included. At the processing facility, some components may be removed partially or completely before the remainder flows further downstream. We also demonstrate that modelling the physical components directly makes it relatively straightforward to add nonlinear composite quality constraints. We compare this formulation with other global optimization formulations in literature, both for a continuous, bilinear model and for the discretized MILP. We also investigate the performance on our added test-cases motivated from natural gas transport on the Norwegian Continental Shelf (NCS). The advantage of these test instances is that they are more representative of the typical problem instances we encounter in practice than randomly generated cases.

## 6.3 Formulation S

In this section we will first present the generalized pooling formulation, Formulation S in Section 6.3, then describe the transformation of standard test instances in Section 6.3, before we add the model of processing facilities in Section 6.3. Finally, we consider the discretizations D1 and D2 of this model in Section 6.3 and Section 6.3, respectively.

### The Generalized Pooling Model

We consider a directed acyclical graph $\mathcal{P} = (\mathcal{N}, \mathcal{A})$ with a set of nodes $\mathcal{N}$ and a set of arcs $\mathcal{A}$. The nodes downstream of a node $i$ are members of the set $\mathcal{O}(i) = \{j \in \mathcal{N} : (i,j) \in \mathcal{A}\}$. For each set $\mathcal{O}(i)$ there is a corresponding set $\mathcal{O}^-(i)$ where one arbitrary element $j \in \mathcal{O}(i)$ is removed such that $\mathcal{O}(i) \setminus \mathcal{O}^-(i) = \{j\}$. The nodes upstream of a node $j$ are members of the set $\mathcal{I}(j) = \{i \in \mathcal{N} : (i,j) \in \mathcal{A}\}$. Let there be non-empty sets $\mathcal{S}, \mathcal{B}, \mathcal{T} \subset \mathcal{N}$, denominating the sources, split nodes and terminals (sinks). Split nodes $b \in \mathcal{B}$ are nodes where $|\mathcal{O}(b)| > 1$ and $\mathcal{B} \subset \mathcal{N} \setminus \mathcal{S}$. Associated with each split node $b$ and each node $j \in \mathcal{O}^-(b)$ are split variables $y_{bj} \in [0,1]$ The lower and upper bounds on flow through each node $i \in \mathcal{N}$ are $F_i^L$

and $F_i^U$, respectively. The net unit cost $c_{ij}$ includes both cost and revenue that apply on each arc $(i,j)$. The set of flow components is denoted $\mathcal{C}$ and the flow $f_{ij}$ on the arc $(i,k)$ can be broken into component flows $f_{ij}^c$ where $c \in \mathcal{C}$. The quality parameters $Q_{ic}^L$ and $Q_{ic}^U$ are lower and upper bounds, respectively, on the proportion of component $c$ relative to the total flow entering (leaving) the sink (source) $i$.

This formulation differs from the formulations traditionally used for the pooling problem in the sense that we trace the content of each component through the network explicitly, and all qualities as such are fractions of the total flow. Other quality measures may be emulated as virtual components through a simple linear transformation (see Section 6.3), which makes the formulations equivalent.

Minimize negative profit:

$$\min_{f,f^c,y} \sum_{(i,j)\in\mathcal{A}} c_{ij} f_{ij} \tag{6.1}$$

Subject to the following constraints:

The quality from a source is given by the quality attribute of the source:

$$f_{sj}^c = Q_{sc}^L f_{sj}, \quad \forall s \in \mathcal{S}, j \in \mathcal{O}(s), c \in \mathcal{C} \tag{6.2}$$

Flow through a node is bounded:

$$F_i^L \leq \sum_{j\in\mathcal{O}(j)} f_{ij} \leq F_i^U, \quad \forall i \in \mathcal{N} \tag{6.3}$$

$$F_j^L \leq \sum_{i\in\mathcal{I}(j)} f_{ij} \leq F_j^U, \quad \forall j \in \mathcal{N} \tag{6.4}$$

Quality bounds:

$$\sum_{i\in\mathcal{I}(j)} f_{ij}^c \leq \sum_{i\in\mathcal{I}(j)} f_{ij} Q_{jc}^U, \quad \forall j \in \mathcal{N}, \forall c \in \mathcal{C} \tag{6.5}$$

$$\sum_{i\in\mathcal{I}(j)} f_{ij}^c \geq \sum_{i\in\mathcal{I}(j)} f_{ij} Q_{jc}^L, \quad \forall j \in \mathcal{N}, \forall c \in \mathcal{C} \tag{6.6}$$

Quality mass balance:

$$\sum_{c\in\mathcal{C}} f_{ij}^c = f_{ij}, \quad \forall (i,j) \in \mathcal{A} \tag{6.7}$$

$$\sum_{i\in\mathcal{I}(l)} f_{il}^c = \sum_{j\in\mathcal{O}(l)} f_{lj}^c, \quad \forall l \in \mathcal{N}, \forall c \in \mathcal{C} \tag{6.8}$$

Splits (bilinear terms):

$$\sum_{i \in \mathcal{I}(b)} f_{ib}^c y_{bj} = f_{bj}^c, \quad \forall b \in \mathcal{B}, j \in \mathcal{O}^-(b), \forall c \in \mathcal{C} \tag{6.9}$$

$$\sum_{i \in \mathcal{I}(b)} f_{ib} y_{bj} = f_{bj}, \quad \forall b \in \mathcal{B}, j \in \mathcal{O}^-(b) \tag{6.10}$$

Sum of fractions:

$$\sum_{j \in \mathcal{O}(b)} y_{bj} \leq 1, \quad \forall b \in \mathcal{B} \tag{6.11}$$

Non-negativity:

$$f_{ij} \geq 0, f_{ij}^c \geq 0, \quad \forall (i,j) \in \mathcal{A}, c \in \mathcal{C} \tag{6.12}$$

We make some modelling choices in the above equations, and other, equivalent formulations are possible. For example, we include both (6.9) and (6.10) although (6.10) is redundant due to (6.7). Including redundant constraints may sometimes be advantageous, but we did not test these alternative formulations. We also choose to define these equations over $j \in \mathcal{O}^-(b)$ (rather than over $j \in \mathcal{O}(b)$) to reduce the number of bilinear terms. This gives inequality in (6.11) as the split fraction for $\mathcal{O}(b) \setminus \mathcal{O}^-(b)$ is implicit, while it would be defined with equality if the equations mentioned above were defined over $j \in \mathcal{O}(b)$.

## Transformation

Solving problems involving network flows is an important part of planning and operating the natural gas transport system from producing fields to market nodes. Depending on the reservoir, the gas may have distinct characteristics in terms of composition, and thereby energy content, as well as the amount of contaminants such as $CO_2$ and $H_2S$. The composition of gas may affect how much can be transported, how much it must be treated in processing facilities and whether it can be sold in a market with defined quality requirements.

Standard pooling models in literature assume linear blending. This applies well to several important quality considerations, such as the proportion of an undesired pollutant or the heating value (GCV) where the linear approximation is good. However, it does not apply to quality aspects that are non-linear in the flow composition, such as Wobbe Number (WN), Incomplete Combustion Factor (ICF) and Soot Index (SI), see e.g. UK Government [1996], Hornemann [1998], formulae are given in Section 6.3.

By modeling the physical component flows directly, we assume that the component flows blend linearly in all blending nodes, but we may still add constraints on quality aspects that depend non-linearly on the flow composition. This is of course also possible using other formulations if all component proportions are added as quality parameters. Modeling the component flows directly facilitates the addition of a simple representation of processing facilities as described in Section 6.3. Note that this violates the typical redundant constraint of tighter formulation which requires that the proportion through a pool stemming from each source will sum to one, as some amount may be removed at the processing plant.

In contrast, the test instances from literature consider quality without unit. Formulation S (Section 6.3) assumes the quality to signify the fraction of the flow for each component flow. We can easily transform the more general quality attribute $Q_{ic}$ to a virtual component flow $Q'_{ic}$ with the following linear transformation:

$$Q'_{ic} = \frac{Q_{ic}}{\sum_{c' \in \mathcal{C}} \max_{j \in \mathcal{N}} Q_{jc'}}, \forall i \in \mathcal{N}, c \in \mathcal{C} \tag{6.13}$$

This ensures that the fractions are kept within the interval $[0, 1]$ and that for all component flows $f^c_{ij}$, the sum of all component flow fractions is at most 1, $\sum_c f^c_{ij} \leq 1, \forall (i, j) \in \mathcal{A}$.

This approach may be more suitable in cases where the qualities are of very different dimensions. Methane is usually a dominating component in natural gas transport, with component flow fractions in the range of $[0.8-1.0]$. The pollutant $H_2S$, on the other hand, is usually measured in parts per million (PPM) due to the low concentration, and this difference in magnitude could lead to numerical difficulties.

The implementation of Alfaki and Haugland [2013b] only imposes upper bounds on quality, and introduces new quality attributes for lower bounds. We transform these upper bounds on the extra attributes back to lower bounds in the original quality attributes in our implementation of Formulation S .

## Processing

We add to the pooling model presented above a simple model of a processing facility $p \in \mathcal{P}$, which for each component $c \in \mathcal{C}$ has defined a removal factor $\rho_{pc} \in [0, 1]$ which is the fraction removed from the incoming flow to the processing facility $p$ of component $c$. The removed quantity is led to a designated sink $t_{pc}$. This represents a separation of the components in the mixed flow into its individual components very different from split nodes $b \in \mathcal{B}$ ($\mathcal{P} \cap \mathcal{B} = \emptyset$ and equations (6.9–6.11) do not apply). The extent to which component flows may be routed independently depends on the nature of the processing facility $p$, which is modelled through the constraints described below:

The following equations enforce that a fraction of each component flow is removed and sent to the corresponding tank downstream of $p$:

$$\sum_{i\in\mathcal{I}(p)} f_{ip}^c \rho_{pc} = f_{pt}^c, t = t_{pc}, p \in \mathcal{P}, c \in \mathcal{C} \tag{6.14}$$

For other components, the flow to a given tank must be zero:

$$f_{ptc'} = 0, t = t_{pc}, p \in \mathcal{P}, c' \neq c \in \mathcal{C} \tag{6.15}$$

The following constraints lets whatever remaining after processing proceed in the network, downstream of the processing facility $p$:

$$\sum_{i\in\mathcal{I}(p)} f_{ip}^c (1-\rho_{pc}) = \sum_{l\in\mathcal{O}(p)\setminus\{t_{pc}\}} f_{pl}^c, \forall p \in \mathcal{P}, c \in \mathcal{C} \tag{6.16}$$

## Discretization D1

We compare the performance of Formulation S with versions where the split fraction is discretized. There are several options available for discretization, and in the following we use the discretization D1, which is described in further detail in Hellemo and Werner [2014]. D1 uses $Z$ evenly spaced discretization points implemented with binary variables or as SOS1. We introduce the allowed split fractions $\alpha_{bj}^z \in [0,1]$ and auxiliary flow variables $e_{bj}^{cz}$. $E_{bj}$ denotes the maximum allowed flow through the arc $(b,j)$ and is used for the discretizations that follow. $E_{bj}^z$ is set to a value higher than the maximum flow to not further constrain the flow from the problem formulation above.

The total component flow equals the sum of the auxiliary flow variables:

$$f_{bj}^c = \sum_{z=1}^{Z} e_{bj}^{cz}, \ \forall\ b \in \mathcal{B}, j \in \mathcal{O}(b). \tag{6.17}$$

Equations (6.18a) and (6.18b) ensure that auxiliary flow variables $e_{bj}^{cz}$ corresponding to active $\lambda_{bj}^z$ are allowed to take non-zero value. Constraints (6.18b) are not strictly necessary, but improve the numerical accuracy of the component flows as the constraints are applied to individual flows in addition to the sum of flows. The candidate split fractions $\alpha_{bj}^z$ are given by $\alpha_{bj}^z = \frac{z-1}{Z-1}$ where $z \in \{1,2,\ldots,Z\}$.

$$\sum_{c\in\mathcal{C}} e_{bj}^{cz} \leq \alpha_{bj}^z E_{bj} \lambda_{bj}^z, \quad \forall b \in \mathcal{B}, j \in \mathcal{O}(b), \lambda_{bj}^z \in \{0,1\}, \tag{6.18a}$$

$$0 \leq e_{bj}^{cz} \leq \lambda_{bj}^z \alpha_{bj}^z E_{bj}, \quad \forall b \in \mathcal{B}, j \in \mathcal{O}(b), c \in \mathcal{C}, \lambda_{bj}^z \in \{0,1\} \tag{6.18b}$$

Each partial flow of a component flow equals the split fraction times the total inflow of this component:

$$e_{bj}^{cz} \leq \alpha_{bj}^{z} \sum_{i \in \mathcal{I}(b)} f_{ib}^{c}, \forall b \in \mathcal{B}, j \in \mathcal{O}(b), z \in \{1, 2, \dots, Z\} \tag{6.19}$$

$$\begin{gathered} \alpha_{bj}^{z} \sum_{i \in \mathcal{I}(b)} f_{ib}^{c} - e_{bj}^{cz} \leq \alpha_{bj}^{z} E_{bj} (1 - \lambda_{bj}^{z}), \\ \forall b \in \mathcal{B}, j \in \mathcal{O}(b), c \in \mathcal{C}, z \in \{1, 2, \dots, Z\} \end{gathered} \tag{6.20}$$

We make sure exactly one $\lambda_{bj}^{z}$ is positive for each node (this may be specified as an SOS1 over $z \in \{1, 2, \dots, Z\}$).

$$\sum_{z=1}^{Z} \lambda_{bj}^{z} = 1, \ \forall \ b \in \mathcal{B}, j \in \mathcal{O}(b). \tag{6.21}$$

## Discretization D2

Discretization D2 is also taken from Hellemo and Werner [2014] and uses split fractions based on binary numbers, where one or more selection variables $\lambda_{bj}$ may be equal to 1. In this way, several weights may be combined to form possible split fraction with a higher resolution than discretization D1, using the same number of binary variables.

The weight of each fraction is defined as follows:

$$\alpha_{bj}^{z} = \begin{cases} \frac{1}{2^{z}}, z \in \{1, 2, ..., Z - 1\} \\ \frac{1}{2^{z-1}}, z = Z \end{cases} \tag{6.22}$$

Otherwise the formulation is similar to D1, but does not include the constraints 6.21.

## Composite quality constraints

One advantage of modelling the physical component flow directly is that addition of nonlinear composite quality constraints is relatively straightforward. This is because the blending of the component flows in the pools is linear. Modelling the composite quality parameters directly would break this fundamental assumption of the model.

As an illustration of such nonlinear composite quality constraints we have added the quality constraints introduced in the UK market: Wobbe Number (WN), Incomplete Combustion Factor (ICF) and Soot Index (SI).

The Wobbe Index is given by the Gross Calorific Value (Upper Heating Value) GCV of the gas mixture divided by the square root of the relative density (specific gravity) of the gas mixture $\rho_{rel}$, see e.g. Hornemann [1998]:

$$\mathrm{WI} = \frac{GCV}{\sqrt{\rho_{rel}}} \tag{6.23}$$

The Incomplete Combustion Factor (ICF) and Soot Index (SI) are defined by UK Government [1996] as:

$$\mathrm{ICF} = \frac{\mathrm{WI} - 50.73 + 0.03\mathrm{PN}}{1.56} \tag{6.24}$$

$$\mathrm{SI} = 0.896 \tan^{-1}(0.0255 C_3H_8 - 0.0233 N_2 + 0.617) \tag{6.25}$$

where PN means the sum of the percentages by volume of propane and nitrogen in the equivalent mixture, $C_3H_8$ means the percentage by volume of propane in the equivalent mixture and $N_2$ means the percentage by volume of nitrogen in the equivalent mixture. UK Government [1996] defines equivalent mixture as a mix of methane, propane and nitrogen having the same characteristics as the original mix. The equivalent mixture is to be calculated such that the average number of carbon atoms and the ideal volumes is the same as for the original mix. The equivalent mix shall also have identical Wobbe Number, given a normalization of the volume.

We add the following expressions in the optimization model for markets $\mathcal{M}^{\mathrm{comp}}$ with composite constraints, using variables $e_{ci}$ to denote the flow of the equivalent mixture; $\forall i \in \mathcal{M}^{\mathrm{comp}} \subset \mathcal{M}$:

**Ideal volume** We add the following constraints to make sure the ideal volume of the equivalent mix $\mathrm{Vol}_i^{\mathrm{em}}$ equals the ideal volume of the original mix $\mathrm{Vol}_i^{\mathrm{om}}$. These equations make use of the constants $\mathrm{volC}_c$ for the ideal volume of gas component $c$.

$$\mathrm{Vol}_i^{\mathrm{om}} = \sum_{c \in \mathcal{C}} \mathrm{volC}_c f_{ci} \tag{6.26}$$

$$\mathrm{Vol}_i^{\mathrm{em}} = \sum_{c \in \mathcal{C}^{\mathrm{em}}} \mathrm{volC}_c e_{ci} \tag{6.27}$$

$$\mathrm{Vol}_i^{\mathrm{om}} = \mathrm{Vol}_i^{\mathrm{em}} \tag{6.28}$$

**Carbon atoms** We also require the average number of carbon atoms in the equivalent mix $\mathrm{AvgNumC}_i^{\mathrm{em}}$ equals the average number of carbon atoms in the original mix $\text{AvgNumC}_i^{\text{om}}$. The number of carbon atoms for each component is given by the constant $\text{numC}_c$.

$$\text{AvgNumC}_i^{\text{om}} = \sum_{c \in \mathcal{C}} \text{numC}_c f_{ci} \tag{6.29}$$

$$\text{AvgNumC}_i^{\text{em}} = \sum_{c \in \mathcal{C}^{\text{em}}} \text{numC}_c e_{ci} \tag{6.30}$$

$$\text{AvgNumC}_i^{\text{om}} = \text{AvgNumC}_i^{\text{em}} \tag{6.31}$$

**Wobbe Number** The Wobbe number of the equivalent mix $\text{wn}_i^{\text{em}}$ must match the Wobbe number of the original mix $\text{wn}_i^{\text{om}}$, respectively. The relative density of each component is given in the constant $\rho_c$ and the flows are normalized using normalization factor $\eta_i$:

$$\text{wn}_i^{\text{em}} = \frac{\sum_{c \in \mathcal{C}} \text{GCV}_i e_{ci} \eta_i}{\sqrt{\sum_{c \in \mathcal{C}} \rho_c e_{ci} \eta_i}} \tag{6.32}$$

$$\text{wn}_i^{\text{om}} = \frac{\sum_{c \in \mathcal{C}^{\text{om}}} \text{GCV}_i f_{ci}}{\sqrt{\sum_{c \in \mathcal{C}} \rho_c f_{ci}}} \tag{6.33}$$

$$\text{wn}_i^{\text{em}} = \text{wn}_i^{\text{om}}, \forall i \in \mathcal{M} \tag{6.34}$$

**ICF** The ICF is computed following the definition, using the variables for the equivalent flow for the components corresponding to $C_3H_8$, $e_{i,C_3H_8}$ and $N_2$, $e_{i,N_2}$.

$$\text{icf}_i = \frac{\text{wn}_i - 50.73 + 0.03(e_{i,C_3H_8} + e_{i,N_2})}{1.56} \tag{6.35}$$

**SI** As described above, the exact formula for calculating SI is:

$$\text{si}_i = 0.896 tan^{-1}(0.0255 e_{C_3H_8} - 0.0233 e_{N_2} + 0.617) \tag{6.36}$$

However, as BARON does not currently support trigonometric functions, we have replaced $tan^{-}1(x)$ in Equation (6.36) with the approximation $f(x)$ in Equation (6.37):

$$f(x) = \frac{8x}{3 + \sqrt{25 + (\frac{16x}{\pi})^2}} \tag{6.37}$$

To allow zero flow and avoid adding binary variables, we multiply both sides with the total flow to the terminal. Alternatively, the constraint may be added directly, but this will imply positive flow constraints on the terminals. We apply the constrains on all terminal nodes for which they are defined, $\forall i \in \mathcal{M}^{\mathrm{comp}} \subset \mathcal{M}$:

$$\sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{wn}_i \leq \sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{WN}_i^U \tag{6.38}$$

$$\sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{wn}_i \geq \sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{WN}_i^L \tag{6.39}$$

$$\sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{icf}_i \leq \sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{ICF}_i^U \tag{6.40}$$

$$\sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{si}_i \leq \sum_{k \in \mathcal{I}(i)} f_{ki}\mathrm{SI}_i^U \tag{6.41}$$

where $\mathrm{WN}_i^U$ and $\mathrm{WN}_i^L$ denote upper and lower bounds on Wobbe number, and $\mathrm{ICF}_i^U$ denotes upper bound on ICF and $\mathrm{SI}_i^U$ is the upper bound for SI for terminal $i$.

The constraints described in this section are only applied to the nonlinear Formulation S . To apply them to the MILP formulations D1 and D2, the constraints would need to be replaced by linear or piecewise linear approximations.

## 6.4 New Test Instances

We provide a set of new test instances based on the network for natural gas transport on the Norwegian Continental Shelf (NCS). The transport system on the NCS consists of almost 8000 km of subsea transport pipelines, operating under high pressure. The volumes transported through this system account for close to 20 % of European Natural Gas consumption.

Complete data sets from industry typically include business sensitive information that may not be published. We have constructed test instances that resemble real cases, using estimations based on openly available data. The network structure used in our NCS test instances is taken from Fodstad et al [2013]. In addition, we specify natural gas quality data estimated from openly available data in The Norwegian Ministry of Petroleum and Energy and The Norwegian Petroleum Directorate [2012]. To increase the number of test instances, we have drawn permutations of the allocation of quality data to entry nodes and increased the production capacity at each node randomly within 20%. This is intended to simulate possible variations over time as new fields are added or taken out of production, or as production from certain fields is increased or decreased.

We consider several cases: The NCS as multi-level pooling problems with and without processing facility added. For both cases we may add the nonlinear compostite quality constraints.

Where the NCS test case is extended with additional composite quality constraints, such constrains are added on only one sink. These constraints are valid in the UK, and are applied to node 16 in the network, the node representing the UK terminals. The parameter values of the bounds on these quality attributes are taken from UK Government [1996]. We also include test cases where we add a single processing node (node 12) to the network. This processing node removes a proportion ranging from 0.1 to 1.0 of 10 out of 12 components. We also consider the NCS case with both processing plant added and composite constraints for node 16. Note that the test instances where processing plant is added, production is a bit more flexible in allowing turning down production from fields.

See Figure 6.1 for an illustration of the network. The GAMS specifications of the test instances we add are available for download at `http://iot.ntnu.no/users/hellemo/Pooling/`.

## 6.5 Numerical Results

In the following section, we present the results from our numerical experiments. All problem instances were run with a maximum allowed time of one hour and with stopping criteria relative gap $1 \times 10^{-3}$ and absolute gap $1 \times 10^{-3}$ . See Section 6.A for hardware and software details. We first present the results from the new NCS motivated cases. For reference, we also include tests using standard multi-level pooling problems provided by Alfaki and Haugland [2013a]. The mean and median cpu times for all instances for a given case and formulation are given in Table 6.1. For the NCS case and the instances from the literature, we compare Formulation S and the MCF formulation of Alfaki and Haugland [2013a], for the NCS cases with processing facilities with and without composite constraints, we only consider Formulation S .

### NCS Case (NCS)

For the NCS base case, the full results are listed in Table 6.6. The performance penalty from applying the more general formulation is small. We see that our formulation is faster for some problem instances, while generally a bit slower. In the best case, our formulation is 0.14 seconds faster (See Table 6.2). The median performance penalty is 0.015 seconds and one instance (27) is not solved to optimality within the time limit. The average performance penalty is about 90 seconds compared with the MCF formulation of Alfaki and Haugland [2013a]. See Table 6.3 for an overview of the final gap for all formulations.

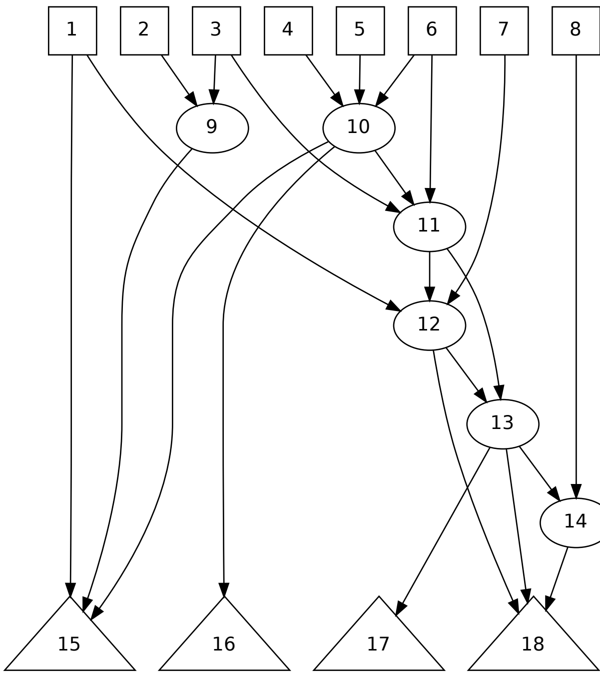



**Figure 6.1:** Test case, aggregated model of Norwegian Continental Shelf Gas Transport System. Sources are shown as rectangles, pools and intermediate nodes shown as ovals, and terminals shown as triangles.

## NCS Case with Processing Facilities (NCS P)

The full results are given in Table 6.4. For the NCS test instances with added processing facility at node 12 (NCS P), we may note that the overall objective function value improves. This is to be expected, as this model allows some components to be extracted at the processing facility and to be sold at a premium price. It also allows for contaminants to be removed, which may allow increased deliveries to downstream terminals. The value of the solution increases by about 3 % on average by adding the processing facility.

The computation time increases when processing facilities are added, but all instances except five are solved to optimality within the time limit of one hour. The average increase in computation time (including instances that did not complete within the time limit) is 364 seconds, while the median increase in computation time is about 3 second, compared with the instances in NCS without composite constraints.

When comparing the continuous formulation with the two discretized versions, we see that for most instances, the continuous formulation is solved much faster than D1 or D2 (median cpu time 3 or 4 times shorter). However, some cases are not solved to completion within the time limit, and the average time spent on each instance is much longer for the continuous formulation. All the discretized formulations complete within the time limit of one hour. The maximum cpu time for formulation D1 is just above 20 seconds, and for formulation D2 267 seconds.

## NCS Case with Composite Constraints (NCS C)

In the case where we add both processing facilities and composite constraints for one terminal, we see that the objective value decreases approximately by 4% on average for the cases where both have a solution available within the time limit. The penalty incurred in terms of increased computational effort is substantial. Half of the problem instances are not solved within the time limit of one hour. A few instances are solved faster than for the standard pooling instance with processing facility. Among the cases solved within the time limit, the average extra computational time amounts to 552 seconds, the median extra computational time is only 8 seconds. See Table 6.5 for details.

## Literature Test Instances (GPP)

For reference, we compare our formulation with state-of-the art formulations on a number of test instances from the literature. These test instances are taken from Alfaki and Haugland [2013a], and are all generalized pooling problems with multiple levels of pools. We expected our formulation to perform worse on these difficult test instances, and this is confirmed by the results in Table 6.9. In terms

**Table 6.1:** Overview of results per case and formulation. Mean and median cpu time in seconds. The cases shown are NCS traditional pooling problem (NCS), with composite constraints (NCS C), with processing (NCS P), with processing and composite constraints (NCS PC) and generalized pooling problem test intances from Alfaki and Haugland [2013a] (GPP).

| Formulation | NCS | NCS C | NCS P | NCS PC | GPP |
|---|---|---|---|---|---|
| MCF | 0.12 / 0.11 | | | | 200 / 0.58 |
| S | 91 / 0.11 | 1082 / 0.73 | 455 / 3 | 1204 / 23 | 1843 / 2377 |
| S-D1 | 5 / 0.56 | | 10 / 10 | | 2145 / 3600 |
| S-D2 | 6 / 0.31 | | 28 / 13 | | 2350 / 3600 |

**Table 6.2:** Difference in cpu time for NCS base case compared with MCF. Positive values mean slower than MCF.

| | S | S-D1 | S-D2 |
|---|---|---|---|
| Mean | 90.494 | 5.102 | 5.475 |
| Median | 0.015 | 0.460 | 0.215 |
| Min | -0.140 | -0.070 | -0.110 |
| Max | 3599.830 | 55.380 | 65.480 |

of computational runtime, our formulation is faster for two problem instances (L9 and L10), but generally slower, and for half of the instances, BARON has not converged within the one hour time limit. In the best case, our formulation is 132 seconds faster, the mean extra computation time is 165 seconds, and the average extra computational time 1642 seconds longer using our formulation. See Table 6.9 for the final gap for each formulation or Table 6.10 for full results.

## 6.6 Conclusions

We have presented a general formulation of the pooling problem that allows multiple levels of interconnected pools, and that is easily extended to include processing facilities. We compared the performance in BARON using the GAMS formulation of our Formulation S , a more general formulation that is easily extended with processing facilities and nonlinear composite quality constraints, as well as the state-of-the-art MCF formulation of Alfaki and Haugland [2013b].

We provide test instances motivated natural gas transport problems from the Norwegian Continental Shelf (NCS) with the network structure not represented in the existing body of test instances. We find that our Formulation S performs comparable to the tighter MCF formulation on most of our test cases. On the problems presented by Alfaki and Haugland [2013b], convergence is generally slower using Formulation S than the MCF formulation.

Formulation S is comparable, but inferior to formulation MCF in most of our test instances. In contrast with the MCF formulation, it solves more general problems including processing facilities. Such problems often arise in analysis of the network on the NCS, which makes Formulation S useful. For hard pooling problems without processing facilites, previous formulations may be preferrable.

When comparing continuous and discretized formulations of the pooling problems with processing facilities, we found that the continuous formulation usually outperforms the discretized versions. However, for several continuous instances, computations did not complete within the time limit of one hour. All problem instances completed well before the time limit using the discretized formulations. It is difficult to predict which formulation will be the best for a given test instance, and one may have to choose between better expected performance with the continuous formulation, or better worst case performance using the discretized, approximate, formulations.

Adding composite constraints to one terminal node is relatively straightforward when the quality is modelled as physical components. As expected, we see from our computational results that the effort required solving the problem instances increases substantially. Not all NCS instances with processing and composite constraints are solved to optimality within the time limit of one hour.

## Bibliography

Adhya N, Tawarmalani M, Sahinidis N (1999) A Lagrangian approach to the pooling problem. Industrial & Engineering Chemistry Research 38(5):1956–1972

Alfaki M, Haugland D (2013a) A multi-commodity flow formulation for the generalized pooling problem. Journal of Global Optimization 56:917–937

Alfaki M, Haugland D (2013b) Strong formulations for the pooling problem. Journal of Global Optimization 56(3):897–916, DOI 10.1007/s10898-012-9875-6

Androulakis I, Maranas C, Floudas C (1995) $\alpha$bb: A global optimization method for general constrained nonconvex problems. Journal of Global Optimization 7(4):337–363

Audet C, Hansen P, Jaumard B, Savard G (2000) A branch and cut algorithm for nonconvex quadratically constrained quadratic programming. Mathematical Programming 87(1):131–152

Audet C, Brimberg J, Hansen P, Le Digabel S, Mladenović N (2004) Pooling problem: Alternate formulations and solution methods. Management Sscience pp 761–776

Baker T, Lasdon L (1985) Successive linear programming at Exxon. Management Science pp 264–274

Ben-Tal A, Eiger G, Gershovitz V (1994) Global minimization by reducing the duality gap. Mathematical Programming 63(1):193–212

Castro PM, Teles JP (2013) Comparison of global optimization algorithms for the design of water-using networks. Computers & Chemical Engineering 52:249–261

Dey SS, Gupte A (2015) Analysis of MILP techniques for the pooling problem. Operations Research 62(2):412–427

Fodstad M, Midthun KT, Tomasgard A (2013) Adding flexibility in a natural gas transportation network using interruptible transportation services, in Review

Foulds LR, Haugland D, Jørnsten K (1992) A bilinear approach to the pooling problem. Optimization 24:165–180

Frimannslund L, Haugland D (2009) Parallel solution of the pooling problem with application to the cell broadband engine architecture. In: International Conference on Computers & Industrial Engineering, 2009. CIE 2009, IEEE, pp 354–359

Gounaris C, Misener R, Floudas C (2009) Computational comparison of piecewise-linear relaxations for pooling problems. Industrial & Engineering Chemistry Research 48(12):5742–5766

Gupte A (2012) Mixed integer bilinear programming with applications to the pooling problem, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, 2012. URL `https://smartech.gatech.edu/handle/1853/45761`

Gupte A, Ahmed S, Dey SS, Cheon MS (2012) Pooling problem. URL `www.optimization-online.org/DB_FILE/2012/10/3658.pdf`

Haugland D (2010) An overview of models and solution methods for pooling problems. Energy, Natural Resources and Environmental Economics pp 459–469

Haverly C (1978) Studies of the behavior of recursion for the pooling problem. ACM SIGMAP Bulletin 25:19–28

Haverly C (1979) Behavior of recursion model—more studies. ACM SIGMAP Bulletin (26):22–28

Haverly C (1980) Recursion model behavior: more studies. ACM SIGMAP Bulletin (28):39–41

Hellemo L, Werner A (2014) Discretizations of natural gas pooling problems, in Review

Hellemo L, Midthun K, Tomasgard A, Werner A (2012a) Multi-stage stochastic programming for natural gas infrastructure design with a production perspective. In: Ziemba WT, Wallace SW, Gassman HI (eds) Stochastic programming – Applications in finance, energy, planning and logistics, World Scientific series in finance, vol 4, World Scientific, 2012, pp 259–288

Hellemo L, Midthun K, Tomasgard A, Werner A (2012b) Natural Gas Infrastructure Design with an Operational Perspective. Energy Procedia 26:67–73

Hornemann JAT (1998) Method for determining the calorific value of a gas and/or the wobbe index of a natural gas. US Patent 5,807,749

Kocis G, Grossmann I (1989) A modelling and decomposition strategy for the MINLP optimization of process flowsheets. Computers & Chemical Engineering 13(7):797–819

Lee S, Grossmann I (2003) Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks. Computers & Chemical Engineering 27(11):1557–1575

Li X, Armagan E, Tomasgard A, Barton PI (2011a) Stochastic pooling problem for natural gas production network design and operation under uncertainty. AIChE Journal 57:2120–2135

Li X, Tomasgard A, Barton P (2011b) Decomposition strategy for the stochastic pooling problem. Journal of Global Optimization 54:765–790

Lodwick W (1992) Preprocessing nonlinear functional constraints with applications to the pooling problem. ORSA Journal on Computing 4(2):119–131

Meyer C, Floudas C (2006) Global optimization of a combinatorially complex generalized pooling problem. AIChE journal 52(3):1027–1037

Misener R, Floudas C (2009) Advances for the pooling problem: Modeling, global optimization, and computational studies. Applied and Computational Mathematics 8(1):3–22

Misener R, Thompson JP, Floudas CA (2011) Apogee: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. Computers & Chemical Engineering 35(5):876–892

Quesada I, Grossmann I (1995) Global optimization of bilinear process networks with multicomponent flows. Computers & Chemical Engineering 19(12):1219–1242

Rømo F, Tomasgard A, Hellemo L, Fodstad M, Eidesen B, Pedersen B (2009) Optimizing the Norwegian natural gas production and transport. Interfaces 39(1):46–56

Selot A, Kuok L, Robinson M, Mason T, Barton P (2008) A short-term operational planning model for natural gas production systems. AIChE Journal 54(2):495–515

Tawarmalani M, Sahinidis NV (2002) Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming. Kluwer Academic Publishers, Norwell, MA, USA

Teles JP, Castro PM, Matos HA (2012) Global optimization of water networks design using multiparametric disaggregation. Computers & Chemical Engineering 40:132–147

The Norwegian Ministry of Petroleum and Energy and The Norwegian Petroleum Directorate (2012) Facts 2012 — The Norwegian petroleum sector. http://www.npd.no/en/Publications/Facts/Facts-2012/

Tomasgard A, Rømo F, Fodstad M, Midthun K (2007) Optimization models for the natural gas value chain. In: Hasle G, Lie KA, Quak E (eds) Geometric Modelling, Numerical Simulation and Optimization, Springer Verlag, pp 521–558

UK Government (1996) Gas safety (management) regulations 1996. UK Statutory Instruments

Ulstein NL (2000) Short Term Planning of Gas Production (In Norwegian). Masters thesis, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology

Ulstein NL, Nygreen B, Sagli JR (2007) Tactical planning of offshore petroleum production. European Journal of Operational Research 176(1):550–564

## 6.A Software and Hardware Specifications

All numerical tests were performed using GAMS version 23.7.2 with BARON using CPLEX in combination with Conopt or GAMS in combination with CPLEX for MILPs on an HP bl685c G7 4x AMD Opteron™ 6274, 16 core, 2.2GHz with 128Gb memory running Linux 2.6.32-358 (Rocks 6.1.1).

## 6.B Tables of numerical results

**Table 6.3:** NCS instances (no processing). Gap with time limit 1 hour. Instances with no solution or gap > 100 are dashed.

| Instance | MCF | S | S-D1 | S-D2 | S w/Comp |
|---|---|---|---|---|---|
| NCS001 | 0.0 | 0.0 | 0.0 | $7.9 \cdot 10^{-4}$ | – |
| NCS002 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| NCS003 | $3.5 \cdot 10^{-4}$ | $8.4 \cdot 10^{-4}$ | $9 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ | $9.5 \cdot 10^{-4}$ |
| NCS004 | 0.0 | 0.0 | 0.0 | $8.4 \cdot 10^{-4}$ | 0.0 |
| NCS005 | 0.0 | $2.7 \cdot 10^{-5}$ | $7.9 \cdot 10^{-4}$ | $5.4 \cdot 10^{-4}$ | $2.2 \cdot 10^{-4}$ |
| NCS006 | 0.0 | 0.0 | $2.7 \cdot 10^{-4}$ | 0.0 | 0.0 |
| NCS007 | $3.8 \cdot 10^{-4}$ | $6.2 \cdot 10^{-4}$ | $9.8 \cdot 10^{-4}$ | $9.2 \cdot 10^{-4}$ | – |
| NCS008 | $1.6 \cdot 10^{-4}$ | $7.1 \cdot 10^{-4}$ | $8.4 \cdot 10^{-4}$ | $8.1 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ |
| NCS009 | $8.1 \cdot 10^{-5}$ | $4.9 \cdot 10^{-4}$ | $8.1 \cdot 10^{-4}$ | $8.1 \cdot 10^{-4}$ | – |
| NCS010 | $1 \cdot 10^{-3}$ | $9.8 \cdot 10^{-4}$ | $9.8 \cdot 10^{-4}$ | $9.8 \cdot 10^{-4}$ | – |
| NCS011 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| NCS012 | 0.0 | 0.0 | 0.0 | 0.0 | – |
| NCS013 | 0.0 | 0.0 | $5.4 \cdot 10^{-4}$ | 0.0 | – |
| NCS014 | 0.0 | $1.9 \cdot 10^{-4}$ | $9.5 \cdot 10^{-4}$ | $9.2 \cdot 10^{-4}$ | – |
| NCS015 | 0.0 | 0.0 | 0.0 | 0.0 | – |
| NCS016 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| NCS017 | 0.0 | 0.0 | 0.0 | $2.7 \cdot 10^{-4}$ | – |
| NCS018 | 0.0 | 0.0 | 0.0 | 0.0 | – |
| NCS019 | 0.0 | 0.0 | $1.9 \cdot 10^{-4}$ | 0.0 | – |
| NCS020 | $6.3 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | $9.8 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ |
| NCS021 | $1.9 \cdot 10^{-4}$ | $6.8 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ | $8.4 \cdot 10^{-4}$ | – |
| NCS022 | 0.0 | 0.0 | $3.8 \cdot 10^{-4}$ | 0.0 | – |

**Table 6.3:** NCS instances (no processing). Gap with time limit 1 hour. Instances with no solution or gap > 100 are dashed.

| Instance | MCF | S | S-D1 | S-D2 | S w/Comp |
|---|---|---|---|---|---|
| NCS023 | 0.0 | 0.0 | 0.0 | 0.0 | $8.1 \cdot 10^{-5}$ |
| NCS024 | 0.0 | 0.0 | $1.1 \cdot 10^{-4}$ | 0.0 | $1 \cdot 10^{-3}$ |
| NCS025 | $2.7 \cdot 10^{-5}$ | $6.2 \cdot 10^{-4}$ | $7.6 \cdot 10^{-4}$ | $8.7 \cdot 10^{-4}$ | $6.2 \cdot 10^{-4}$ |
| NCS026 | 0.0 | 0.0 | $2.2 \cdot 10^{-4}$ | $8.1 \cdot 10^{-4}$ | – |
| NCS027 | $1 \cdot 10^{-3}$ | $1.4 \cdot 10^{-1}$ | – | $9.3 \cdot 10^{-4}$ | – |
| NCS028 | $1.9 \cdot 10^{-4}$ | $4.3 \cdot 10^{-4}$ | $9.8 \cdot 10^{-4}$ | $6.8 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ |
| NCS029 | $5.4 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ | $9.8 \cdot 10^{-4}$ | $9.8 \cdot 10^{-4}$ | – |
| NCS030 | $2.7 \cdot 10^{-4}$ | $8.1 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ | $9.8 \cdot 10^{-4}$ | $9.8 \cdot 10^{-4}$ |
| NCS031 | 0.0 | 0.0 | 0.0 | $2.7 \cdot 10^{-5}$ | 0.0 |
| NCS032 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| NCS033 | 0.0 | $6.2 \cdot 10^{-4}$ | $6.5 \cdot 10^{-4}$ | $6.5 \cdot 10^{-4}$ | $9.9 \cdot 10^{-4}$ |
| NCS034 | $2.7 \cdot 10^{-5}$ | $2.4 \cdot 10^{-4}$ | $2.7 \cdot 10^{-4}$ | $4.6 \cdot 10^{-4}$ | – |
| NCS035 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| NCS036 | 0.0 | 0.0 | 0.0 | 0.0 | – |
| NCS037 | 0.0 | 0.0 | 0.0 | $2.7 \cdot 10^{-4}$ | – |
| NCS038 | 0.0 | 0.0 | $2.7 \cdot 10^{-5}$ | 0.0 | – |
| NCS039 | $1.4 \cdot 10^{-4}$ | $6.5 \cdot 10^{-4}$ | $8.7 \cdot 10^{-4}$ | $6.2 \cdot 10^{-4}$ | $9.9 \cdot 10^{-4}$ |
| NCS040 | $9.5 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | – |

**Table 6.4:** Full results from NCS processing cases. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel-nocomp | NCS001 | Optimal | 3.26 | 216 | 0 | 533 | 525 | −380.9 | −380.5 |
| sd1model | NCS001 | Optimal | 13.85 | 0 | 198 | 3,180 | 7,876 | −362.4 | −362.1 |
| sd2model | NCS001 | Optimal | 12.34 | 0 | 180 | 2,946 | 7,192 | −368.1 | −367.7 |
| smodel-nocomp | NCS002 | Optimal | 5.44 | 216 | 0 | 533 | 515 | −376.3 | −376.0 |
| sd1model | NCS002 | Optimal | 34.89 | 0 | 198 | 3,180 | 7,866 | −362.1 | −362.1 |
| sd2model | NCS002 | Optimal | 9.17 | 0 | 180 | 2,946 | 7,182 | −368.1 | −367.7 |
| smodel-nocomp | NCS003 | Optimal | 1.57 | 216 | 0 | 533 | 520 | −381.3 | −380.9 |
| sd1model | NCS003 | Optimal | 22.99 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS003 | Optimal | 12.49 | 0 | 180 | 2,946 | 7,187 | −368.0 | −367.7 |
| smodel-nocomp | NCS004 | Optimal | 0.83 | 216 | 0 | 533 | 517 | −373.2 | −372.8 |
| sd1model | NCS004 | Optimal | 83.45 | 0 | 198 | 3,180 | 7,868 | −287.3 | −287.3 |
| sd2model | NCS004 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,184 | −301.2 | −287.4 |
| smodel-nocomp | NCS005 | Unfinished | – | 216 | 0 | 533 | 520 | −376.2 | −373.6 |
| sd1model | NCS005 | Optimal | 53.56 | 0 | 198 | 3,180 | 7,871 | −301.5 | −301.3 |
| sd2model | NCS005 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,187 | −304.1 | −301.4 |
| smodel-nocomp | NCS006 | Optimal | 3.98 | 216 | 0 | 533 | 517 | −379.3 | −378.9 |
| sd1model | NCS006 | Optimal | 58.78 | 0 | 198 | 3,180 | 7,868 | −361.4 | −361.0 |
| sd2model | NCS006 | Optimal | 31.49 | 0 | 180 | 2,946 | 7,184 | −367.1 | −366.8 |
| smodel-nocomp | NCS007 | Unfinished | – | 216 | 0 | 533 | 525 | −381.1 | −374.6 |
| sd1model | NCS007 | Optimal | 11.47 | 0 | 198 | 3,180 | 7,876 | −361.2 | −361.0 |
| sd2model | NCS007 | Optimal | 38.49 | 0 | 180 | 2,946 | 7,192 | −367.1 | −366.8 |
| smodel-nocomp | NCS008 | Optimal | 1.7 | 216 | 0 | 533 | 517 | −379.0 | −378.7 |
| sd1model | NCS008 | Optimal | 62.23 | 0 | 198 | 3,180 | 7,868 | −362.1 | −362.1 |

**Table 6.4:** Full results from NCS processing cases. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd2model | NCS008 | Optimal | 8.19 | 0 | 180 | 2,946 | 7,184 | −368.0 | −367.6 |
| smodel-nocomp | NCS009 | Optimal | 15.45 | 216 | 0 | 533 | 512 | −381.3 | −380.9 |
| sd1model | NCS009 | Optimal | 42.54 | 0 | 198 | 3,180 | 7,863 | −362.2 | −362.1 |
| sd2model | NCS009 | Optimal | 10.31 | 0 | 180 | 2,946 | 7,179 | −368.0 | −367.7 |
| smodel-nocomp | NCS010 | Optimal | 1.54 | 216 | 0 | 533 | 517 | −380.6 | −380.2 |
| sd1model | NCS010 | Optimal | 50.31 | 0 | 198 | 3,180 | 7,868 | −352.7 | −352.7 |
| sd2model | NCS010 | Optimal | 49.5 | 0 | 180 | 2,946 | 7,184 | −367.3 | −367.0 |
| smodel-nocomp | NCS011 | Optimal | 1.7 | 216 | 0 | 533 | 515 | −377.4 | −377.0 |
| sd1model | NCS011 | Optimal | 89.57 | 0 | 198 | 3,180 | 7,866 | −362.3 | −362.1 |
| sd2model | NCS011 | Optimal | 31.09 | 0 | 180 | 2,946 | 7,182 | −368.0 | −367.7 |
| smodel-nocomp | NCS012 | Optimal | 9.62 | 216 | 0 | 533 | 512 | −374.4 | −374.0 |
| sd1model | NCS012 | Optimal | 35.74 | 0 | 198 | 3,180 | 7,863 | −362.4 | −362.1 |
| sd2model | NCS012 | Optimal | 17.32 | 0 | 180 | 2,946 | 7,179 | −367.9 | −367.7 |
| smodel-nocomp | NCS013 | Optimal | 0.52 | 216 | 0 | 533 | 520 | −381.0 | −380.7 |
| sd1model | NCS013 | Optimal | 37.2 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS013 | Optimal | 36.41 | 0 | 180 | 2,946 | 7,187 | −368.0 | −367.7 |
| smodel-nocomp | NCS014 | Optimal | 0.78 | 216 | 0 | 533 | 525 | −375.0 | −374.6 |
| sd1model | NCS014 | Optimal | 72.54 | 0 | 198 | 3,180 | 7,876 | −359.0 | −358.7 |
| sd2model | NCS014 | Optimal | 36.62 | 0 | 180 | 2,946 | 7,192 | −366.9 | −366.6 |
| smodel-nocomp | NCS015 | Optimal | 12.91 | 216 | 0 | 533 | 520 | −380.2 | −379.9 |
| sd1model | NCS015 | Optimal | 47.29 | 0 | 198 | 3,180 | 7,871 | −345.2 | −345.2 |
| sd2model | NCS015 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,187 | −354.9 | −352.6 |
| smodel-nocomp | NCS016 | Optimal | 0.66 | 216 | 0 | 533 | 517 | −380.8 | −380.4 |

**Table 6.4:** Full results from NCS processing cases. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd1model | NCS016 | Optimal | 58.25 | 0 | 198 | 3,180 | 7,868 | −362.1 | −362.1 |
| sd2model | NCS016 | Optimal | 34.76 | 0 | 180 | 2,946 | 7,184 | −368.1 | −367.7 |
| smodel-nocomp | NCS017 | Optimal | 1.75 | 216 | 0 | 533 | 515 | −375.6 | −375.2 |
| sd1model | NCS017 | Optimal | 13.37 | 0 | 198 | 3,180 | 7,866 | −362.1 | −362.1 |
| sd2model | NCS017 | Optimal | 14 | 0 | 180 | 2,946 | 7,182 | −368.0 | −367.7 |
| smodel-nocomp | NCS018 | Unfinished | – | 216 | 0 | 533 | 520 | −380.9 | −379.0 |
| sd1model | NCS018 | Optimal | 35.18 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS018 | Optimal | 17.72 | 0 | 180 | 2,946 | 7,187 | −368.1 | −367.7 |
| smodel-nocomp | NCS019 | Optimal | 5.32 | 216 | 0 | 533 | 525 | −378.9 | −378.5 |
| sd1model | NCS019 | Optimal | 47.37 | 0 | 198 | 3,180 | 7,876 | −362.1 | −362.1 |
| sd2model | NCS019 | Optimal | 30.57 | 0 | 180 | 2,946 | 7,192 | −368.1 | −367.7 |
| smodel-nocomp | NCS020 | Optimal | 1.11 | 216 | 0 | 533 | 520 | −381.3 | −380.9 |
| sd1model | NCS020 | Optimal | 7.54 | 0 | 198 | 3,180 | 7,871 | −361.5 | −361.3 |
| sd2model | NCS020 | Optimal | 26.1 | 0 | 180 | 2,946 | 7,187 | −367.4 | −367.0 |
| smodel-nocomp | NCS021 | Optimal | 9.58 | 216 | 0 | 533 | 525 | −379.3 | −378.9 |
| sd1model | NCS021 | Optimal | 29.5 | 0 | 198 | 3,180 | 7,876 | −362.1 | −362.1 |
| sd2model | NCS021 | Optimal | 49.37 | 0 | 180 | 2,946 | 7,192 | −368.1 | −367.7 |
| smodel-nocomp | NCS022 | Optimal | 0.74 | 216 | 0 | 533 | 517 | −381.3 | −380.9 |
| sd1model | NCS022 | Optimal | 11.78 | 0 | 198 | 3,180 | 7,868 | −362.1 | −362.1 |
| sd2model | NCS022 | Optimal | 28.82 | 0 | 180 | 2,946 | 7,184 | −367.9 | −367.6 |
| smodel-nocomp | NCS023 | Optimal | 1.05 | 216 | 0 | 533 | 525 | −373.5 | −373.2 |
| sd1model | NCS023 | Optimal | 85.74 | 0 | 198 | 3,180 | 7,876 | −310.3 | −310.1 |
| sd2model | NCS023 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,192 | −326.2 | −310.3 |

**Table 6.4:** Full results from NCS processing cases. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel-nocomp | NCS024 | Optimal | 12.08 | 216 | 0 | 533 | 525 | −379.7 | −379.3 |
| sd1model | NCS024 | Optimal | 43.12 | 0 | 198 | 3,180 | 7,876 | −288.5 | −288.3 |
| sd2model | NCS024 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,192 | −295.4 | −288.4 |
| smodel-nocomp | NCS025 | Unfinished | – | 216 | 0 | 533 | 512 | – | – |
| sd1model | NCS025 | Optimal | 48.95 | 0 | 198 | 3,180 | 7,863 | – | 0.0 |
| sd2model | NCS025 | Unfinished | 3,600.02 | 0 | 180 | 2,946 | 7,179 | – | 0.0 |
| smodel-nocomp | NCS026 | Optimal | 22.72 | 216 | 0 | 533 | 512 | −375.9 | −375.6 |
| sd1model | NCS026 | Optimal | 26.13 | 0 | 198 | 3,180 | 7,863 | −362.4 | −362.1 |
| sd2model | NCS026 | Optimal | 12.3 | 0 | 180 | 2,946 | 7,179 | −368.1 | −367.7 |
| smodel-nocomp | NCS027 | Optimal | 2.48 | 216 | 0 | 533 | 520 | −379.7 | −379.3 |
| sd1model | NCS027 | Optimal | 24.02 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS027 | Optimal | 6.65 | 0 | 180 | 2,946 | 7,187 | −368.1 | −367.7 |
| smodel-nocomp | NCS028 | Optimal | 3.48 | 216 | 0 | 533 | 525 | −380.3 | −380.0 |
| sd1model | NCS028 | Optimal | 39.63 | 0 | 198 | 3,180 | 7,876 | −362.1 | −362.0 |
| sd2model | NCS028 | Optimal | 34.29 | 0 | 180 | 2,946 | 7,192 | −367.9 | −367.5 |
| smodel-nocomp | NCS029 | Optimal | 2.9 | 216 | 0 | 533 | 520 | −379.7 | −379.3 |
| sd1model | NCS029 | Optimal | 33.5 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS029 | Optimal | 14.66 | 0 | 180 | 2,946 | 7,187 | −368.0 | −367.7 |
| smodel-nocomp | NCS030 | Optimal | 1.62 | 216 | 0 | 533 | 512 | −375.5 | −375.1 |
| sd1model | NCS030 | Optimal | 25.71 | 0 | 198 | 3,180 | 7,863 | −362.3 | −362.1 |
| sd2model | NCS030 | Optimal | 26.21 | 0 | 180 | 2,946 | 7,179 | −368.1 | −367.7 |
| smodel-nocomp | NCS031 | Optimal | 3.55 | 216 | 0 | 533 | 520 | −379.3 | −378.9 |

**Table 6.4:** Full results from NCS processing cases. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd1model | NCS031 | Optimal | 38.49 | 0 | 198 | 3,180 | 7,871 | −284.6 | −284.6 |
| sd2model | NCS031 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,187 | −299.7 | −284.7 |
| smodel-nocomp | NCS032 | Optimal | 1.7 | 216 | 0 | 533 | 517 | −379.1 | −378.8 |
| sd1model | NCS032 | Optimal | 35.07 | 0 | 198 | 3,180 | 7,868 | −362.1 | −362.1 |
| sd2model | NCS032 | Optimal | 29.56 | 0 | 180 | 2,946 | 7,184 | −368.1 | −367.7 |
| smodel-nocomp | NCS033 | Optimal | 8.12 | 216 | 0 | 533 | 512 | −372.9 | −372.5 |
| sd1model | NCS033 | Optimal | 32.06 | 0 | 198 | 3,180 | 7,863 | −362.1 | −362.1 |
| sd2model | NCS033 | Optimal | 8.81 | 0 | 180 | 2,946 | 7,179 | −368.0 | −367.7 |
| smodel-nocomp | NCS034 | Optimal | 0.76 | 216 | 0 | 533 | 512 | −380.8 | −380.4 |
| sd1model | NCS034 | Optimal | 72.97 | 0 | 198 | 3,180 | 7,863 | −362.1 | −362.1 |
| sd2model | NCS034 | Optimal | 29.29 | 0 | 180 | 2,946 | 7,179 | −368.1 | −367.7 |
| smodel-nocomp | NCS035 | Optimal | 12.21 | 216 | 0 | 533 | 525 | −380.8 | −380.5 |
| sd1model | NCS035 | Optimal | 58.52 | 0 | 198 | 3,180 | 7,876 | −361.6 | −361.3 |
| sd2model | NCS035 | Optimal | 123.14 | 0 | 180 | 2,946 | 7,192 | −367.4 | −367.0 |
| smodel-nocomp | NCS036 | Optimal | 31.46 | 216 | 0 | 533 | 517 | −380.2 | −379.8 |
| sd1model | NCS036 | Optimal | 32.93 | 0 | 198 | 3,180 | 7,868 | −340.5 | −340.5 |
| sd2model | NCS036 | Unfinished | 3,600.03 | 0 | 180 | 2,946 | 7,184 | −348.7 | −341.6 |
| smodel-nocomp | NCS037 | Optimal | 2.85 | 216 | 0 | 533 | 517 | −381.8 | −381.4 |
| sd1model | NCS037 | Optimal | 7.23 | 0 | 198 | 3,180 | 7,868 | −362.1 | −362.1 |
| sd2model | NCS037 | Optimal | 30.85 | 0 | 180 | 2,946 | 7,184 | −368.1 | −367.7 |
| smodel-nocomp | NCS038 | Optimal | 0.61 | 216 | 0 | 533 | 520 | −380.8 | −380.4 |
| sd1model | NCS038 | Optimal | 79.82 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS038 | Optimal | 25.9 | 0 | 180 | 2,946 | 7,187 | −368.0 | −367.7 |

**Table 6.4:** Full results from NCS processing cases. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel-nocomp | NCS039 | Optimal | 0.75 | 216 | 0 | 533 | 520 | −380.9 | −380.5 |
| sd1model | NCS039 | Optimal | 34.81 | 0 | 198 | 3,180 | 7,871 | −362.1 | −362.1 |
| sd2model | NCS039 | Optimal | 27.36 | 0 | 180 | 2,946 | 7,187 | −368.0 | −367.7 |
| smodel-nocomp | NCS040 | Unfinished | – | 216 | 0 | 533 | 512 | −379.8 | −375.8 |
| sd1model | NCS040 | Optimal | 30.96 | 0 | 198 | 3,180 | 7,863 | −362.1 | −362.1 |
| sd2model | NCS040 | Optimal | 23.1 | 0 | 180 | 2,946 | 7,179 | −368.0 | −367.7 |

**Table 6.5:** Full results from NCS processing cases with composite constraints using smodel. Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|
| NCS001 | Optimal | 0.81 | 304 | 0 | 732 | 696 | −120.0 | – |
| NCS002 | Optimal | 3.06 | 304 | 0 | 732 | 686 | −121.1 | – |
| NCS003 | Unfinished | – | 304 | 0 | 732 | 691 | −410.3 | −380.8 |
| NCS004 | Optimal | 2,707.05 | 304 | 0 | 732 | 688 | −120.3 | – |
| NCS005 | Optimal | 6.91 | 304 | 0 | 732 | 691 | −372.8 | −372.5 |
| NCS006 | Unfinished | – | 304 | 0 | 732 | 688 | – | – |
| NCS007 | Unfinished | – | 304 | 0 | 732 | 696 | – | – |
| NCS008 | Optimal | 1.39 | 304 | 0 | 732 | 688 | −119.9 | – |
| NCS009 | Optimal | 16.27 | 304 | 0 | 732 | 683 | −381.1 | −380.7 |
| NCS010 | Optimal | 1.73 | 304 | 0 | 732 | 688 | −347.3 | −346.9 |
| NCS011 | Optimal | 1.78 | 304 | 0 | 732 | 686 | −120.3 | – |
| NCS012 | Optimal | 53.79 | 304 | 0 | 732 | 683 | −374.4 | −374.0 |
| NCS013 | Optimal | 1.02 | 304 | 0 | 732 | 691 | −120.0 | – |
| NCS014 | Unfinished | – | 304 | 0 | 732 | 696 | – | – |
| NCS015 | Optimal | 100.25 | 304 | 0 | 732 | 691 | −380.1 | −379.7 |
| NCS016 | Unfinished | – | 304 | 0 | 732 | 688 | −332.6 | −300.6 |
| NCS017 | Optimal | 9.93 | 304 | 0 | 732 | 686 | −375.6 | −375.2 |
| NCS018 | Unfinished | – | 304 | 0 | 732 | 691 | – | – |
| NCS019 | Optimal | 24.39 | 304 | 0 | 732 | 696 | −378.8 | −378.4 |
| NCS020 | Optimal | 0.58 | 304 | 0 | 732 | 691 | −381.1 | −380.8 |
| NCS021 | Unfinished | – | 304 | 0 | 732 | 696 | – | – |
| NCS022 | Optimal | 8.4 | 304 | 0 | 732 | 688 | −381.2 | −380.8 |
| NCS023 | Optimal | 6.31 | 304 | 0 | 732 | 696 | −373.5 | −373.2 |

**Table 6.5:** Full results from NCS processing cases with composite constraints using smodel. Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|
| NCS024 | Optimal | 85.14 | 304 | 0 | 732 | 696 | −378.8 | −378.4 |
| NCS025 | Optimal | 3.18 | 304 | 0 | 732 | 683 | −122.1 | – |
| NCS026 | Optimal | 11.86 | 304 | 0 | 732 | 683 | −375.9 | −375.6 |
| NCS027 | Optimal | 15.37 | 304 | 0 | 732 | 691 | −379.7 | −379.3 |
| NCS028 | Optimal | 1,803.26 | 304 | 0 | 732 | 696 | −120.3 | – |
| NCS029 | Unfinished | – | 304 | 0 | 732 | 691 | – | – |
| NCS030 | Unfinished | – | 304 | 0 | 732 | 683 | −374.0 | −346.2 |
| NCS031 | Optimal | 1.95 | 304 | 0 | 732 | 691 | −379.2 | −378.8 |
| NCS032 | Optimal | 24.22 | 304 | 0 | 732 | 688 | −119.9 | – |
| NCS033 | Optimal | 9.48 | 304 | 0 | 732 | 683 | −325.9 | −325.6 |
| NCS034 | Unfinished | – | 304 | 0 | 732 | 683 | −357.5 | −317.4 |
| NCS035 | Optimal | 0.92 | 304 | 0 | 732 | 696 | −120.0 | – |
| NCS036 | Optimal | 36.31 | 304 | 0 | 732 | 688 | −314.5 | −314.2 |
| NCS037 | Unfinished | – | 304 | 0 | 732 | 688 | – | – |
| NCS038 | Optimal | 2.21 | 304 | 0 | 732 | 691 | −120.3 | – |
| NCS039 | Unfinished | – | 304 | 0 | 732 | 691 | – | – |
| NCS040 | Optimal | 21.46 | 304 | 0 | 732 | 683 | −120.8 | – |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| mcfmodel | NCS001 | Optimal | 0.12 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS001 | Optimal | 0.1 | 264 | 0 | 306 | 332 | −368.5 | −368.5 |
| smodel | NCS001 | Unfinished | – | 352 | 0 | 361 | 383 | – | – |
| sd1model | NCS001 | Optimal | 0.52 | 0 | 66 | 1,164 | 2,776 | −368.5 | −368.5 |
| sd2model | NCS001 | Optimal | 0.14 | 0 | 36 | 774 | 1,660 | −368.5 | −368.2 |
| mcfmodel | NCS002 | Optimal | 0.14 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS002 | Optimal | 0.12 | 264 | 0 | 306 | 329 | −368.5 | −368.5 |
| smodel | NCS002 | Optimal | 0.28 | 352 | 0 | 361 | 380 | −368.5 | −368.5 |
| sd1model | NCS002 | Optimal | 0.53 | 0 | 66 | 1,164 | 2,773 | −368.5 | −368.5 |
| sd2model | NCS002 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,657 | −368.5 | −368.5 |
| mcfmodel | NCS003 | Optimal | 0.14 | 112 | 0 | 108 | 165 | −368.3 | −368.2 |
| smodel-nocomp | NCS003 | Optimal | 0.32 | 264 | 0 | 306 | 342 | −368.6 | −368.3 |
| smodel | NCS003 | Optimal | 0.51 | 352 | 0 | 361 | 393 | −368.6 | −368.2 |
| sd1model | NCS003 | Optimal | 7.52 | 0 | 66 | 1,164 | 2,786 | −368.5 | −368.2 |
| sd2model | NCS003 | Optimal | 5.42 | 0 | 36 | 774 | 1,670 | −368.6 | −368.2 |
| mcfmodel | NCS004 | Optimal | 0.14 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS004 | Optimal | 0.12 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |
| smodel | NCS004 | Optimal | 0.25 | 352 | 0 | 361 | 388 | −368.5 | −368.5 |
| sd1model | NCS004 | Optimal | 0.38 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.5 |
| sd2model | NCS004 | Optimal | 0.59 | 0 | 36 | 774 | 1,665 | −368.5 | −368.2 |
| mcfmodel | NCS005 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS005 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | NCS005 | Optimal | 0.21 | 352 | 0 | 361 | 388 | −368.5 | −368.4 |
| sd1model | NCS005 | Optimal | 1.19 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.2 |
| sd2model | NCS005 | Optimal | 0.52 | 0 | 36 | 774 | 1,665 | −368.5 | −368.3 |
| mcfmodel | NCS006 | Optimal | 0.1 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS006 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |
| smodel | NCS006 | Optimal | 0.23 | 352 | 0 | 361 | 388 | −368.5 | −368.5 |
| sd1model | NCS006 | Optimal | 0.45 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.4 |
| sd2model | NCS006 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,665 | −368.5 | −368.5 |
| mcfmodel | NCS007 | Optimal | 0.11 | 112 | 0 | 108 | 165 | −368.4 | −368.3 |
| smodel-nocomp | NCS007 | Optimal | $9{\cdot}10^{-2}$ | 264 | 0 | 306 | 337 | −368.5 | −368.3 |
| smodel | NCS007 | Unfinished | – | 352 | 0 | 361 | 388 | – | – |
| sd1model | NCS007 | Optimal | 9.17 | 0 | 66 | 1,164 | 2,781 | −368.4 | −368.1 |
| sd2model | NCS007 | Optimal | 1.81 | 0 | 36 | 774 | 1,665 | −368.5 | −368.2 |
| mcfmodel | NCS008 | Optimal | $6{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.3 | −368.2 |
| smodel-nocomp | NCS008 | Optimal | 0.12 | 264 | 0 | 306 | 342 | −368.5 | −368.2 |
| smodel | NCS008 | Optimal | 0.59 | 352 | 0 | 361 | 393 | −368.3 | −367.9 |
| sd1model | NCS008 | Optimal | 12.25 | 0 | 66 | 1,164 | 2,786 | −368.4 | −368.1 |
| sd2model | NCS008 | Optimal | 1.4 | 0 | 36 | 774 | 1,670 | −368.5 | −368.2 |
| mcfmodel | NCS009 | Optimal | 0.16 | 112 | 0 | 108 | 165 | −368.4 | −368.3 |
| smodel-nocomp | NCS009 | Optimal | 0.1 | 264 | 0 | 306 | 342 | −368.5 | −368.3 |
| smodel | NCS009 | Optimal | 5.77 | 352 | 0 | 361 | 393 | −133.4 | – |
| sd1model | NCS009 | Optimal | 1.93 | 0 | 66 | 1,164 | 2,786 | −368.5 | −368.2 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd2model | NCS009 | Optimal | 1.19 | 0 | 36 | 774 | 1,670 | −368.5 | −368.2 |
| mcfmodel | NCS010 | Optimal | 0.31 | 112 | 0 | 108 | 165 | −367.6 | −367.2 |
| smodel-nocomp | NCS010 | Optimal | 10.11 | 264 | 0 | 306 | 337 | −367.6 | −367.2 |
| smodel | NCS010 | Optimal | 1.02 | 352 | 0 | 361 | 388 | −184.8 | – |
| sd1model | NCS010 | Optimal | 19.71 | 0 | 66 | 1,164 | 2,781 | −367.0 | −366.6 |
| sd2model | NCS010 | Optimal | 65.79 | 0 | 36 | 774 | 1,665 | −367.5 | −367.1 |
| mcfmodel | NCS011 | Optimal | $6{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS011 | Optimal | 0.1 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |
| smodel | NCS011 | Optimal | 0.25 | 352 | 0 | 361 | 388 | −368.5 | −368.5 |
| sd1model | NCS011 | Optimal | 0.36 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.5 |
| sd2model | NCS011 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,665 | −368.5 | −368.5 |
| mcfmodel | NCS012 | Optimal | $8{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS012 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.6 | −368.6 |
| smodel | NCS012 | Unfinished | – | 352 | 0 | 361 | 388 | – | – |
| sd1model | NCS012 | Optimal | 0.51 | 0 | 66 | 1,164 | 2,781 | −368.6 | −368.6 |
| sd2model | NCS012 | Optimal | 0.31 | 0 | 36 | 774 | 1,665 | −368.6 | −368.6 |
| mcfmodel | NCS013 | Optimal | 0.16 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS013 | Optimal | 0.1 | 264 | 0 | 306 | 342 | −368.5 | −368.5 |
| smodel | NCS013 | Unfinished | – | 352 | 0 | 361 | 393 | – | – |
| sd1model | NCS013 | Optimal | 0.55 | 0 | 66 | 1,164 | 2,786 | −368.5 | −368.3 |
| sd2model | NCS013 | Optimal | 0.1 | 0 | 36 | 774 | 1,670 | −368.5 | −368.5 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| mcfmodel | NCS014 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.4 | −368.4 |
| smodel-nocomp | NCS014 | Optimal | 0.11 | 264 | 0 | 306 | 342 | −368.5 | −368.4 |
| smodel | NCS014 | Unfinished | – | 352 | 0 | 361 | 393 | – | – |
| sd1model | NCS014 | Optimal | 1.23 | 0 | 66 | 1,164 | 2,786 | −368.5 | −368.2 |
| sd2model | NCS014 | Optimal | 1.13 | 0 | 36 | 774 | 1,670 | −368.5 | −368.2 |
| mcfmodel | NCS015 | Optimal | $9{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS015 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.6 | −368.6 |
| smodel | NCS015 | Optimal | 3.22 | 352 | 0 | 361 | 388 | −133.4 | – |
| sd1model | NCS015 | Optimal | 0.86 | 0 | 66 | 1,164 | 2,781 | −368.6 | −368.6 |
| sd2model | NCS015 | Optimal | 0.18 | 0 | 36 | 774 | 1,665 | −368.6 | −368.6 |
| mcfmodel | NCS016 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS016 | Optimal | 0.12 | 264 | 0 | 306 | 329 | −368.6 | −368.6 |
| smodel | NCS016 | Optimal | 0.24 | 352 | 0 | 361 | 380 | −368.6 | −368.6 |
| sd1model | NCS016 | Optimal | 0.12 | 0 | 66 | 1,164 | 2,773 | −368.6 | −368.6 |
| sd2model | NCS016 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,657 | −368.6 | −368.6 |
| mcfmodel | NCS017 | Optimal | 0.15 | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS017 | Optimal | 0.12 | 264 | 0 | 306 | 337 | −368.6 | −368.6 |
| smodel | NCS017 | Optimal | 26.2 | 352 | 0 | 361 | 388 | −133.4 | – |
| sd1model | NCS017 | Optimal | $8{\cdot}10^{-2}$ | 0 | 66 | 1,164 | 2,781 | −368.6 | −368.6 |
| sd2model | NCS017 | Optimal | 0.27 | 0 | 36 | 774 | 1,665 | −368.6 | −368.5 |
| mcfmodel | NCS018 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS018 | Optimal | 0.1 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | NCS018 | Unfinished | – | 352 | 0 | 361 | 388 | – | – |
| sd1model | NCS018 | Optimal | 0.1 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.5 |
| sd2model | NCS018 | Optimal | 0.3 | 0 | 36 | 774 | 1,665 | −368.5 | −368.5 |
| mcfmodel | NCS019 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS019 | Optimal | 0.11 | 264 | 0 | 306 | 334 | −368.6 | −368.6 |
| smodel | NCS019 | Unfinished | – | 352 | 0 | 361 | 385 | – | – |
| sd1model | NCS019 | Optimal | 0.15 | 0 | 66 | 1,164 | 2,778 | −368.6 | −368.5 |
| sd2model | NCS019 | Optimal | 0.21 | 0 | 36 | 774 | 1,662 | −368.6 | −368.6 |
| mcfmodel | NCS020 | Optimal | 0.12 | 112 | 0 | 108 | 165 | −368.1 | −367.8 |
| smodel-nocomp | NCS020 | Optimal | 9.53 | 264 | 0 | 306 | 329 | −368.2 | −367.8 |
| smodel | NCS020 | Optimal | 0.66 | 352 | 0 | 361 | 380 | −368.1 | −367.7 |
| sd1model | NCS020 | Optimal | 48.47 | 0 | 66 | 1,164 | 2,773 | −367.8 | −367.4 |
| sd2model | NCS020 | Optimal | 60.47 | 0 | 36 | 774 | 1,657 | −368.1 | −367.8 |
| mcfmodel | NCS021 | Optimal | 0.12 | 112 | 0 | 108 | 165 | −368.1 | −368.1 |
| smodel-nocomp | NCS021 | Optimal | 0.1 | 264 | 0 | 306 | 337 | −368.3 | −368.1 |
| smodel | NCS021 | Unfinished | – | 352 | 0 | 361 | 388 | – | – |
| sd1model | NCS021 | Optimal | 6.78 | 0 | 66 | 1,164 | 2,781 | −368.3 | −367.9 |
| sd2model | NCS021 | Optimal | 3.19 | 0 | 36 | 774 | 1,665 | −368.3 | −368.0 |
| mcfmodel | NCS022 | Optimal | 0.13 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS022 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |
| smodel | NCS022 | Optimal | 5.98 | 352 | 0 | 361 | 388 | −133.4 | – |
| sd1model | NCS022 | Optimal | 0.12 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.4 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd2model | NCS022 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,665 | −368.5 | −368.5 |
| mcfmodel | NCS023 | Optimal | 0.14 | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS023 | Optimal | $9{\cdot}10^{-2}$ | 264 | 0 | 306 | 337 | −368.6 | −368.6 |
| smodel | NCS023 | Optimal | 0.18 | 352 | 0 | 361 | 388 | −368.6 | −368.6 |
| sd1model | NCS023 | Optimal | 0.14 | 0 | 66 | 1,164 | 2,781 | −368.6 | −368.6 |
| sd2model | NCS023 | Optimal | 0.1 | 0 | 36 | 774 | 1,665 | −368.6 | −368.6 |
| mcfmodel | NCS024 | Optimal | 0.14 | 112 | 0 | 108 | 165 | −368.4 | −368.4 |
| smodel-nocomp | NCS024 | Optimal | 0.12 | 264 | 0 | 306 | 337 | −368.4 | −368.4 |
| smodel | NCS024 | Optimal | 0.6 | 352 | 0 | 361 | 388 | −361.5 | −361.2 |
| sd1model | NCS024 | Optimal | 0.13 | 0 | 66 | 1,164 | 2,781 | −368.4 | −368.4 |
| sd2model | NCS024 | Optimal | $8{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,665 | −368.4 | −368.4 |
| mcfmodel | NCS025 | Optimal | 0.19 | 112 | 0 | 108 | 165 | −368.2 | −368.1 |
| smodel-nocomp | NCS025 | Optimal | 0.14 | 264 | 0 | 306 | 332 | −368.4 | −368.1 |
| smodel | NCS025 | Optimal | 0.27 | 352 | 0 | 361 | 383 | −368.4 | −368.1 |
| sd1model | NCS025 | Optimal | 1.81 | 0 | 66 | 1,164 | 2,776 | −368.4 | −368.1 |
| sd2model | NCS025 | Optimal | 1.01 | 0 | 36 | 774 | 1,660 | −368.4 | −368.1 |
| mcfmodel | NCS026 | Optimal | $9{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS026 | Optimal | 0.1 | 264 | 0 | 306 | 342 | −368.6 | −368.6 |
| smodel | NCS026 | Optimal | 17.8 | 352 | 0 | 361 | 393 | −133.4 | – |
| sd1model | NCS026 | Optimal | 0.1 | 0 | 66 | 1,164 | 2,786 | −368.6 | −368.5 |
| sd2model | NCS026 | Optimal | 0.25 | 0 | 36 | 774 | 1,670 | −368.6 | −368.3 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| mcfmodel | NCS027 | Optimal | 0.17 | 112 | 0 | 108 | 165 | −323.9 | −323.5 |
| smodel-nocomp | NCS027 | Unfinished | – | 264 | 0 | 306 | 337 | −367.3 | −323.5 |
| smodel | NCS027 | Unfinished | – | 352 | 0 | 361 | 388 | – | – |
| sd1model | NCS027 | Optimal | 3.04 | 0 | 66 | 1,164 | 2,781 | – | 0.0 |
| sd2model | NCS027 | Optimal | 13.67 | 0 | 36 | 774 | 1,665 | −322.4 | −322.1 |
| mcfmodel | NCS028 | Optimal | $9{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.3 | −368.3 |
| smodel-nocomp | NCS028 | Optimal | 0.12 | 264 | 0 | 306 | 332 | −368.4 | −368.3 |
| smodel | NCS028 | Optimal | 0.66 | 352 | 0 | 361 | 383 | −333.8 | −333.5 |
| sd1model | NCS028 | Optimal | 2.55 | 0 | 66 | 1,164 | 2,776 | −368.4 | −368.0 |
| sd2model | NCS028 | Optimal | 0.69 | 0 | 36 | 774 | 1,660 | −368.4 | −368.2 |
| mcfmodel | NCS029 | Optimal | 0.15 | 112 | 0 | 108 | 165 | −368.1 | −367.9 |
| smodel-nocomp | NCS029 | Optimal | 0.4 | 264 | 0 | 306 | 334 | −368.2 | −367.9 |
| smodel | NCS029 | Optimal | 3.02 | 352 | 0 | 361 | 385 | −190.2 | – |
| sd1model | NCS029 | Optimal | 55.53 | 0 | 66 | 1,164 | 2,778 | −367.8 | −367.4 |
| sd2model | NCS029 | Optimal | 9.96 | 0 | 36 | 774 | 1,662 | −368.2 | −367.9 |
| mcfmodel | NCS030 | Optimal | 0.1 | 112 | 0 | 108 | 165 | −368.3 | −368.2 |
| smodel-nocomp | NCS030 | Optimal | 0.13 | 264 | 0 | 306 | 342 | −368.6 | −368.3 |
| smodel | NCS030 | Optimal | 0.22 | 352 | 0 | 361 | 393 | −368.6 | −368.2 |
| sd1model | NCS030 | Optimal | 10.01 | 0 | 66 | 1,164 | 2,786 | −368.6 | −368.2 |
| sd2model | NCS030 | Optimal | 3.78 | 0 | 36 | 774 | 1,670 | −368.6 | −368.2 |
| mcfmodel | NCS031 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS031 | Optimal | 0.12 | 264 | 0 | 306 | 334 | −368.5 | −368.5 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| MODEL | INSTANCE | STATUS | CPU | #NL | #D | ROWS | COLUMNS | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | NCS031 | Optimal | 0.2 | 352 | 0 | 361 | 385 | −368.5 | −368.5 |
| sd1model | NCS031 | Optimal | 0.12 | 0 | 66 | 1,164 | 2,778 | −368.5 | −368.5 |
| sd2model | NCS031 | Optimal | $9{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,662 | −368.5 | −368.5 |
| mcfmodel | NCS032 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS032 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |
| smodel | NCS032 | Optimal | 0.21 | 352 | 0 | 361 | 388 | −368.5 | −368.5 |
| sd1model | NCS032 | Optimal | 0.56 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.5 |
| sd2model | NCS032 | Optimal | $9{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,665 | −368.5 | −368.5 |
| mcfmodel | NCS033 | Optimal | 0.23 | 112 | 0 | 108 | 165 | −368.2 | −368.2 |
| smodel-nocomp | NCS033 | Optimal | $9{\cdot}10^{-2}$ | 264 | 0 | 306 | 334 | −368.4 | −368.2 |
| smodel | NCS033 | Optimal | 0.65 | 352 | 0 | 361 | 385 | −334.5 | −334.2 |
| sd1model | NCS033 | Optimal | 4.89 | 0 | 66 | 1,164 | 2,778 | −368.4 | −368.2 |
| sd2model | NCS033 | Optimal | 0.43 | 0 | 36 | 774 | 1,662 | −368.4 | −368.2 |
| mcfmodel | NCS034 | Optimal | 0.1 | 112 | 0 | 108 | 165 | −368.4 | −368.4 |
| smodel-nocomp | NCS034 | Optimal | 0.11 | 264 | 0 | 306 | 334 | −368.5 | −368.4 |
| smodel | NCS034 | Unfinished | – | 352 | 0 | 361 | 385 | – | – |
| sd1model | NCS034 | Optimal | 1.75 | 0 | 66 | 1,164 | 2,778 | −368.5 | −368.4 |
| sd2model | NCS034 | Optimal | 0.81 | 0 | 36 | 774 | 1,662 | −368.5 | −368.3 |
| mcfmodel | NCS035 | Optimal | 0.18 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS035 | Optimal | 0.1 | 264 | 0 | 306 | 337 | −368.5 | −368.5 |
| smodel | NCS035 | Optimal | 0.28 | 352 | 0 | 361 | 388 | −368.5 | −368.5 |
| sd1model | NCS035 | Optimal | 0.27 | 0 | 66 | 1,164 | 2,781 | −368.5 | −368.5 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| MODEL | INSTANCE | STATUS | CPU | #NL | #D | ROWS | COLUMNS | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd2model | NCS035 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,665 | −368.5 | −368.5 |
| mcfmodel | NCS036 | Optimal | 0.1 | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS036 | Optimal | 0.11 | 264 | 0 | 306 | 337 | −368.6 | −368.6 |
| smodel | NCS036 | Unfinished | – | 352 | 0 | 361 | 388 | – | – |
| sd1model | NCS036 | Optimal | 0.34 | 0 | 66 | 1,164 | 2,781 | −368.6 | −368.6 |
| sd2model | NCS036 | Optimal | 0.23 | 0 | 36 | 774 | 1,665 | −368.6 | −368.6 |
| mcfmodel | NCS037 | Optimal | 0.11 | 112 | 0 | 108 | 165 | −368.6 | −368.6 |
| smodel-nocomp | NCS037 | Optimal | 0.1 | 264 | 0 | 306 | 334 | −368.6 | −368.6 |
| smodel | NCS037 | Optimal | 0.57 | 352 | 0 | 361 | 385 | −133.4 | – |
| sd1model | NCS037 | Optimal | 0.11 | 0 | 66 | 1,164 | 2,778 | −368.6 | −368.6 |
| sd2model | NCS037 | Optimal | $7{\cdot}10^{-2}$ | 0 | 36 | 774 | 1,662 | −368.6 | −368.5 |
| mcfmodel | NCS038 | Optimal | 0.11 | 112 | 0 | 108 | 165 | −368.5 | −368.5 |
| smodel-nocomp | NCS038 | Optimal | 0.12 | 264 | 0 | 306 | 342 | −368.5 | −368.5 |
| smodel | NCS038 | Unfinished | – | 352 | 0 | 361 | 393 | – | – |
| sd1model | NCS038 | Optimal | 0.3 | 0 | 66 | 1,164 | 2,786 | −368.5 | −368.5 |
| sd2model | NCS038 | Optimal | 0.11 | 0 | 36 | 774 | 1,670 | −368.5 | −368.5 |
| mcfmodel | NCS039 | Optimal | 0.1 | 112 | 0 | 108 | 165 | −368.1 | −368.1 |
| smodel-nocomp | NCS039 | Optimal | 0.13 | 264 | 0 | 306 | 337 | −368.3 | −368.1 |
| smodel | NCS039 | Optimal | 0.6 | 352 | 0 | 361 | 388 | −355.4 | −355.1 |
| sd1model | NCS039 | Optimal | 3.02 | 0 | 66 | 1,164 | 2,781 | −368.3 | −368.0 |
| sd2model | NCS039 | Optimal | 7.76 | 0 | 36 | 774 | 1,665 | −368.3 | −368.0 |

**Table 6.6:** Full results from NCS benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| mcfmodel | NCS040 | Optimal | $7{\cdot}10^{-2}$ | 112 | 0 | 108 | 165 | −367.8 | −367.5 |
| smodel-nocomp | NCS040 | Optimal | 0.39 | 264 | 0 | 306 | 334 | −367.8 | −367.5 |
| smodel | NCS040 | Optimal | 0.8 | 352 | 0 | 361 | 385 | −165.3 | – |
| sd1model | NCS040 | Optimal | 11.19 | 0 | 66 | 1,164 | 2,778 | −367.7 | −367.3 |
| sd2model | NCS040 | Optimal | 41.19 | 0 | 36 | 774 | 1,662 | −367.8 | −367.4 |

**Table 6.7:** Full results from NCS benchmarks with processing facility. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel-nocomp | NCS001 | Optimal | 3.3 | 216 | 0 | 533 | 525 | −380.9 | −380.5 |
| smodel-nocomp | NCS002 | Optimal | 5.5 | 216 | 0 | 533 | 515 | −376.3 | −376.0 |
| smodel-nocomp | NCS003 | Optimal | 1.6 | 216 | 0 | 533 | 520 | −381.3 | −380.9 |
| smodel-nocomp | NCS004 | Optimal | $8.3 \cdot 10^{-1}$ | 216 | 0 | 533 | 517 | −373.2 | −372.8 |
| smodel-nocomp | NCS005 | Unfinished | – | 216 | 0 | 533 | 520 | −376.2 | −373.6 |
| smodel-nocomp | NCS006 | Optimal | 4.0 | 216 | 0 | 533 | 517 | −379.3 | −378.9 |
| smodel-nocomp | NCS007 | Unfinished | – | 216 | 0 | 533 | 525 | −381.1 | −374.6 |
| smodel-nocomp | NCS008 | Optimal | 1.7 | 216 | 0 | 533 | 517 | −379.0 | −378.7 |
| smodel-nocomp | NCS009 | Optimal | 15.4 | 216 | 0 | 533 | 512 | −381.3 | −380.9 |
| smodel-nocomp | NCS010 | Optimal | 1.5 | 216 | 0 | 533 | 517 | −380.6 | −380.2 |
| smodel-nocomp | NCS011 | Optimal | 1.7 | 216 | 0 | 533 | 515 | −377.4 | −377.0 |
| smodel-nocomp | NCS012 | Optimal | 9.7 | 216 | 0 | 533 | 512 | −374.4 | −374.0 |
| smodel-nocomp | NCS013 | Optimal | $5.2 \cdot 10^{-1}$ | 216 | 0 | 533 | 520 | −381.0 | −380.7 |
| smodel-nocomp | NCS014 | Optimal | $7.8 \cdot 10^{-1}$ | 216 | 0 | 533 | 525 | −375.0 | −374.6 |
| smodel-nocomp | NCS015 | Optimal | 12.8 | 216 | 0 | 533 | 520 | −380.2 | −379.9 |
| smodel-nocomp | NCS016 | Optimal | $6.6 \cdot 10^{-1}$ | 216 | 0 | 533 | 517 | −380.8 | −380.4 |
| smodel-nocomp | NCS017 | Optimal | 1.8 | 216 | 0 | 533 | 515 | −375.6 | −375.2 |
| smodel-nocomp | NCS018 | Unfinished | – | 216 | 0 | 533 | 520 | −380.9 | −379.0 |
| smodel-nocomp | NCS019 | Optimal | 5.3 | 216 | 0 | 533 | 525 | −378.9 | −378.5 |
| smodel-nocomp | NCS020 | Optimal | 1.1 | 216 | 0 | 533 | 520 | −381.3 | −380.9 |
| smodel-nocomp | NCS021 | Optimal | 9.5 | 216 | 0 | 533 | 525 | −379.3 | −378.9 |
| smodel-nocomp | NCS022 | Optimal | $7.5 \cdot 10^{-1}$ | 216 | 0 | 533 | 517 | −381.3 | −380.9 |

**Table 6.7:** Full results from NCS benchmarks with processing facility. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel-nocomp | NCS023 | Optimal | 1.1 | 216 | 0 | 533 | 525 | −373.5 | −373.2 |
| smodel-nocomp | NCS024 | Optimal | 12.1 | 216 | 0 | 533 | 525 | −379.7 | −379.3 |
| smodel-nocomp | NCS025 | Unfinished | – | 216 | 0 | 533 | 512 | – | – |
| smodel-nocomp | NCS026 | Optimal | 22.7 | 216 | 0 | 533 | 512 | −375.9 | −375.6 |
| smodel-nocomp | NCS027 | Optimal | 2.5 | 216 | 0 | 533 | 520 | −379.7 | −379.3 |
| smodel-nocomp | NCS028 | Optimal | 3.5 | 216 | 0 | 533 | 525 | −380.3 | −380.0 |
| smodel-nocomp | NCS029 | Optimal | 2.9 | 216 | 0 | 533 | 520 | −379.7 | −379.3 |
| smodel-nocomp | NCS030 | Optimal | 1.6 | 216 | 0 | 533 | 512 | −375.5 | −375.1 |
| smodel-nocomp | NCS031 | Optimal | 3.6 | 216 | 0 | 533 | 520 | −379.3 | −378.9 |
| smodel-nocomp | NCS032 | Optimal | 1.7 | 216 | 0 | 533 | 517 | −379.1 | −378.8 |
| smodel-nocomp | NCS033 | Optimal | 8.1 | 216 | 0 | 533 | 512 | −372.9 | −372.5 |
| smodel-nocomp | NCS034 | Optimal | $7.6 \cdot 10^{-1}$ | 216 | 0 | 533 | 512 | −380.8 | −380.4 |
| smodel-nocomp | NCS035 | Optimal | 12.2 | 216 | 0 | 533 | 525 | −380.8 | −380.5 |
| smodel-nocomp | NCS036 | Optimal | 31.5 | 216 | 0 | 533 | 517 | −380.2 | −379.8 |
| smodel-nocomp | NCS037 | Optimal | 2.9 | 216 | 0 | 533 | 517 | −381.8 | −381.4 |
| smodel-nocomp | NCS038 | Optimal | $6.1 \cdot 10^{-1}$ | 216 | 0 | 533 | 520 | −380.8 | −380.4 |
| smodel-nocomp | NCS039 | Optimal | $7.6 \cdot 10^{-1}$ | 216 | 0 | 533 | 520 | −380.9 | −380.5 |
| smodel-nocomp | NCS040 | Unfinished | – | 216 | 0 | 533 | 512 | −379.8 | −375.8 |

**Table 6.8:** Full results from NCS benchmarks with processing facility and composite constraints. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | NCS001 | Optimal | 0.81 | 304 | 0 | 732 | 696 | −120.0 | – |
| smodel | NCS002 | Optimal | 3.06 | 304 | 0 | 732 | 686 | −121.1 | – |
| smodel | NCS003 | Unfinished | – | 304 | 0 | 732 | 691 | −410.3 | −380.8 |
| smodel | NCS004 | Optimal | 2,707.05 | 304 | 0 | 732 | 688 | −120.3 | – |
| smodel | NCS005 | Optimal | 6.91 | 304 | 0 | 732 | 691 | −372.8 | −372.5 |
| smodel | NCS006 | Unfinished | – | 304 | 0 | 732 | 688 | – | – |
| smodel | NCS007 | Unfinished | – | 304 | 0 | 732 | 696 | – | – |
| smodel | NCS008 | Optimal | 1.39 | 304 | 0 | 732 | 688 | −119.9 | – |
| smodel | NCS009 | Optimal | 16.27 | 304 | 0 | 732 | 683 | −381.1 | −380.7 |
| smodel | NCS010 | Optimal | 1.73 | 304 | 0 | 732 | 688 | −347.3 | −346.9 |
| smodel | NCS011 | Optimal | 1.78 | 304 | 0 | 732 | 686 | −120.3 | – |
| smodel | NCS012 | Optimal | 53.79 | 304 | 0 | 732 | 683 | −374.4 | −374.0 |
| smodel | NCS013 | Optimal | 1.02 | 304 | 0 | 732 | 691 | −120.0 | – |
| smodel | NCS014 | Unfinished | – | 304 | 0 | 732 | 696 | – | – |
| smodel | NCS015 | Optimal | 100.25 | 304 | 0 | 732 | 691 | −380.1 | −379.7 |
| smodel | NCS016 | Unfinished | – | 304 | 0 | 732 | 688 | −332.6 | −300.6 |
| smodel | NCS017 | Optimal | 9.93 | 304 | 0 | 732 | 686 | −375.6 | −375.2 |
| smodel | NCS018 | Unfinished | – | 304 | 0 | 732 | 691 | – | – |
| smodel | NCS019 | Optimal | 24.39 | 304 | 0 | 732 | 696 | −378.8 | −378.4 |
| smodel | NCS020 | Optimal | 0.58 | 304 | 0 | 732 | 691 | −381.1 | −380.8 |
| smodel | NCS021 | Unfinished | – | 304 | 0 | 732 | 696 | – | – |
| smodel | NCS022 | Optimal | 8.4 | 304 | 0 | 732 | 688 | −381.2 | −380.8 |

**Table 6.8:** Full results from NCS benchmarks with processing facility and composite constraints. Model, Instance, Solver Status, Cpu Time in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Columns | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | NCS023 | Optimal | 6.31 | 304 | 0 | 732 | 696 | −373.5 | −373.2 |
| smodel | NCS024 | Optimal | 85.14 | 304 | 0 | 732 | 696 | −378.8 | −378.4 |
| smodel | NCS025 | Optimal | 3.18 | 304 | 0 | 732 | 683 | −122.1 | – |
| smodel | NCS026 | Optimal | 11.86 | 304 | 0 | 732 | 683 | −375.9 | −375.6 |
| smodel | NCS027 | Optimal | 15.37 | 304 | 0 | 732 | 691 | −379.7 | −379.3 |
| smodel | NCS028 | Optimal | 1,803.26 | 304 | 0 | 732 | 696 | −120.3 | – |
| smodel | NCS029 | Unfinished | – | 304 | 0 | 732 | 691 | – | – |
| smodel | NCS030 | Unfinished | – | 304 | 0 | 732 | 683 | −374.0 | −346.2 |
| smodel | NCS031 | Optimal | 1.95 | 304 | 0 | 732 | 691 | −379.2 | −378.8 |
| smodel | NCS032 | Optimal | 24.22 | 304 | 0 | 732 | 688 | −119.9 | – |
| smodel | NCS033 | Optimal | 9.48 | 304 | 0 | 732 | 683 | −325.9 | −325.6 |
| smodel | NCS034 | Unfinished | – | 304 | 0 | 732 | 683 | −357.5 | −317.4 |
| smodel | NCS035 | Optimal | 0.92 | 304 | 0 | 732 | 696 | −120.0 | – |
| smodel | NCS036 | Optimal | 36.31 | 304 | 0 | 732 | 688 | −314.5 | −314.2 |
| smodel | NCS037 | Unfinished | – | 304 | 0 | 732 | 688 | – | – |
| smodel | NCS038 | Optimal | 2.21 | 304 | 0 | 732 | 691 | −120.3 | – |
| smodel | NCS039 | Unfinished | – | 304 | 0 | 732 | 691 | – | – |
| smodel | NCS040 | Optimal | 21.46 | 304 | 0 | 732 | 683 | −120.8 | – |

**Table 6.9:** Gap with time limit 1 hour for cases from the literature (GPP). Instances with no solution or gap > 100 are dashed.

| Instance | MCF | S | S-D1 | S-D2 |
|---|---|---|---|---|
| gppA1 | 0 | 0 | 0.0 | 0.0 |
| gppA2 | 0 | 0 | 0.0 | 0.0 |
| gppA3 | 0 | 0 | 0.0 | 0.0 |
| gppA4 | 0 | $1.2 \cdot 10^{-1}$ | $9.8 \cdot 10^{-4}$ | $3.7 \cdot 10^{-2}$ |
| gppA5 | 0 | 1.6 | 0.0 | 0.0 |
| gppB1 | 0 | $8.7 \cdot 10^{-1}$ | 0.0 | $8.4 \cdot 10^{-4}$ |
| gppB2 | 0 | 0 | 0.0 | 0.0 |
| gppB3 | 0 | $6.6 \cdot 10^{-2}$ | $6.6 \cdot 10^{-2}$ | $6.6 \cdot 10^{-2}$ |
| gppB4 | 0 | 0 | 0.0 | 0.0 |
| gppB5 | 0 | $2 \cdot 10^{-2}$ | $2 \cdot 10^{-2}$ | $2 \cdot 10^{-2}$ |
| gppC1 | 0 | $1.5 \cdot 10^{-1}$ | $1.5 \cdot 10^{-1}$ | $1.5 \cdot 10^{-1}$ |
| gppC2 | $1 \cdot 10^{-3}$ | $8.1 \cdot 10^{-1}$ | $9 \cdot 10^{-1}$ | $1 \cdot 10^{0}$ |
| gppC3 | 0 | $1.9 \cdot 10^{-1}$ | $3.6 \cdot 10^{-1}$ | $3.9 \cdot 10^{-1}$ |
| gppC4 | 0 | $1.5 \cdot 10^{-2}$ | $3.5 \cdot 10^{-2}$ | $3.2 \cdot 10^{-2}$ |
| gppC5 | 0 | $4.7 \cdot 10^{-1}$ | $5 \cdot 10^{-1}$ | $4.9 \cdot 10^{-1}$ |
| gppD1 | $2.5 \cdot 10^{-3}$ | $2.7 \cdot 10^{-1}$ | $4.7 \cdot 10^{-1}$ | 1.6 |
| gppD2 | 0 | $3.9 \cdot 10^{-1}$ | $5.7 \cdot 10^{-1}$ | 1.0 |
| gppD3 | 0 | $9.5 \cdot 10^{-3}$ | $1.7 \cdot 10^{-2}$ | $7.4 \cdot 10^{-2}$ |
| gppD4 | 0 | 1.1 | 1.2 | 1.2 |
| gppD5 | 0 | $4.2 \cdot 10^{-2}$ | $3.2 \cdot 10^{-1}$ | $3.7 \cdot 10^{-1}$ |
| gppE1 | 0 | – | 2.3 | 2.3 |
| gppE2 | 0 | – | 1.9 | 1.7 |
| gppE3 | 0 | – | 5.8 | 5.8 |
| gppE4 | 0 | – | 6.2 | 6.2 |
| gppE5 | 0 | – | 7.6 | 7.6 |
| gppL1 | $1.2 \cdot 10^{-3}$ | $1.2 \cdot 10^{-3}$ | $9.9 \cdot 10^{-4}$ | 0.0 |
| gppL2 | $1 \cdot 10^{-3}$ | $6.9 \cdot 10^{-1}$ | $9.6 \cdot 10^{-4}$ | $3 \cdot 10^{-1}$ |
| gppL3 | $1 \cdot 10^{-3}$ | $4.5 \cdot 10^{-1}$ | $9.1 \cdot 10^{-4}$ | $3.8 \cdot 10^{-1}$ |
| gppL4 | $2.1 \cdot 10^{-2}$ | $5.7 \cdot 10^{-1}$ | $5.8 \cdot 10^{-1}$ | $6 \cdot 10^{-1}$ |
| gppL5 | $9.9 \cdot 10^{-4}$ | $2 \cdot 10^{-1}$ | $1.8 \cdot 10^{-1}$ | $2.2 \cdot 10^{-1}$ |
| gppL6 | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | 0.0 | 0.0 |
| gppL7 | 0 | 0 | 0.0 | 0.0 |
| gppL8 | 0 | $1 \cdot 10^{-3}$ | $7.9 \cdot 10^{-4}$ | $1 \cdot 10^{-3}$ |
| gppL9 | 0 | 0 | 15.0 | 5.7 |
| gppL10 | 0 | 0 | 7.5 | 3.1 |
| gppL11 | 0 | 0 | 3.0 | 3.4 |
| gppL12 | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | 0.0 | 0.0 |

**Table 6.9:** Gap with time limit 1 hour for cases from the literature (GPP). Instances with no solution or gap > 100 are dashed.

| Instance | MCF | S | S-D1 | S-D2 |
|---|---|---|---|---|
| gppL13 | $1 \cdot 10^{-3}$ | $3 \cdot 10^{-3}$ | 0.0 | $1 \cdot 10^{-3}$ |
| gppL14 | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | 0.0 | $9.2 \cdot 10^{-4}$ |
| gppL15 | $1 \cdot 10^{-3}$ | $1 \cdot 10^{-3}$ | $9.5 \cdot 10^{-4}$ | $9.5 \cdot 10^{-4}$ |

Table 6.10 shows the full results from comparing several formulations for the NCS test instances without processing facilities or composite quality constraints. Note that the number of binary variables for formulation S-D1 is shown as 0. This is because SOS1 variables are not reported as binary variables by GAMS.

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| mcfmodel | gppA1 | Optimal | $3{\cdot}10^{-2}$ | 36 | 0 | 40 | 53 | −1,175.0 | −1,175.0 |
| smodel | gppA1 | Optimal | $3{\cdot}10^{-2}$ | 54 | 0 | 65 | 82 | −1,175.0 | −1,175.0 |
| sd1model | gppA1 | Optimal | 0.66 | 0 | 0 | 13,579 | 34,118 | −1,175.0 | −1,175.0 |
| sd2model | gppA1 | Optimal | 0.36 | 0 | 24 | 8,559 | 19,094 | −1,175.0 | −1,175.0 |
| mcfmodel | gppA2 | Optimal | $3{\cdot}10^{-2}$ | 20 | 0 | 29 | 38 | −641.0 | −641.0 |
| smodel | gppA2 | Optimal | $3{\cdot}10^{-2}$ | 24 | 0 | 55 | 72 | −641.0 | −641.0 |
| sd1model | gppA2 | Optimal | 0.27 | 0 | 0 | 7,306 | 17,090 | −641.0 | −641.0 |
| sd2model | gppA2 | Optimal | 0.14 | 0 | 12 | 4,796 | 9,578 | −641.0 | −641.0 |
| mcfmodel | gppA3 | Optimal | $3{\cdot}10^{-2}$ | 24 | 0 | 31 | 44 | −420.6 | −420.6 |
| smodel | gppA3 | Optimal | $3{\cdot}10^{-2}$ | 27 | 0 | 51 | 69 | −420.6 | −420.6 |
| sd1model | gppA3 | Optimal | 0.28 | 0 | 0 | 7,549 | 17,087 | −420.6 | −420.6 |
| sd2model | gppA3 | Optimal | 0.2 | 0 | 12 | 5,039 | 9,575 | −420.6 | −420.6 |
| mcfmodel | gppA4 | Optimal | $3{\cdot}10^{-2}$ | 42 | 0 | 45 | 57 | −599.0 | −599.0 |
| smodel | gppA4 | Unfinished | – | 60 | 0 | 74 | 90 | −669.4 | −599.0 |
| sd1model | gppA4 | Optimal | 223.49 | 0 | 0 | 16,349 | 42,635 | −599.6 | −599.0 |
| sd2model | gppA4 | Unfinished | – | 0 | 30 | 10,074 | 23,855 | −620.9 | −599.0 |
| mcfmodel | gppA5 | Optimal | $3{\cdot}10^{-2}$ | 34 | 0 | 41 | 51 | −198.0 | −198.0 |
| smodel | gppA5 | Unfinished | – | 57 | 0 | 78 | 98 | −508.8 | −198.0 |
| sd1model | gppA5 | Optimal | 82.14 | 0 | 0 | 16,353 | 42,643 | −198.0 | −198.0 |
| sd2model | gppA5 | Optimal | 185.63 | 0 | 30 | 10,078 | 23,863 | −198.0 | −198.0 |
| mcfmodel | gppB1 | Optimal | $4{\cdot}10^{-2}$ | 70 | 0 | 78 | 101 | −427.4 | −427.4 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | gppB1 | Unfinished | – | 112 | 0 | 142 | 173 | −797.3 | −427.4 |
| sd1model | gppB1 | Optimal | 507.75 | 0 | 0 | 21,136 | 51,221 | −427.4 | −427.4 |
| sd2model | gppB1 | Optimal | 1,053.22 | 0 | 36 | 13,606 | 28,685 | −427.7 | −427.4 |
| mcfmodel | gppB2 | Optimal | $4{\cdot}10^{-2}$ | 94 | 0 | 100 | 116 | −210.0 | −210.0 |
| smodel | gppB2 | Optimal | $6{\cdot}10^{-2}$ | 196 | 0 | 195 | 222 | −210.0 | −210.0 |
| sd1model | gppB2 | Optimal | 1.31 | 0 | 0 | 30,948 | 76,794 | −210.0 | −210.0 |
| sd2model | gppB2 | Optimal | 0.64 | 0 | 54 | 19,653 | 42,990 | −210.0 | −210.0 |
| mcfmodel | gppB3 | Optimal | 0.12 | 160 | 0 | 139 | 162 | −932.0 | −932.0 |
| smodel | gppB3 | Unfinished | – | 296 | 0 | 203 | 230 | −993.3 | −932.0 |
| sd1model | gppB3 | Unfinished | – | 0 | 0 | 40,223 | $1.02 \cdot 10^5$ | −993.3 | −932.0 |
| sd2model | gppB3 | Unfinished | – | 0 | 72 | 25,163 | 57,254 | −993.3 | −932.0 |
| mcfmodel | gppB4 | Optimal | $5{\cdot}10^{-2}$ | 150 | 0 | 136 | 156 | −912.8 | −912.8 |
| smodel | gppB4 | Optimal | 0.13 | 288 | 0 | 212 | 234 | −912.8 | −912.8 |
| sd1model | gppB4 | Optimal | 3.8 | 0 | 0 | 36,733 | 93,822 | −912.8 | −912.8 |
| sd2model | gppB4 | Optimal | 1.77 | 0 | 66 | 22,928 | 52,506 | −912.8 | −912.8 |
| mcfmodel | gppB5 | Optimal | $6{\cdot}10^{-2}$ | 150 | 0 | 139 | 155 | −439.0 | −439.0 |
| smodel | gppB5 | Unfinished | – | 328 | 0 | 233 | 266 | −447.7 | −439.0 |
| sd1model | gppB5 | Unfinished | – | 0 | 0 | 40,991 | $1.02 \cdot 10^5$ | −447.7 | −439.0 |
| sd2model | gppB5 | Unfinished | – | 0 | 72 | 25,931 | 57,290 | −447.7 | −439.0 |
| mcfmodel | gppC1 | Optimal | 0.15 | 210 | 0 | 198 | 247 | −1,352.7 | −1,352.7 |
| smodel | gppC1 | Unfinished | – | 420 | 0 | 359 | 406 | −1,549.6 | −1,352.7 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd1model | gppC1 | Unfinished | – | 0 | 0 | 53,110 | $1.37 \cdot 10^5$ | −1,549.6 | −1,352.7 |
| sd2model | gppC1 | Unfinished | – | 0 | 96 | 33,030 | 76,422 | −1,549.6 | −1,352.7 |
| mcfmodel | gppC2 | Optimal | 0.69 | 388 | 0 | 307 | 356 | −674.5 | −673.9 |
| smodel | gppC2 | Unfinished | – | 795 | 0 | 450 | 499 | −1,222.7 | −673.9 |
| sd1model | gppC2 | Unfinished | – | 0 | 0 | 76,693 | $1.96 \cdot 10^5$ | −1,222.7 | −644.8 |
| sd2model | gppC2 | Unfinished | – | 0 | 138 | 47,828 | $1.1 \cdot 10^5$ | −1,222.7 | −611.5 |
| mcfmodel | gppC3 | Optimal | 0.97 | 476 | 0 | 356 | 409 | −1,716.6 | −1,716.6 |
| smodel | gppC3 | Unfinished | – | 1,040 | 0 | 481 | 513 | −2,045.8 | −1,716.6 |
| sd1model | gppC3 | Unfinished | – | 0 | 0 | 97,276 | $2.56 \cdot 10^5$ | −2,045.8 | −1,508.0 |
| sd2model | gppC3 | Unfinished | – | 0 | 180 | 59,626 | $1.43 \cdot 10^5$ | −2,045.8 | −1,467.8 |
| mcfmodel | gppC4 | Optimal | 2.06 | 416 | 0 | 331 | 374 | −1,512.1 | −1,512.1 |
| smodel | gppC4 | Unfinished | – | 1,010 | 0 | 522 | 545 | −1,535.2 | −1,512.1 |
| sd1model | gppC4 | Unfinished | – | 0 | 0 | 95,536 | $2.47 \cdot 10^5$ | −1,535.2 | −1,483.8 |
| sd2model | gppC4 | Unfinished | – | 0 | 174 | 59,141 | $1.38 \cdot 10^5$ | −1,535.2 | −1,487.4 |
| mcfmodel | gppC5 | Optimal | 0.46 | 576 | 0 | 415 | 467 | −1,071.8 | −1,071.8 |
| smodel | gppC5 | Unfinished | – | 1,310 | 0 | 527 | 570 | −1,573.6 | −1,071.8 |
| sd1model | gppC5 | Unfinished | – | 0 | 0 | $1.1 \cdot 10^5$ | $2.9 \cdot 10^5$ | −1,573.6 | −1,049.2 |
| sd2model | gppC5 | Unfinished | – | 0 | 204 | 66,921 | $1.62 \cdot 10^5$ | −1,573.6 | −1,058.8 |
| mcfmodel | gppD1 | Unfinished | – | 934 | 0 | 673 | 783 | −1,994.0 | −1,989.1 |
| smodel | gppD1 | Unfinished | – | 1,986 | 0 | 842 | 901 | −2,503.4 | −1,965.8 |
| sd1model | gppD1 | Unfinished | – | 0 | 0 | $1.41 \cdot 10^5$ | $3.67 \cdot 10^5$ | −2,503.4 | −1,707.0 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd2model | gppD1 | Unfinished | – | 0 | 258 | 87,316 | $2.05 \cdot 10^5$ | −2,503.4 | −948.9 |
| mcfmodel | gppD2 | Optimal | 5.87 | 1,076 | 0 | 785 | 881 | −1,356.5 | −1,356.5 |
| smodel | gppD2 | Unfinished | – | 2,490 | 0 | 1,039 | 1,100 | −1,883.7 | −1,356.5 |
| sd1model | gppD2 | Unfinished | – | 0 | 0 | $1.49 \cdot 10^5$ | $3.75 \cdot 10^5$ | −1,883.7 | −1,201.3 |
| sd2model | gppD2 | Unfinished | – | 0 | 264 | 94,143 | $2.1 \cdot 10^5$ | −1,883.7 | −929.8 |
| mcfmodel | gppD3 | Optimal | 12.1 | 1,118 | 0 | 796 | 894 | −2,071.0 | −2,071.0 |
| smodel | gppD3 | Unfinished | – | 2,388 | 0 | 1,015 | 1,065 | −2,090.7 | −2,071.0 |
| sd1model | gppD3 | Unfinished | – | 0 | 0 | $1.59 \cdot 10^5$ | $4.09 \cdot 10^5$ | −2,090.7 | −2,055.1 |
| sd2model | gppD3 | Unfinished | – | 0 | 288 | 98,679 | $2.29 \cdot 10^5$ | −2,090.7 | −1,947.3 |
| mcfmodel | gppD4 | Optimal | 7.33 | 1,364 | 0 | 952 | 1,038 | −637.9 | −637.9 |
| smodel | gppD4 | Unfinished | – | 3,372 | 0 | 1,222 | 1,252 | −1,341.7 | −637.9 |
| sd1model | gppD4 | Unfinished | – | 0 | 0 | $1.94 \cdot 10^5$ | $5.03 \cdot 10^5$ | −1,341.7 | −597.4 |
| sd2model | gppD4 | Unfinished | – | 0 | 354 | $1.2 \cdot 10^5$ | $2.82 \cdot 10^5$ | −1,341.7 | −607.1 |
| mcfmodel | gppD5 | Optimal | 3.35 | 1,472 | 0 | 1,007 | 1,099 | −1,641.8 | −1,641.8 |
| smodel | gppD5 | Unfinished | – | 3,756 | 0 | 1,227 | 1,296 | −1,711.0 | −1,641.8 |
| sd1model | gppD5 | Unfinished | – | 0 | 0 | $2.1 \cdot 10^5$ | $5.46 \cdot 10^5$ | −1,711.0 | −1,300.0 |
| sd2model | gppD5 | Unfinished | – | 0 | 384 | $1.3 \cdot 10^5$ | $3.05 \cdot 10^5$ | −1,711.0 | −1,252.2 |
| mcfmodel | gppE1 | Optimal | 5.73 | 2,120 | 0 | 1,342 | 1,592 | −463.2 | −463.2 |
| smodel | gppE1 | Optimal | 11.58 | 11,089 | 0 | 2,450 | 2,791 | 0.0 | 0.0 |
| sd1model | gppE1 | Unfinished | – | 0 | 0 | $3 \cdot 10^5$ | $8.19 \cdot 10^5$ | −1,512.1 | −461.7 |
| sd2model | gppE1 | Unfinished | – | 0 | 576 | $1.79 \cdot 10^5$ | $4.58 \cdot 10^5$ | −1,512.1 | −461.7 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| mcfmodel | gppE2 | Optimal | 4.94 | 1,852 | 0 | 1,188 | 1,439 | −556.0 | −556.0 |
| smodel | gppE2 | Optimal | 17.47 | 9,464 | 0 | 2,276 | 2,634 | 0.0 | 0.0 |
| sd1model | gppE2 | Unfinished | – | 0 | 0 | $2.83 \cdot 10^5$ | $7.76 \cdot 10^5$ | −1,480.9 | −515.6 |
| sd2model | gppE2 | Unfinished | – | 0 | 546 | $1.69 \cdot 10^5$ | $4.34 \cdot 10^5$ | −1,480.9 | −542.9 |
| mcfmodel | gppE3 | Optimal | 16.14 | 2,440 | 0 | 1,540 | 1,768 | −78.7 | −78.7 |
| smodel | gppE3 | Optimal | 23.46 | 17,784 | 0 | 2,960 | 3,280 | 0.0 | 0.0 |
| sd1model | gppE3 | Unfinished | – | 0 | 0 | $3.52 \cdot 10^5$ | $9.55 \cdot 10^5$ | −536.1 | −78.7 |
| sd2model | gppE3 | Unfinished | – | 0 | 672 | $2.12 \cdot 10^5$ | $5.35 \cdot 10^5$ | −536.1 | −78.7 |
| mcfmodel | gppE4 | Optimal | 8.97 | 2,860 | 0 | 1,752 | 1,999 | −891.3 | −891.3 |
| smodel | gppE4 | Optimal | 41.78 | 18,096 | 0 | 3,007 | 3,365 | 0.0 | 0.0 |
| sd1model | gppE4 | Unfinished | – | 0 | 0 | $4.11 \cdot 10^5$ | $1.13 \cdot 10^6$ | −1,587.1 | −219.4 |
| sd2model | gppE4 | Unfinished | – | 0 | 798 | $2.45 \cdot 10^5$ | $6.34 \cdot 10^5$ | −1,587.1 | −219.4 |
| mcfmodel | gppE5 | Optimal | 22.04 | 2,960 | 0 | 1,829 | 2,054 | −221.4 | −221.4 |
| smodel | gppE5 | Optimal | 36.02 | 24,063 | 0 | 3,363 | 3,716 | 0.0 | 0.0 |
| sd1model | gppE5 | Unfinished | – | 0 | 0 | $4.32 \cdot 10^5$ | $1.18 \cdot 10^6$ | −1,892.8 | −221.4 |
| sd2model | gppE5 | Unfinished | – | 0 | 828 | $2.59 \cdot 10^5$ | $6.58 \cdot 10^5$ | −1,892.8 | −221.4 |
| mcfmodel | gppL1 | Optimal | $6 \cdot 10^{-2}$ | 20 | 0 | 25 | 38 | −42.6 | −42.6 |
| smodel | gppL1 | Optimal | 0.1 | 12 | 0 | 30 | 47 | −42.6 | −42.6 |
| sd1model | gppL1 | Optimal | 0.77 | 0 | 0 | 6,792 | 17,067 | −40.4 | −40.4 |
| sd2model | gppL1 | Optimal | 0.49 | 0 | 12 | 4,282 | 9,555 | −41.8 | −41.8 |
| mcfmodel | gppL2 | Optimal | 292.58 | 100 | 0 | 76 | 110 | −550.4 | −549.8 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| smodel | gppL2 | Unfinished | – | 180 | 0 | 99 | 132 | −928.8 | −549.8 |
| sd1model | gppL2 | Optimal | 1,337.26 | 0 | 0 | 25,372 | 68,188 | −550.1 | −549.6 |
| sd2model | gppL2 | Unfinished | – | 0 | 48 | 15,332 | 38,140 | −712.3 | −549.7 |
| mcfmodel | gppL3 | Optimal | 241.19 | 100 | 0 | 76 | 118 | −550.4 | −549.8 |
| smodel | gppL3 | Unfinished | – | 252 | 0 | 129 | 170 | −799.3 | −549.8 |
| sd1model | gppL3 | Optimal | 2,745.1 | 0 | 0 | 25,376 | 68,210 | −550.1 | −549.6 |
| sd2model | gppL3 | Unfinished | – | 0 | 48 | 15,336 | 38,162 | −760.8 | −549.6 |
| mcfmodel | gppL4 | Unfinished | – | 288 | 0 | 195 | 253 | −572.9 | −561.0 |
| smodel | gppL4 | Unfinished | – | 595 | 0 | 224 | 272 | −882.8 | −561.0 |
| sd1model | gppL4 | Unfinished | – | 0 | 0 | 47,228 | $1.28 \cdot 10^5$ | −882.8 | −558.9 |
| sd2model | gppL4 | Unfinished | – | 0 | 90 | 28,403 | 71,507 | −882.8 | −552.4 |
| mcfmodel | gppL5 | Optimal | 4.84 | 192 | 0 | 133 | 178 | −878.5 | −877.7 |
| smodel | gppL5 | Unfinished | – | 300 | 0 | 131 | 172 | −1,051.7 | −877.7 |
| sd1model | gppL5 | Unfinished | – | 0 | 0 | 32,151 | 85,242 | −1,005.0 | −852.0 |
| sd2model | gppL5 | Unfinished | – | 0 | 60 | 19,601 | 47,682 | −1,044.5 | −856.6 |
| mcfmodel | gppL6 | Optimal | 0.16 | 48 | 0 | 43 | 59 | −450.5 | −450.0 |
| smodel | gppL6 | Optimal | 127.65 | 32 | 0 | 35 | 51 | −450.5 | −450.0 |
| sd1model | gppL6 | Optimal | 20.01 | 0 | 0 | 13,063 | 34,091 | −450.0 | −450.0 |
| sd2model | gppL6 | Optimal | 14.47 | 0 | 24 | 8,043 | 19,067 | −450.0 | −450.0 |
| mcfmodel | gppL7 | Optimal | 0.11 | 168 | 0 | 135 | 156 | −3,500.0 | −3,500.0 |
| smodel | gppL7 | Optimal | 0.15 | 378 | 0 | 171 | 203 | −3,500.0 | −3,500.0 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd1model | gppL7 | Optimal | 1,329.58 | 0 | 0 | 57,279 | $1.53 \cdot 10^5$ | −3,500.0 | −3,500.0 |
| sd2model | gppL7 | Optimal | 2,273.71 | 0 | 108 | 34,689 | 85,757 | −3,500.0 | −3,500.0 |
| mcfmodel | gppL8 | Optimal | 0.2 | 80 | 0 | 71 | 85 | −1,100.0 | −1,100.0 |
| smodel | gppL8 | Optimal | 202.44 | 64 | 0 | 75 | 97 | −1,101.1 | −1,100.0 |
| sd1model | gppL8 | Optimal | 71.61 | 0 | 0 | 25,139 | 68,177 | −1,100.9 | −1,100.0 |
| sd2model | gppL8 | Optimal | 46.07 | 0 | 48 | 15,099 | 38,129 | −1,101.1 | −1,100.0 |
| mcfmodel | gppL9 | Optimal | 139.53 | 4,048 | 0 | 2,329 | 2,444 | −8.0 | −8.0 |
| smodel | gppL9 | Optimal | 7.21 | 4,224 | 0 | 825 | 900 | −8.0 | −8.0 |
| sd1model | gppL9 | Unfinished | – | 0 | 0 | $5.38 \cdot 10^5$ | $1.5 \cdot 10^6$ | −8.0 | $-5 \cdot 10^{-1}$ |
| sd2model | gppL9 | Unfinished | – | 0 | 1,056 | $3.17 \cdot 10^5$ | $8.38 \cdot 10^5$ | −8.0 | −1.2 |
| mcfmodel | gppL10 | Optimal | 23.34 | 4,048 | 0 | 2,329 | 2,444 | −8.0 | −8.0 |
| smodel | gppL10 | Optimal | 2.89 | 4,224 | 0 | 825 | 900 | −8.0 | −8.0 |
| sd1model | gppL10 | Unfinished | – | 0 | 0 | $5.38 \cdot 10^5$ | $1.5 \cdot 10^6$ | −8.0 | $-9.4 \cdot 10^{-1}$ |
| sd2model | gppL10 | Unfinished | – | 0 | 1,056 | $3.17 \cdot 10^5$ | $8.38 \cdot 10^5$ | −8.0 | −1.9 |
| mcfmodel | gppL11 | Optimal | 0.77 | 1,672 | 0 | 989 | 1,052 | −8.0 | −8.0 |
| smodel | gppL11 | Optimal | 1.15 | 1,728 | 0 | 397 | 456 | −8.0 | −8.0 |
| sd1model | gppL11 | Unfinished | – | 0 | 0 | $2.25 \cdot 10^5$ | $6.13 \cdot 10^5$ | −8.0 | −2.0 |
| sd2model | gppL11 | Unfinished | – | 0 | 432 | $1.35 \cdot 10^5$ | $3.43 \cdot 10^5$ | −8.0 | −1.8 |
| mcfmodel | gppL12 | Optimal | $9 \cdot 10^{-2}$ | 36 | 0 | 34 | 48 | −400.4 | −400.0 |
| smodel | gppL12 | Optimal | 37.85 | 28 | 0 | 32 | 47 | −400.4 | −400.0 |
| sd1model | gppL12 | Optimal | 19.76 | 0 | 0 | 12,812 | 34,087 | −400.0 | −400.0 |

**Table 6.10:** Full results from benchmark. Model, Instance, Solver Status, Cpu Time in seconds, Number of non-linear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | Instance | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| sd2model | gppL12 | Optimal | 14.96 | 0 | 24 | 7,792 | 19,063 | −400.0 | −400.0 |
| mcfmodel | gppL13 | Optimal | 0.12 | 36 | 0 | 34 | 48 | −600.6 | −600.0 |
| smodel | gppL13 | Unfinished | – | 28 | 0 | 32 | 47 | −601.8 | −600.0 |
| sd1model | gppL13 | Optimal | 79.25 | 0 | 0 | 12,812 | 34,087 | −600.0 | −600.0 |
| sd2model | gppL13 | Optimal | 257.4 | 0 | 24 | 7,792 | 19,063 | −600.6 | −600.0 |
| mcfmodel | gppL14 | Optimal | 0.2 | 36 | 0 | 34 | 48 | −750.8 | −750.0 |
| smodel | gppL14 | Optimal | 44.98 | 28 | 0 | 32 | 47 | −750.8 | −750.0 |
| sd1model | gppL14 | Optimal | 11.8 | 0 | 0 | 12,812 | 34,087 | −750.0 | −750.0 |
| sd2model | gppL14 | Optimal | 14.09 | 0 | 24 | 7,792 | 19,063 | −750.7 | −750.0 |
| mcfmodel | gppL15 | Optimal | 0.17 | 48 | 0 | 49 | 79 | −4,396.2 | −4,391.8 |
| smodel | gppL15 | Optimal | 1,153.3 | 150 | 0 | 187 | 187 | −4,396.2 | −4,391.8 |
| sd1model | gppL15 | Optimal | 109.22 | 0 | 0 | 19,645 | 51,229 | −4,356.4 | −4,352.3 |
| sd2model | gppL15 | Optimal | 102.6 | 0 | 36 | 12,115 | 28,693 | −4,373.7 | −4,369.6 |

Paper VI

Lars Hellemo, Paul I. Barton, and Asgeir Tomasgard:

# Stochastic Programming with Decision-Dependent Probabilities

*Submitted to an international, peer-reviewed journal*

# Chapter 7

# Stochastic Programs with Decision-Dependent Probabilities

**Abstract:**
Stochastic programming with recourse usually assumes uncertainty to be exogenous. We discuss modelling and application of decision-dependent uncertainty in mathematical programming and present a taxonomy of stochastic programming approaches with decision-dependent uncertainty.
We present several ways of incorporating direct or indirect manipulation of underlying probability distributions through decision variables in two-stage stochastic programming problems. We formulate two-stage models where prior probabilities are distorted through an affine transformation, or combined using a convex combination of several probability distributions. Additionally, we present models where the parameters of the probability distribution are first stage decision variables. The probability distributions are either incorporated in the model using the exact expression or by using a rational approximation.
Test instances for each formulation are solved with a commercial solver, BARON, using selective branching.

## 7.1 Introduction

Most practical decision problems involve uncertainty at some level, and stochastic programming was introduced by Dantzig [1955] and Beale [1955] to handle uncertain parameters in mathematical programs. Their approach was to model a discrete time decision process where uncertain parameters are represented by scenarios and their respective probabilities. In a scenario-based stochastic program, decisions are made and uncertain values are revealed at discrete points in time. Some decisions are made before the actual values of uncertain parameters are known, but the realization of the stochastic parameters is independent of the decisions. This framework will later be referred to as stochastic programs with exogenous uncertainty or stochastic programming with decision independent uncertainty. In recent years stochastic programs with endogenous uncertainty or decision-dependent uncertainty have received increased attention. Some early examples of papers with decision-dependent uncertainty are Jonsbråten [1998], Jonsbråten et al. [1998] and Goel & Grossmann [2004]. We will use the terms decision-dependent uncertainty and endogenous uncertainty interchangeably.

The main contribution of our paper is to provide new formulations for endogenous stochastic programming models where the probabilities of future events depend on decision variables in the optimization model, in the following called stochastic programs with decision-dependent probabilities. This is a subclass of endogenous stochastic programming models that has received little attention in the literature. There are some examples in the existing literature of problems where a decision may shift from one predefined set of probabilities to another. To the best of our knowledge, there are no examples in the literature where the relation is modeled as a continuous function. In Section 7.2 we give a more thorough description of problem classes with endogenous uncertainty and discuss the choices a problem owner or modeler needs to make. We provide an extended taxonomy for stochastic programs with endogenous uncertainty and a literature review in Section 7.3. Our new formulations for models with decision-dependent probabilities are found in Section 7.4. We provide several test instances of models using these formulations in Section 7.5. We give some computational results in Section 7.6 and conclude in Section 7.7.

## 7.2 Decision Problems with Decision-Dependent Uncertainty

To discuss the concept of decision-dependent uncertainty, it is useful to first make distinctions between the real world, the description of the real world presented to the modeler as a problem and the actual mathematical model formulation. A problem description belongs to one of these classes:

**Deterministic problems** are problems where there is no substantial uncertainty, there may for example be available precise measurements of all parameters, or there may be some official values available, such as the prices for today's operations.

**Exogenous uncertainty problems** are problems with substantial uncertainty, where we know the distribution of the stochastic parameters, for example based on historical data or expert opinion. The information structure and the probability distributions do not depend on any decisions in the model. Rather, the model will seek a solution that does well in expectation. Some models also include different risk attitudes or use a risk measure.

**Endogenous uncertainty problems** are problems where decisions at one point in time will have a substantial impact on the uncertainty faced later, either in terms of when information about the actual value of a stochastic parameter becomes available, or the probability that a certain realization of a

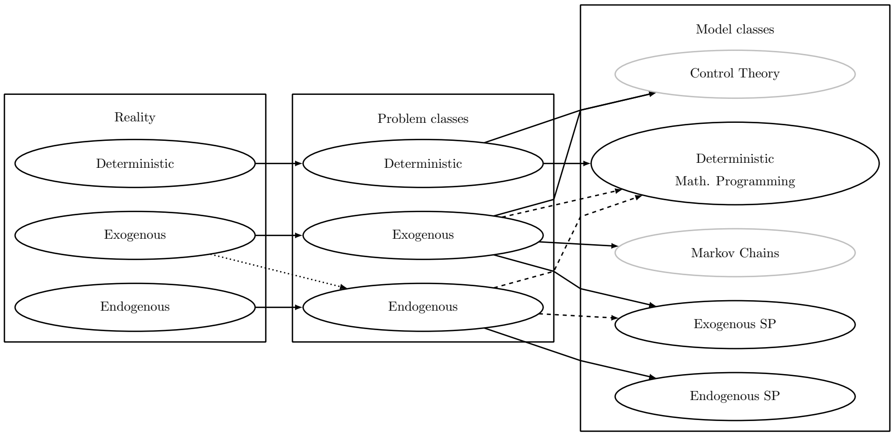



**Figure 7.1:** Classification problem and model classes. Stochastic programming models in black, examples of other modelling paradigms in grey. Dashed line indicates problem relaxation, whereas the full stroke indicates a mapping.

parameter occurs. We classify a problem as having endogenous uncertainty when decisions that are part of the problem to be solved, influences the uncertainty of parameters that are also part of the problem.

Note that there is not a one to one mapping between reality and the problem description or between the problem and the model choice. Figure 7.1 shows some alternative mappings. In the following we will illustrate this with some examples.

First, consider a river where a dam is to be built and the design parameters of the dam are to be determined. The risk of a dam break has to be balanced against the extra cost of further reinforcing it. The stochastic inflow is not influenced by the way the dam is built, rather the dam's resistance to various inflows is. In this case a problem description may focus on the stochastic inflow, and describe this as a design problem with exogenous uncertainty. The risk of a dam failure would depend on the stochastic inflow, but the design decision would not affect the stochastic parameter in the model. Alternatively, we could decide to model the probability of a dam break directly as uncertainty depending on the dam design. The computational tractability of the two models would be one important criterion in selecting between the two.

Next consider a petroleum reservoir where there is some uncertainty about the properties of the reservoir, and the decisions are the technology used for drilling wells, where to drill wells, as well as when the wells should be drilled if drilled at all. The actual petroleum content of the reservoir is fixed, but not known precisely. The decision to drill test wells does not change the content of the reservoir as such, but it may provide more information about the reservoir. This information is not revealed unless we actually drill the test wells, which incurs a substantial cost. This is a situation where the underlying reservoir content is deterministic, but unknown to the decision maker. The information structure (when and what information is revealed) is affected by the decisions, calling for a model that handles decision-dependent uncertainty. Note that the underlying uncertain parameter is not affected. In the same reservoir case, the choice between alternative drilling technologies is another, similar, consideration. Some drilling approaches may jeopardize the reservoir itself by introducing leaks between layers in the ground, something that could render part of the resources unrecoverable. In this way, our decisions may actually change the recoverable volume from said reservoir. This aspect of the real world situation may be included or ignored in the decision problem and later in a mathematical model, giving very different models.

For this oil reservoir with fixed but unknown petroleum content we may choose to ignore information structure or operational uncertainty or assume we know the true state of the reservoir already and apply a deterministic model. Alternatively we may choose to see it as an endogenous problem, where our decisions in part determine the uncertainty of the recoverable volumes or change the information structure, or we may see it as an exogenous problem, where we get to know the true state of the reservoir at some future point in time.

The right-hand part of Figure 7.1 shows examples of model classes. Moving on to formulating a specific model to aid the solution of a certain problem, some relaxations or approximations will usually have to be made, often to reduce cognitive load of model users or to improve computational tractability, or both. While in this paper we focus on stochastic programming, also other modeling paradigms exist such as control theory, game theory and several others that may be considered for stochastic problems. In the following literature review and taxonomy description we limit the scope to include problems described with endogenous uncertainty and where the model choice is stochastic programming with recourse.

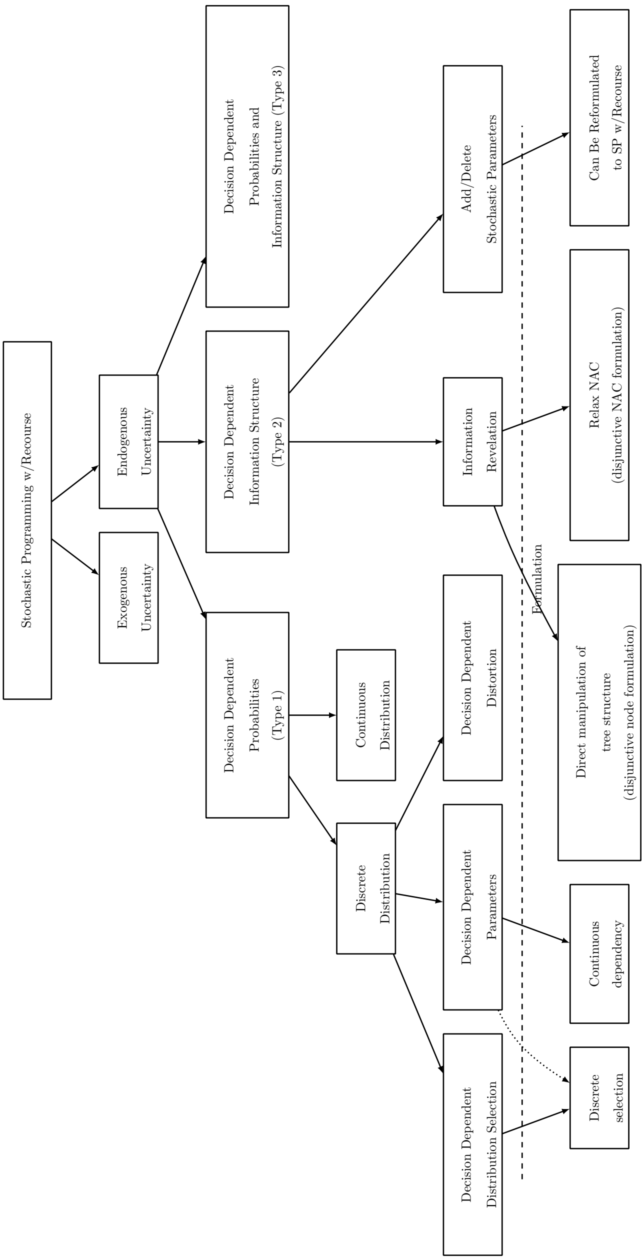



**Figure 7.2:** Classification of endogenous SPs

## 7.3 Taxonomy

In this section we will present a taxonomy and literature review for stochastic programs with decision-dependent uncertainty. Our taxonomy expands previously presented classifications of such problems. We summarize the taxonomy in Figure 7.2.

The literature on endogenous uncertainty in stochastic programming is sparse. This should come as no surprise as one quickly departs from the domains where well performing solution techniques are available, notably for convex programming in general and linear programming in particular, as noted by Varaiya & Wets [1989]. Jonsbråten [1998] and Jonsbråten et al. [1998] proposed a generalized formulation of stochastic programs with recourse of which the standard SP is a special case (Equation (7.1)), and suggested the classification of stochastic programs into two sub classes: endogenous and exogenous uncertainty.

$$\min\ E^p f(x) = \int_{\Xi} f(\xi; x)p(d\xi) \text{ such that } (p, x) \in \mathcal{K} \subset \mathcal{P} \times \mathbb{R}^N \tag{7.1}$$

$\mathcal{P}$ is a subset of the probability measures on $\Xi$ and $\mathcal{K}$ are the constraints linking the decision $x$ to the choice of $p$.

The problems discussed in the paper by Jonsbråten et al., concern situations where the time that information becomes available is determined by the decisions in preceding stages. As an example they use stochastic production costs. Only after making the decision of which product to make, is the uncertainty of this particular product revealed. The other possible products' true costs remain hidden (stochastic) until a decision to produce them is made.

Several authors (Dupačová [2006], Tarhan et al. [2009]) identify two subclasses within endogeneous stochastic programs. One class of problems is where the probabilities are decision-dependent, and we will denote this class of problems as *Decision-Dependent Probabilities* or Type 1. Equation (7.2) further generalizes Equation (7.1) to include the possibility that the probability measure also depends on $x$:

$$\min\ E^p f(x) = \int_{\Xi} f(\xi; x)p(x; d\xi) \text{ such that } (p, x) \in \mathcal{K} \subset \mathcal{P} \times \mathbb{R}^N \tag{7.2}$$

Problems with decision-dependent probabilities are discussed further in Section 7.3. The other subclass concerns information revelation, where the time of the information revelation is decision-dependent.

We suggest a more general categorization, *Decision-Dependent Information Structure* or Type 2. Problems with decision-dependent information structure are discussed further in Section 7.3. Some problems may have both kinds of decision-dependent uncertainty, and we add a Type 3 to include such problems. To the best of our knowledge, problems of Type 3 have not yet been discussed in the literature. For an overview of the different problem classes and their subclasses, see Figure 7.2.

## Decision-Dependent Information Structure

By decision-dependent information structure we mean all ways of altering the time dynamics of a stochastic program. This includes the time of information revelation, as in endogenous problems of Type 2, as well as the addition of stochastic parameters, and deletion of stochastic parameters. Another example is problems for which the time when uncertainty is redefined/refined is a decision variable, such as in using sensors or in acquisition of information. We include in this category all stochastic programs with endogenous uncertainty were nonanticipativity constraints (NAC) can be manipulated by decision variables, whereas the probabilities remain fixed.

### Information Revelation

The subcategory of information revelation has received most attention in the literature, following Jonsbråten [1998], Jonsbråten et al. [1998] and Goel & Grossmann [2004]. The most used technique is to relax the nonanticipativity constraints of a stochastic program, allowing selection of the times of branching of the tree (when scenarios become distinguishable), see discussion below.

Goel & Grossmann [2004] formulated a model for development of natural gas resources where the time of exploitation can be selected in the model. This introduces endogenous uncertainty as the information revelation depends on which wells are drilled and when, and it is formulated as a disjunctive programming problem where the nonanticipativity constraints depend on the decision variables related to drilling. They first considered a model with pure decision-dependent uncertainty, and later generalized it to a hybrid model including both endogenous and exogenous uncertainty (Goel & Grossmann [2006]). This form of endogenous uncertainty arises in multi-stage models, where the decisions to explore a field unravels the true parameter values of the field that is explored, but not the others. As this decision can be made at different times (stages), it is only relevant in a multi-stage environment. Effectively their approach is a model with decision-dependent nonanticipativity constraints, and they develop several theoretical results demonstrating redundancy in the constraints and that the number of nonanticipativity constraints can be reduced accordingly. This improves the practicality of the model by making it more readily solvable. The models are still quite large, though, and they propose a branch and bound solution procedure based on Lagrangian duality.

Solak [2007] presents a portfolio optimization problem where the timings of the realizations are dependent on the decisions to invest in the projects. The application is from R&D in the aviation industry where a technology development portfolio is to be optimized. Solak introduces gradual resolution of uncertainty, where the amount invested in a project increases the resolution of the uncertainty regarding that project up to a point where all uncertainty has been resolved. The author proposes solution approaches for the multi-stage stochastic integer programming model with focus on decomposability, sample average approximation and Lagrangian relaxation with lower bounding heuristics.

A model with gradual resolution of information is also presented by Tarhan et al. [2009], another petroleum application with a multi-stage non-convex stochastic model, solved by a duality-based branch and bound method. Colvin & Maravelias [2009] build on the work by Goel and Grossmann. Their application is from the pharmaceutical industry and clinical trials. They further improve on a reformulation with redundant nonanticipativity constraints removed, and observe that few of the remaining are binding. They add the constraints only as needed through a customized branch-and-cut algorithm. The model is formulated as a pure MIP. Boland et al. [2008] also build on the work of Goel and Grossman in their open pit mining application where geological properties of the mining blocks (quality) varies, and there is a mix of already mined blocks, and blocks where the quality is uncertain until the point of development. They find that they can reuse existing variables for nonanticipativity constraints and thus reduce the size of the problem. They exploit the problem structure to omit a significant proportion of the nonanticipativity constraints. Boland et al. implemented a version of their model with “lazy” constraints, but found that this did not improve performance for their model instances.

The latest improvement on the work by Goel and Grossman is by Gupta & Grossmann [2011], and they also propose new methods for obtaining a more compact representation of the nonanticipativity constraints. In addition, they propose three solution procedures. One is based on a relaxation of the problem in what they call a k-stage constraints problem, where only nonanticipativity constraints for a given number of stages are included. Secondly, they propose an iterative procedure for nonanticipativity constraint relaxation, and third they present a Lagrangian decomposition algorithm. The application is the same as in Goel & Grossmann [2006].

An alternative and equivalent way of formulating stochastic programming problems with recourse is using a node formulation of the scenario tree. As an alternative to the disjunctive nonanticipativity constraints (NAC) formulation with relaxation of NAC, problems with decision-dependent information revelation may be formulated using a disjunctive node formulation. However, to our knowledge, such a model has never been presented in the literature.

In an early paper, Artstein & Wets [1994] present a framework for analysis where a decision maker can seek more information through what they call a sensor. By using a sensor, they allow a redefinition of the probability distribution that is used in the stochastic program. This refines the decision process in that it acknowledges that the inquiry process may itself introduce errors. They solve an example based on a variant of the newsboy problem where the newsboy may perform a poll/sampling to gain information about the probability distribution, possibly at a cost. They provide a general approach to the situation when the underlying uncertainty is not known, and decisions may influence the accuracy of the uncertainty in a stochastic program.

#### Problems that May be Reformulated as Ordinary SP

In addition to problems with decision-dependent information revelation, other structures are conceivable, that may be reformulated as stochastic programs with recourse. This includes deleting stochastic variables, adding stochastic variables, and modifying the support. This may be achieved through the use of binary variables. For a recent example, see Ntaimo et al. [2012] where a two-stage stochastic program for wildfire initial attacks is presented. The cost incurred by each wildfire is one of two possible outcomes for each scenario, depending on whether the fire can be contained through an effective attack or not. The model is formulated as a two-stage stochastic (integer) program with recourse, with binary variables to select which set of recourse costs is incurred in stage two based on the selection of attack means available as a consequence of decisions in stage one. The scenarios are based on fire simulations, giving a large number of scenarios. The model size is reduced by applying sample average approximation (SAA).

### Decision-Dependent Probabilities

The first attempt to model explicitly the relationship between the probability measure and the decision variable was made by Ahmed [2000]. He formulates single-stage stochastic programs that are applied to network design, server selection and p-choice facility location. Ahmed uses Luce's choice axiom to develop an expression for the probability that, e.g., a path is used, and this probability depends on the design variables of the network. The resulting model is 0-1 hyperbolic program, which he solves by a binary reformulation and by genetic programming in addition to a customized branch and bound algorithm.

For some problems with decision-dependent probabilities, the decision dependency may be removed through an appropriate transformation of the probability measure, which is called the push-in technique by Rubinstein & Shapiro [1993, 214f], see also Pflug [1996, 143ff]. Dupačová [2006] notes that in some cases, dependence of distribution $\mathcal{P}$ on decision variable $x$ can be removed by a suitable transformation of the decision-dependent probability distribution (push-in technique).

Escudero et al. [2014] have developed a multi-stage stochastic model including both exogenous and endogenous uncertainty. They also include risk considerations in the form of stochastic dominance constraints. The resulting model is a mixed-integer quadratic program where the weights (probabilities) of each scenario group and/or outcomes of the stochastic parameters may be determined by decision variables from previous stages. To be able to solve large problem instances the authors apply a customized Branch and Fix Coordination (BFC) parallel algorithm.

For the problems in this section, only probabilities depend on the decision variables, while the information structure is fixed. To be specific, nonanticipativity constraints are not manipulated by decision variables. Dupačová [2006] identifies two fundamental classes of problems with endogenous probabilities. One where the probability distribution is known and the decisions influence the parameters and one where some decision will cause the probability distribution to be chosen between a finite set of probability distributions Dupačová [2006]. We extend her taxonomy with a third category, decision-dependent distribution distortion.

In principle, both discrete and continuous distributions may be considered, where the use of discrete scenarios as an approximation also for continuous distributions is the most used method for modeling such problems. We are not familiar with any attempts to model and solve problems with decision-dependent probabilities using continuous probability distributions, and in the following we only consider problems with discrete probability distributions, using a set of discrete scenarios.

### Decision-Dependent Distribution Selection

Viswanath et al. [2004] consider the design of a robust transportation network where links can be reinforced by investing in additional measures. By investing, the probability of survival of a disruptive event is improved. The model is an investment model with a choice between a finite number of sets of probabilities, typically two, $p_e$ and $q_e$ where $p_e$ is used if there is investment, $q_e$ otherwise. The random variables take values 0 or 1 with probabilities given above. Dupačová [2006] also discusses the subset of problems where available techniques from binary and integer programming can be can be applied to choosie between a finite number of set of probability distributions with fixed parameters.

#### Decision-Dependent Parameters

Selection between a discrete number of parameter values can be implemented using a generalization of the technique described above. We suggest some models where parameters are continuous decision variables in this paper, see Section 7.4. We show an example of using the exact expression for a probability distribution in Section 7.4 and a rational approximation in Section 7.4. We are not aware of any other papers that include models of Type 1 where the probability distribution parameters can be set continuously.

#### Distortion

We also include some models where we distort some prior set of probabilities for a distribution with known parameters. We then introduce a distortion of these probabilities controlled by decision variables. This distortion could be applied in form of a transformation of one set of probabilities or by combining several sets of probabilities. We give examples of linear transformations in Section 7.4, distorting one set of prior probabilities in Section 7.4 and using the convex combination of several sets of probabilities in Section 7.4.

We are not aware of any other publications to present this kind of model, however Dupačová [2006] makes notes on the stability of optimal solutions, using probability distribution contamination to investigate the case where a convex combination of several distributions can be applied for convex problems.

#### Related Work

A bit on the side, Held & Woodruff [2005] consider a multi stage stochastic network interdiction problem. The goal is to maximize the probability of sufficient disruption, in terms of maximizing the probability that the minimum path length exceeds a certain value. They present an exact (full enumeration) algorithm and a heuristic solution procedure.

Another approach to uncertainty in optimization is to search for solutions that are robust in the sense that they are good for the most disadvantageous outcomes of the stochastic parameters. Several research groups are working with robust optimization, going back to Ben-Tal et al. [1994], Ben-Tal & Nemirovski [1998], Bertsimas & Sim [2003] and Bertsimas & Sim [2004]. Also, rather than taking a worst-case approach, introducing some ambiguity to the underlying probability distribution has been demonstrated in the works of Pflug & Wozabal [2007] and Pflug & Pichler [2011].

Finally, while the optimization over a number of discrete scenarios is the dominant approach within stochastic programming, Kuhn [2009] and Kuhn et al. [2011] optimize linear decision rules over a continuous probability distribution.

## 7.4 Decision-Dependent Probabilities

We will give here several formulations of stochastic programs with decision-dependent probabilities. The formulations allow the probabilities of scenarios $s \in \mathcal{S}$ to be altered by some decision variable $y$, typically a first-stage variable in a two-stage stochastic program. In this section we will only consider the case where the function $p_s : \mathbb{R} \to [0,1]$ is an affine function.

$$\begin{aligned} \min c_x^\top x + c_y^\top y + \sum_s p_s(y) q_s^\top z_s \\ s.t. \\ x \in X, y \in Y, z_s \in Z_s(x,y). \end{aligned} \tag{7.3}$$

### Affine $p_s$

In this formulation we do not directly manipulate the parameters of the probability distribution, but apply a transformation to one or more predetermined probability distributions. We will first examine some special cases where the function $p_s$ is an affine function. This is primarily motivated by computational tractability, as it will yield mathematical programs where, in the case where the rest of the model is linear, the only nonlinearities are bilinear terms related to variables controlling scenario probabilities. This can easily be generalized to non-linear transformations and non-linear stochastic programs.

#### Linear Scaling

Let $s \in \mathcal{S}$ be discrete scenarios, each with probability $p_{0s} > 0, \sum_{s\in\mathcal{S}} p_{0s} = 1$. For each $s \in \hat{\mathcal{S}} \subset \mathcal{S}$ we let the variable $y$ scale the probability linearly, whereas we adjust the remaining scenarios $s \in \mathcal{S} \setminus \hat{\mathcal{S}}$:

$$p_s(y) = \begin{cases} p_{0s} y, & s \in \hat{\mathcal{S}} \subset \mathcal{S} \\ \frac{1 - y\sum_{s'\in\hat{\mathcal{S}}} p_{0s'}}{\sum_{s'\in\mathcal{S}\setminus\mathcal{S}} p_{0s'}} p_{0s}, & s \in \mathcal{S} \setminus \hat{\mathcal{S}}. \end{cases} \tag{7.4}$$

In the special case where the original distribution is uniform, this gives the function $p_s$:

$$p_s(y) = \begin{cases} \frac{1}{|\mathcal{S}|} y, & s \in \hat{\mathcal{S}} \subset \mathcal{S} \\ \frac{1 - y\sum_{\hat{\mathcal{S}}} \frac{1}{|\mathcal{S}|}}{|\mathcal{S}| - |\hat{\mathcal{S}}|}, & s \in \mathcal{S} \setminus \hat{\mathcal{S}}. \end{cases} \tag{7.5}$$

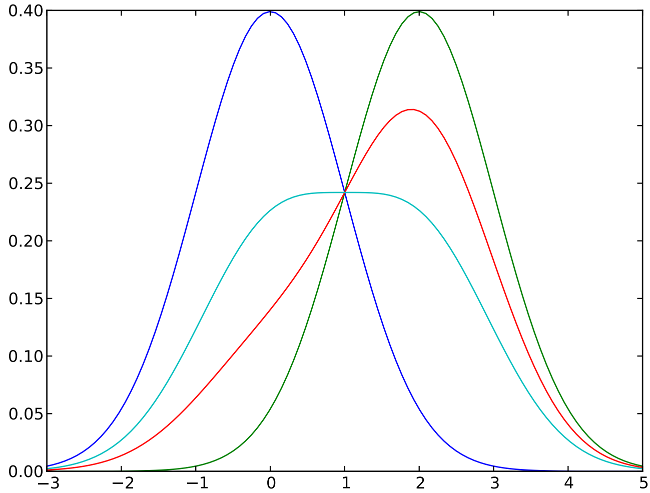



**Figure 7.3:** Example of convex combination of normal distributions.

### Convex Combination of Distributions

Let $i \in \mathcal{I}$ be discrete distributions with probabilities $p_{i,s}, \sum_{s \in \mathcal{S}} p_{i,s} = 1, \forall i \in \mathcal{I}$ associated to each scenario $s \in \mathcal{S}$.

Then define

$$p_s = \sum_{i \in \mathcal{I}} p_{i,s} y_i, \forall s \in \mathcal{S}. \tag{7.6}$$

A distribution defined like this is often called a mixture distribution, see, e.g., Feller [1943], Behboodian [1970], and Frühwirth-Schnatter [2006]. One interpretation would be that the final outcome is selected at random from the underlying distributions, with a certain probability $y_i$ associated with each of them. In our model the mixture weights $y_i \geq 0$ are decision variables, but of course the sum of weights need to be 1. See Figure 7.3 for some examples of convex combinations of normal distributions.

Mixture distributions are often used when subsets of the data have specific characteristics, for example where subpopulations exist in a population. Our model then gives the opportunity to influence the weights of the different subpopulations, potentially at a cost.

To reduce the number of $y$-variables, we may let the one $y_u$ is uniquely determined by the remaining $i \in \mathcal{I} \setminus \{u\}$ such that:

$$p_s = \sum_{i \in \mathcal{I} \setminus \{u\}} p_{i,s} y_i + (1 - \sum_{i \in \mathcal{I} \setminus \{u\}} y_i) p_{u,s}, u \in \mathcal{I}, \forall s \in \mathcal{S}. \tag{7.7}$$

## Parameterization of Distribution

In this formulation we change the parameters of a probability distribution directly, rather than distorting or combining some preexisting probability distributions. Taking a known probability distribution and letting the model choose the mean, or variability, for example, would allow for a range of interesting applications. This formulation gives the ability to model general properties such as an increase of the expected value or reduction of variability. It is often desirable to apply continuous distributions. In order to stay within the frameworks of scenario based recourse models we then have to discretize the distribution.

Here we provide short description of the discretization we use: For a stochastic parameter $x$, we define an allowed interval $[X^L, X^U]$ which is divided into $|S|$ subintervals, one for each scenario $s \in S$. The subintervals are $[x_{L,s}, x_{U,s}], X^L \leq x_{L,s}, x_{U,s} \leq X^U, \forall s \in S$, using a representative value $x_{M,s}$ for each scenario, normally $x_{M,s} = \frac{x_{L,s}+x_{U,s}}{2}$. The probability of a scenario $p_s$ is given by the cumulative probability (cumulative density function, cdf) of the upper value less the cumulative probability of the lower value of each subinterval: $p_s = \mathit{cdf}(x_{U,s}) - \mathit{cdf}(x_{L,s})$.

We will first give a formulation using a discretization of a probability distribution with closed form cdf Section 7.4, then a discretization of an approximation of normal distributions in Section 7.4.

### Kumaraswamy Distribution

The double bounded pdf proposed by Kumaraswamy [1980] for better matching observed values in hydrology has the nice property that the cdf is available in closed form.

The Kumaraswamy probability density function with parameters $a, b > 0, x \in [0, 1]$ is given as:

$$f(x|a, b) = abx^{a-1}(1 - x^a)^{b-1}. \tag{7.8}$$

While the cumulative density function is:

$$F(x|a, b) = 1 - (1 - x^a)^b. \tag{7.9}$$

Note that the original formulation allows parameters $a, b \geq 0$, but as this would allow situations where the probability of all scenarios equal to 0, we exclude

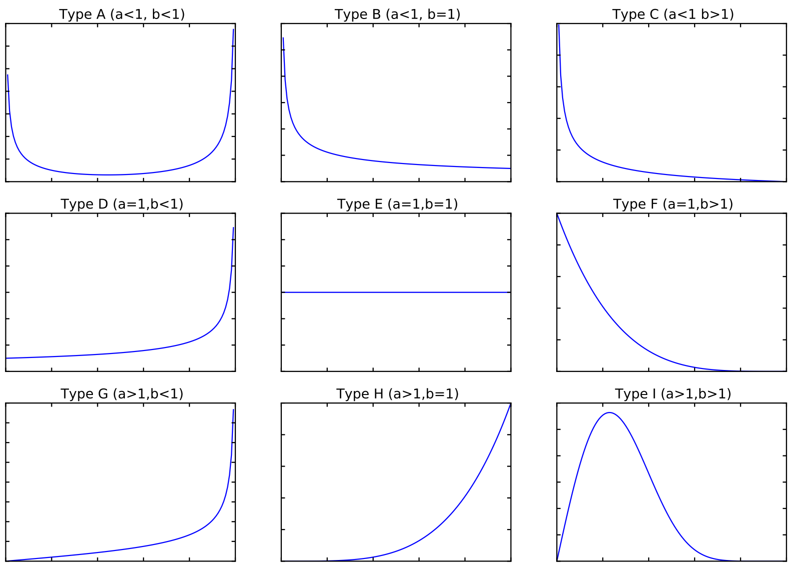



**Figure 7.4:** Examples of Kumaraswamy probability density function (pdf) with different parameters a and b.

this possibility. When parameters $a$ or $b$ pass from a value less then 1.0 to a value greater than 1.0, the shape of the probability density function changes, see Figure 7.4 for examples.

More general general (bounded) random variables $z$ can be normalized:

$$x = \frac{z - z_{\min}}{z_{\max} - z_{\min}} \tag{7.10}$$

With the cumulative probability given as a closed form expression, the discretized Kumaraswamy distribution can be directly included in an optimization model as follows, see also example in Section 7.5:

$$\begin{aligned} p_s(a,b) = F(x_{U,s}|a,b) - F(x_{L,s}|a,b) = \\ 1 - (1 - x_{U,s}^a)^b - 1 + (1 - x_{L,s}^a)^b = (1 - x_{L,s}^a)^b - (1 - x_{U,s}^a)^b, \forall s \in \mathcal{S}. \end{aligned} \tag{7.11}$$

### Approximation of Normal Distribution

The widely applied normal distribution has no closed form cdf, which makes it difficult to apply directly. Fortunately, there are polynomial and rational approximations to the standard normal distribution. For example, the cdf of the standard normal distribution can be approximated for $x \geq 0$ with the following expression (Abramowitz & Stegun [1964, 26.2.19]):

$$\begin{aligned} P(x) = 1 - \frac{1}{2}(1 + d_1x + d_2x^2 + d_3x^3 + d_4x^4 + d_5x^5 + d_6x^6)^{-16} + \epsilon(x) \\ |\epsilon(x)| < 1.5 \times 10^{-7} \end{aligned} \tag{7.12}$$

$$\begin{array}{lll} d_1 = 0.0498673470 & d_3 = 0.0032776263 & d_5 = 0.0000488906 \\ d_2 = 0.0211410061 & d_4 = 0.0000380036 & d_6 = 0.0000053830. \end{array}$$

We can find the cdf of a standard distribution with mean $a$ through a change of variables $x = x' - a$ (see Figure 7.5) for an example). As the approximation is only valid for positive $x$, we exploit the symmetry of the standard normal distribution and use $P^-(x) = P(-x)$ for $x < 0$ to approximate the normal distribution $N(a,1)$ and use this approximation in the mathematical program. This disjunct formulation will require the use of binary variables, yielding a MINLP.

To express the split formulation of Equation (7.12), we split the expression into denominators $\mathrm{divL}_s^+$ and $\mathrm{divU}_s^+$ for $x_{M,s} - a > 0$ and $\mathrm{divL}_s^-$ and $\mathrm{divU}_s^-$ for $x_{M,s} - a \leq 0$:

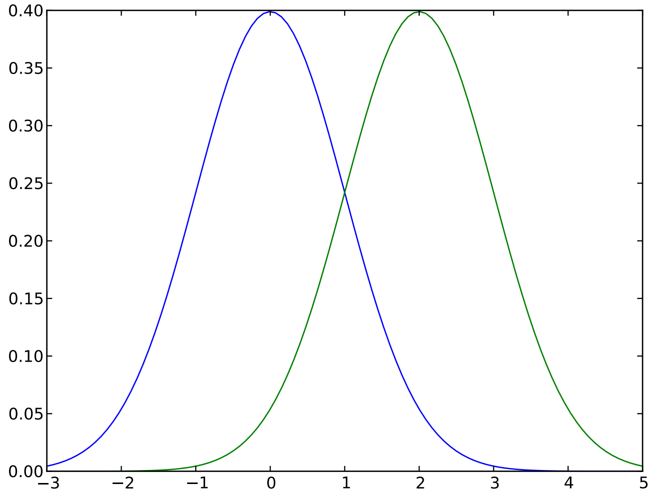



**Figure 7.5:** Examples of Normal distribution shifted mean by change of variables.

$$\begin{aligned}\mathrm{divL}_s^+ = (1 + d_1(x_{L,s} - a) + d_2(x_{L,s} - a)^2 + d_3(x_{L,s} - a)^3 \\ +d_4(x_{L,s} - a)^4 + d_5(x_{L,s} - a)^5 + d_6(x_{L,s} - a)^6)^{16}\end{aligned} \tag{7.13}$$

$$\begin{aligned}\mathrm{divU}_s^+ = (1 + d_1(x_{U,s} - a) + d_2(x_{U,s} - a)^2 + d_3(x_{U,s} - a)^3 \\ +d_4(x_{U,s} - a)^4 + d_5(x_{U,s} - a)^5 + d_6(x_{U,s} - a)^6)^{16}\end{aligned} \tag{7.14}$$

$$\begin{aligned}\mathrm{divL}_s^- = (1 + d_1(-x_{L,s} + a) + d_2(-x_{L,s} + a)^2 + d_3(-x_{L,s} + a)^3 \\ +d_4(-x_{L,s} + a)^4 + d_5(-x_{L,s} + a)^5 + d_6(-x_{L,s} + a)^6)^{16}\end{aligned} \tag{7.15}$$

$$\begin{aligned}\mathrm{divU}_s^- = (1 + d_1(-x_{U,s} + a) + d_2(-x_{U,s} + a)^2 + d_3(-x_{U,s} + a)^3 \\ +d_4(-x_{U,s} + a)^4 + d_5(-x_{U,s} + a)^5 + d_6(-x_{U,s} + a)^6)^{16}\end{aligned} \tag{7.16}$$

We combine the expressions above to express the resulting interval probabilities for $p_s^-$ and $lp_s^+$:

$$p_s^- \leq (1-\delta_s)\frac{1}{2}\left(\frac{1}{\mathrm{divL}_s^-} - \frac{1}{\mathrm{divU}_s^-}\right) \tag{7.17}$$

$$p_s^+ \leq \delta_s\frac{1}{2}\left(\frac{1}{\mathrm{divU}_s^+} - \frac{1}{\mathrm{divL}_s^+}\right) \tag{7.18}$$

For all discrete scenarios $s \in \mathcal{S}$ with corresponding possible realization of the variable $x_{M,s} \in [x_{L,s}, x_{U,s}]$ we use $x_{M,s}$ and binary indicator variables $\delta_s \in \{0,1\}, \forall s \in \mathcal{S}$ to indicate the location of the interval. This will give some inaccuracy for the interval spanning both definitions. For improved accuracy, separate indicator variables may be used for upper and lower interval values, doubling the number of binary variables.

Note that in order to calculate the cumulative probabilities correctly for the tail scenarios, we use extreme values for the end points $x_{L,1}$ and $x_{U,|S|}$.

$$p_s = p_s^- + p_s^+, \tag{7.19}$$

$$p_s^+ \leq 1 - \delta_s, \tag{7.20}$$

$$p_s^- \leq \delta_s. \tag{7.21}$$

Ensure appropriate $\delta_s$ is set to 1 with big M constraints using constants $M^+$ and $M^-$:

$$x_{M,s} - a \leq (1-\delta_s)M^+, \tag{7.22}$$

$$x_{M,s} - a \geq \delta_s M^-. \tag{7.23}$$

Bound probabilities to 1:

$$\sum_s p_s = 1, \sum_s p_s^+ \leq 1, \sum_s p_s^- \leq 1, \forall s \in \mathcal{S}. \tag{7.24}$$

Only allow one shift from negative to positive:

$$\delta_s \leq \delta_{s-1}, \forall s = 2 \dots S. \tag{7.25}$$

This model includes a complex polynomial expression as well as a binary variable resulting in a non-convex mixed integer non-linear formulation.

## 7.5 Test Instances and Example

We have implemented a few test instances to investigate how hard they will be to solve. All test models are implemented as GAMS models and can be downloaded from `http://iot.ntnu.no/users/hellemo/DDP/`. We have tried different data sets with different numbers of scenarios. The results of these experiments can be seen in Section 7.6. In our test case we look at capacity expansion of power generation. The investor seeks to minimize the cost of meeting a given demand. Either unit cost or demand is stochastic. In addition to the available production technologies, we assume it to be possible to invest in an activity or technology that will alter the probabilities of the discrete scenarios occurring. By investing in such a technology or activity, it is possible to alter the probability distribution as discussed in the subsections of Section 7.4.

### Test instances

The mathematical formulations of each test model follow here. We first present the base model in Section 7.5, in the following sections the deviations from the base model are presented in accordance with the models discussed above. These modifications mostly concern the objective function.

#### Base Model

| | |
|---|---|
| $B$ | Total investment budget, |
| $\mathcal{P}$ | set of baseline probability distributions (index $g$), |
| $\mathcal{I}$ | set of available technologies (index $i$), |
| $\mathcal{J}$ | set of modes of electricity demand (index $j$), |
| $\mathcal{S}$ | set of scenarios (index $s$), |
| $p_{gs}$ | baseline probability of scenario $s$ for probability distribution $g$, |
| $\pi_{js}$ | price of electricity in mode $j$ in scenario $s$, |
| $x_i$ | new capacity of $i$, decided in first stage, |
| $c_i$ | unit investment cost of $i$, |
| $c_g$ | unit investment cost of increasing weight to probabilities $g$, |
| $d_{js}$ | electricity demand in mode $j$ in scenario $s$ (if stochastic), |
| $q_{is}$ | unit production cost of $i$ in scenario $s$ (if stochastic) |
| $y_g$ | weight assigned to group $g$, |
| $z_{ijs}$ | production rate from $i$ for mode $j$ in scenario $s$, |
| $\overline{X_i}$ | Upper bound on $x_i$, |
| $\underline{X_i}$ | Lower bound on $x_i$, |
| $\overline{Y_g}$ | Upper bound on $y_g$, |
| $\underline{Y_g}$ | Lower bound on $y_g$, |

$\overline{Z_{ij}}$ Upper bound on $z_{ij}$,
$\underline{Z_{ij}}$ Lower bound on $z_{ij}$.

**Nomenclature**

$$\min \sum_{i \in \mathcal{I}} c_i x_i + \sum_{g \in \mathcal{P}} c_g y_g + \sum_{s \in \mathcal{S}} \sum_{g \in \mathcal{P}} p_{gs} y_g \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (q_{is} - \pi_{js}) z_{ijs}, \tag{7.26}$$

subject to:

$$\sum_{i \in \mathcal{I}} z_{ijs} = d_{js} \qquad \forall s \in \mathcal{S}, j \in \mathcal{J}, \tag{7.27}$$

$$\sum_{j \in \mathcal{J}} z_{ijs} \leq x_i \qquad \forall s \in \mathcal{S}, i \in \mathcal{I}, \tag{7.28}$$

$$\sum_{i \in \mathcal{I}} c_i x_i + \sum_{g \in \mathcal{P}} c_g y_g \leq B, \tag{7.29}$$

$$\sum_{g \in \mathcal{P}} y_g = 1, \tag{7.30}$$

$$\underline{X_i} \leq x_i \leq \overline{X_i}, \forall i \in \mathcal{I}, \tag{7.31}$$

$$\underline{Y_g} \leq y_g \leq \overline{Y_g}, \forall g \in \mathcal{P}, \tag{7.32}$$

$$\underline{Z_{ij}} \leq z_{ijs} \leq \overline{Z_{ij}}, \forall i \in \mathcal{I}, j \in \mathcal{J}. \tag{7.33}$$

**Model description** This model takes inspiration from the model of Louveaux & Smeers [1988], an investment problem from the electricity sector. There are $I$ technologies available to invest in. The demand for electricity in $J$ modes is given by the parameter $d_{js}$. The model is formulated as a two stage stochastic recourse model.

New capacity of technology $i$ is decided upon and installed in the first stage, determined by variables $x_i$. The total capacity available in stage two is limited by the investments in the first stage. A technology $i$ is characterized by the following parameters: The costs are determined by $c_i$ for the unit investment cost of type $i$. The unit production cost is $q_{is}$. The demand for electricity in each mode $j$ is given by parameter $d_{js}$ and the production rate using technology $i$ to serve demand in mode $j$ is $z_{ijs}$. For the convex combination of scenario probabilities, the scenario probabilities from each distribution are weighted by $y_g$, and the associated cost is given by parameter $c_g$.

The objective is to minimize the expected total cost while meeting demand. The objective function is given in Equation (7.26), investment costs $c_i x_i$ and production costs $q_{is}$ in scenarios occurring with weighted probabilities $p_{gs} y_g$.

Total production $\sum_{i \in \mathcal{I}} z_{ijs}$ needs to satisfy demand $d_{js}$ for each mode $j \in \mathcal{J}$ and scenario $s \in \mathcal{S}$ (Equation (7.27)). The total capacity investment for a given technology in the first time period equals the investment in that period $x_i$ and total production for each technology must be lower than available capacity for that technology (Equation (7.28)).

To enforce relatively complete recourse, Louveaux and Smeers Louveaux & Smeers [1988] make sure there is a technology $i_{\text{rcr}} \in \mathcal{I}$ with high production cost which simulates purchases in the market to balance supply $\sum\limits_{i \in \mathcal{I}} z_{ijs}$ and demand $d_{js}$. The sum of the weights to all probability distributions $\sum_{g \in \mathcal{P}} y_g$ must equal 1 (Equation (7.30)).

All variables are bounded, Equations (7.31) to (7.33).

#### Scalable Subsets of Scenarios

Here we adjust the probability for a subset of scenarios where each scenario in the subset has equal probability. We adjust the probability of the remaining scenarios proportionally in the opposite direction. By adjusting a fixed proportion of scenarios, we maintain the same structure independent of the number of scenarios chosen. The practical interpretation is that by investing in a technology or activity, it is possible to increase the probability of some scenarios, while reducing the probability of the remaining scenarios, or vice versa.

Replace Equation (7.26) with Equation (7.34):

$$\begin{aligned} \min \sum_{i \in \mathcal{I}} c_i x_i + \sum_{g \in \mathcal{P}} c_g y_g + \sum_{g \in \mathcal{P}} \sum_{s \in \hat{\mathcal{S}}} \frac{1}{|\mathcal{S}|} y_g \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (q_{is} - \pi_{js}) z_{ijs} + \\ \sum_{s \in \mathcal{S} \setminus \hat{\mathcal{S}}} \frac{1 - \sum_{g \in \mathcal{P}} y_g \sum_{s \in \hat{\mathcal{S}}} \frac{1}{|\mathcal{S}|}}{|\mathcal{S}| - |\hat{\mathcal{S}}|} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (q_{is} - \pi_{js}) z_{ijs}. \end{aligned} \tag{7.34}$$

#### Convex Combination of Probabilities

Here we apply the mixture distribution, modeling the possibility to change the weights of the underlying probability distributions for the subsets of outcomes. This can be used to model heterogeneous populations where the relative size of each subpopulation $g \in \mathcal{P}$ can be influenced by a decision variable $y_g$, determining the relative probability $p_{gs}$ for each scenario $s \in \mathcal{S}$.

Replace Equation (7.26) with Equation (7.35):

$$\min \sum_{i\in\mathcal{I}} c_i x_i + \sum_{g\in\mathcal{P}} c_g y_g +$$
$$\sum_{s\in\mathcal{S}} \left\{ \sum_{g=1}^{G-1} p_{gs} y_g \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} (q_{is} - \pi_{js}) z_{ijs} + \left[1 - \sum_{g=1}^{G-1} y_g\right] p_{Gs} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} (q_{is} - \pi_{js}) z_{ijs} \right\}. \tag{7.35}$$

#### Kumaraswamy

In this formulation the decision maker can change directly parameters $a$ and $b$ in the distribution, possibly at a cost. For this specific problem, it can for example be interpreted as changing the characteristics of the cost uncertainty. See Figure 7.6 for an example of scenario probabilities with parameters chosen in the example model. Replace the expression for $p_s$ in Equation (7.26) with Equation (7.36):

$$\min \sum_{i\in\mathcal{I}} c_i x_i + c_a a + c_b b + \sum_{s\in\mathcal{S}} p_s(a,b) \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} (q_{is} - \pi_{js}) z_{ijs}$$
$$= \sum_{i\in\mathcal{I}} c_i x_i + c_a a + c_b b + \sum_{s\in\mathcal{S}} \left[(1 - x_{L,s}^a)^b - (1 - x_{U,s}^a)^b\right] \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} (q_{is} - \pi_{js}) z_{ijs} \tag{7.36}$$

Also replace the budget constraint Equation (7.29) with Equation (7.40):

$$\sum_{i\in\mathcal{I}} c_i x_i + c_a a + c_b b \le B \tag{7.37}$$

Parameters $a$ and $b$ should be positive:

$$a, b > 0. \tag{7.38}$$

#### Approximation of Normal Distribution

We can find the cdf of a standard distribution with mean $a$ through a change of variables $x = x' - a$ and using $P^-(x) = P(-x)$ for $x < 0$, we can approximate the normal distribution $N(a, 1)$ and use this approximation in the mathematical

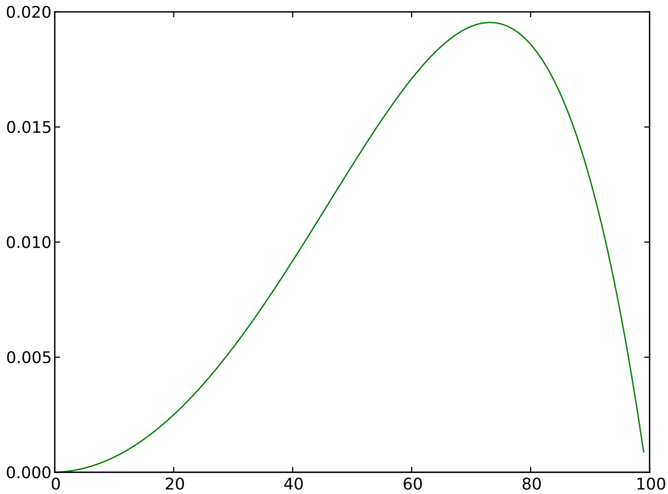



**Figure 7.6:** Probabilities from solution of GAMS implementation of model using 100 scenarios and Kumaraswamy distribution

program. See Figure 7.7 for an example of resulting probabilities in the test model.

Replace the objective function given in Equation (7.26) with Equation (7.39) and use the discrete scenarios $s \in \mathcal{S}$ with corresponding possible realization of the variable $x_{M,s} \in [x_{L,s}, x_{U,s}]$:

$$\min \sum_{i \in \mathcal{I}} c_i x_i + c_a a + \sum_{s \in \mathcal{S}} p_s \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (q_{is} - \pi_{js}) z_{ijs} \tag{7.39}$$

where $p_s$ follows the definition from Section 7.4 and is defined by Equation (7.13) to Equation (7.25).

Also replace the budget constraint Equation (7.29) with Equation (7.40):

$$\sum_{i \in \mathcal{I}} c_i x_i + c_a a \leq B \tag{7.40}$$

Parameter $a$ should be positive:

$$a > 0. \tag{7.41}$$

## Example of the Effects of DDP

To illustrate the effects of decision dependent probabilities in our models, we will look at the results from one particular test instance with the approximation of the Normal distribution from Section 7.5. This instance has stochastic demand, and we assume the demand can be increased by engaging in some activity, for example by investing in campaigning, improving the safety or by reducing emissions from production if the demand is sensitive to these parameters. In the model, demand is influenced by shifting the mean of the probability distribution by $a$.

In this instance the mean may be shifted by $a \in [-1.0, 0]$. We shift the mean in the opposite direction from the original model, hence $a \leq 0$. The uncertain parameters are discretized with 10 scenarios. The outcomes for the stochastic parameters are fixed for each scenario, while the probabilities for each scenario occurring are determined by selecting the mean of the distribution. The investment decisions are whether to invest in any of the 10 available technologies $x_i, i \in 1, 2, ..., 10$.

The results are summarized in Figure 7.8. In the figure we show the optimal expected profit for different values of $a$ in the upper pane, while the corresponding investment levels of technologies $x_8$, $x_9$ and $x_{10}$ are shown in the lower pane. Expected profit increases with more negative $a$ (increasing demand), and so does investment in the different technologies. As demand shifts it becomes profitable to invest in more technologies, also the ones with higher operating costs as the

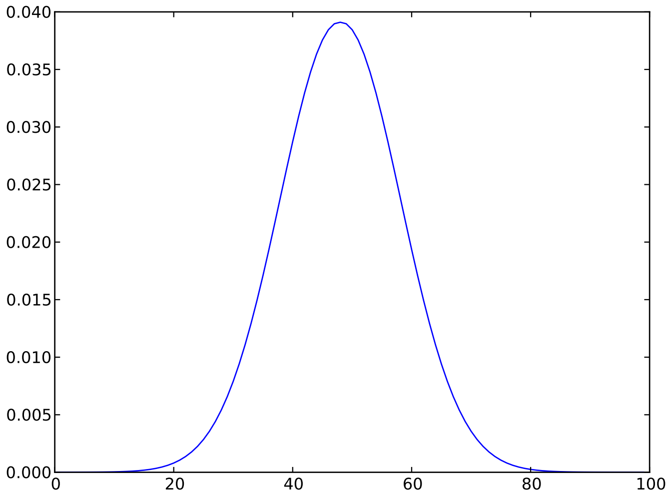



**Figure 7.7:** Probabilities from optimal solution of GAMS implementation of model using 100 scenarios and approximation of normal distribution

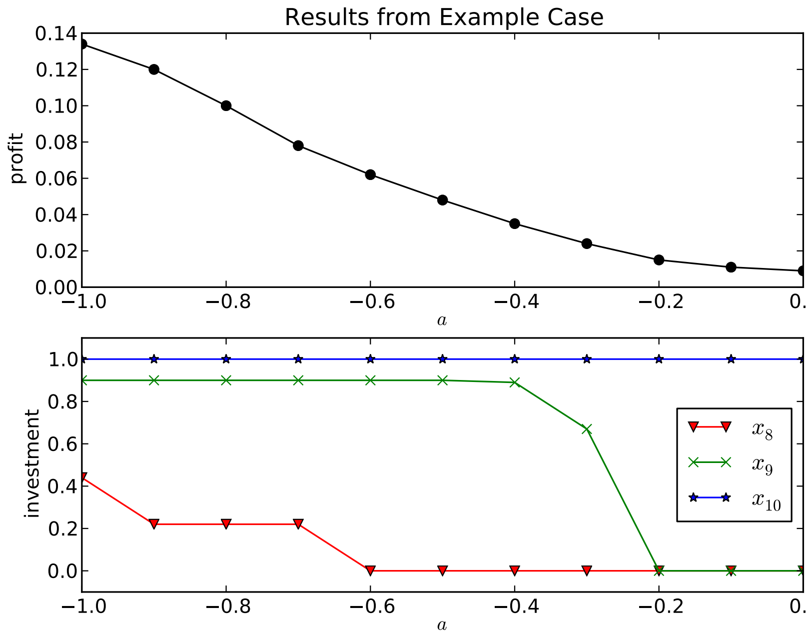



**Figure 7.8:** Results from example model. Investments and profit increase with more negative $a$.

maximum investment level is reached for technologies with lower operating cost. See also Table 7.1 for details.

This simple example shows how the inclusion of decision-dependent probabilities changes the problem. Note that for fixed $a$, the resulting problem is a traditional stochastic program with recourse. While finding the optimal solution of the problem with DDP is easy to do by inspection for this simple example this is of course in general not a practical solution approach for such non-convex models where decision-dependencies are linked to several variables.

The test instances for which we provide computational results in the next section, are all based on synthetic data. We provide the aggregated results from a series of test instances to illustrate the computational difficulty of this class of problems.

**Table 7.1:** Summarized results from one test instance with different values of a, the objective function value, and values for investment in technologies $x_8$, $x_9$ and $x_{10}$.

| $a$ | PROFIT | $x_8$ | $x_9$ | $x_{10}$ |
|---|---|---|---|---|
| 0 | 0.009 | 0 | 0 | 1 |
| -0.1 | 0.011 | 0 | 0 | 1 |
| -0.2 | 0.015 | 0 | 0 | 1 |
| -0.3 | 0.024 | 0 | 0.67 | 1 |
| -0.4 | 0.035 | 0 | 0.89 | 1 |
| -0.5 | 0.048 | 0 | 0.9 | 1 |
| -0.6 | 0.062 | 0 | 0.9 | 1 |
| -0.7 | 0.078 | 0.22 | 0.9 | 1 |
| -0.8 | 0.1 | 0.22 | 0.9 | 1 |
| -0.9 | 0.12 | 0.22 | 0.9 | 1 |
| -1.0 | 0.134 | 0.44 | 0.9 | 1 |

## 7.6 Computational Results

In this section we present the computational results from all four variations of the base model. The models are all implemented in GAMS. We first present our solution strategy, followed by a summary of the computational results.

### Solution Strategy

All of the formulations presented above introduce many non-linear terms in the corresponding optimization problems. In general, this gives a non-convex bilinear program. Many of the potential applications of such models involve investment decisions. Fixed investment costs often necessitate the use of discrete variables. Hence, the models where these modeling techniques should be applied will often already have integer variables, yielding a deterministic equivalent that is a mixed integer non-linear (non-convex) model.

Global optimization techniques must be applied to guarantee an optimal solution. BARON is the state-of-the-art global optimization solver, using convex relaxations for non-convex terms. A widely applied technique is to use McCormick relaxations to construct convex relaxations of factorable functions. BARON also applies techniques for constraint propagation to reduce the search space Tawarmalani & Sahinidis [2002]. We have tested BARON using three different approaches Baron1: the problem were fed into BARON without information about structure and bilinear expressions expanded; Baron2: the same as the previous but using a selective branching strategy on the complicating variables motivated by Epperly & Pistikopoulos [1997]; Baron3: The problem was fed into BARON using the original un-expanded bilinear expressions in GAMS and solved directly. In addition we tested the instances using an approach combining relaxations of algorithms (Mitsos et al. [2009]) and generalized Benders decomposition (Benders [1962], Geoffrion [1972]) that we implemented for the purpose (GGBD).

We observed initially what appeared to be good results decomposing these stochastic programs based on GGBD and comparing it to the Baron1 approach. The Baron3 approach with a GAMS implementation of the same model showed much better performance, though. We tested Baron2 using a selective branching strategy inferred from the problem structure, and achieved the same behavior as Baron3. Our conclusion is that solution times for these problems can be dramatically improved by using a selective branching strategy on the complicating variables controlling the scenario probabilities. Selective branching can be readily implemented through setting branching priority in BARON. Interestingly, using the original, un-expanded formulation achieved similar results to the selective branching strategy. Note that when fixing the variables that influence the probabilities of the scenarios, the resulting sub problems are much easier to solve. For the affine formulations given in Section 7.4, the remaining problem is a standard linear or mixed integer stochastic program.

## Solution Times for Test Instances

**Table 7.2:** Full results from benchmarks. Model, Number of scenarios, Solver Status, Cputime in seconds, Number of nonlinear terms, number of discrete terms, rows and columns, and lower and upper bound after stop criterion.

| Model | #Scen. | Status | CPU | #NL | #D | Rows | Cols | LB | UB |
|---|---|---|---|---|---|---|---|---|---|
| Subsets | 10 | Unfinished | 3,600 | 602 | 0 | 1,013 | 203 | $-3.2 \cdot 10^{-1}$ | $-3.2 \cdot 10^{-1}$ |
| Combination | 10 | Optimal | 0.5 | 901 | 0 | 1,013 | 213 | $-6.8 \cdot 10^{-1}$ | $-6.8 \cdot 10^{-1}$ |
| Kumaraswamy | 10 | Unfinished | 3,600 | 1,020 | 0 | 1,034 | 232 | $-5.9 \cdot 10^{-1}$ | $-5.9 \cdot 10^{-1}$ |
| Rational | 10 | Optimal | 4.27 | 1,110 | 10 | 1,092 | 334 | $-5.2 \cdot 10^{-1}$ | $-5.2 \cdot 10^{-1}$ |
| Subsets | 20 | Optimal | 2.25 | 1,302 | 0 | 2,013 | 403 | $-3.4 \cdot 10^{-1}$ | $-3.4 \cdot 10^{-1}$ |
| Combination | 20 | Optimal | 1.48 | 1,901 | 0 | 2,013 | 423 | $-6.6 \cdot 10^{-1}$ | $-6.6 \cdot 10^{-1}$ |
| Kumaraswamy | 20 | Unfinished | 3,600.01 | 2,040 | 0 | 2,054 | 462 | $-5.9 \cdot 10^{-1}$ | $-5.9 \cdot 10^{-1}$ |
| Rational | 20 | Unfinished | 3,601.02 | 2,220 | 20 | 2,172 | 664 | $-5.9 \cdot 10^{-1}$ | $-5.2 \cdot 10^{-1}$ |
| Subsets | 30 | Unfinished | 3,600 | 1,902 | 0 | 3,013 | 603 | $-3.4 \cdot 10^{-1}$ | $-3.4 \cdot 10^{-1}$ |
| Combination | 30 | Optimal | 2.98 | 2,901 | 0 | 3,013 | 633 | $-6.6 \cdot 10^{-1}$ | $-6.6 \cdot 10^{-1}$ |
| Kumaraswamy | 30 | Unfinished | 3,600.02 | 3,060 | 0 | 3,074 | 692 | $-5.9 \cdot 10^{-1}$ | $-5.9 \cdot 10^{-1}$ |
| Rational | 30 | Unfinished | 3,600.02 | 3,330 | 30 | 3,252 | 994 | $-6.8 \cdot 10^{-1}$ | $-5.2 \cdot 10^{-1}$ |
| Subsets | 50 | Unfinished | 3,600.04 | 3,302 | 0 | 5,013 | 1,003 | $-3.5 \cdot 10^{-1}$ | $-3.5 \cdot 10^{-1}$ |
| Combination | 50 | Optimal | 9.08 | 4,901 | 0 | 5,013 | 1,053 | $-6.5 \cdot 10^{-1}$ | $-6.5 \cdot 10^{-1}$ |
| Kumaraswamy | 50 | Unfinished | 3,600.03 | 5,100 | 0 | 5,114 | 1,152 | $-5.9 \cdot 10^{-1}$ | $-5.9 \cdot 10^{-1}$ |
| Rational | 50 | Optimal | 93.21 | 5,550 | 50 | 5,412 | 1,654 | $-5.2 \cdot 10^{-1}$ | $-5.2 \cdot 10^{-1}$ |
| Subsets | 100 | Optimal | 66.59 | 6,602 | 0 | 10,013 | 2,003 | $-3.5 \cdot 10^{-1}$ | $-3.5 \cdot 10^{-1}$ |
| Combination | 100 | Optimal | 45.27 | 9,801 | 0 | 10,013 | 2,103 | $-6.5 \cdot 10^{-1}$ | $-6.5 \cdot 10^{-1}$ |
| Kumaraswamy | 100 | Unfinished | 3,600 | 10,200 | 0 | 10,214 | 2,302 | $-5.3$ | $-5.9 \cdot 10^{-1}$ |
| Rational | 100 | Optimal | 429 | 11,100 | 100 | 10,812 | 3,304 | $-5 \cdot 10^{-1}$ | $-5 \cdot 10^{-1}$ |

In Table 6.10 we present results from running our test instances. Each test instance is run with different numbers of scenarios. The resulting problem sizes, both in terms of number of rows, columns and number of discrete and non-linear variables are all reported in the table. All problems were run with a time limit of one hour, and most test instances were close to optimal after one hour, although not as close as the stopping criterion of a relative gap less than $1 \times 10^{-5}$ . All numbers presented are from Baron2 (Baron3 gave similar results).

Our numerical experiments show that BARON is generally able to solve the instances of the Convex combination of probabilities from Section 7.5 as well as the instances using the approximation of the normal distribution from Section 7.5. The scalable subsets of scenarios from Section 7.5 and the Kumaraswamy models from Section 7.5 proved harder to solve, and while the solver has found a good solution, optimality remains to be proved within the time limit. BARON is able to solve relatively large problems in reasonable time if the problem formulation provides enough structure for the solver to choose an efficient solution strategy. In cases where we provided an unstructured problem without a selective branching strategy, BARON would often end up doing a lot of unnecessary branching, which made convergence very slow and in general slower than our GGBD (Results not included).

For larger problems in the harder categories, specialized solution techniques may be necessary, and we hope that our test instances may come of use in future research in this area.

## 7.7 Conclusions and Further Work

Little work has been done on stochastic programming problems with decision-dependent probabilities. We extend previous taxonomies of stochastic programming problems with decision-dependent uncertainty, and present some examples of models with decision-dependent probabilities. We show how direct or indirect manipulation of probability distribution can be incorporated in stochastic programs with recourse. We demonstrate that such problems may be solved by the commercial solver BARON, using selective branching in the complicating variables. For the test instances we considered, choosing a selective branching strategy for the scenario probability variables proved much more efficient than the decomposition method we implemented and tested. We provide a set of test cases for this class of problems.

We only considered linear dependency between cost and a change on the underlying probability distribution. An extension would be to introduce some non-linear cost such as diminishing return to scale.

Our test cases were based on a risk neutral approach. Investigating the effects of different risk attitudes on decision-dependent probabilities is another area of research that would be very interesting to pursue.

Finally, as these large scale non-convex problems grow more complex, finding good and robust decomposition techniques would greatly improve the scale at which such techniques could be applied. We hope that the test problems we provide can be a starting point for further research on solution methods for stochastic programming problems with decision-dependent probabilities.

## Acknowledgements

This research was supported by The Norwegian Research Council, project number 176089.

## Bibliography

Abramowitz, M., & Stegun, I. (1964). *Handbook of mathematical functions with formulas, graphs, and mathematical tables* volume 55. New York: Dover publications.

Ahmed, S. (2000). *Strategic Planning under Uncertainty – Stochastic Integer Programming Approaches*. Ph.D. thesis Graduate College, University of Illinois at Urbana-Champaign Urbana, IL, USA.

Artstein, Z., & Wets, R. (1994). Stability results for stochastic programs and sensors, allowing for discontinuous objective functions. *SIAM Journal on Optimization*, *4*, 537.

Beale, E. M. L. (1955). On minimizing a convex function subject to linear inequalities. *Journal of the Royal Statistical Society, Series B*, *17*, 173–184.

Behboodian, J. (1970). On the modes of a mixture of two normal distributions. *Technometrics*, *12*, 131–139.

Ben-Tal, A., Eiger, G., & Gershovitz, V. (1994). Global minimization by reducing the duality gap. *Mathematical Programming*, *63*, 193–212.

Ben-Tal, A., & Nemirovski, A. (1998). Robust convex optimization. *Mathematics of Operations Research*, *23*, 769–805.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, *4*, 238–252.

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming*, *98*, 49–71.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations Research*, *52*, 35–53.

Boland, N., Dumitrescu, I., & Froyland, G. (2008). A multistage stochastic programming approach to open pit mine production scheduling with uncertain geology. In *7th joint Australia-New Zealand Mathematics Convention (ANZMC2008)*. Christchurch, New Zealand.

Colvin, M., & Maravelias, C. (2009). A Branch and Cut Framework for Multi-Stage Stochastic Programming Problems Under Endogenous Uncertainty. *Computer Aided Chemical Engineering*, *27*, 255–260.

Dantzig, G. B. (1955). Linear programming under uncertainty. *Management Science*, *1*, 197–206.

Dupačová, J. (2006). Optimization under exogenous and endogenous uncertainty. In L. Lukáš (Ed.), *Proc. of MME06* (pp. 131 – 136). University of West Bohemia in Pilsen.

Epperly, T. G., & Pistikopoulos, E. N. (1997). A reduced space branch and bound algorithm for global optimization. *Journal of Global Optimization*, *11*, 287–311.

Escudero, L. F., Garín, M. A., Merino, M., & Pérez, G. (2014). On multistage mixed 0-1 optimization under a mixture of Exogenous and Endogenous Uncertainty in a risk averse environment. *Working Paper*, .

Feller, W. (1943). On a general class of "contagious" distributions. *The Annals of Mathematical Statistics*, *14*, 389–400.

Frühwirth-Schnatter, S. (2006). *Finite mixture and Markov switching models*. New York: Springer.

Geoffrion, A. (1972). Generalized Benders Decomposition. *Journal of Optimization Theory and Applications*, *10*, 237–260.

Goel, V., & Grossmann, I. (2004). A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. *Computers and Chemical Engineering*, *28*, 1409–1429.

Goel, V., & Grossmann, I. (2006). A class of stochastic programs with decision dependent uncertainty. *Mathematical Programming*, *108*, 355–394.

Gupta, V., & Grossmann, I. E. (2011). Solution strategies for multistage stochastic programming with endogenous uncertainties. *Computers & Chemical Engineering*, *35*, 2235–2247.

Held, H., & Woodruff, D. (2005). Heuristics for multi-stage interdiction of stochastic networks. *Journal of Heuristics*, *11*, 483–500.

Jonsbråten, T. (1998). Oil field optimization under price uncertainty. *Journal of the Operational Research Society*, *49*, 811–818.

Jonsbråten, T., Wets, R., & Woodruff, D. (1998). A class of stochastic programs with decision dependent random elements. *Annals of Operations Research*, *82*, 83–106.

Kuhn, D. (2009). An information-based approximation scheme for stochastic optimization problems in continuous time. *Mathematics of Operations Research*, *34*, 428–444.

Kuhn, D., Wiesemann, W., & Georghiou, A. (2011). Primal and dual linear decision rules in stochastic and robust optimization. *Mathematical Programming*, *130*, 177–209.

Kumaraswamy, P. (1980). A generalized probability density function for double-bounded random processes. *Journal of Hydrology*, *46*, 79–88.

Louveaux, F. V., & Smeers, Y. (1988). Optimal investments for electricity generation: A stochastic model and a test problem. In Y. Ermoliev, & R. J.-B. Wets (Eds.), *Numerical Techniques for Stochastic Optimization* (pp. 445–454). Berlin: Springer-Verlag.

Mitsos, A., Chachuat, B., & Barton, P. (2009). McCormick-based relaxations of algorithms. *SIAM Journal on Optimization*, *20*, 573–601.

Ntaimo, L., Arrubla, J. A. G., Stripling, C., Young, J., & Spencer, T. (2012). A stochastic programming standard response model for wildfire initial attack planning. *Canadian Journal of Forest Research*, *42*, 987–1001.

Pflug, G. (1996). *Optimization of stochastic models: the interface between simulation and optimization*. Boston: Kluwer Academic.

Pflug, G., & Wozabal, D. (2007). Ambiguity in portfolio selection. *Quantitative Finance*, *7*, 435–442.

Pflug, G. C., & Pichler, A. (2011). Approximations for probability distributions and stochastic optimization problems. In *Stochastic Optimization Methods in Finance and Energy* (pp. 343–387). New York: Springer.

Rubinstein, R. Y., & Shapiro, A. (1993). *Discrete event systems: Sensitivity analysis and stochastic optimization by the score function method* volume 346. New York: Wiley.

Solak, S. (2007). *Efficient solution procedures for multistage stochastic formulations of two problem classes*. Ph.D. thesis Georgia Institute of Technology Atlanta.

Tarhan, B., Grossmann, I., & 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*, 3078–3097.

Tawarmalani, M., & Sahinidis, N. V. (2002). *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming*. Norwell, MA, USA: Kluwer Academic Publishers.

Varaiya, P., & Wets, R. J.-B. (1989). Stochastic dynamic optimization, approaches and computation. In *Mathematical Programming, Recent Developments and Applications, M. Iri* (pp. 309–332). Boston: Kluwer Academic Publisher.

Viswanath, K., Peeta, S., & Salman, S. F. (2004). *Investing in the Links of a Stochastic Network to Minimize Expected Shortest Path Length*. Technical Report Purdue University, Department of Economics West Lafayette.

## 7.A Hardware and Software Used

All computations were performed on a Six-Core AMD Opteron processor 2431 with 24Gb memory. The computer was running Linux 2.6.18 (Rocks 5.3).

GAMS versions 23.6.2 and 23.7.2 with BARON using CPLEX in combination with CONOPT or MINOS.