Robust Simulation and Optimization Methods for Natural Gas Liquefaction Processes

by

Harry Alexander James Watson

M.S.C.E.P., Massachusetts Institute of Technology (2014)
B.E., Vanderbilt University (2012)

Submitted to the Department of Chemical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Chemical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2018

Author ..... Signature redacted
Department of Chemical Engineering
November 16, 2017

Certified by ..... Signature redacted
Paul I. Barton
Lammot du Pont Professor of Chemical Engineering
Thesis Supervisor

Accepted by ..... Signature redacted
Patrick S. Doyle
Chairman, Department Committee on Graduate Theses

Robust Simulation and Optimization Methods for Natural Gas Liquefaction Processes

by

Harry Alexander James Watson

Submitted to the Department of Chemical Engineering
on November 16, 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Chemical Engineering

Abstract

Natural gas is one of the world's leading sources of fuel in terms of both global production and consumption. The abundance of reserves that may be developed at relatively low cost, paired with escalating societal and regulatory pressures to harness low carbon fuels, situates natural gas in a position of growing importance to the global energy landscape. However, the nonuniform distribution of readily-developable natural gas sources around the world necessitates the existence of an international gas market that can serve those regions without reasonable access to reserves. International transmission of natural gas via pipeline is generally cost-prohibitive beyond around two thousand miles, and so suppliers instead turn to the production of liquefied natural gas (LNG) to yield a tradable commodity. While the production of LNG is by no means a new technology, it has not occupied a dominant role in the gas trade to date. However, significant growth in LNG exports has been observed within the last few years, and this trend is expected to continue as major new liquefaction operations have and continue to become operational worldwide.

Liquefaction of natural gas is an energy-intensive process requiring specialized cryogenic equipment, and is therefore expensive both in terms of operating and capital costs. However, optimization of liquefaction processes is greatly complicated by the inherently complex thermodynamic behavior of process streams that simultaneously change phase and exchange heat at closely-matched cryogenic temperatures. The determination of optimal conditions for a given process will also generally be nontransferable information between LNG plants, as both the specifics of design (e.g. heat exchanger size and configuration) and the operation (e.g. source gas composition) may have significantly variability between sites. Rigorous evaluation of process concepts for new production facilities is also challenging to perform, as economic objectives must be optimized in the presence of constraints involving equipment size and safety precautions even in the initial design phase. The absence of reliable and versatile software to perform such tasks was the impetus for this thesis project.

To address these challenging problems, the aim of this thesis was to develop new models, methods and algorithms for robust liquefaction process simulation and opti-

mization, and to synthesize these advances into reliable and versatile software. Recent advances in the sensitivity analysis of nondifferentiable functions provided an advantageous foundation for the development of physically-informed yet compact process models that could be embedded in established simulation and optimization algorithms with strong convergence properties. Within this framework, a nonsmooth model for the core unit operation in all industrially-relevant liquefaction processes, the multistream heat exchanger, was first formulated. The initial multistream heat exchanger model was then augmented to detect and handle internal phase transitions, and an extension of a classic vapor-liquid equilibrium model was proposed to account for the potential existence of solutions in single-phase regimes, all through the use of additional nonsmooth equations.

While these initial advances enabled the simulation of liquefaction processes under the conditions of simple, idealized thermodynamic models, it became apparent that these methods would be unable to handle calculations involving nonideal thermophysical property models reliably. To this end, robust nonsmooth extensions of the celebrated inside-out algorithms were developed. These algorithms allow for challenging phase equilibrium calculations to be performed successfully even in the absence of knowledge about the phase regime of the solution, as is the case when model parameters are chosen by a simulation or optimization algorithm. However, this still was not enough to equip realistic liquefaction process models with a completely reliable thermodynamics package, and so new nonsmooth algorithms were designed for the reasonable extrapolation of density from an equation of state under conditions where a given phase does not exist. This procedure greatly enhanced the ability of the nonsmooth inside-out algorithms to converge to physical solutions for mixtures at very high temperature and pressure.

These models and submodels were then integrated into a flowsheeting framework to perform realistic simulations of natural gas liquefaction processes robustly, efficiently and with extremely high accuracy. A reliable optimization strategy using an interior-point method and the nonsmooth process models was then developed for complex problem formulations that rigorously minimize thermodynamic irreversibilities. This approach significantly outperforms other strategies proposed in the literature or implemented in commercial software in terms of the ease of initialization, convergence rate and quality of solutions found. The performance observed and results obtained suggest that modeling and optimizing such processes using nondifferentiable models and appropriate sensitivity analysis techniques is a promising new approach to these challenging problems. Indeed, while liquefaction processes motivated this thesis, the majority of the methods described herein are applicable in general to processes with complex thermodynamic or heat transfer considerations embedded. It is conceivable that these models and algorithms could therefore inform a new, robust generation of process simulation and optimization software.

Thesis Supervisor: Paul I. Barton
Title: Lammot du Pont Professor of Chemical Engineering

Acknowledgments

It still seems somewhat unreal that my time at MIT is actually coming to a close, and on reflection, what a long, strange trip it has been! There have been so many times over the past five years that I found myself doing something I never thought I would do: living for months in both Norway and Ireland, spending time in about a dozen other countries, skiing my first mountain, running my first half-marathon; and then many things that I only hoped I might experience: making lifelong friendships, finding an incredible partner and, of course, actually finishing my PhD thesis. Little, if any, of this could have happened without the immense support and encouragement I've received from so many people along the way. I hope I can convey my gratitude for all your efforts with these acknowledgments; though I know that words alone will certainly fall short as reciprocity.

I begin, of course, with expressing my sincerest gratitude to my thesis advisor, Paul Barton. Paul was an exemplary advisor, capable of educating me more efficiently and efficaciously than any other single person in my academic career. Looking back, I can see how much I've grown as a researcher and engineer under his guidance as the inevitable result of many years of candid and incisive feedback, whether I wanted it or not. I also greatly appreciate the degree to which I was able to pursue those aspects of this project that most interested me and in which I felt I could make the greatest impact. I'm proud of what I've been able to achieve during my time at MIT, but I know I wouldn't have made a fraction of the progress that I did without Paul's support and supervision. Lastly, I must also thank Paul for the smaller, subtler lessons he instilled in me along the way, and as a result I move onwards to the next stage of my life knowing full well to always strive to set an example for others, to never split infinitives, and most importantly, to never, ever, use the Oxford comma.

I want to recognize my thesis committee, Professors William Green and John Brisson, for their valuable input throughout this thesis project. Their feedback certainly helped me to evaluate and reflect on my progress throughout and to really sharpen up the presentation of my work. I am also extremely grateful to Professor Truls Gundersen and his research group at NTNU. Trondheim became something of a home-away-from-home for me over these past few years, and Truls, my advisor-away-from-advisor, was always so welcoming, supportive and insightful. It has been a great pleasure and privilege to have worked with him throughout this project. Then there's Matias, whom I first met on a wintry June day in Trondheim, shortly after learning that I was to be one of his Master's thesis supervisors. Needless to say, my expectations were not high. Yet, as unlikely as it seemed, Matias proved to be an excellent student and, moreover, a true friend. I consider myself very lucky to have known him in both capacities these last few years.

I also attribute a great deal of my success at MIT to the Practice School, both for my experience as a student of the program and for the opportunity to act as a station director, and in particular, I acknowledge Professor Alan Hatton and Bob Hanlon. To Alan, I say thank you for the trust you placed in me to run a station of my own – the Practice School is such a unique, demanding and important experience and I hope you know that I will be a staunch advocate of it to anyone who cares to listen.

As for Bob, well, in keeping with one of the unofficial practice school mottos: "it's amazing what you can do in a month", it must be said that it was astounding how much impact Bob had on me in the course of just two months. His crash course in problem solving, technical communication and professional development was one the most important experiences I had while at MIT and I will certainly carry his lessons with me as I start my career. Lastly, to my own students – you were absolutely excellent to work with and I sincerely hope that at least some of the tutelage that I received made its way onwards to you.

I have met a number of incredible people within the chemical engineering department at MIT whom I am proud to call my friends. Firstly, I want to thank everyone whose time I overlapped with while in the lab. In particular I want to mention Jose, Garrett, Rohit, Kamil, Michael, Paul and Peter, who were so helpful to be around throughout my PhD project, both in terms of being there for discussing ideas and for making the lab such an enjoyable place to spend large amounts of time. To all my classmates whom I began this journey at MIT with, I thank you for all the great times you've shown me and the support you've offered, even (and especially!) right from the beginning during the first semester here. Within that group, I want to express tremendous gratitude towards Kevin, Nick and Isaac, both for being dear friends and also for motivating me to get on board with this new-fangled "diet-and-exercise" phenomenon – losing the better part of fifty pounds in a year drastically improved my life and I am forever grateful to them for setting me on the right track. I also want to thank another group that now consists mostly of distinguished alumni from our first-year class: Chad, Justin and Isaac, but also Abel. It's been a fantastic experience having such esteemed companions (and, again, also Abel) with whom to travel the world and consume its culture. Lastly, no acknowledgments section in an MIT thesis should be complete without a mention of one's D&D group, and mine has certainly earned that. Andrew, Carlos, Dan, Jimmy, Orpheus and Zsigi, you have been incredible to adventure with, even if we are now two years in and still as dysfunctional as ever. Thank you all for the endless hilarity and all the late nights with the best company anyone could ask for.

The deepest debts of gratitude are those I owe to my family and loved ones. Whitney, you are everything I've looked for in a partner and more, and you've made me so extraordinarily happy these past two and a half years. I can't wait to see where our future takes us. To my family overseas, I thank you all for your unending love and encouragement despite the many miles between us. And above all, to my mum and dad, I can hardly express just how thankful I am for every opportunity you have afforded me that has led me to this point. Absolutely none of this would have been possible without your ceaseless love and support, and I do my best to remember that every single day. I love you both beyond words and I hope I will continue to make you proud, wherever the future leads me.

Finally, I would like to acknowledge Statoil for providing funding and support for this research project.

Contents

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

This thesis develops and demonstrates the efficacy of a new paradigm for the simulation and optimization of natural gas liquefaction processes. Moreover, while liquefaction processes have been the motivation and primary examples for this body of work, the majority of the methods described in this thesis are in fact applicable to many other processes that have complex thermodynamic and heat transfer considerations embedded. In summation, this work represents the first concerted effort in modeling and optimizing chemical processes using nondifferentiable models in conjunction with exact sensitivity analysis techniques, and the results herein suggest that this is a viable and efficacious approach that could become the foundation for a new, robust generation of process flowsheeting software. The contributions of this thesis include a new nonsmooth model for multiphase multistream heat exchangers, robust nonsmooth algorithms for vapor-liquid equilibrium (VLE or "flash") calculations even at extreme conditions and a simulation and optimization framework that integrates these nonsmooth modeling elements in order to solve complex liquefaction process flowsheeting problems reliably. The majority of the material that appears in this thesis has been either submitted for publication or published in peer-reviewed journals. This introductory chapter elaborates upon the motivation, novelty, objective, scope, structure and contributions of this thesis.

1.1 Project motivation

Natural gas is the third-most consumed energy source in the world, ranking behind only oil and coal and far ahead of all remaining sources (nuclear, renewables, hydropower, etc.) combined.[18] While new technologies continue to prolong the age of oil's dominance as a fuel source, global production of coal has decreased substantially in recent years, due both to the increasing availability of natural gas and in response to societal and regulatory pressures to shift towards cleaner, lower carbon fuels. Natural gas, it therefore seems, is poised to be a major source of global energy for the foreseeable future. However, growth of the international gas trade highlights a key issue with the transportation of natural gas – namely, that it is in a gaseous state at ambient conditions. Liquefaction processes are therefore necessary to produce liquefied natural gas (LNG), so that large volumes of fuel can be delivered economically across long distances. Such processes require energy-intensive refrigeration over a wide temperature range, including at cryogenic temperatures. Naturally, this translates to processes with both high capital investment costs and significant operating costs. Optimization of both the design and operation of liquefaction processes is therefore necessary, and rigorous, accurate methods for such problems are highly coveted.

While traditional liquefaction plants, located onshore and with high capacity, are typically optimized with the goal of improving throughput or energy efficiency and thus improving profit margins, recent interest in new process concepts, especially for remote gas production and floating operations, mean that objectives and constraints related to compactness, environmental impact, safety, and flexibility may also factor into optimal design and operation problems. As such, a versatile and reliable framework for the optimization of these process concepts is needed. As an additional challenge, the composition of natural gas also varies quite dramatically between sources worldwide, such that the optimal conditions for a liquefaction process that is designed and optimized for LNG production in, say, northern Norway are likely to be suboptimal (or even entirely unusable) for LNG production in the continental United States. In addition, the quality of natural gas entering a liquefaction plant can potentially

change on a temporal basis, both in the long term as the act of gas production itself impacts the source over time, and in the short term due to disturbances or equipment train reassignment. In all cases, the need for optimization methods that can determine operating conditions to match specific process conditions is critical, and in the latter cases it is important that these methods are able to perform flexibly, reliably and efficiently.

A basic example of a liquefaction process to visualize throughout this exposition is the Polyrefrigerated Integrated Cycle Operations (PRICO) process[76] that is shown in Figure 1-1. In the PRICO process, pressurized, preconditioned natural gas feed at

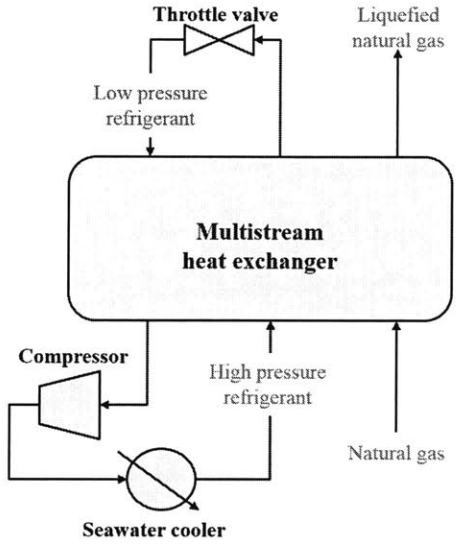


Figure 1-1: An example of a simple natural gas liquefaction process.

ambient temperature enters a multistream heat exchanger (MHEX), and exits as a subcooled liquid, the LNG product. In a process such as PRICO that, in the present day, is used for small-scale production, this heat exchanger is usually a plate-and-fin type exchanger, while larger scale and more complex processes will most often usef spiral-wound heat exchangers with strictly proprietary (and often customized) internal configurations.[48] After liquefaction, the LNG product may be re-expanded to a lower pressure as necessary for storage and transport. The necessary cooling is provided by a refrigerant mixture consisting of nitrogen and light hydrocarbons. This mixture is cooled and partially liquefied in a condenser that rejects heat to a large sink

such as seawater before entering the MHEX. In the exchanger, it is cooled to the same temperature as the LNG product, then exits to be expanded adiabatically through a throttle valve to a lower pressure. This expansion further lowers the temperature of this stream, which is fed back to the heat exchanger to provide refrigeration for the other streams, which is possible because the expansion to low pressure substantially increases the heat capacity of the refrigerant stream. The low-pressure evaporated refrigerant then exits the MHEX and is compressed to restart the cycle. The PRICO process may therefore be viewed as a traditional vapor compression refrigeration cycle that absorbs and removes heat from the natural gas stream. Further details about liquefaction plants and other process concepts are given in the next chapter, and the PRICO process itself is studied in many examples throughout this thesis. Note that while this process appears quite simple on initial inspection due to the lack of chemical reactors or multistage separation trains, the underlying thermodynamic considerations for streams simultaneously exchanging heat and changing phase at closely-matched cryogenic temperatures lead to complex process models and imply complicated economic trade-offs that preclude straightforward design and evaluation. Accordingly, rigorous thermodynamic models are required to ensure realism and feasibility of the process design or operating state, further complicating simulation and optimization problems involving liquefaction.

The most important and challenging unit operation to model in such a process is the multiphase multistream heat exchanger. However, none of the widely-used process simulation software suites include rigorous simulation-based models for this critical unit operation. Running simulations of the PRICO process in software such as Aspen Plus® or Aspen HYSYS® often involves a trial-and-error strategy due to the models being overconstrained and therefore unable to guarantee satisfaction of the second law of thermodynamics. This means that many parameter combinations chosen by the user will result in infeasible heat transfer at the model "solution", with no feedback given as to how to specify more reasonable conditions. Furthermore, the optimization routines in these commercial products usually consist of a local optimization method for smooth nonlinear programs (usually based on the sequential-quadratic program-

ming (SQP) method) that acquires local sensitivity information about the flowsheet outputs by perturbation of the inputs in a finite-differencing scheme. As will be shown in this thesis, these processes are truly described by inherently nonsmooth behavior and the performance of optimization and simulation algorithms are highly sensitive to inaccuracies in derivative (or generalizations thereof) evaluation. As such, a key element that sets the work detailed in this thesis apart from all of the existing literature on the topic of liquefaction process optimization is the use of nonsmooth models and exact sensitivity analysis methods in the constituent flowsheet units. Nonsmooth or nondifferentiable models have experienced somewhat niche usage within the process systems engineering (PSE) community until very recently, despite their many applications and benefits that will become apparent in the course of this thesis.

However, the use of nonsmooth models is by no means the only method for simulating and optimizing liquefaction processes. Many frameworks for process and system modeling have been proposed and championed in the PSE literature over the course of the past 40-50 years that can and have been applied to such problems. Each of these approaches fall somewhere on a spectrum that represents a level of trade-off between applicability and ease of formulation and solution, as shown in Figure 1-2.

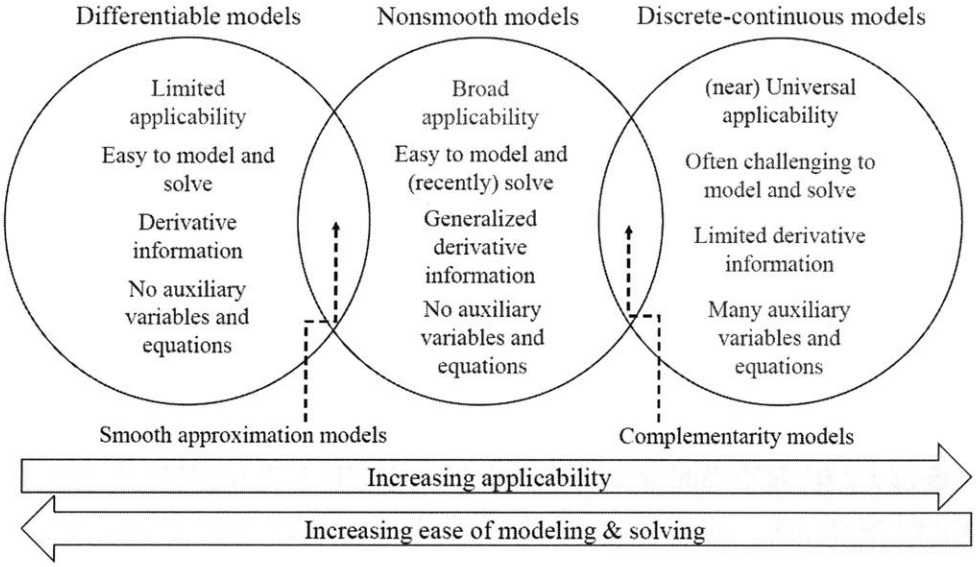


Figure 1-2: Modeling strategies in Process Systems Engineering.

For those problems that can be completely described by differentiable (smooth)

models, doing so is virtually always the best approach. The majority of efficient algorithms for equation solving and optimization exploit derivative information to achieve robust and rapid convergence to solutions. Such algorithms show the benefits of many decades of research, having been studied and modified continuously throughout, resulting in highly reliable methods and implementations thereof. Unfortunately, not all PSE problems can be modeled in this framework, and natural gas liquefaction processes are one such example of this.

The next-most widely used modeling archetype is the far more applicable (nearly universally so) discrete-continuous framework. In terms of steady-state process optimization, there are two main types of discrete-continuous problems. The first of these are superstructure formulations, in which many (or all) of the alternative configurations for a proposed system are explicitly modeled and then the optimizer chooses the optimal configuration or design. Most relevant to this thesis, this formulation is commonly used for heat exchanger network synthesis. Other applications abound however, including process synthesis, reactor network design and separation train sequencing. A comprehensive description of superstructure models may be found in e.g. Grossmann *et al.*[44] The other major type of discrete-continuous model used in chemical engineering applications is known as a generalized disjunctive program. In these models, subsets of the problem constraints are either active (enforced) or ignored depending on the values taken by discrete decision variables that are implied by logical propositions (or reformulations thereof). Generalized disjunctive programming has been used for many of the same problem archetypes as the superstructure methodology, in addition to finding use in the simulation and optimization of multiphase equilibrium systems, which will be further explored later in this thesis. The reader is referred to the article by Grossmann and Trespalacios[45] for a detailed survey of these methods. Both of these archetypes are formulated as mixed-integer programs, which can be reasonably simple to solve if the remaining continuous constraints are linear or affine, but potentially very difficult to solve otherwise. These latter mathematical programs are known as mixed-integer nonlinear programs (MINLPs) and pose many challenges to optimization algorithms, particularly if the nonlinear functions are also

nonconvex. In all cases, a global optimization algorithm is required to solve a mixed-integer problem, which are conjectured to have worst-case exponential complexity in the number of variables. Such models are also prone to exhibiting pathological behaviors during the solution procedure that are difficult to exclude *a priori*. However, for describing truly discrete or discontinuous model behavior, such formulations are the only viable approach. Fortunately, the thermodynamic regime changes and heat transfer considerations that must be included in a framework for simulating or optimizing liquefaction processes need not be modeled in this way, for in actuality, these are continuous (yet nondifferentiable) transitions and phenomena. Nevertheless, the discrete-continuous approach is the predominant modeling philosophy for such problems in the literature.

In the comparatively-unexplored middle ground between the previous categories sits the nonsmooth modeling paradigm. Modeling with nonsmooth functions has traditionally been avoided because the classic measure of local sensitivity, the derivative, is undefined for those points at which a nonsmooth function instantaneously and discontinuously changes slope. This behavior defeats most of the aforementioned established methods for equation solving and optimization. However, notions of "generalized derivatives" and numerical methods that can exploit such information have existed in the literature for some time, though until very recently, these objects have remained impractical to compute. However, owing to the recent advances in automatable nonsmooth sensitivity analysis by Khan and Barton,[64] described in detail in Chapter 2, nonsmooth equations now represent essentially no greater challenge than do smooth equations, at least for the purposes of equation-solving problems (algorithms for reliable nonsmooth optimization remain somewhat in their infancy). As will be demonstrated throughout this thesis, this enables the development of compact equation-based models for complex systems that would otherwise be modeled as challenging, large-scale MINLPs.

As indicated in Figure 1-2, there are other strategies that attempt to balance some of the tradeoffs between the previously mentioned approaches, notably to avoid tackling nonsmoothness directly. As the name implies, smooth approximation models

represent an attempt to relax the nonsmoothness in a model by replacing nondifferentiable terms with smooth ones that exhibit similar behavior. The article by Gopal and Biegler[41] gives an overview of the applications of smoothing methods in the context of PSE calculations. Complementarity constraints, on the other hand, recast nonsmooth problems in an optimization context and generally also use approximations to yield nonlinear programs that are solvable with established techniques. Baumrucker *et al.*[14] provide an overview of complementarity-constrained mathematical programming for chemical engineering applications. A concrete illustration of these different modeling approaches is now given in the context of a simple unit operation model.

Example 1.1. Consider the model of a one-way (check) valve in a process. The flow, F, through the valve is related to the pressure difference across the valve, ΔP, and for the purposes of this simple example, assume this relationship is described by a locally Lipschitz function $f : \mathbb{R} \to \mathbb{R}$ that returns positive values when the pressure difference is positive, negative values when it is negative, and zero when $\Delta P = 0$. However, a check valve closes when the pressure difference across the valve is negative to prevent reversal of flow direction. Using nonsmooth functions, this continuous switching behavior is very simply modeled with the following equation:

$$F = \max\left(0, f(\Delta P)\right).$$

The mechanism of the equation is obvious by inspection, it includes only the relevant physical quantities and it models the intended behavior exactly. It is therefore a compact and accurate model of the valve, as desired.

However, as many authors have not had the mathematical tools to use nonsmooth functions directly, one tactic for avoiding them has been to attempt to cast the problem as a differentiable one using smoothing approximations. Possible models for the check valve using this strategy are

$$F = \frac{\sqrt{f(\Delta P)^2 + \beta^2} + f(\Delta P)}{2},$$

as suggested by Balakrishna and Biegler[12] or

$$F = f(\Delta P) + \beta \ln(1 + \exp(-\Delta P/\beta)),$$

as suggested by Chen and Mangasarian,[25] where in each case, β is a user-defined parameter that represents a tradeoff between accuracy and numerical conditioning. At $\Delta P = 0$, the error in the first approximation is 0.5β and the error in the second is $\ln(2)\beta$, though the latter equation decays more rapidly to the true function as ΔP moves away from zero.[41] In either case, it is no longer so immediately obvious what the model represents by inspection, and, in addition, the model includes a nonphysical parameter β that must be tuned appropriately (by trial and error) and is always inaccurate around the point of switching.

Instead of transforming the nonsmooth problem into a smooth one, another approach is to reformulate the nonsmooth terms in the form of complementarity constraints and then solve the model as an optimization problem. The constraints of such a model for the check valve are as follows:

$$\begin{aligned} F &= f(\Delta P) + s_B, \\ f(\Delta P) &= s_A - s_B, \\ 0 \leq s_A &\perp s_B \geq 0, \end{aligned}$$

where s_A and s_B are nonphysical slack variables that have been added to the problem and $\perp$ is the complementarity operator that is equivalent to requiring $s_A s_B = 0$. This problem may be solved by minimizing the product $\rho s_A s_B$ subject to the first two equations and the variable bound constraints, where ρ is a user-defined penalty parameter that must be tuned to balance accuracy and solvability. Once again, this model obfuscates the problem with the addition of additional variables and tuning parameters, while requiring far more complex machinery to solve and still not achieving the exactness of the basic nonsmooth formulation.

Finally, a discrete-continuous modeling framework may be applied to this problem, resulting in the following set of constraints for an optimization problem with a

constant objective function:

$$
\begin{aligned}
f^L \leq f(\Delta P) \leq f^U \\
F \geq f(\Delta P), \\
F \leq f^U(1 - y_1), \\
F \leq f(\Delta P) + (f^U - f^L)(1 - y_2), \\
y_1 + y_2 = 1, \\
P^L \leq \Delta P \leq P^U, \\
0 \leq F \leq F^U, \\
\mathbf{y} \in \{0, 1\}^2,
\end{aligned}
$$

where now the values of ΔP and F are constrained to fall between prescribed lower and upper bounds, as is the value of the function f (the bounds on which are, for instance, chosen based the bounds on ΔP) and nonphysical binary variables $\mathbf{y}$ are used to determine the state of the valve, that is, $y_1 = 1$ (true) when the valve is closed and $y_2 = 1$ (true) when the valve is open. While this model is exact assuming the bounds on f are properly chosen, the model complexity has now increased to the point where an optimization algorithm that can handle mixed-integer problems is required.

Even in this simple example, the potential advantages of handling model nonsmoothness directly are extremely evident. Curiously, the latter two formulations of the problem are those most commonly used in the literature despite the significant increases in complexity. This is likely due to the fact that reliable, practical methods for obtaining the necessary local sensitivity information about the nonsmooth formulation have, until recently, been unavailable. New advances in this area have been used throughout this thesis to create compact and accurate models for such processes where other authors have resorted to far more complicated frameworks.

1.2 Objective and scope

The main objective of this work was to develop models, algorithms and software to enable robust simulation and optimization of steady-state natural gas liquefaction processes. These methods were to be designed so that they would be able to handle both simple and complex liquefaction process concepts with thermodynamics described by realistic models. The project also intended to demonstrate that a modeling strategy based around the use of nonsmooth functions was a viable and altogether effective approach to solving complex and realistic chemical engineering problems, capable of avoiding many of the pitfalls associated with the other frameworks highlighted previously.

Other operations in the LNG production chain, including both upstream of liquefaction (e.g. dehydration and purification of the source gas) and downstream of liquefaction (e.g. transportation, storage and regasification) have not been considered herein. Dynamic simulation and optimization of events such as process startup, shutdown and disturbance rejection were also not in the scope of this project, nor was determining optimal control strategies. Development of new optimization methods themselves was also not a focus in this project. Additionally, validation of results with either experimental or plant data was not possible during this project, and so results are instead compared against models and data from Aspen Plus v8.4.[5]

1.3 Thesis structure and summary of contributions

The primary contribution of this thesis is the development of new optimization methods for natural gas liquefaction processes based on nondifferentiable process models. Achieving this required the development of nonsmooth simulation-based flowsheeting strategies, which themselves required new advances in models for the constituent unit operations commonly found in liquefaction processes. A concept map for the main topics in this thesis is shown in Figure 1-3. In brief, the work flow of the project proceeded as follows. Building upon the development of efficient mathematical techniques

for sensitivity analysis of nonsmooth functions, a new nonsmooth simulation-based model for MHEXs was developed. This model was then augmented with methods for automatically detecting phase changes in the streams exchanging heat and robustly handling vapor-liquid equilibrium calculations with (possibly) single-phase results, enabling the simulation of LNG production processes. This framework was useful for simulations involving simple descriptions of mixture thermodynamics, but proved challenging to use with more realistic equation of state (EOS) models. To address this, new nonsmooth extensions of the celebrated inside-out algorithms for flash calculations were developed that extended the automatic handling of flash outlet conditions to cases involving complex thermodynamics. However, even this was insufficient to equip LNG process simulations with realistic EOS models, and so a nonsmooth density extrapolation method was developed to help avoid failures in flash calculations at high temperatures and/or pressures. Combining all these techniques, a new framework for flowsheeting with nonsmooth models and generalized derivatives was established and liquefaction process simulation and optimization with thermodynamics described by a cubic equation of state was performed successfully and reliably.

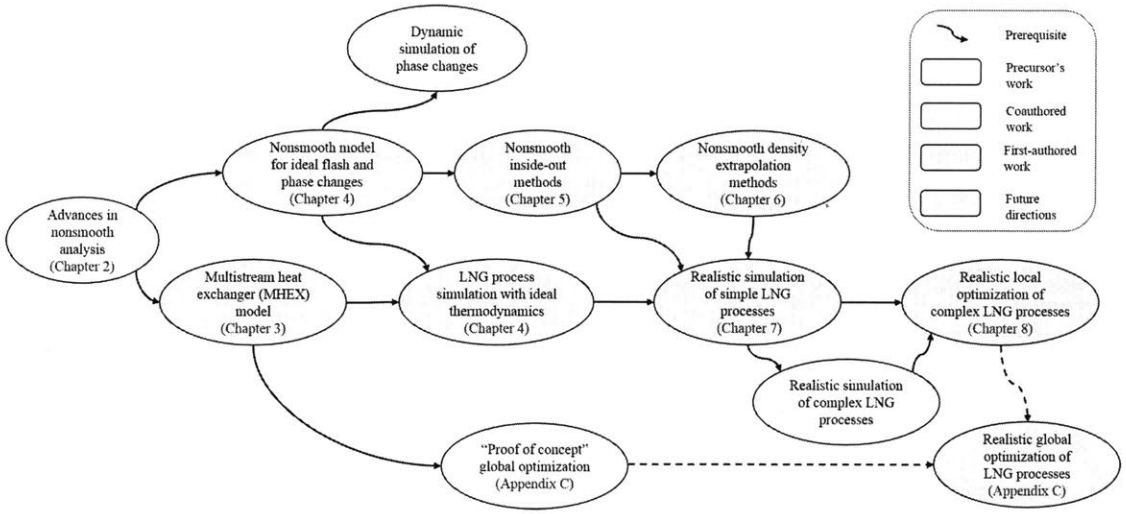


Figure 1-3: Concept map of this thesis project. Chapter numbers indicate where in this thesis document the corresponding work is described in detail.

The specific contents and contributions of each chapter of this thesis are briefly summarized in the following. The published or submitted articles associated with

each chapter are also indicated by citation after the corresponding chapter title.

Chapter 2 – Background: This preliminary chapter gives a more in-depth introduction to natural gas, LNG and liquefaction processes. It also introduces the relevant mathematical background in nonsmooth analysis necessary to understand the methods and algorithms developed in the remainder of this thesis.

Chapter 3 – A nonsmooth model for multistream heat exchanger simulation and design:[137] A nonsmooth simulation-based model for MHEXs is developed in analogy to traditional models for countercurrent two-stream heat exchangers and pinch analysis techniques. In contrast to the simulation models found in most commercial software that only require energy balance in the MHEX, the nonsmooth model includes equations that rigorously enforce heat transfer feasibility from both second law and and equipment size perspectives. Flowsheets of processes involving MHEXs are simulated in the absence of explicit thermophysical property models.

Chapter 4 – Modeling phase changes in multistream heat exchangers:[136] The necessary modifications to the basic MHEX model of the previous chapter to allow for the incorporation of thermodynamic models are detailed. This primarily includes provisions for automatically detecting and handling streams changing phase. A nonsmooth formulation of vapor-liquid equilibrium equations is also developed that allows flash calculations to converge automatically to single-phase or two-phase solutions. Initial case studies involving the PRICO process simulated with idealized thermophysical property models are also presented.

Chapter 5 – Nonsmooth inside out algorithms for robust flash calculations:[140] An improvement on the nonsmooth flash formulation from the previous chapter is developed in the form of nonsmooth inside-out algorithms. Building on the classic two-phase inside-out algorithms that allow for extremely robust and efficient nonideal flash calculations, the addition of nonsmooth functions allows for reliable convergence to solutions regardless of the true phase regime prescribed by the flash parameters.

Chapter 6 – A nonsmooth method for density extrapolation and pseudoproperty evaluation:[135] A procedure for augmenting the nonsmooth inside-out

algorithms from the previous chapter with density extrapolation methods is presented. These methods handle instances that arise in the process of solving a flash calculation where thermophysical property models are queried for the density (or equivalently, volume or compressibility) of a phase that does not exist at the conditions of a given iteration. The method improves on algorithms found in the literature for extrapolating reasonable properties for nonexistent phases by recasting them in terms of nonsmooth functions that are amenable to exact sensitivity analysis.

Chapter 7 – Process flowsheeting with nonsmooth models and generalized derivatives:[139] The nonsmooth modeling elements of the previous chapters are combined into a flowsheeting strategy that generalizes the sequential-modular framework for process simulation. Elements of computationally-relevant generalized derivatives are calculated and communicated throughout flowsheets containing nonsmooth models and submodels, including those solved with the nonsmooth inside-out methods. The PRICO process is again simulated, this time with the Peng-Robinson cubic EOS providing the thermodynamic model.

Chapter 8 – An optimization strategy for liquefied natural gas production processes:[138] Flowsheets for the PRICO process and more complex liquefaction processes described by the method of the previous chapter are optimized using a reliable interior-point method and a constraint formulation that results in optimal MHEX area utilization. Highly accurate descriptions of the processes are included in compact optimization formulations that can be automatically initialized and optimized with little *a priori* information. The results indicate that the strategies developed in this thesis project show great promise for the future of optimizing large-scale liquefaction processes.

Chapter 9 – Conclusions and future research directions: This final chapter provides a summary of the work completed in this thesis project and gives suggestions for future work based on the contributions herein.

Appendices: This thesis contains three appendices. Appendix A is a summary of the abbreviations and notation used throughout the manuscript. Appendix B details the primary physical property methods used in Chapters 4-8. Appendix C describes

some of the initial efforts in applying deterministic global optimization techniques to liquefaction processes and their constituent submodels.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

Background

This preliminary chapter introduces several of the major concepts that will be studied throughout this thesis. Firstly, a brief introduction to natural gas, liquefied natural gas and liquefaction processes is given to further contextualize and motivate the project. The remainder of the chapter is devoted to mathematical background in the area of nonsmooth analysis, which will be used heavily throughout the rest of this thesis. More specialized preliminary information pertaining to the content of an individual chapter of this thesis may also be found within the given chapter.

2.1 Natural gas

Natural gas is an odorless, colorless and noncorrosive hydrocarbon mixture composed primarily of methane. Generally, natural gas also contains appreciable amounts of nitrogen and heavier alkanes such as ethane, propane and butanes, along with lesser amounts of heavier hydrocarbons depending on the exact gas source. Water, oxygen, carbon dioxide, hydrogen sulfide and other sulfurous compounds, and trace contaminants such as mercury may also be present in untreated natural gas. Table 2.1 shows the typical ranges of abundance for the compounds that comprise dry natural gas. As a result of its light hydrocarbon composition, natural gas is also the cleanest fossil fuel product in regards to CO_2 emissions as shown in Table 2.2, while also producing virtually no sulfur oxides and only trace amounts of nitrogen oxides during combustion.

In light of the overwhelming scientific consensus on anthropogenic climate change, the development of greener energy sources is more vital than ever to meet the world's future energy needs.

Table 2.1: Typical component composition ranges in dry natural gas.[127]

Component	Typical composition range (mol%)
Methane	87.0 - 97.0
Ethane	1.5 - 7.0
Propane	0.1 - 1.5
n-Butane	0.01 - 0.3
iso-Butane	0.01 - 0.3
n-Pentane	trace - 0.04
iso-Pentane	trace - 0.04
Heavier hydrocarbons	trace - 0.06
Nitrogen	0.2 - 5.5
Carbon dioxide	0.1 - 1.0
Oxygen	0.01 - 0.1
Hydrogen sulfide	trace - 0.02

Table 2.2: CO_2 emissions from burning various fuels (MMBTU = million British thermal units).[129]

Fuel type	kg CO_2 / MMBTU
Coal	95.3
Crude Oil	74.5
Diesel Fuel	73.2
Gasoline	71.3
Natural Gas	53.1

At present, natural gas is the world's third largest source of fuel, accounting for 24.0% and 24.1% of global energy consumption in 2015 and 2016, respectively.[18] Production and consumption of natural gas continue to grow year over year and proven reserves of over 186 trillion cubic meters of unrecovered natural gas exist on the planet.[18] The U.S. Energy Information Administration (EIA) projects that in the timeframe from 2017 to 2040, natural gas usage (in terms of energy consumption) will increase more than any other single fuel source, driven primarily by the industrial and electric power sectors.[130] By 2040, a full 40% of the energy production in the U.S. is

expected to be attributed to natural gas.[130] In terms of the global energy markets, in 2016, over one trillion cubic meters of natural gas was traded (up nearly 5% from 2015), of which 68% was transported by pipeline and 32% was transported as LNG.[18] However, pipeline transmission's share of this trade is decreasing steadily and global production of LNG is projected to increase by almost 30% by 2020 as major facilities come on-line worldwide.[18] The U.S. EIA expects LNG to be the dominant mode of export for natural gas produced in the U.S. by the year 2020, accounting for over 70% of the nation's gas trade. This is in large part due to the five new major LNG export hubs that are scheduled to be operational by that time.[130]

The inherent economic challenges of importing or exporting large volumes of a gas make the production and transportation of LNG an attractive alternative, despite the required cost and energy expenditure. Liquefied natural gas occupies about 1/600th of its vapor phase volume at 15°C and atmospheric pressure, which enables bulk transportation in specialized vessels. Once natural gas has been liquefied, the marginal cost of transportation as a function of distance is much lower than that of pipeline transmission. While affected by many factors, the breakeven distance at which it becomes more economical to liquefy natural gas prior to transport is on the order of 2,200 miles (or 700 miles for the case of offshore gas production).[85]

However, the production of LNG incurs significantly higher capital and operating costs than preparing natural gas for pipeline transmission. Liquefaction operations on average account for around 50% of the total investment in a plant,[85;90] and on average, liquefaction also increases the total supply cost of natural gas by around $1.50-$2.00 per MMBTU.[86] Table 2.3 summarizes typical ranges for costs incurred in the LNG production chain. Some portion of the gas itself is also usually burned to provide energy (e.g. to run the compressors in the liquefaction process), and these losses are also shown in Table 2.3. As this operation accounts for such substantial costs, and given the optimistic projections for increased production of LNG worldwide, optimization of liquefaction process design and operation is of high importance to many suppliers.

Table 2.3: Typical module contributions to total capital cost, supply cost and gas loss along the LNG production chain.[58;86;90]

	Upstream	Liquefaction	Shipping	Regasification
Capital cost	20-30%	40-60%	5-25%	5-15%
Supply cost	20-30%	35-55%	10-25%	5-10%
Gas loss (as fuel)	–	10-14%	1.5-3.5%	1-2%

2.2 Natural gas liquefaction processes

The first patent for a natural gas liquefaction process was granted to Godfrey Cabot in 1914, however, it wasn't until 1941 that the first commercial LNG production plant was built in Cleveland, Ohio. Despite initially successful operation, public perception and general acceptance of LNG as a primary fuel source was shaken three years later in 1944, when a storage tank that had been newly added to the Cleveland facility ruptured. The resulting LNG spill ignited, killing 128 people. The ensuing investigation revealed that the incident was caused by brittle fracture of the inner wall of the tank resulting from the use of an inappropriate material of construction, which led to stringent regulations.[85] Since then however, no fatal incidents related to LNG have been recorded in the United States, and only one additional fatality-producing event has ever occurred worldwide, as the result of a fire in an Algerian LNG plant in 2004. Despite these two isolated events, LNG has gained acceptance as a reliable and safe fuel source, and the associated production facilities do not represent significant safety risks.

The goal of any natural gas liquefaction process is to use refrigeration to liquefy and then subcool the feed gas stream to between -163°C and -155°C (110.15 K – 118.15 K), such that the mixture remains in the liquid state upon expansion to storage pressure. As in any refrigeration process, heat is transferred from a source to a sink, where the temperature of the sink is higher than that of the source. In accordance with the second law of thermodynamics, heat transfer in this direction requires power input. As mentioned in the first chapter, the refrigeration cycle most commonly used for liquefaction processes is the vapor-compression cycle. This cycle is shown on a

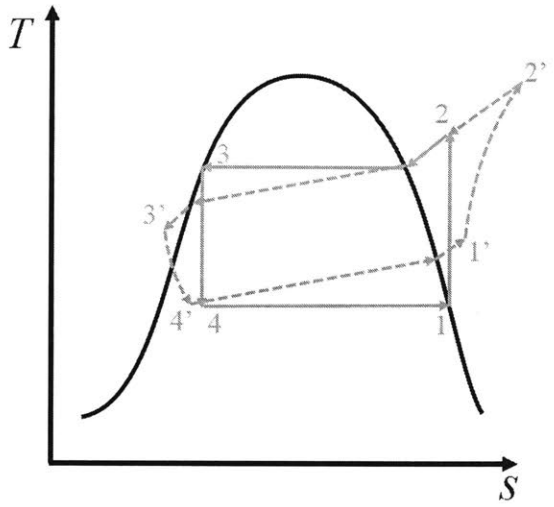


Figure 2-1: Temperature-entropy diagram for a vapor-compression refrigeration cycle. Green solid lines show the ideal cycle and blue dashed lines indicate the nonideal cycle.

temperature-entropy ($T - s$) diagram in Figure 2-1 for both an ideal case and a more realistic case. The ideal cycle has four basic steps:

- $1 \rightarrow 2$: isentropic compression,
- $2 \rightarrow 3$: desuperheating and isothermal condensation to the bubble point,
- $3 \rightarrow 4$: isentropic expansion,
- $4 \rightarrow 1$: isothermal evaporation to the dew point.

However, for practical reasons, the real cycle will proceed more similarly to the following:

- $1' \rightarrow 2'$: irreversible compression,
- $2' \rightarrow 3'$: desuperheating, nonisothermal condensation and subcooling,
- $3' \rightarrow 4'$: isenthalpic expansion (throttling),
- $4' \rightarrow 1'$: nonisothermal evaporation past the dew point (to avoid formation of liquid droplets in the compressor).

Note that while a pure refrigerant will condense and evaporate isothermally in the absence of pressure drop (as in the ideal cycle), the condensation and evaporation of a refrigerant mixture are inherently nonisothermal processes that take place over a temperature range.

There are three broad categories of liquefaction processes currently used for LNG production: pure-refrigerant cascade processes, mixed-refrigerant processes and turbine-based processes. More complex technologies may also incorporate elements from more than one category in distinct stages, as discussed later.

Pure-refrigerant cascade processes are currently only used in a small number of liquefaction plants, despite being the original process concept for LNG production. In a cascade-style process, the natural gas feed is cooled and liquefied using different pure refrigerants in different refrigeration cycles. Using industry terminology, "vertical stages" are (generally closed-loop) refrigeration cycles that operate with a single type of refrigerant, whereas "horizontal stages" are subcycles within a vertical stage that operate at, for example, different pressure levels. Cascade processes often consist of three vertical stages, using propane or propylene for desuperheating, ethane or ethylene for liquefying, and methane for subcooling the natural gas feed.[131] Each of these stages consists of multiple horizontal stages so that each refrigerant is evaporated at multiple pressure levels to provide cooling that matches temperatures along the cooling curve of natural gas as closely as possible. As each horizontal stage requires a heat exchanger, complex configurations result in a large number of small MHEX units being needed for such a process. At present, the most successful commercial cascade process is the ConocoPhillips Optimized Cascade®, which uses either propane or propylene in the first stage, ethylene in the second stage and replaces the traditional third stage with an open-loop process involving methane and boil-off gas from LNG expansion for the final stage.[28]

Currently, the vast majority of liquefaction plants operate using some form of a mixed-refrigerant liquefaction processes. The key difference between pure-refrigerant cascade and mixed-refrigerant processes is illustrated in Figure 2-2. As natural gas is a mixture, it liquefies over a temperature range rather than isothermally. The

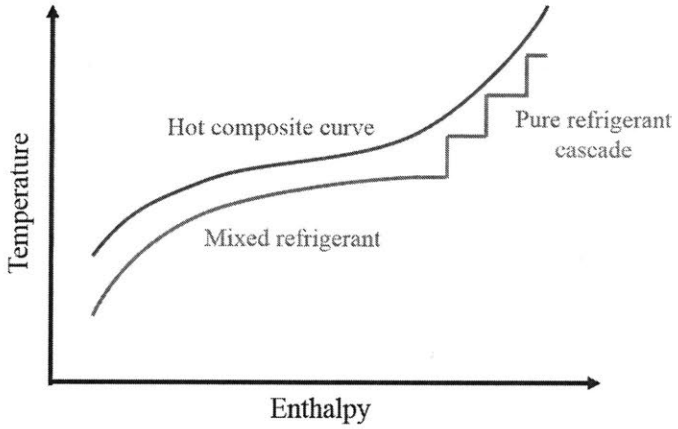


Figure 2-2: Typical hot and cold composite curve shapes for mixed refrigerant and pure refrigerant cascade processes.

use of pure refrigerants that evaporate isothermally at fixed pressure levels therefore necessitates many vertical and horizontal stages to provide refrigeration at small temperature differences, in a procedure analogous to how a simple numerical integration technique would approximate the area under a curve. If a refrigerant mixture is used, however, refrigeration of the natural gas can be feasibly performed in a single stage by attempting to match nonlinear cooling curves directly. The PRICO process introduced in the previous chapter is one such example of a single-stage, single mixed-refrigerant (SMR) process. These processes may be enhanced by the addition of horizontal stages, which in this case are created by phase separation of the refrigerant mixture to produce liquid- and vapor-phase mixtures with distinct compositions within the same cycle. An example of such a process is studied in Chapter 8 (see Figure 8-10).

Multiple vertical stages may also be used in mixed-refrigerant processes to yield processes that are more appropriate for high-throughput (base-load) LNG production facilities. As the name implies, dual mixed-refrigerant (DMR) processes use two refrigerant mixtures in a cascade, the first for desuperheating the natural gas and the second for liquefaction and subcooling. An example of a simple DMR process concept is shown in Figure 2-3 in which the second vertical stage includes phase separation of the main refrigerant to create two horizontal stages within the refrigerant cycle.

The original DMR process concept was patented in 1985,[89] though more recently both Shell and Air Products and Chemicals, Inc. (APCI) have licensed DMR process technologies.[43;103] A different two-stage mixed-refrigerant cycle is Technip/Air Liquide's TEALARC process.[80;74] In its most-studied configuration, the first mixed refrigerant is used only to provide cooling to the second. The natural gas stream is then cooled only through heat exchange with the second refrigerant mixture. The only commercially-licensed mixed-refrigerant process technology with more than two vertical stages is the Statoil/Linde Mixed-Fluid Cascade (MFC®)[118] that uses three cycles in a cascade. The process design is analogous to that of the traditional cascade concept with three pure refrigerants, except that here, each is replaced by a distinct refrigerant mixture. The addition of mixed-refrigerant stages beyond three is unlikely to be advantageous in practice, as significant diminishing returns are observed on the improvement in process efficiency, while the cost and complexity of operation increase substantially.[131;74]

Turbine-based processes are based on the reverse-Brayon cycle instead of the vapor-recompression cycle. In these processes, a refrigerant, typically nitrogen or a mixture of nitrogen and methane, is first compressed to very high pressure (greater than 10 MPa). The high pressure refrigerant is then precooled (along with the natural gas) in a multistream heat exchanger and expanded to low pressure through a turbine, reducing the temperature further. The cold, low pressure refrigerant is then used to liquefy and subcool the natural gas stream.[131;74] A example schematic of a turbine-based LNG production process is shown in Figure 2-4. A notable difference between these processes and those based on the vapor-recompression cycle is that here the refrigerant is always in in the vapor phase. Additionally, the use of a turbine in place of a throttle valve causes a much larger temperature drop and allows for the extraction of work from the process, though such a process will still require a net input of work to drive the compressor(s). However, turbine-based processes typically have low efficiency, though they have much faster start-up times compared to the other process concepts. This means that they are used primarily in peak-shaving plants, which are smaller facilities connected to existing gas supply networks that

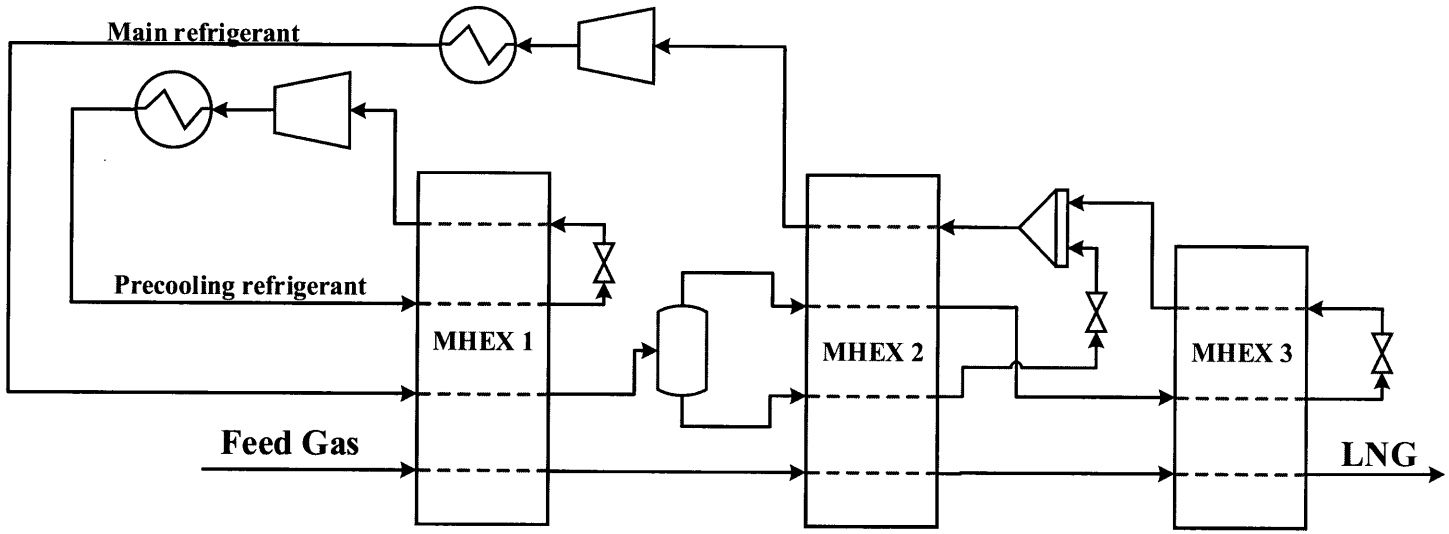


Figure 2-3: Flowsheet of a DMR process concept.

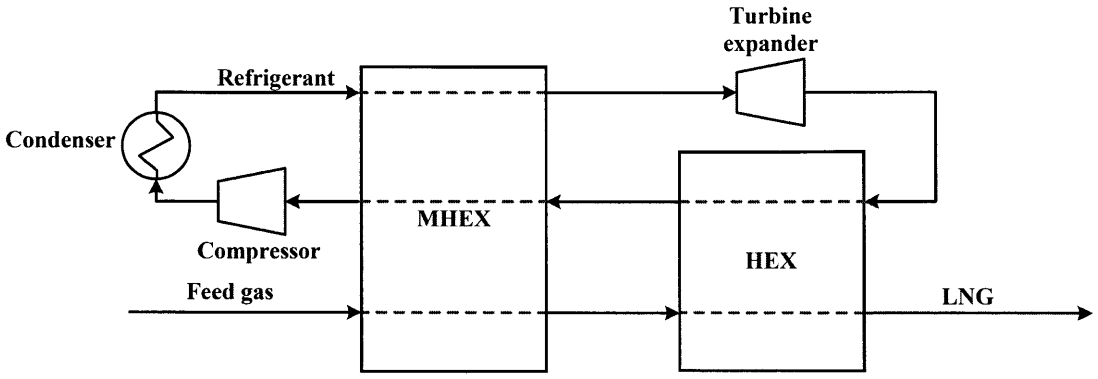


Figure 2-4: Turbine-based liquefaction process concept based on the reverse-Brayton cycle.

store natural gas when fuel demand is low and then supply LNG to help meet peak consumption demands.[131] Multiple vertical stages are possible in such processes by including, for instance, both a nitrogen and a methane expansion cycle, whereas multiple horizontal stages are possible with the inclusion of multiple turbines that expand to different pressure levels. Turbine-based processes have also found niche usage for offshore and remote gas production, wherein the plant area is tightly constrained, the use of inert nitrogen poses far fewer safety hazards than hydrocarbon mixtures and the gas phase-only heat transfer simplifies heat exchanger design and maintenance.[21;8]

Hybrids of these three process archetypes may also exist, as shown in Figure 2-5. One such widely-used process concept is APCI's propane-precooled mixed-refrigerant process (C3MR).[96] In this process, a pure-component propane stage (possibly with multiple horizontal pressure stages) is combined with a mixed-refrigerant stage. APCI's AP-X® process combines all three concepts, using a propane cycle for desuperheating, a mixed-refrigerant stage for liquefaction and a turbine-based nitrogen stage for subcooling the natural gas stream.[104] Due to their prevalence, this thesis exclusively considers mixed-refrigerant liquefaction processes. However, there is no reason that the methods and algorithms developed in this thesis could not be applied to these other process concepts.

It is worth noting that liquefaction is usually the last operation that is performed

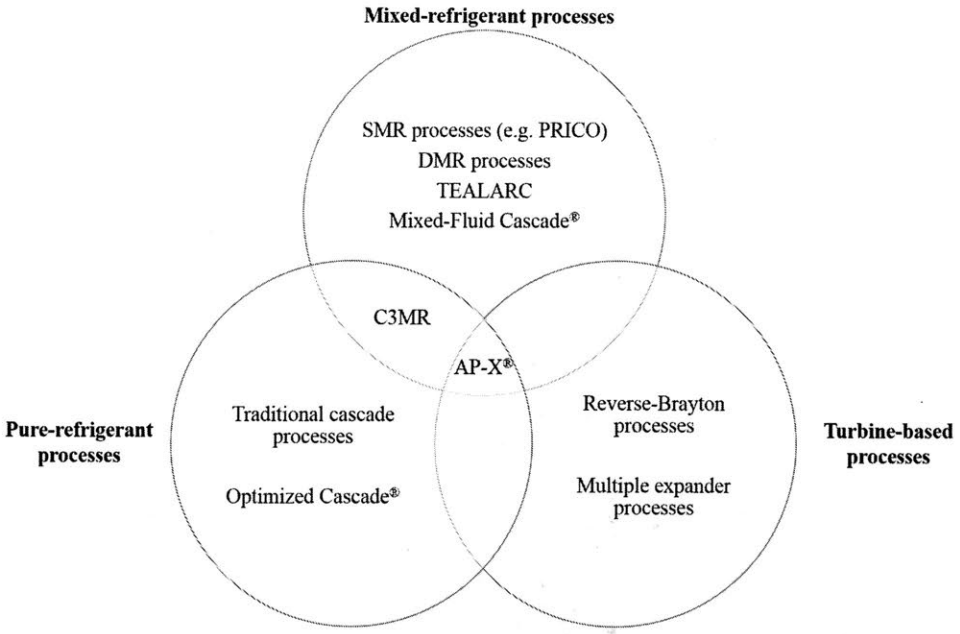


Figure 2-5: Venn diagram of liquefaction process concepts and technologies.

in an LNG processing plant, as pretreatment of the natural gas is virtually always required. In brief, hydrocarbon condensates, carbon dioxide, hydrogen sulfide, water and mercury are sequentially removed from the raw gas stream. Depending on the desired product and process configuration, heavier hydrocarbons may also be removed by flashing the purified natural gas stream prior to liquefaction. The liquid products of this operation are called natural gas liquids (NGLs) and can both be sold either as separate products (after fractionation) or used to create the refrigerant mixture for liquefaction. This process can also be integrated into the liquefaction process by first precooling the natural gas and then performing the liquid separation at a lower temperature level to control the component split ratios. An example of an integrated NGL extraction and liquefaction process is studied in Chapter 8 (see Figure 8-13). As noted earlier, purification operations that occur prior to liquefaction are outside the scope of the present thesis project, and a more detailed description of these other operations may be found in the literature, e.g. in the reference by Mokhatab *et al.*[85]

The abundance of liquefaction technologies invites comparison between processes. However, it must be kept in mind that process designs may have specific and non-

transferable goals and/or operating assumptions. SMR processes, for instance, are generally designed to run with lower product throughput than C3MR processes, which in turn run with lower capacity than DMR processes. Turbine-based processes operate with lower efficiency than other processes because the ability to start-up and shut-down rapidly to capitalize on peak demand or high-frequency fluctuations in the price of electricity more than mitigates the additional energy expenditure. Some of the process technologies are designed to run most efficiency in particular environments and will be suboptimal in less well-suited conditions. The MFC® process, for instance, was designed for Statoil's Snøvhit project and takes advantage of the extremely low ambient temperature and cold seawater. Available plot area for a new plant will also limit choices in liquefaction process configuration, as the higher capacity processes also generally have larger footprints. However, there are certainly cases where comparisons between optimized processes would be desirable, and so a tool that could be used robustly and flexibly in order to optimize the full range of technologies discussed here would have great value. The work detailed in this thesis represents a significant step towards building such a tool that takes advantage of recent advances in nonsmooth analysis.

2.3 Nonsmooth analysis

This section describes the nonsmooth numerical toolkit needed for the sensitivity analysis implemented in the process simulation and optimization case studies throughout this thesis. It reviews recent advances in the calculation of exact generalized derivative information for nondifferentiable functions, including implicit functions, and methods for solving nonsmooth equation systems. Notation and definitions largely follow from the recent review article on the subject by Barton *et al.*[13]

The motivation for the development of tractable and automatic methods for nonsmooth sensitivity analysis stems from the need for this information in equation-solving and optimization methods. Consider the traditional form of Newton's method for solving a square equation system involving a continuously differentiable function

$\mathbf{f} : X \to \mathbb{R}^n$, with $X \subset \mathbb{R}^n$ an open set, for a solution $\mathbf{x}^* \in X$, which is a vector that satisfies $\mathbf{f}(\mathbf{x}^*) = \mathbf{0}_n$. Given the current estimate for the solution on iteration k, $\mathbf{x}^{(k)}$, the iterations of Newton's method involve solving the following equation for $\mathbf{x}^{(k+1)}$:

$$\mathbf{Df}(\mathbf{x}^{(k)})(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\mathbf{f}(\mathbf{x}^{(k)}), \tag{2.1}$$

where $\mathbf{Df}(\mathbf{z})$ denotes the classical Jacobian matrix of $\mathbf{f}$ at some $\mathbf{z} \in X$. Under the condition that $\mathbf{Df}(\mathbf{x}^*)$ is nonsingular, Newton's method converges to $\mathbf{x}^*$ at a Q-quadratic rate in a neighborhood $N(\mathbf{x}^*)$ of $\mathbf{x}^*$, that is, the sequence of iterates $\{\mathbf{x}^{(k)}\}$ tending to the limit $\mathbf{x}^*$ obeys the relationship:[34]

$$\limsup_{k \to \infty} \frac{\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\right\|}{\left\|\mathbf{x}^{(k)} - \mathbf{x}^*\right\|^2} \leq +\infty \tag{2.2}$$

for $\mathbf{x}^{(k)} \in N(\mathbf{x}^*) \setminus \mathbf{x}^*$. This simple, elegant method and its many variants have been used reliably for hundreds of years for solving nonlinear systems of equations.

However, consider the case in which the function $\mathbf{f}$ is not continuously differentiable, but is instead a locally Lipschitz function on its domain, as defined next.

Definition 2.1. A function $\mathbf{f} : X \subset \mathbb{R}^n \to \mathbb{R}^m$, with X an open set, is *Lipschitz* near (in a neighborhood of) $\mathbf{x} \in X$ if there exist constants $\delta > 0$ and $L > 0$ such that whenever $\|\mathbf{x} - \mathbf{z}\| < \delta$, $\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{z})\| \leq L \|\mathbf{x} - \mathbf{z}\|$.

Definition 2.2. A function $\mathbf{f} : X \subset \mathbb{R}^n \to \mathbb{R}^m$, with X an open set, is *locally Lipschitz* on X if it is Lipschitz near $\mathbf{x}$ for all $\mathbf{x} \in X$.

A function satisfying Definition 2.2 is continuous but not necessarily differentiable everywhere on its domain. Rademacher's theorem shows that a locally Lipschitz continuous function $\mathbf{f}$ is in fact differentiable at each point in $X \setminus Z_{\mathbf{f}}$, where $Z_{\mathbf{f}} \subset X$ has zero Lebesgue measure. An example of this is the absolute value function $\text{abs} : x \mapsto |x|$, which is differentiable for all $x \neq 0$ with $\text{abs}'(x) = \text{sign}(x)$, but has no classical derivative defined at $x = 0$. Hence, if such a function were to appear in an equation system, then Newton's method would fail upon reaching any $\mathbf{x}^{(k)} \in Z_{\mathbf{f}}$. However, if instead of the Jacobian matrix, an appropriate generalization of the derivative

that provided useful local sensitivity information were available, then the method could conceivably continue. The following section discusses some useful "generalized derivatives" and how they may be calculated.

2.3.1 Notions of the generalized derivative

There have been numerous definitions offered in the literature for generalized derivatives of functions that are locally Lipschitz continuous on their domains which vary greatly in ease of computation and range of applicability. For the purposes of this thesis, the primary interest is in generalized derivatives that have elements that are both easy (automatic) to compute and have desirable properties when used in equation-solving methods. To this end, two extremely useful generalized derivatives are defined as follows:

Definition 2.3. [34;27] Let the function $\mathbf{f} : X \to \mathbb{R}^m$, where X is an open subset of $\mathbb{R}^n$, be locally Lipschitz continuous. The *B-subdifferential* of $\mathbf{f}$ at $\mathbf{x}$, $\partial_B \mathbf{f}(\mathbf{x})$, is the set:

$$\partial_{\mathrm{B}}\mathbf{f}(\mathbf{x}) := \left\{ \mathbf{H} \in \mathbb{R}^{m\times n} : \lim_{i\to\infty} \mathbf{Df}(\mathbf{x}^{(i)}) = \mathbf{H}, \quad \lim_{i\to\infty} \mathbf{x}^{(i)} = \mathbf{x}, \quad \mathbf{x}^{(i)} \in X \setminus Z_{\mathbf{f}}, \forall i \in \mathbb{N} \right\} \tag{2.3}$$

where $Z_{\mathbf{f}}$ is the set of points of measure zero where $\mathbf{f}$ is not differentiable. The *Clarke Jacobian* of $\mathbf{f}$ at $\mathbf{x}$ is defined as:

$$\partial_{\mathrm{C}}\mathbf{f}(\mathbf{x}) := \mathrm{conv}\left(\partial_{\mathrm{B}}\mathbf{f}(\mathbf{x})\right), \tag{2.4}$$

where conv$(\cdot)$ returns the convex hull of its argument.

Note that these sets are always non-empty and are singletons with $\partial_{\mathrm{B}}\mathbf{f}(\mathbf{x}) = \partial_{\mathrm{C}}\mathbf{f}(\mathbf{x}) = \{\mathbf{Df}(\mathbf{x})\}$ at any point $\mathbf{x} \in X$ for which $\mathbf{f}$ is continuously differentiable, so that they are truly generalizations of the Jacobian matrix. As an example, consider again the absolute value function. The B-subdifferential of this function at $x = 0$ is the set $\{-1, 1\}$ and the Clarke Jacobian is the interval $[-1, 1]$. For all other $x \in \mathbb{R}$, each of these generalized derivatives is the set $\{\mathrm{sign}(x)\}$.

Two important subsets of locally Lipschitz continuous functions are now introduced: the class of lexicographically-smooth functions and the class of piecewise differentiable functions.

Lexicographically-smooth functions

The concept of lexicographic smoothness is a generalization of classical directional differentiability as shown in the following definition.

Definition 2.4. [87] Given a locally Lipschitz continuous mapping $\mathbf{f} : X \subset \mathbb{R}^n \to \mathbb{R}^m$, X open, $\mathbf{f}$ is called lexicographically smooth (L-smooth) at $\mathbf{x} \in X$ if, for any $k \in \mathbb{N}$ and directions matrix $\mathbf{M} = [\mathbf{m}_{(1)} \ \cdots \ \mathbf{m}_{(k)}] \in \mathbb{R}^{n\times k}$, the following directional derivatives are well-defined:

$$\begin{aligned} &\mathbf{f}^{(0)}_{\mathbf{x},\mathbf{M}} : \mathbb{R}^n \to \mathbb{R}^m : \mathbf{d} \mapsto \mathbf{f}'(\mathbf{x};\mathbf{d}), \\ &\mathbf{f}^{(j)}_{\mathbf{x},\mathbf{M}} : \mathbb{R}^n \to \mathbb{R}^m : \mathbf{d} \mapsto [\mathbf{f}^{(j-1)}_{\mathbf{x},\mathbf{M}}]'(\mathbf{m}_{(j)};\mathbf{d}), \quad \forall j \in \{1,\ldots,k\}, \end{aligned} \tag{2.5}$$

where the directional derivative of $\mathbf{f}$ at $\mathbf{x}$ in the direction $\mathbf{d} \in \mathbb{R}^n$ is given by

$$\mathbf{f}'(\mathbf{x};\mathbf{d}) := \lim_{\delta \downarrow 0} \frac{\mathbf{f}(\mathbf{x}+\delta\mathbf{d}) - \mathbf{f}(\mathbf{x})}{\delta}. \tag{2.6}$$

The function $\mathbf{f}$ is said to be L-smooth on X if it is L-smooth at each point $\mathbf{x} \in X$.

Pertinent to this work, the class of L-smooth functions includes all continuously differentiable and piecewise differentiable functions, all convex functions and all compositions thereof. L-smoothness is also inherited by implicitly-defined functions,[65] which will be covered in detail in a later section. Another generalization of the derivative may be defined for general L-smooth functions in terms of a matrix of directions $\mathbf{M} \in \mathbb{R}^{n\times k}$ as follows:

Definition 2.5. [87] Let the function $\mathbf{f} : X \subset \mathbb{R}^n \to \mathbb{R}^m$, X open, be L-smooth at $\mathbf{x} \in X$ and let $\mathbf{M} \in \mathbb{R}^{n\times k}$ have full row rank. The mapping $\mathbf{f}^{(k)}_{\mathbf{x},\mathbf{M}} : \mathbb{R}^n \to \mathbb{R}^m$ is linear and the matrix

$$\mathbf{D}_{\mathrm{L}}\mathbf{f}(\mathbf{x};\mathbf{M}) := \mathbf{D}\mathbf{f}^{(k)}_{\mathbf{x},\mathbf{M}}(\mathbf{0}_n) \in \mathbb{R}^{m\times n} \tag{2.7}$$

is called the *lexicographic (L-) derivative* of $\mathbf{f}$ at $\mathbf{x}$ in the directions $\mathbf{M}$. The collection of these objects for all possible full row rank matrices is called the *lexicographic (L-) subdifferential* of $\mathbf{f}$ at $\mathbf{x}$:

$$\partial_{\mathrm{L}}\mathbf{f}(\mathbf{x}) := \{\mathbf{D}_{\mathrm{L}}\mathbf{f}(\mathbf{x};\mathbf{M}) : k \in \mathbb{N},\ \mathbf{M} \in \mathbb{R}^{n\times k},\ \mathbf{M} \text{ has full row rank}\}. \tag{2.8}$$

Despite its less intuitive definition, the L-subdifferential is in fact closely related to the other definitions of a generalized derivative. In general, the L-subdifferential is a subset of the plenary hull of the Clarke Jacobian,[64] and a subset of Clarke Jacobian itself in the case of scalar-valued functions.[87] If $\mathbf{f}$ is differentiable at $\mathbf{x}$, then the L-subdifferential is a singleton corresponding to the Jacobian matrix of $\mathbf{f}$ at $\mathbf{x}$, i.e., $\partial_{\mathrm{L}}\mathbf{f}(\mathbf{x}) = \{\mathbf{Df}(\mathbf{x})\}$.[87] These facts imply that calculating an element of the L-subdifffential is no less useful than calculating an element of the Clarke Jacobian itself for all applications that require a (generalized) Jacobian-vector product, e.g. all efficient equation-solving procedures. The L-subdifferential can also be directly related to the B-subdifferential in the case of piecewise differentiable functions, as indicated in the next subsection.

Piecewise differentiable functions

The nonsmooth functions used in the primary applications of this thesis are piecewise differentiable functions. As the name perhaps implies, a piecewise differentiable function is a continuous function which consists of a set of "pieces" in its domain, each of which is described by a continuously differentiable function. This is formalized in the following definition.

Definition 2.6. [111] Given a locally Lipschitz continuous mapping $\mathbf{f} : X \subset \mathbb{R}^n \to \mathbb{R}^m$ with X open, $\mathbf{f}$ is called *piecewise differentiable* (PC^1) at $\mathbf{x} \in X$ if there exists a neighborhood $N(\mathbf{x}) \subset X$ of $\mathbf{x}$ and a finite collection of continuously differentiable (C^1) selection functions on $N(\mathbf{x})$, $\mathcal{F}_{\mathbf{f}}(\mathbf{x}) \equiv \{\mathbf{f}_{(1)}, \ldots, \mathbf{f}_{(k)}\}$ for some $k \in \mathbb{N}$, such that $\mathbf{f}$

is continuous on $N(\mathbf{x})$, and

$$\mathbf{f}(\eta) \in \{\mathbf{f}^{(i)}(\eta) : i \in \{1, \ldots, k\}\}, \quad \forall \eta \in N.$$

The function $\mathbf{f}$ is said to be PC^1 on X if $\mathbf{f}$ is PC^1 at each $\mathbf{x} \in X$. Given a family of selection functions $\mathcal{F}_\mathbf{f}(\mathbf{x})$, the set of essentially active indices of $\mathbf{f}$ at $\mathbf{x}$ with respect to $\mathcal{F}_\mathbf{f}(\mathbf{x})$ is given by:

$$I_\mathbf{f}^{\text{ess}}(\mathbf{x}) := \{i \in \{1, \ldots, k\} : \mathbf{x} \in \text{cl}(\text{int}\{\eta \in N(\mathbf{x}) : \mathbf{f}(\eta) = \mathbf{f}^{(i)}(\eta)\})\}.$$

The set of essentially active indices is also associated with a set of essentially active selection functions, given by:

$$\mathcal{E}_\mathbf{f}(\mathbf{x}) := \{\mathbf{f}^{(i)} : i \in I_\mathbf{f}^{\text{ess}}(\mathbf{x})\}.$$

For a given $\mathbf{x} \in X$, the sets of essentially active indices and essentially active selection functions must be nonempty by definition (else there would be no function defined at this point), and in the case that $|I_\mathbf{f}^{\text{ess}}(\mathbf{x})| = 1$, then $\mathbf{f}$ is C^1 at this point. Additionally, all compositions of PC^1 functions are also PC^1 functions.[111] The most widely-used PC^1 functions in PSE applications are functions such as $(x, y) \mapsto \max(x, y)$, $(x, y) \mapsto \min(x, y)$, $x \mapsto |x|$ and compositions thereof. The function $(x, y, z) \mapsto \text{mid}(x, y, z)$ that returns the median value of its arguments is a less commonly-used PC^1 function that is nevertheless very important for the work in this thesis. The function is potentially not differentiable at any point where at least two of its arguments are equal while differentiable elsewhere and may be expressed directly in terms of more familiar PC^1 functions as follows:

$$\text{mid}(a, b, c) = \max(\min(a, b), \min(\max(a, b), c)), \tag{2.9}$$

which can be obtained as the result of applying a bubble-sort procedure to the list of

three arguments, or alternatively as follows:

$$\text{mid}(a,b,c) = a + b + c - \max(\max(a,b),c) - \min(\min(a,b),c), \tag{2.10}$$

which is essentially the statement that the median value remains after the maximum and minimum are removed from a list of three values.

A key result in Khan and Barton[64] states that if $\mathbf{f}$ is PC^1 at $\mathbf{x}$, then $\mathbf{f}$ is also L-smooth at $\mathbf{x}$, and, $\partial_{\text{L}}\mathbf{f}(\mathbf{x}) \subset \partial_{\text{B}}\mathbf{f}(\mathbf{x})$. This is particularly useful from an applications perspective, as there exists an automatic and tractable method for calculating elements of the L-subdifferential using lexicographic directional (LD-) derivatives, discussed next.

Lexicographic directional derivatives

While useful theoretically, the definition of the L-derivative (Definition 2.5) is cumbersome and an automatic method for calculating the necessary high-order directional derivatives in the definition is not obvious. Fortunately, recent work by Khan and Barton[64] established a link between the L-derivative and another object that is convenient to calculate, the so-called lexicographic directional derivative, defined as follows:

Definition 2.7. [64] Given $k \in \mathbb{N}$, any $\mathbf{M} = [\mathbf{m}_{(1)} \ \cdots \ \mathbf{m}_{(k)}] \in \mathbb{R}^{n\times k}$, and $\mathbf{f} : X \to \mathbb{R}^m$ L-smooth at $\mathbf{x} \in X$ with $X \subset \mathbb{R}^n$ open, the *lexicographic directional (LD-) derivative* of $\mathbf{f}$ at $\mathbf{x}$ in the directions $\mathbf{M}$ is

$$\begin{aligned}\mathbf{f}'(\mathbf{x};\mathbf{M}) &\equiv \left[\mathbf{f}^{(0)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(1)}) \quad \mathbf{f}^{(1)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(2)}) \quad \cdots \quad \mathbf{f}^{(k-1)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(k)})\right], \\ &= \left[\mathbf{f}^{(k)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(1)}) \quad \mathbf{f}^{(k)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(2)}) \quad \cdots \quad \mathbf{f}^{(k)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(k)})\right].\end{aligned} \tag{2.11}$$

The LD-derivative is a generalization of the classical directional derivative that is resolved sequentially along k directions instead of just one. This sequence of directions systematically probes the local sensitivity of the function, and, assuming that the set of directions is sufficiently well-chosen (e.g. $\mathbf{M}$ spans $\mathbb{R}^n$), will eventually step away from points of nonsmoothness and return a linear mapping, as is expected for a

derivative-like object. Note that if $k = 1$, then $\mathbf{f}'(\mathbf{x}; \mathbf{M}) = \mathbf{f}'(\mathbf{x}; \mathbf{m}_{(1)})$, which is just the classical directional derivative in the direction $\mathbf{m}_{(1)}$. Continuing this analogy, it is therefore also the case that if $\mathbf{f}$ is differentiable at $\mathbf{x}$, then:

$$\mathbf{f}'(\mathbf{x}; \mathbf{M}) = \mathbf{Df}(\mathbf{x})\mathbf{M}. \tag{2.12}$$

The LD-derivative also satisfies several useful relationships with regards to the L-derivative. For one, if $\mathbf{M}$ has full row rank, then:

$$\mathbf{f}'(\mathbf{x}; \mathbf{M}) = \mathbf{D}_{\mathrm{L}}\mathbf{f}(\mathbf{x}; \mathbf{M})\mathbf{M}. \tag{2.13}$$

However, the directions matrix need not be full row rank in the case of the LD-derivative, which is particularly important for intermediate calculations involving compositions of functions. Critically, the LD-derivative also satisfies a sharp chain rule, as shown in the following theorem:

Theorem 2.1. [64] Let $X \subset \mathbb{R}^n$ be open and let the function $\mathbf{f} : X \to \mathbb{R}^m$ be L-smooth at $\mathbf{x} \in X$. Let $k \in \mathbb{N}$ and $\mathbf{M} \in \mathbb{R}^{n \times k}$. Let $Z \subset \mathbb{R}^m$ be open and introduce a second function $\mathbf{g} : Z \to \mathbb{R}^q$ that is L-smooth at $\mathbf{f}(\mathbf{x}) \in Z$, then:

$$[\mathbf{g} \circ \mathbf{f}]'(\mathbf{x}; \mathbf{M}) = \mathbf{g}'(\mathbf{f}(\mathbf{x}); \mathbf{f}'(\mathbf{x}; \mathbf{M})). \tag{2.14}$$

Additionally, for L-smooth functions $\mathbf{u}$ and $\mathbf{v}$ (with appropriate domains and ranges), then defining $\mathbf{f} : (\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x} + \mathbf{y}$ and $\mathbf{g} : \mathbf{x} \mapsto (\mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{x}))$ yields a summation rule:

$$[\mathbf{u} + \mathbf{v}]'(\mathbf{x}; \mathbf{M}) = \mathbf{u}'(\mathbf{x}; \mathbf{M}) + \mathbf{v}'(\mathbf{x}; \mathbf{M}), \tag{2.15}$$

and for scalar-valued L-smooth functions u and v (with appropriate domains and ranges) defining $f : (x, y) \mapsto xy$ and $\mathbf{g} : \mathbf{x} \mapsto (u(\mathbf{x}), v(\mathbf{x}))$ yields a product rule:

$$[uv]'(\mathbf{x}; \mathbf{M}) = v(\mathbf{x})u'(\mathbf{x}; \mathbf{M}) + u(\mathbf{x})v'(\mathbf{x}; \mathbf{M}). \tag{2.16}$$

To date, the LD-derivative is the only known object that obeys a sharp chain rule

(Equation (2.14)) and also furnishes useful generalized derivatives for nonsmooth functions. As a consequence of the relationship between the L- and B-subdifferential for PC^1 functions noted in the previous section and the relationship given in Equation (2.13), it can be seen that given an open set $X \subset \mathbb{R}^n$ and $\mathbf{f} : X \to \mathbb{R}^n$ that is PC^1 at $\mathbf{x} \in X$,

$$\begin{aligned}\mathbf{f}'(\mathbf{x}; \mathbf{I}_{n\times n}) &= \mathbf{D}_{\mathrm{L}}\mathbf{f}(\mathbf{x}; \mathbf{I}_{n\times n})\mathbf{I}_{n\times n}, \\ &= \mathbf{D}_{\mathrm{L}}\mathbf{f}(\mathbf{x}; \mathbf{I}_{n\times n}) \in \partial_{\mathrm{L}}\mathbf{f}(\mathbf{x}) \subset \partial_{\mathrm{B}}\mathbf{f}(\mathbf{x}),\end{aligned}$$

where $\mathbf{I}_{n\times n}$ is the identity matrix in $\mathbb{R}^{n\times n}$. Therefore, for a PC^1 function, an element of the B-subdifferential at a point in the domain can be obtained by calculating the LD-derivative of the function at that point in the identity directions.

For clarity, a simple example that highlights many aspects of the previous definitions and theorems is presented.

Example 2.1. Let the function $f : \mathbb{R}^2 \to \mathbb{R}$ be given by

$$f(x_1, x_2) = \min(x_1, x_2).$$

Then f is a PC^1 function with $\mathcal{F}_{\mathbf{f}}(\mathbf{x}) = \{\mathbf{x} \mapsto x_1\}$ whenever $x_1 < x_2$, $\mathcal{F}_{\mathbf{f}}(\mathbf{x}) = \{\mathbf{x} \mapsto x_2\}$ whenever $x_2 < x_1$ and $\mathcal{F}_{\mathbf{f}}(\mathbf{x}) = \{\mathbf{x} \mapsto x_1, \mathbf{x} \mapsto x_2\}$ whenever $x_1 = x_2$. For each $\mathbf{x} \in \mathbb{R}^2$ such that $x_1 = x_2$, the directional derivative of f in the direction $\mathbf{d} = (d_1, d_2)$ is:

$$\begin{aligned}f'(\mathbf{x}; \mathbf{d}) &= \lim_{\delta \downarrow 0} \frac{f(x_1 + \delta d_1, x_2 + \delta d_2) - f(\mathbf{x})}{\delta}, \\ &= \lim_{\delta \downarrow 0} \frac{\min(x_1 + \delta d_1, x_2 + \delta d_2) - \min(x_1, x_2)}{\delta}, \\ &= \lim_{\delta \downarrow 0} \frac{\delta \min(d_1, d_2)}{\delta}, \\ &= \min(d_1, d_2).\end{aligned}$$

The mapping $\mathbf{d} \mapsto f'(\mathbf{x}; \mathbf{d})$ is clearly also directionally differentiable. Therefore, given

an appropriate directions matrix, the first higher-order directional derivative given in Definition 2.4 will be well-defined. Let $k := 2$ and for a full row rank matrix $\mathbf{M} = [\mathbf{m}_{(1)}\ \mathbf{m}_{(2)}]$, then first-order directional derivative for $\mathbf{x}$ with $x_1 = x_2$ is $f^{(0)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(1)}) = \min(m_{11}, m_{21})$, as calculated earlier. If $m_{11} = m_{21}$, then nonsmoothness is still present and the higher-order directional derivatives are required. Since $\mathbf{M}$ is of full rank, if $m_{11} = m_{21}$ then $m_{12} \neq m_{22}$ necessarily. If $m_{11} \neq m_{21}$ then $f^{(0)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(1)})$ evaluates to either m_{11} or m_{21}, depending on which is smaller. For illustration, assume that $m_{11} = m_{21}$ but $m_{12} < m_{22}$, then:

$$\begin{aligned} f^{(1)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(2)}) &= \left[f^{(0)}_{\mathbf{x},\mathbf{M}}\right]'(\mathbf{m}_{(1)};\mathbf{m}_{(2)}) \\ &= \lim_{\delta\downarrow 0} \frac{\min(m_{11}+\delta m_{12}, m_{21}+\delta m_{22}) - \min(m_{11}, m_{21})}{\delta}, \\ &= \lim_{\delta\downarrow 0} \frac{\delta \min(m_{12}, m_{22})}{\delta} = \min(m_{12}, m_{22}) = m_{12}. \end{aligned}$$

and

$$\begin{aligned} f^{(2)}_{\mathbf{x},\mathbf{M}}(\mathbf{d}) &= \left[f^{(1)}_{\mathbf{x},\mathbf{M}}\right]'(\mathbf{m}_{(2)};\mathbf{d}) \\ &= \lim_{\delta\downarrow 0} \frac{m_{12}+\delta d_1 - m_{12}}{\delta} \\ &= d_1. \end{aligned}$$

The lexicographic derivative of f in the directions $\mathbf{M}$ at $\mathbf{x}$ with $x_1 = x_2$ is then:

$$\mathbf{D}f^{(2)}_{\mathbf{x},\mathbf{M}}(\mathbf{0}_2) = \left[\frac{\partial f^{(2)}_{\mathbf{x},\mathbf{M}}}{\partial d_1}(\mathbf{0}_2) \quad \frac{\partial f^{(2)}_{\mathbf{x},\mathbf{M}}}{\partial d_2}(\mathbf{0}_2)\right] = [1\ \ 0].$$

Now, from the earlier calculations and Definition 2.7, the LD-derivative is given by:

$$\begin{aligned} f'(\mathbf{x};\mathbf{M}) &= \left[\mathbf{f}^{(2)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(1)}) \quad \mathbf{f}^{(2)}_{\mathbf{x},\mathbf{M}}(\mathbf{m}_{(2)})\right], \\ &= [m_{11}\ \ m_{12}]. \end{aligned}$$

The directions matrix is invertible and so the L-derivative can be recovered from the LD-derivative and $\mathbf{M}$ by solving Equation (2.13) (illustrated in a naïve manner)

$$\begin{aligned}
&\mathbf{D}_{\mathrm{L}} f(\mathbf{x}, \mathbf{M})\, \mathbf{M} = f'(\mathbf{x}; \mathbf{M}) \\
\Longrightarrow\ &\mathbf{D}_{\mathrm{L}} f(\mathbf{x}, \mathbf{M}) = f'(\mathbf{x}; \mathbf{M})\mathbf{M}^{-1}, \\
&= \frac{[m_{11}\ \ m_{12}]}{\det(\mathbf{M})} \begin{bmatrix} m_{22} & -m_{12} \\ -m_{21} & m_{11} \end{bmatrix}, \\
&= \begin{bmatrix} \dfrac{m_{11}m_{22} - m_{12}m_{21}}{\det(\mathbf{M})} & \dfrac{m_{11}m_{12} - m_{11}m_{12}}{\det(\mathbf{M})} \end{bmatrix}, \\
&= \begin{bmatrix} \dfrac{\det(\mathbf{M})}{\det(\mathbf{M})} & \dfrac{0}{\det(\mathbf{M})} \end{bmatrix}, \\
&= [1\ \ 0],
\end{aligned}$$

which is the same result as obtained previously. This is an element of the lexicographic subdifferential of f at those $\mathbf{x}$ with $x_1 = x_2$. The B-subdifferential of f at $\mathbf{x}$ with $x_1 = x_2$ is the set $\{[1\ \ 0], [0\ \ 1]\}$, so this vector is also an element of the function's B-subdifferential for these $\mathbf{x}$. Note that if $\mathbf{M} := \mathbf{I}_{2\times 2}$, then $f'(\mathbf{x}; \mathbf{I}_{2\times 2}) = [0\ \ 1]$, as $m_{21} < m_{11}$, so it is also the case that $f'(\mathbf{x}; \mathbf{I}_{2\times 2}) \in \partial_{\mathrm{B}} f(\mathbf{x})$, as expected.

As Example 2.1 shows, manual computation of elements of the L-subdifferential is quite tedious. However, the computation of LD-derivatives can be made surprisingly straightforward using the automatic algorithm first developed in Khan and Barton.[64] Automatic differentiation (AD) techniques may be applied to any function that is factorable, which includes all functions that can be represented finitely on a computer in terms of compositions of so-called elemental functions (e.g. standard univariate functions such as sin, exp, abs, etc. and multivariate functions such as $+, -, \times$, max, min, etc.) without the need for conditional statements (e.g. `if` statements) or loops without a fixed number of iterations (e.g. `while` loops).[42] The vector forward mode of AD for evaluating LD-derivatives of an L-smooth factorable function developed by Khan and Barton[64] is a generalization of the vector forward mode of AD for smooth functions described by Griewank and Walther[42]. When the elemental functions that

appear in a factored representation of $\mathbf{f}$ are all classically differentiable, the method is identical. Otherwise, the overarching mechanism is the same, but the rules needed to evaluate the LD-derivatives of the elemental functions must be included in the underlying library of calculus rules. For the presentation herein, let $\mathbf{f} : X \to \mathbb{R}^m$ be a factorable L-smooth function with domain X as an open subset of $\mathbb{R}^n$ and let the $\mathbf{M}$ be a directions matrix in $\mathbb{R}^{n\times k}$.

A factored representation of $\mathbf{f}$ consists of $l+1$ quantities $\mathbf{v}_0 \ldots \mathbf{v}_l$, where $\mathbf{v}_0 \equiv \mathbf{x}$ and $\mathbf{v}_l \equiv \mathbf{f}(\mathbf{x})$. The intermediate quantities $\mathbf{v}_1 \ldots \mathbf{v}_l$ are each obtained by applying an elemental function $\boldsymbol{\varphi}_i : X_{\boldsymbol{\varphi}_i} \to \mathbb{R}^{m_{\boldsymbol{\varphi}_i}}$, $(X_{\boldsymbol{\varphi}_i} \subset \mathbb{R}^{n_{\boldsymbol{\varphi}_i}}$ open) with $i \in \{1, \ldots, l\}$, to some subset of the other intermediate quantities $\mathbf{v}_j$ with $j \in \{0, \ldots, l-1\}$ and $j < i$. This members of this subset are indicated using the Boolean precedence operator "$\prec$", defined by Griewank and Walther,[42] which indicates functional dependency, i.e., which subset of intermediate values $\mathbf{v}_j$ are necessary for the evaluation of elemental function $\boldsymbol{\varphi}_i$. In context, $j \prec i$ evaluates to `true` if quantity $\mathbf{v}_j$ is needed to evaluate an input to function $\boldsymbol{\varphi}_i$ and `false` otherwise. The argument of the function $\boldsymbol{\varphi}_i$ can therefore be denoted by $\mathbf{u}_i \equiv (\mathbf{v}_j)_{j\prec i}$, which is the vertical concatenation of all elements of the set $\{\mathbf{v}_j : j \prec i\}$ in order of increasing j. A brief example is now given to illustrate this concept.

Example 2.2. Consider the function $f : \mathbb{R}^2 \to \mathbb{R} : \mathbf{x} \mapsto \exp(\max(x_1, x_2))$. Here, $l = 2$, φ_1 is the bivariate max function and φ_2 is the exponential function. A factored representation of f in terms of intermediate quantities and elemental function arguments is as follows:

$$
\begin{aligned}
\mathbf{v}_0 &= (x_1, x_2), \\
\mathbf{u}_1 &= \mathbf{v}_0, \\
v_1 &= \max(\mathbf{u}_1), \\
u_2 &= v_1, \\
v_2 &= \exp(u_2), \\
f(\mathbf{x}) &= v_2.
\end{aligned}
$$

Now, assume that a calculus rule describing the local sensitivity of each elemental function $\boldsymbol{\varphi}_i$ is known, e.g. for the directional derivative in the classical vector forward mode and for the LD-derivative in the nonsmooth version. Then the intermediate sensitivities, $\dot{\mathbf{V}}_j$, and the sensitivity argument matrix $\dot{\mathbf{U}}_j \equiv (\dot{\mathbf{V}}_j)_{j \prec i}$ can be calculated analogously to $\mathbf{v}_j$ and $\mathbf{u}_j$ by application of these calculus rules along the same computational sequence as the function value calculation (due to the sharp chain rule). In this case, $\dot{\mathbf{V}}_0 \equiv \mathbf{M}$ and $\dot{\mathbf{V}}_l \equiv \mathbf{f}'(\mathbf{x}; \mathbf{M})$ $(= \mathbf{Df}(\mathbf{x})\mathbf{M}$ in the smooth case). An outline of this procedure is given in Algorithm 2.1.[64] In practice, the evaluation of the function value and its LD-derivative given $\mathbf{x}$ and $\mathbf{M}$ can be performed simultaneously with the use of operator overloading and a custom data type that holds both a value and the associated sensitivity vector. More advanced and efficient numerical techniques involving source code transformation are also possible, though outside the scope of the implementation described in this thesis.

Algorithm 2.1: Evaluate the value $\mathbf{y} \equiv \mathbf{f}(\mathbf{x})$ and LD-derivative $\dot{\mathbf{Y}} \equiv \mathbf{f}'(\mathbf{x}; \mathbf{M})$ of a factorable L-smooth function $\mathbf{f}$ given $\mathbf{x}$ and $\mathbf{M}$

$\mathbf{v}_0 \leftarrow \mathbf{x}$,
$\dot{\mathbf{V}}_0 \leftarrow \mathbf{M}$,
for $i = 1, \ldots, l$ **do**
  $\mathbf{u}_i \leftarrow (\mathbf{v}_j)_{j \prec i}$,
  $\mathbf{v}_i \leftarrow \boldsymbol{\varphi}_i(\mathbf{u}_i)$,
  $\dot{\mathbf{U}}_i \leftarrow (\dot{\mathbf{V}}_j)_{j \prec i}$,
  $\dot{\mathbf{V}}_i \leftarrow \boldsymbol{\varphi}_i'(\mathbf{u}_i; \dot{\mathbf{U}}_i)$,
end for
$\mathbf{y} \leftarrow \mathbf{v}_l$,
$\dot{\mathbf{Y}} \leftarrow \dot{\mathbf{V}}_l$,
Return $\mathbf{y}, \dot{\mathbf{Y}}$.

As noted before, if all elemental functions $\boldsymbol{\varphi}_i$ are differentiable, then this is exactly the algorithm for the classical vector forward mode of AD. Therefore, the only additions that must be made to an existing AD library are the elemental calculus rules for the LD-derivatives of the nondifferentiable functions. A full overview of these rules is given in the review article by Barton *et al.*,[13] but as an example, the LD-derivatives

for the absolute value function may be calculated as follows:

$$\text{abs}'(x, \mathbf{m}) = \text{fsign}(x, \mathbf{m}^\intercal)\mathbf{m}, \tag{2.17}$$

for $\mathbf{m} \in \mathbb{R}^{1\times k}$ and where here the fsign function maps from $\mathbb{R}^{k+1}$ to $\{-1, 0, 1\}$ and returns the first nonzero sign of its argument list, or zero if the argument is the $(k+1)$-dimensional zero vector. Consideration of Equation (2.17) reveals that if $x \neq 0$, i.e. the function is evaluated at a point of differentiability, then $\text{abs}'(x, \mathbf{m}) = \text{sign}(x)\mathbf{m}$, which is exactly the directional derivative given by classical calculus rules. At $x = 0$, the calculus rule (the sign function) is instead applied hierarchically to the direction vector $\mathbf{m}$. Similar insight yields the calculus rules for min, max, mid and the Euclidean norm functions.[64;13]

Approximations of LD-derivatives

Despite the apparent similarity at first inspection, directional derivatives in the coordinate directions are not guaranteed to furnish B-subdifferential elements in the case of PC^1 functions (or Clarke Jacobian elements in the case of general locally Lipschitz functions), i.e., in general:

$$\mathbf{f}'(\mathbf{x}; \mathbf{I}_{n\times n}) \neq [\mathbf{f}'(\mathbf{x}; \mathbf{e}_{(1)}) \;\; \mathbf{f}'(\mathbf{x}; \mathbf{e}_{(2)}) \;\; \cdots \;\; \mathbf{f}'(\mathbf{x}; \mathbf{e}_{(n)})] \tag{2.18}$$

at points of nondifferentiability, though they are equivalent elsewhere.[13] Numerical examples in Chapter 4 show that in challenging equation-solving problems, the set of nondifferentiable points of a function implemented in finite precision arithmetic is indeed reachable, and the differences are noticeable between using LD-derivatives and the approximation with directional derivatives in the coordinate directions. The right-hand side of Equation (2.18) also requires n applications of the forward mode of AD to compute (or a vector forward pass), as compared to the LD-derivative in the $\mathbf{I}_{n\times n}$ directions, which has its total computational cost bounded above (usually weakly) by $3n + 1$ times the function evaluation cost.[64] Given that the LD-derivative

is both more accurate and is at least as efficient to compute, it seems that there is no convincing reason to use this approximation.

As noted in the previous chapter, finite difference approximations of derivatives are ubiquitous in process simulation and optimization software. The commonly-used forward finite difference approximation is as follows:

$$\mathbf{f}'(\mathbf{x}; \mathbf{e}_{(j)}) = \frac{\partial \mathbf{f}}{\partial x_j}(\mathbf{x}) \approx \frac{\mathbf{f}(\mathbf{x} + \delta \mathbf{e}_{(j)}) - \mathbf{f}(\mathbf{x})}{\delta}, \tag{2.19}$$

where $\mathbf{e}_{(j)}$ is the jth Cartesian basis vector and δ is the perturbation in variable x_j chosen by the modeler. This approach may incur either significant truncation errors if δ is too large or significant round-off errors if δ is too small. Optimal values for δ are challenging to ascertain and only offer at best a tradeoff between these classes of errors. While sometimes useful for classical derivative approximations, finite differences are particularly prone to calculating erroneous generalized derivatives when used in nonsmooth sensitivity analysis. As finite differences only approximate directional derivatives, the aforementioned shortcomings of the approach are further compounded with the fact that the directional derivatives themselves do not in general return correct sensitivity information at points of nondifferentiability. Note also that Equation (2.19) implies that $n+1$ function evaluations are needed to obtain the full approximate Jacobian matrix in the standard application of the method, so it is also not significantly cheaper to compute than the LD-derivative in general. Several examples throughout this thesis show the potentially dramatic differences in performance obtained between using LD-derivatives, exact directional derivatives in the coordinate directions and finite difference approximations when solving challenging process simulation problems.

2.3.2 Equation-solving methods

Methods for solving square systems of locally Lipschitz continuous equations are now discussed. First, methods for fixed-point problems that do not require sensitivity information are introduced. Afterwards, methods for solving nonlinear equation sys-

tems with the aid of generalized derivatives are discussed.

Fixed-point methods

The fixed-point problem is stated as follows: given $\mathbf{g} : X \subset \mathbb{R}^n \to \mathbb{R}^n$, with X an open set, find $\mathbf{x}^* \in X$ such that $\mathbf{x}^* = \mathbf{g}(\mathbf{x}^*)$. If $\mathbf{g}$ is a contractive mapping on a closed subset X_0 of X, meaning that there exists an $\lambda < 1$ such that $\|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{z})\| \leq \lambda \|\mathbf{x} - \mathbf{z}\|$ for all $\mathbf{x}, \mathbf{z} \in X_0$, then there is a unique fixed point in X_0 that may be found using the method of successive substitution.[91] The successive substitution method is particularly simple: starting from an initial guess $\mathbf{x}^0 \in X_0$, the following sequence $\{\mathbf{x}^{(k)}\}$ is calculated:

$$\mathbf{x}^{(k+1)} := \mathbf{g}(\mathbf{x}^{(k)}), \tag{2.20}$$

with $\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}^*$ under the previous assumptions. While this iteration scheme is very simple and requires neither assumptions on the differentiability of $\mathbf{g}$ nor any local sensitivity information, it achieves only a Q-linear convergence rate, i.e. the sequence of iterates obeys the following relationship:[34]

$$\limsup_{k\to\infty} \frac{\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\right\|}{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|} \leq +\infty \tag{2.21}$$

for $\mathbf{x}^{(k)}$ in a neighborhood of $\mathbf{x}^*$ (excluding $\mathbf{x}^*$ itself). This convergence rate is potentially unacceptably slow (c.f. Equation (2.2)), especially when the evaluation of $\mathbf{g}$ is computationally expensive. Therefore, in practice, accelerated successive substitution methods are generally used instead. For the algorithms that will be discussed in Chapter 5, the method of Anderson acceleration was found to be particularly adept at promoting rapid convergence for the embedded fixed-point problem. The Anderson acceleration (also called Anderson mixing) technique considers a particular linear combination of the residuals from up to m previous iterations of the algorithm to attempt to accelerate the solution procedure, where m is a parameter chosen by the user.[2;134] If $m := 0$, then the method reduces to the successive substitution iteration. Otherwise, the iteration takes the following form: on iteration k, $m_k := \min(m, k)$

and the following unconstrained least squares problem is solved:

$$\min_{\boldsymbol{\gamma}} \left\| \mathbf{f}(\mathbf{x}^{(k)}) - \mathbf{F}(\mathbf{x}^{(k)})\boldsymbol{\gamma} \right\|^2, \tag{2.22}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^{m_k}$, $\mathbf{f}(\mathbf{x}^{(k)}) \equiv \mathbf{g}(\mathbf{x}^{(k)}) - \mathbf{x}^{(k)}$, $\mathbf{F}(\mathbf{x}^{(k)}) \equiv \left(\Delta\mathbf{f}(\mathbf{x}^{(k-m_k)}), \ldots, \Delta\mathbf{f}(\mathbf{x}^{(k-1)})\right)$ and $\Delta\mathbf{f}(\mathbf{x}^{(i)}) \equiv \mathbf{f}(\mathbf{x}^{(i+1)}) - \mathbf{f}(\mathbf{x}^{(i)})$. The solution of Equation (2.22) is denoted by $\boldsymbol{\gamma}^{(k)}$ and the current iterate is updated by the following equation:

$$\mathbf{x}^{(k+1)} = \mathbf{g}(\mathbf{x}^{(k)}) - \sum_{i=0}^{m_k-1} \gamma_i^{(k)} \left(\mathbf{g}(\mathbf{x}^{(k-m_k+i+1)}) - \mathbf{g}(\mathbf{x}^{(k-m_k+i)})\right). \tag{2.23}$$

As detailed in Walker and Ni,[134] there are techniques that reduce the computational burden of solving the least squares problem. Most important among these is the idea of solving Equation (2.22) with the QR decomposition technique while storing and updating the QR factors of the matrix $\mathbf{F}(\mathbf{x}^{(k)})$ on each iteration. This matrix only changes throughout the algorithm by either adding a new rightmost column and/or dropping the leftmost column, and these operations may be performed efficiently ($O(n^2)$ flops to delete a column and $O(mn)$ flops to add a column for an $m \times n$ matrix, as compared to the $O(2mn^2)$ flops needed to recompute the QR factors naïvely).[39] Another useful modification that has been used in this work is to drop columns of the matrix $\mathbf{F}(\mathbf{x}^{(k)})$ when its condition number becomes large (which can again be done inexpensively by checking the condition number of its R factor). Examples in Chapter 5 highlight the efficacy of this acceleration technique compared to basic successive substitution.

Sensitivity-based methods

Generalizations of Newton's method (Equation (2.1)) may also be used to solve nonsmooth equation systems. If the function $\mathbf{f} : X \subset \mathbb{R}^n \to \mathbb{R}^n$ (X open) is a semismooth function as defined by Facchinei and Pang,[34] a class of function that encompasses all PC^1 functions, then the semismooth Newton method[99] may be used to find a solution of the equation system $\mathbf{f}(\mathbf{x}^*) = \mathbf{0}_n$ for $\mathbf{x}^* \in X$ starting from an initial guess $\mathbf{x}^{(0)} \in X$.

The method proceeds identically to Equation (2.1), except that the Jacobian matrix is replaced with an element of a generalized derivative of $\mathbf{f}$ at the current iterate $\mathbf{x}^{(k)}$ denoted $\mathbf{G}(\mathbf{x}^{(k)})$ as follows:

$$\mathbf{G}(\mathbf{x}^{(k)})(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\mathbf{f}(\mathbf{x}^{(k)}). \tag{2.24}$$

When elements of the Clarke Jacobian are used for each $\mathbf{G}(\mathbf{x}^{(k)})$, then the method converges Q-superlinearly provided that the Clarke Jacobian at the solution contains no singular matrices, which is to say that the iterates will obey the following:[34]

$$\lim_{k\to\infty} \frac{\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\right\|}{\left\|\mathbf{x}^{(k)} - \mathbf{x}^*\right\|} \leq 0. \tag{2.25}$$

When $\mathbf{f}$ is PC^1 and $\mathbf{G}(\mathbf{x}^{(k)})$ is an element of the B-subdifferential of $\mathbf{f}$ at $\mathbf{x}^{(k)}$ for each k, the method converges Q-quadratically in a neighborhood of a solution, just as the classical Newton's method does, under the assumption that all elements of the B-subdifferential at the solution are nonsingular.[34] As a result of these convergence guarantees, these generalized derivatives are considered the most useful from a computational standpoint.

While generally highly effective, the semismooth Newton method will fail if $\mathbf{G}(\mathbf{x}^{(k)})$ is singular, which may be problematic for some nonsmooth equation systems, e.g. those with residuals containing expressions such as $\max(0, x)$. In such cases, an algorithm that is amenable to these problems at the expense of increased computational cost is the linear programming (LP) Newton method,[33] in which the following linear program is solved at each iteration in place of Equation (2.24):

$$\begin{aligned} &\min_{\gamma,\mathbf{x}} \ \gamma \\ &\text{s.t.} \ \left\|\mathbf{f}(\mathbf{x}^{(k)}) + \mathbf{G}(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)})\right\|_\infty \leq \gamma \min\left(\left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty, \left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty^2\right), \\ &\qquad \left\|(\mathbf{x} - \mathbf{x}^{(k)})\right\|_\infty \leq \gamma \left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty, \\ &\qquad \mathbf{x} \in X, \end{aligned} \tag{2.26}$$

where γ is an auxiliary variable used to drive convergence to the solution and X is a

set of known polyhedral bounds on the solution. The $\mathbf{x}$ part of the solution is used as the next iterate $\mathbf{x}^{(k+1)}$. Alternatively, the right-hand side of the first constraint may be rewritten simply as $\gamma \left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty^2$ and the step-size constraint may instead be replaced with

$$\left\|(\mathbf{x}-\mathbf{x}^{(k)})\right\|_\infty \leq \gamma \max\left(\left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty, \left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty^2\right),$$

for the same effect of allowing larger steps to be taken far from the solution.[35] The LP-Newton method can be further adapted to guarantee global convergence for C^1 and PC^1 equation systems with the addition of a backtracking linesearch, as described by Fischer *et al.*[35] As with the semismooth Newton method, local quadratic convergence can be achieved when $\mathbf{G}(\mathbf{x}^{(k)}) \in \partial_{\mathrm{B}}\mathbf{f}(\mathbf{x}^{(k)})$. It is also possible to combine the strengths of these two methods into a single method by taking steps with the LP-Newton (or globalized LP-Newton) method if $\mathbf{G}(\mathbf{x}^{(k)})$ is singular or poorly conditioned and by taking steps with the semismooth Newton method otherwise. This "hybrid" nonsmooth Newton method is shown in full in Algorithm 2.2.

2.3.3 Nonsmooth implicit functions

Methods for calculating generalized derivative information for implicitly defined functions are needed in order to introduce several of the models described in this thesis into simulation or optimization problems. In the case of differentiable functions, the sensitivity analysis implied by the classical implicit function theorem can be applied to obtain the derivatives of implicit functions. Fortunately, generalizations of the classical result now exist for L-smooth and PC^1 functions. For L-smooth functions this result is stated in the following theorem.

Theorem 2.2. [65;13] Let $W \subset \mathbb{R}^n \times \mathbb{R}^m$ be open, and $\mathbf{g} : W \to \mathbb{R}^m$ be L-smooth at $(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*)) \in W$. Then, if $\mathbf{g}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*)) = \mathbf{0}_m$ and $\left\{\mathbf{X} \in \mathbb{R}^{m\times m} : \exists[\mathbf{P} \quad \mathbf{X}] \in \partial_{\mathrm{C}}\mathbf{g}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\right\}$ contains no singular matrices, then there exists a neighborhood $N \subset \mathbb{R}^n$ of $\mathbf{p}^*$ and a Lipschitz continuous function $\mathbf{x} : N \to \mathbb{R}^m$ such that, for each $\boldsymbol{\eta} \in N$, $(\boldsymbol{\eta}, \mathbf{x}(\boldsymbol{\eta}))$ is the unique vector in a neighborhood of $(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ satisfying

Algorithm 2.2: Solve a square system of locally Lipschitz equations $\mathbf{f}(\mathbf{x}) = \mathbf{0}_n$

Choose $\mathbf{x}^{(0)} \in X$.
$k \leftarrow 0$.
while $\left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty > \varepsilon_{\text{tol}}$ and $k < k_{\max}$ **do**
  Calculate $\mathbf{f}(\mathbf{x}^{(k)})$ and $\mathbf{G}(\mathbf{x}^{(k)}) \equiv \mathbf{f}'(\mathbf{x}^{(k)}; \mathbf{I}_{n\times n})$.
  if $\text{cond}(\mathbf{G}(\mathbf{x}^{(k)})) < N_{\text{cond}}$ **then**
    Solve $\mathbf{G}(\mathbf{x}^{(k)})\mathbf{d}^{(k)} = -\mathbf{f}(\mathbf{x}^{(k)})$ for $\mathbf{d}^{(k)}$.
    $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)} + \mathbf{d}^{(k)}$.
  else
    Solve Equation (2.26) for $\gamma^{(k)}, \mathbf{x}^{(k+1)}$.
    $\mathbf{d}^{(k)} \leftarrow \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$.
    $\alpha \leftarrow 1$.
    $\Delta \leftarrow -\left(1 - \gamma^{(k)} \left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty\right) \left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|$.
    while $\left\|\mathbf{f}(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)})\right\|_\infty > \left(\left\|\mathbf{f}(\mathbf{x}^{(k)})\right\|_\infty + \sigma\alpha\Delta\right)$ and $\alpha > \alpha_{\min}$ **do**
      $\alpha \leftarrow \theta\alpha$.
    end while
    $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}$.
  end if
  $k \leftarrow k + 1$.
end while
Return $\mathbf{x}^{(k)}$.

The parameters in the algorithm are as follows. $k_{\max}$: maximum number of iterations allowed. ε_{tol}: overall error tolerance for the solution. N_{cond}: maximum condition number for which semismooth Newton step is taken. $\alpha_{\min}$: minimum step size modifier. σ, θ: linesearch parameters with values in the open interval (0,1). The latter four parameters are fixed in all examples throughout this thesis at the following values: $N_{\text{cond}} := 10^8$, $\alpha_{\min} := 10^{-13}$, $\sigma := 10^{-3}$, $\theta := 0.05$.

$\mathbf{g}(\boldsymbol{\eta}, \mathbf{x}(\boldsymbol{\eta})) = \mathbf{0}_m$. Moreover, $\mathbf{x}$ is L-smooth at $\mathbf{p}^*$; for any $k \in \mathbb{N}$ and any $\mathbf{M} \in \mathbb{R}^{n\times k}$, the LD-derivative $\mathbf{x}'(\mathbf{p}^*; \mathbf{M})$ is the unique solution $\mathbf{N} \in \mathbb{R}^{m\times k}$ of the equation system

$$\mathbf{g}'(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*); (\mathbf{M}, \mathbf{N})) = \mathbf{0}_{m\times k}. \tag{2.27}$$

Equation (2.27) indicates that the generalized derivative information is obtained from the solution of a nonsmooth nonlinear equation system. In the case of general L-smooth functions, Equation (2.27) is solved by applying the following lemma from Stechlinski *et al.*[117]

Lemma 2.3. Let conditions of Theorem 2.2 hold. Then the i^{th} column $\mathbf{n}_{(i)}$ of $\mathbf{N} \equiv$

$\mathbf{x}'(\mathbf{p}^*; \mathbf{M})$ is the unique solution to the set of equations:

$$\mathbf{0} = \mathbf{g}^{(i-1)}\left(\begin{bmatrix}\mathbf{p}^* \\ \mathbf{x}(\mathbf{p}^*)\end{bmatrix}, \begin{bmatrix}\mathbf{M}_{(1:i-1)} \\ \mathbf{N}_{(1:i-1)}\end{bmatrix}, \left(\begin{bmatrix}\mathbf{M}_{(i)} \\ \mathbf{n}_{(i)}\end{bmatrix}\right)\right). \tag{2.28}$$

where the notation $\mathbf{M}_{(1:i-1)}$ denotes the submatrix of $\mathbf{M}$ defined by columns 1 through $i-1$ of $\mathbf{M}$. The residual function for this equation system, defined as

$$\mathbf{h} : \mathbf{v} \to \mathbf{g}^{(i-1)}\left(\begin{bmatrix}\mathbf{p}^* \\ \mathbf{x}(\mathbf{p}^*)\end{bmatrix}, \begin{bmatrix}\mathbf{M}_{(1:i-1)} \\ \mathbf{N}_{(1:i-1)}\end{bmatrix}, \left(\begin{bmatrix}\mathbf{M}_{(i)} \\ \mathbf{v}\end{bmatrix}\right)\right), \tag{2.29}$$

is L-smooth on $\mathbb{R}^m$; for any $\mathbf{v} \in \mathbb{R}^m$ and $\mathbf{A} \in \mathbb{R}^{m \times q}$, $q \in \mathbb{N}$

$$\mathbf{h}'(\mathbf{v}; \mathbf{A}) = \mathbf{g}'\left(\begin{bmatrix}\mathbf{p}^* \\ \mathbf{x}(\mathbf{p}^*)\end{bmatrix}; \begin{bmatrix}\mathbf{M}_{(1:i-1)} & \mathbf{M}_{(i)} & \mathbf{0}_{m \times q} \\ \mathbf{N}_{(1:i-1)} & \mathbf{v} & \mathbf{A}\end{bmatrix}\right)\begin{bmatrix}\mathbf{0}_{i \times q} \\ \mathbf{I}_{q \times q}\end{bmatrix}. \tag{2.30}$$

Lemma 2.3 enables solving for each column of $\mathbf{x}'(\mathbf{p}^*; \mathbf{M})$ sequentially using a non-smooth Newton-type solver. At each step, $\mathbf{n}^{(i)}$ is found as the i^{th}-order directional derivative of $\mathbf{g}$ in direction $\begin{bmatrix}\mathbf{M}^{(i)} & \mathbf{n}^{(i)}\end{bmatrix}^{\mathrm{T}}$ satisfying Equation (2.28). Previously evaluated LD-derivative elements $\mathbf{N}_{(1:i-1)}$ of the implicit function, as well as the directions matrix up to column i, $\mathbf{M}_{(1:i)}$, are used in the calculations, so necessitating a sequential computation. The search direction for the Newton-type solver is provided by the LD-derivative $\mathbf{h}'(\mathbf{v}; \mathbf{A})$ of the residual function $\mathbf{h}$ from Equation (2.29), where $\mathbf{A}$ is an auxiliary directions matrix. Choosing $\mathbf{A}$ to be the $m \times m$ identity matrix is the best choice for efficient solution of the equation system. Algorithm 2.3 presents a procedure for solving Equation (2.27).

In the case of PC^1 functions, the hypotheses and consequences of Theorem 2.2 can be made more specific, in that piecewise differentiability of the implicit function $\mathbf{x}$ at $\mathbf{p}^*$ is inherited from piecewise differentiability of the participating functions (see the implicit function theorem for PC^1 functions given by Ralph and Scholtes).[101] Note

Algorithm 2.3: Evaluate an LD-derivative of a lexicographically smooth implicit function

1 Choose $\mathbf{M} \in \mathbb{R}^{n \times k}$.
2 **for** $i = 1, \ldots, k$ **do**
3 **if** $k = 1$ **then**
4 $\mathbf{M}_{(1)} \leftarrow \mathbf{M}_{(1)}$.
5 **else**
6 $\mathbf{M}_{(1:i)} \leftarrow \left[\mathbf{M}_{(1:i-1)} \quad \mathbf{M}_{(i)}\right]$.
7 **end if**
8 Provide an initial guess $\mathbf{v}^{(0)}$, e.g. $\mathbf{v}^{(0)} \leftarrow \mathbf{0}_n$
9 Solve $\mathbf{h}(\mathbf{v}) = \mathbf{0}_n$ for $\mathbf{v}^*$ with $\mathbf{h}$ defined as in Equation (2.29) with an iterative nonsmooth method. LD-derivatives are given by Equation (2.30), i.e. $\mathbf{h}'\left(\mathbf{v}^{(j)}; \mathbf{I}_{n \times n}\right)$ for iteration j.
10 **if** $k = 1$ **then**
11 $\mathbf{N}_{(1)} \leftarrow \mathbf{v}^*$.
12 **else**
13 $\mathbf{N}_{(1:i)} \leftarrow \left[\mathbf{N}_{(1:i-1)} \quad \mathbf{v}^*\right]$.
14 **end if**
15 **end for**
16 Return $\mathbf{N}$.

also that in this case, the condition from Theorem 2.2 that the projection of Clarke Jacobian of $\mathbf{g}$ must contain no singular matrices at $(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ may be weakened to the condition that $\mathbf{g}$ must be completely coherently oriented with respect to $\mathbf{x}$ at this point, which means that all matrices in the set $\prod_{j=1}^{m} \left\{ \frac{\partial g_{(i),j}}{\partial \mathbf{x}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*)) : i \in I_{\mathbf{g}}^{\text{ess}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*)) \right\}$ must have the same nonvanishing determinant sign (where $\prod$ denotes the Cartesian product). Moreover, if $\mathbf{g}$ is PC^1 at $(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ with known essentially active selection functions $\{\mathbf{g}^{(i)} : i \in I_{\mathbf{g}}^{\text{ess}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\}$, it is often less computationally expensive to use a different algorithm, Algorithm 2.4,[65] to compute the sensitivities in Equation (2.27). This algorithm iterates through the set of known essentially active selection functions and applies the classical implicit function theorem result to each differentiable piece in turn, stopping once the derivatives that satisfy Equation (2.27) are found. In the worst case, Algorithm 2.4 must cycle through all $|I_{\mathbf{g}}^{\text{ess}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))|$ selection functions before returning the sensitivities. A method for improving the robustness of Algorithm 2.4 in the presence of error in the linear solve in Line 5 is discussed in Chapter 7.

Algorithm 2.4: Evaluate an LD-derivative of a PC^1 implicit function

1 Choose $\mathbf{M} \in \mathbb{R}^{n \times k}$.
2 **for** $i = 1, \ldots, |I_{\mathbf{g}}^{\text{ess}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))|$ **do**
3 **if** $\mathbf{g}^{(i)}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))) = \mathbf{0}_m$ **then**
4 **if** $\det(\mathbf{D}_{\mathbf{x}}\mathbf{g}^{(i)}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*)) \neq 0$ **then**
5 Solve the following linear equation system for $\mathbf{N} \in \mathbb{R}^{m \times k}$:

$$\frac{\partial \mathbf{g}^{(i)}}{\partial \mathbf{p}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\mathbf{M} + \frac{\partial \mathbf{g}^{(i)}}{\partial \mathbf{x}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\mathbf{N} = \mathbf{0}_{m \times k}. \tag{2.31}$$

  if $||\mathbf{g}'(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*); (\mathbf{M}, \mathbf{N}))||_1 < \varepsilon_{\text{tol}}^{\text{sens}}$ **then**
6 Return $\mathbf{N}$.
7 **end if**
8 **end if**
9 **end if**
10 **end for**

In Line 5, $||\cdot||_1$ denotes the induced matrix 1-norm and $\varepsilon_{\text{tol}}^{\text{sens}}$ denotes a user-defined tolerance on the accuracy of the LD-derivatives.

Chapter 3

A nonsmooth model for multistream heat exchanger simulation and design

A new model formulation and solution strategy for the design and simulation of processes involving multistream heat exchangers is presented in this chapter. The approach combines an extension of pinch analysis with an explicit dependence on the heat exchange area in a nonsmooth equation system to create a model that solves for up to three unknown variables in an MHEX. The compact nonsmooth formulation keeps the method tractable even for MHEXs with many process streams, and the advances in automatic generation of generalized derivative information for nonsmooth equations reviewed in the previous chapter mean that the model can be solved efficiently. Several illustrative examples and a case study featuring an offshore liquefied natural gas production concept are presented that highlight the flexibility and strengths of the formulation.

3.1 Introduction

Despite being ubiquitous in cryogenic processes, MHEXs are notoriously difficult to model, simulate and design. Such processes are by nature extremely energy intensive,

and therefore stand to benefit greatly from accurate process optimization. However, without effective and flexible models for heat exchange unit operations, accurate simulation and optimization of these processes cannot be performed. Among those cryogenic processes that utilize MHEXs, LNG production plants are of key importance in the current global energy industry and this application presents a compelling case for the development of general, rigorous, and versatile models for multistream heat exchangers for process design and simulation as described in the previous chapter.

The use of process simulation software is common in the literature involving processes with multistream heat exchangers. Commercial simulators employ proprietary models that generally permit solving for a single unknown variable, afforded by the energy balance, typically taken as the one of the exchanger outlet temperatures. However, as an example, the author's experience using the MHEATX block from Aspen Plus suggests there are no rigorous checks in place to avoid heat exchange between two streams at very similar temperatures, or prevent temperature crossovers. This leads to the somewhat frustrating experience of needing to know parameter values that avoid this problem *a priori*, which then leads to a "guess-and-check" iterative approach to the MHEX simulation.

A more rigorous approach to modeling MHEXs involves the use of a superstructure concept.[48;49] This approach works by deriving a network of two-stream heat exchangers that is equivalent to the multistream heat exchanger. This model can also handle phase changes along the length of the heat exchanger, as long as the phase changes are known to happen *a priori*. The major disadvantage of this methodology is that simulating the MHEX involves the solution of a nonconvex MINLP model, which is extremely challenging to find globally, and would be highly undesirable to use within an outer optimization routine when the simulation is needed as part of a repeated function evaluation.

Another method for MHEX modeling borrows heavily from pinch analysis and the analysis of composite curves. A recent paper by Kamath *et al.*[59] showed how to create a fully equation-oriented model for MHEX by considering the unit operation as a heat exchanger network that requires no external utilities. The authors use the

classic Duran and Grossmann[32] formulation for heat integration in their model, which will be more thoroughly discussed in the following section. They also show how their model can detect and handle phase changes through a disjunctive representation of the phase detection problem, and also that their model is amenable to cubic equations of state governing the thermodynamics. Note that the simulation and/or design of MHEX here again cannot be performed independently of solving a hard optimization problem.

A common theme in both the Kamath *et al.* article and much of the literature on pinch analysis is the use of smoothing approximations to remove the nondifferentiability inherent in the model, such as with the formulation presented by Balakrishna and Biegler.[12] Alternatively, some authors choose to use the disjunctive mixed-integer model of the pinch operator from Grossmann *et al.*[46] Both these approaches introduce small, sensitive, user-set, non-physical parameters that can easily create numerical difficulties and inaccuracies. However, with the recent advent of robust methods for solving nonsmooth equation systems and optimizing nonsmooth functions,[75;77;34;33] such approximations are no longer a necessary evil, and the current work develops a model that handles nondifferentiable functions directly. This relies heavily on the use of tractable, automatic methods for calculating derivative-like information, more information about which is found in the following section.

It is also notable that the MHEX modeling literature rarely makes mention of the dependence of heat exchange area on the performance of the operation. Rather, it is often assumed that an exchanger of sufficient size is simply available, and the size is only calculated following determination of the output stream states, if at all. With the above discussion in mind, this chapter develops a new model and solution procedure for MHEX that solves for up to three unknowns, avoids returning infeasible solutions, doesn't rely on approximations or solving a hard optimization problem, and incorporates information about the available heat exchange area into the procedure. This model and the proposed solution method can be used both in a standalone unit operation model for use in a sequential-modular process simulation, or as the solution algorithm and part of the equation system in an equation-oriented simulation. An

example of the latter functionality is shown in a case study for an offshore LNG production process later in the chapter. As a design tool, this model is intended for use at the flowsheeting stage of design for processes involving one or more MHEX units in order to determine stream states and evaluate feasibility. The detailed equipment design necessary before constructing MHEXs is not considered in this chapter (nor is it in any of the previously cited literature).

3.2 Background

This section reviews the traditional model formation for a two-stream heat exchanger in preparation for the generalization to the mutlistream case. The nonsmooth formulation of the pinch point constraints in heat integration problems that will be used in a modified form in the MHEX model is also introduced.

3.2.1 Standard models for heat exchangers

The development of the new model for MHEX begins by analogy to the well-known model of a countercurrent two-stream heat exchanger as shown in Figure 3-1. Here, a hot stream with (assumed) constant heat capacity flow rate F_{C_p} is cooled from inlet temperature T^{in} to outlet temperature T^{out} by exchanging heat with a cold stream with (assumed) constant heat capacity flow rate f_{C_p}, which in turn is heated from inlet temperature t^{in} to outlet temperature t^{out}.

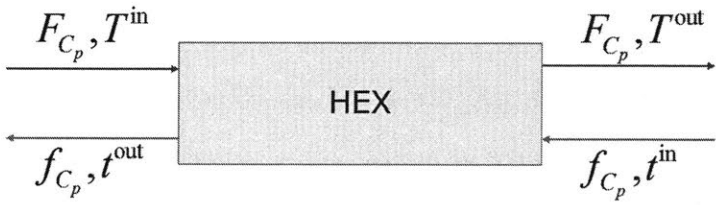


Figure 3-1: Schematic of a countercurrent two-stream heat exchanger.

The standard formulation of the model describing the transfer of heat in the

exchanger shown in Figure 3-1 is given by the following equations:

$$F_{C_p}(T^{\text{in}} - T^{\text{out}}) = f_{C_p}(t^{\text{out}} - t^{\text{in}}), \tag{3.1}$$

$$Q \equiv F_{C_p}(T^{\text{in}} - T^{\text{out}}) = UA\Delta T_{\text{LM}}, \tag{3.2}$$

where Equation (3.1) is the energy balance and Equation (3.2) gives the relationship between the total heat transferred Q and the overall heat transfer coefficient U, the heat transfer area A and the log-mean temperature difference ΔT_{LM} (average driving force of heat transfer). Additionally, there exists a third relationship that governs this system but is generally not explicitly considered:

$$\Delta T_{\text{min}} = \min\{T^{\text{in}} - t^{\text{out}}, T^{\text{out}} - t^{\text{in}}\}, \tag{3.3}$$

with ΔT_{min} referred to as the minimum temperature difference or the minimum approach temperature. Explicitly defining ΔT_{min} in this manner is nonstandard. In practice, this quantity is often considered a parameter set *a priori*, which is then used after Equations (3.1)-(3.2) are solved to judge the feasibility of the result. However, for the current work, it is more useful to think of Equation (3.3) as an additional equation that provides information about the state of the system. Note that $\Delta T_{\text{min}} > 0$ in any physically realizable process design.

Now, consider the case of the multistream heat exchanger model shown in Figure 3-2, in which a set of hot streams, indexed by a set H, exchange heat with a set of cold streams, indexed by a set C. Each hot stream $i \in H$ enters at temperature T_i^{in}, exits at temperature T_i^{out} (with $T_i^{\text{in}} \geq T_i^{\text{out}}$) and has a constant molar heat capacity flowrate $F_{C_p,i}$, which is defined as the product of the molar flowrate F_i and the (assumed constant) molar heat capacity $C_{p,i}$ of the stream at representative conditions. Similarly, each cold stream $j \in C$ enters at temperature t_j^{in}, exits at temperature t_j^{out} (with $t_j^{\text{in}} \leq t_j^{\text{out}}$) and has a constant molar heat capacity flowrate $f_{C_p,j}$. As shown in the figure, assume that the MHEX operates as an ideal countercurrent exchanger, so that all hot streams are codirectional with each other, and all cold streams are

codirectional with each other and oppositely-directed to the hot streams. Beyond this, the internal geometric configuration of the exchanger is not considered in the present work.In addition, it is assumed that the heat transfer consequences of fluid dynamics and the material properties of the MHEX are captured entirely by the value of the overall heat transfer coefficient U. The available heat transfer area in a MHEX is denoted by A, though often the product UA is used instead as the metric for the physical ability of a MHEX to exchange heat to avoid the need for accurate estimation of the value of U at the early stage of process design. For brevity, this quantity UA is referred to as the heat exchanger conductance throughout this thesis.

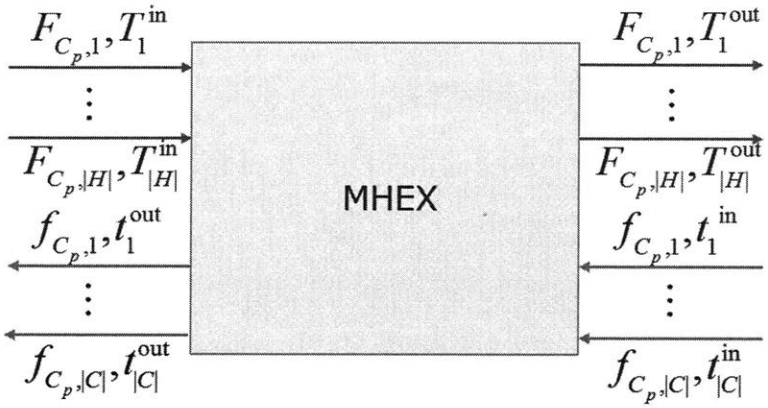


Figure 3-2: Schematic of a multistream heat exchanger with $|H|$ hot streams and $|C|$ cold streams.

Equation (3.1) immediately generalizes to the following energy balance:

$$\sum_{i \in H} F_{C_p,i}(T_i^{\text{in}} - T_i^{\text{out}}) = \sum_{j \in C} f_{C_p,j}(t_j^{\text{out}} - t_j^{\text{in}}). \tag{3.4}$$

However, the MHEX analogues of Equations (3.2) and (3.3) are less obvious. Specifically, the concept of the log-mean temperature difference has no immediate generalization to more than two streams, and the minimum temperature difference could occur at the inlet temperature of any stream. Fortunately, the problem of determining the minimum approach temperature in MHEX can be linked to the well-studied field of pinch analysis for heat exchanger networks.

3.2.2 Pinch analysis for heat integration

Pinch analysis is a methodology for minimizing the energy consumption of a chemical process by optimizing heat recovery between process streams. Interpreted graphically, this translates to shifting the process hot and cold composite curves on a temperature-enthalpy ($T-Q$) graph to maximize their overlap (and hence minimize external utility requirements) while maintaining a certain minimum temperature difference between them (such that a non-negligible thermodynamic driving force exists). The classic formulation of the pinch constraints for simultaneous heat integration and process optimization (due to Duran and Grossmann[32]) is given by:

$$\sum_{i \in H} F_{C_p,i}(T_i^{\text{in}} - T_i^{\text{out}}) - \sum_{j \in C} f_{C_p,j}(t_j^{\text{out}} - t_j^{\text{in}}) + Q_H - Q_C = 0, \tag{3.5}$$

$$AP_C^p - AP_H^p - Q_H \leq 0, \quad \forall p \in \mathcal{P}, \tag{3.6}$$

where $\mathcal{P} = H \cup C$ is the index set of pinch point candidates, Q_H is the heat load of the heating utilities, Q_C is the heat load of the cooling utilities, and AP_H^p and AP_C^p are defined by:

$$AP_H^p = \sum_{i \in H} F_{C_p,i}[\max\{0, T_i^{\text{in}} - T^p\} - \max\{0, T_i^{\text{out}} - T^p\}], \quad \forall p \in \mathcal{P}, \tag{3.7}$$

$$AP_C^p = \sum_{j \in C} f_{C_p,j}[\max\{0, t_j^{\text{out}} - (T^p - \Delta T_{\min})\} - \max\{0, t_j^{\text{in}} - (T^p - \Delta T_{\min})\}], \quad \forall p \in \mathcal{P}. \tag{3.8}$$

The temperatures of the pinch point candidates, T^p, are defined by:

$$T^p = \begin{cases} T_i^{\text{in}}, & \forall p = i \in H, \\ t_j^{\text{in}} + \Delta T_{\min}, & \forall p = j \in C. \end{cases}$$

As noted in Kamath *et al.*,[59] a multistream heat exchanger can be viewed as a special case of a heat exchanger network where the external utilities are not present ($Q_H = 0$ and $Q_C = 0$), which corresponds graphically to there being complete vertical overlap

of the hot and cold composite curves on a $T - Q$ diagram. Under the assumption of maximum heat transfer in a heat exchanger network, it is known that for at least one $p \in \mathcal{P}$, the heat deficit constraint (Equation (3.6)) is active for any feasible set of stream conditions,[32] so the set of inequalities are equivalent to the single equality constraint:

$$\max_{p \in \mathcal{P}} \{AP_C^p - AP_H^p\} = 0. \tag{3.9}$$

Additionally, the value of $AP_C^p - AP_H^p$ is greater than 0 whenever the heating requirements of the cold streams above the assumed pinch p exceed the available heat content of the hot streams above this temperature level, indicating infeasible heat transfer. The left-hand side of (3.9) takes a positive value in this case. Together with the cited result, this implies that the left-hand side of Equation (3.9) is always greater than or equal to zero, with equality whenever all heat exchange is feasible. Additionally, due to the nondifferentiability of the $\max\{0, y\}$ function at $y = 0$, the following smoothing approximation is often used in practice, which is dependent on a user-defined parameter β:[12]

$$\max\{0, f(x)\} \approx \frac{\left(\sqrt{f(x)^2 + \beta^2} + f(x)\right)}{2}. \tag{3.10}$$

However, an attractive feature of the formulation proposed in this chapter is that no such approximation is needed; the nondifferentiable nature of the equation system is instead handled directly.

3.3 Formulation of MHEX minimum approach temperature constraint

To enforce feasible heat transfer in MHEX, a variant of Equation (3.9) is used in the model. However, using Equations (3.4) and (3.9) directly in an equation solving procedure leads to a system that will usually have a nonunique solution; there are many sets of stream conditions that give feasible heat transfer for a multistream heat

exchanger in energy balance. While there are nonsmooth equation solving methods that can tolerate nonuniqueness (one of which is discussed further later), the single solution that is returned becomes dependent on the initial guess provided to the solver.

Instead, the problem is formulated so that its only solution is the one corresponding to the minimum temperature difference between hot and cold streams in the MHEX being exactly $\Delta T_{\min}$. A solution of this form maximizes the heat transferred in the MHEX. Geometrically, this is the problem of attempting to reduce the smallest vertical separation between the hot and cold composite curves to exactly $\Delta T_{\min}$. This is equivalent to the problem of reducing the smallest horizontal distance between the composite curves to zero after applying a temperature shift of $\Delta T_{\min}$ to the cold curve.

To obtain functions that describe the shape of the composite curves, the order of the temperature terms as well as the signs inside the max statements of AP_H^p and AP_C^p are reversed. The resulting expressions therefore account for heat transfer below, rather than above, the pinch. Note that both curves are not defined at every temperature, and so the horizontal distance could be undefined at certain points. This can be solved by creating nonphysical extensions of the curves that extend to the maximum and minimum temperatures existing in the heat exchanger by adding additional terms. With these modifications, the hot and cold composite curve enthalpy values corresponding to each pinch point temperature are defined using the following expressions:

$$\begin{aligned} EBP_H^p = &\sum_{i \in H} F_{C_p,i}[\max\{0, T^p - T_i^{\text{out}}\} - \max\{0, T^p - T_i^{\text{in}}\} \\ &- \max\{0, T^{\min} - T^p\} + \max\{0, T^p - T^{\max}\}], \quad \forall p \in \mathcal{P}, \end{aligned} \tag{3.11}$$

$$\begin{aligned} EBP_C^p = &\sum_{j \in C} f_{C_p,j}[\max\{0, (T^p - \Delta T_{\min}) - t_j^{\text{in}}\} - \max\{0, (T^p - \Delta T_{\min}) - t_j^{\text{out}}\} \\ &+ \max\{0, (T^p - \Delta T_{\min}) - t^{\max}\} - \max\{0, t^{\min} - (T^p - \Delta T_{\min})\}], \quad \forall p \in \mathcal{P}, \end{aligned} \tag{3.12}$$

where $T^{\max}$ is the maximum hot stream inlet temperature, $T^{\min}$ is the minimum hot stream outlet temperature, $t^{\max}$ is the maximum cold stream outlet temperature, and $t^{\min}$ is the minimum cold stream inlet temperature. In each equation, the last two terms correspond to the nonphysical extensions of the curves. Whenever one of these additional terms is nonzero in Equation (3.11), it will be multiplied by the sum of all the hot stream heat capacity flowrates, and similarly for Equation (3.12) with the sum of the cold stream heat capacity flowrates. Figure 3-3 shows the full extended composite curves generated by evaluating Equations (3.11) and (3.12) for an example set of inlet and outlet temperatures for a MHEX. In practice however, the expressions need only be evaluated at the known pinch candidate temperatures. Note that it is entirely possible to transform Equations (3.11) and (3.12) into expressions that preserve the order of terms and signs in Equations (3.7) and (3.8), however, the geometric interpretation for the form of the extensions is less obvious in this formulation.

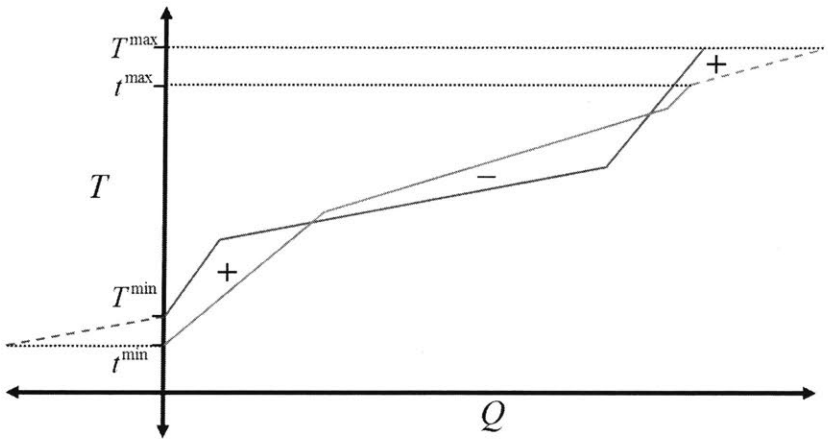


Figure 3-3: Illustration of the extended composite curves generated by the expressions EBP_C^p (blue) and EBP_H^p (red) when $\mathcal{P}$ is expanded to include both inlet and outlet temperatures. The dashed lines indicate the extended portions of the curves added by the last two terms in Equations (3.11) and (3.12). The sign of $EBP_C^p - EBP_H^p$ is indicated in the various regions of the plot.

The smallest horizontal distance between the extended composite curves can now be found by solving:

$$\min_{p \in \mathcal{P}} \{EBP_C^p - EBP_H^p\} = 0. \tag{3.13}$$

It can also be seen that the expression inside the min statement of Equation (3.13) is negative at any pinch temperature where the hot composite curve lies to the right of the shifted cold composite curve, and positive where it lies to the left, as indicated in Figure 3-3. After calculating $EBP_C^p - EBP_H^p, \forall p \in \mathcal{P}$, Equation (3.13) is solved by searching through the finite set $\mathcal{P}$ to find the minimum. This can be done using a PC^1-factorable algorithm in analogy to Algorithm 6.1 from Khan and Barton[63] (first computing $T^{\max}, T^{\min}, t^{\max}, t^{\min}$ and then evaluating Equations (3.11), (3.12), and (3.13) in place of Equations (3.7), (3.8), (3.9)). As such, generalized derivatives can be calculated automatically for Equation (3.13) with respect to the unknown variables. A small example is now presented to examine the method thus far and highlight an additional issue with the solution process.

Example 3.1. Consider the process data in Table 3.1 for two hot streams and two cold streams in a multistream heat exchanger (adapted from a heat exchanger network example in Smith[114]).

Table 3.1: Stream data for Example 3.1.

Stream Name	T^{in} or t^{in} (°C)	T^{out} or t^{out} (°C)	F_{C_p} or f_{C_p} (MW°C^{-1})
H1	250	40	0.15
H2	200	x_1	0.25
C1	20	180	0.20
C2	140	x_2	0.30

Let $\Delta T_{\min} = 10$°C for this example. The equation system that must be solved for the unknown temperatures $x_1 \equiv T_{H2}^{\text{out}}$ and $x_2 \equiv t_{C2}^{\text{out}}$ consists of Equations (3.4) and (3.13). For the solution to be feasible, it must also be that $T_i^{\text{out}} \leq T_i^{\text{in}}, \forall i \in H$ and $t_j^{\text{out}} \geq t_j^{\text{in}}, \forall j \in C$. When modeling a MHEX using this formulation, it is recommended that one unknown correspond to a hot stream outlet and the other correspond to a cold stream outlet, as is the case here. Figure 3-4 shows the residual functions for Equations (3.4) and (3.13) plotted for a range of x_1 and x_2 values around the solution.

The system is clearly nonsmooth, and so Equation (2.24) is applied iteratively to the problem from the initial guess $\mathbf{y}^0 = (80, 230)$, the solution from Smith[114] with utilities present. The infinity norm of the residual functions serves as the basis for

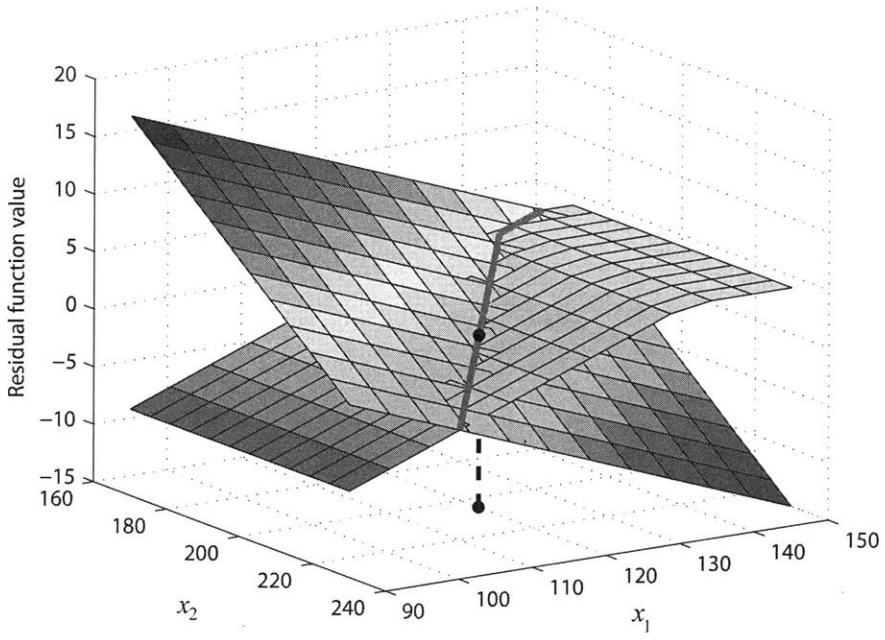


Figure 3-4: Residual of the MHEX model in the vicinity of the solution for Example 3.1. The intersection of the two surfaces is indicated by the solid line. The solution point is shown both on the surfaces and projected onto the x_1-x_2 plane.

termination; here the problem is determined to have converged when the value is less than 10^{-9}. The solution corresponding to the multistream heat exchanger that maximizes heat transfer is found after just a single iteration to be $\hat{\mathbf{y}} = (120, 205)$. The (non-extended) composite curves at the solution are shown in Figure 3-5. Figure 3-6 shows the zero-level contours of Equations (3.4), (3.9), and (3.13) applied to this example over a range of values for $\mathbf{x}$. Note that the solution is unique using the proposed formulation, but that there are infinitely many solutions if Equation (3.9) is used in place of Equation (3.13), since its residual is zero over a large region that partially overlaps with the zero-level contour of the energy balance residual. Note that both of these formulations will have flat regions in the residual plots; such regions are a natural feature of the MHEX problem because unknowns such as inlet and outlet temperatures will only influence the minimum distance between the composite curves over limited ranges of values. However, the extensions in the new expressions have the effect of moving this flat region to a nonzero residual function value, which eliminates

the nonuniqueness of the solution in many cases.

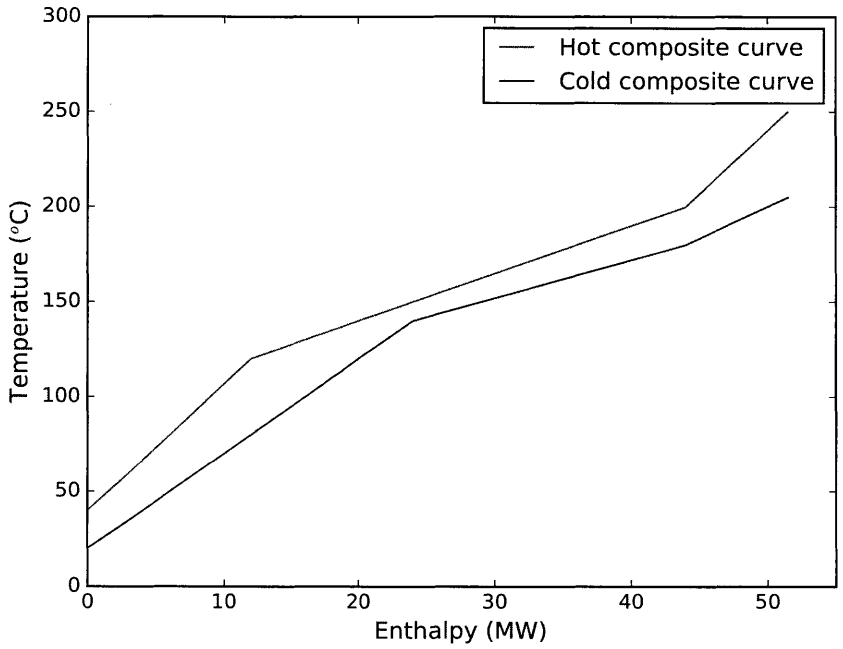


Figure 3-5: The composite curves corresponding to the multistream heat exchanger simulated in Example 3.1.

Finally, note that if a nonsmooth Newton method is started from any point with $x_1 > 130$, where the surface corresponding to Equation (3.13) is flat, the method fails to solve the Newton step for the next iterate because the generalized derivative is a singleton corresponding to the standard Jacobian matrix, which is singular.

As a result of the unavoidable presence of singular Jacobians, the nonsmooth Newton method based on Equation (2.24) cannot solve the problem starting from any possible initial guess. Instead, the LP-Newton method (Equation (2.26)) can be used, which has the added benefit of allowing bounds to be enforced on the solution. An example of a useful bound that could be added in the case that $\Delta T_{\min}$ is an unknown is $\Delta T_{\min} \geq \Delta T_{\text{tol}}$, where $\Delta T_{\text{tol}} > 0$ is the smallest approach temperature that would be tolerated in operation. The bounds on the unknown temperatures should also be enforced, for instance, if some y_k corresponds to a hot stream outlet temperature T_i^{out} for some i, then the constraint $y_k \leq T_i^{\text{in}}$ is added. This prevents

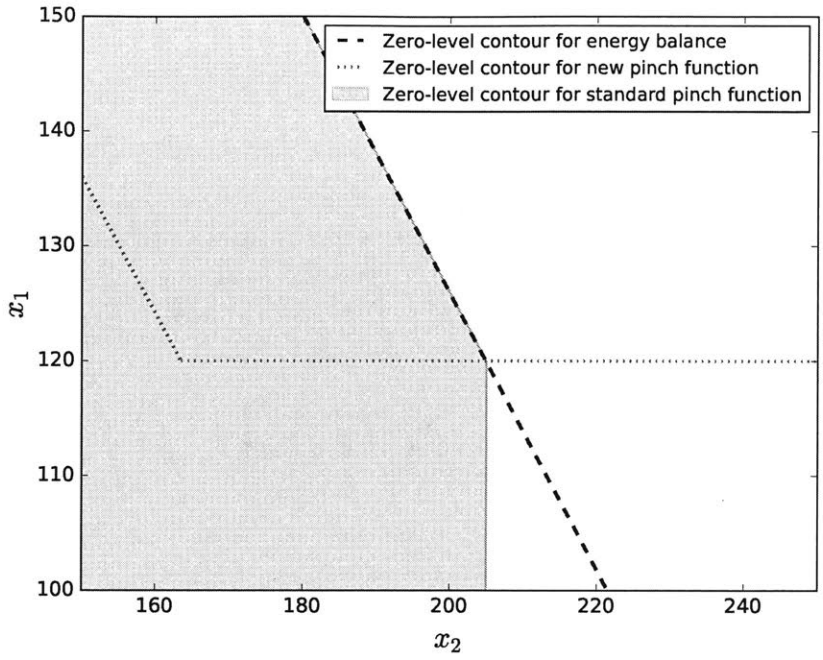


Figure 3-6: Zero-level contours for Equation (3.4) (black dashed line), (3.9) (purple shaded region), and (3.13) (red dotted line) for the problem in Example 3.1 over a range of values for the unknown temperatures.

the hot stream from gaining heat as if it were a cold stream, which would invalidate the model. Analogous constraints can be imposed for variables corresponding to inlet or cold stream temperatures. Using the LP-Newton method on the problem given in Example 3.1 gives the correct solution regardless of whether or not the Jacobian is singular at the initial guess, although the method is no longer exact for linear equations as the previous method was, so the number of iterations needed to converge is greater (around 40 iterations were needed for all initial guesses tested).

3.4 Formulation of MHEX area constraint

Consider now an analogue of Equation (3.2) for the MHEX case. Define K as the index set for the points at which the composite curves are nondifferentiable (kinks), as well as their endpoints, then for $k \in K$, let Q^k denote the enthalpy value at this kink or endpoint, which could occur on either the hot or cold composite curve. Now

suppose a list of triples of the form (Q^k, T^k, t^k), ordered by nondecreasing Q^k value has been calculated. For each of these triples, T^k is the temperature on the hot composite curve at Q^k, and t^k is the temperature on the cold composite curve at Q^k. An adjacent pair of triples in the list demarcates an interval of the composite curves in which part of MHEX can be modeled as a two-stream heat exchanger. A simple expression for the total required heat transfer area of a network of two-stream heat exchangers that can be applied to MHEX is as follows:[114;52]

$$UA = \sum_{\substack{k \in K \\ k \neq |K|}} \frac{\Delta Q^k}{\Delta T^k_{\mathrm{LM}}}, \tag{3.14}$$

where $\Delta Q^k = Q^{k+1} - Q^k$ is the width of enthalpy interval k, $|K| = 2(|H| + |C|)$, and ΔT^k_{LM} is a modified version of the log-mean temperature difference across this same enthalpy interval that is defined later in this section. Figure 3-7 illustrates the definitions of Q^k and ΔQ^k for a sample set of composite curves. Note that for this work, it is assumed that there can be no transverse heat transfer between adjacent enthalpy intervals.

In standard practice, heat transfer area is calculated after heat integration is performed, and the integrated composite curves are used to define the enthalpy intervals. However, in this work, it is desired that Equation (3.14) be included in the system of equations being solved simultaneously, so that an additional unknown is available for simulation pruposes and so that the area may be specified as part of the problem input. This creates some difficulty, as the full sorted list of (Q^k, T^k, t^k) triples must be calculated at each iteration, starting from an incomplete list of just the inlet and outlet temperatures that are arranged in an arbitrary order. Furthermore, this must be done using an algorithm for which valid generalized derivatives can be calculated.

The proposed procedure begin by arbitrarily ordering the set of inlet and outlet temperatures into a list indexed by a set L of size $|K|$. Then for each $l \in L$, the

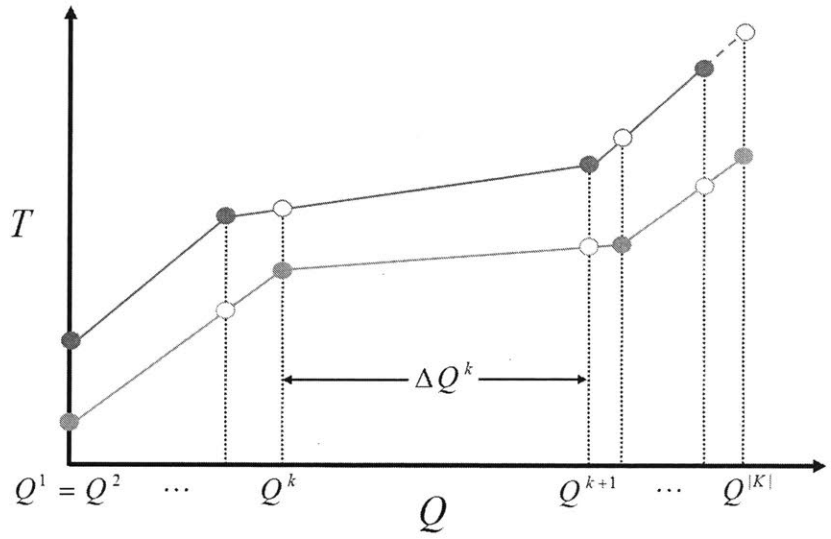


Figure 3-7: Example of the enthalpy intervals used in the calculation of Equation (3.14). Solid circles represent the temperatures that are part of the data for the problem, hollow circles represent those that must be calculated. In the case where the endpoints of the composite curves do not align, one curve must be extrapolated out to the end of the other (dashed line).

pre-sort enthalpy P^l is calculated using one of the following equations:

$$P^l = \sum_{i \in H} F_{C_p,i} \left(\max\{0, T^l - T_i^{\text{out}}\} - \max\{0, T^l - T_i^{\text{in}}\} \right), \quad T^l \in \{T_i^{\text{in/out}} : i \in H\}, \tag{3.15}$$

$$P^l = \sum_{j \in C} f_{C_p,j} \left(\max\{0, t^l - t_j^{\text{in}}\} - \max\{0, t^l - t_j^{\text{out}}\} \right), \quad t^l \in \{t_j^{\text{in/out}} : j \in C\}. \tag{3.16}$$

This provides all the enthalpy values needed to calculate the heat transfer area, but they are out of order and not yet associated with the corresponding pair of temperatures on the composite curves. To correct the order, a list of triples is created, each of the form (P^l, T^l, t^l), in which each P^l is associated with either the hot or cold temperature used in its calculation (and as such one of the temperature entries is currently unknown in each triple). This list is then sorted into nondecreasing order based on enthalpy value to set up the intervals for Equation (3.14) correctly. If the sort is performed using a naïve bubble sort, implemented as shown in Algortihm 3.1, then the only operations involve taking the max or min of two values. The loops are

run over the entire length of the list to ensure that the same number of operations are performed for a given input size, regardless of how well-sorted the input data is. These conditions imply that the sort algorithm has an PC^1-factorable representation and therefore has well-defined LD-derivatives that can be calculated automatically. Therefore, the sorting algorithm may be viewed as a nonsmooth function that maps the unsorted input to the sorted output. Since LD-derivatives obey a sharp chain rule (Theorem 2.1), evaluating the LD-derivatives of the sorting function in the directions set by the LD-derivatives of the inner function that maps $\mathbf{x}$ to the list of P^l values, as given in Equations (3.15) and (3.16), valid LD-derivatives of the composite function in the original directions $\mathbf{I}$ are obtained. Example 3.2 demonstrates this initial part of the procedure.

Algorithm 3.1: The naïve bubble sort algorithm.

Input : An unsorted list, A with entries $A[1], \ldots, A[m]$.
Output: The list A, with the m entries sorted in order of increasing value.

```
for i ← 1 to m do
    for j ← 1 to m − 1 do
        a ← min(A[j], A[j + 1])
        b ← max(A[j], A[j + 1])
        A[j] ← a
        A[j + 1] ← b
    end for
end for
return A
```

Example 3.2. Consider the stream data from the previous example at the solution point $\hat{\mathbf{x}}$. First, the enthalpy values and their LD-derivatives at $\hat{\mathbf{x}}$ in the directions $\mathbf{I}$ are computed. This is performed in the `C++` programming language using the implementation of automatic LD-derivative evaluation described in Chapter 2 from Khan and Barton.[64] Table 3.2 contains these values for this example.

This data is arranged into triples of the form (P^l, T^l, t^l) and the bubble sort algorithm is applied to sort by nondecreasing enthalpy value. The result of this operation is the list of correctly ordered but incomplete triples as shown in Table 3.3.

Table 3.2: Temperature-enthalpy data for the streams in Example 3.1.

l	Temperature ID	Temperature (°C)	$P^l(\hat{\mathbf{x}})$ (MW)	$P^{l\prime}(\hat{\mathbf{x}};\mathbf{I})$
1	T_1^{in}	250	51.5	[-0.25 0]
2	T_1^{out}	40	0.0	[0 0]
3	T_2^{in}	200	44.0	[-0.25 0]
4	T_2^{out}	120	12.0	[0.15 0]
5	t_1^{in}	20	0.0	[0 0]
6	t_1^{out}	180	44.0	[0 0]
7	t_2^{in}	140	24.0	[0 0]
8	t_2^{out}	205	51.5	[0 0.3]

Table 3.3: Results of the bubble sort operation on the triples generated from Table 3.2. The symbol "–" represents those temperatures that have yet to be calculated.

k	$(Q^k(\hat{\mathbf{x}}), T^k, t^k)$	$Q^{k\prime}(\hat{\mathbf{x}};\mathbf{I})$
1	(0, 40, –)	[0 0]
2	(0, –, 20)	[0 0]
3	(12, 120, –)	[0.15 0]
4	(24, –, 140)	[0 0]
5	(44, 200, –)	[-0.25 0]
6	(44, – 180)	[0 0]
7	(51.5, 250, –)	[-0.25 0]
8	(51.5, –, 205)	[0 0.3]

Note that here, the LD-derivatives associated with each P^l prior to the sort remain associated with the corresponding variable in the sorted order. In general, however, this need not be the case. For instance, for $l = 1$, if $P^{l\prime}(\hat{\mathbf{x}};\mathbf{I}) = [0.25\ 0]$, then following the sort, the ordering of triples is identical, but $Q^{7\prime}(\hat{\mathbf{x}};\mathbf{I}) = [0\ 0.3]$, and $Q^{8\prime}(\hat{\mathbf{x}};\mathbf{I}) = [0.25\ 0]$.

Now the missing temperature in each of the triples must be calculated. Given Q^k, if T^k is unknown, (3.17) is solved for T^k. Similarly, if t^k is unknown, then (3.18) is solved for t^k.

$$Q^k - \sum_{i \in H} F_{C_p,i} \left(\max\{0, T^k - T_i^{\text{out}}\} - \max\{0, T^k - T_i^{\text{in}}\} \right) = 0, \tag{3.17}$$

$$Q^k - \sum_{j \in C} f_{C_p,j} \left(\max\{0, t^k - t_j^{\text{in}}\} - \max\{0, t^k - t_j^{\text{out}}\} \right) = 0. \tag{3.18}$$

Unfortunately, there seems to be no way to solve either of these equations for the

unknown temperature without the use of selection statements, and so any such algorithm would not have a factorable representation. Therefore, since it cannot be guaranteed that valid B-subdifferential elements would be obtained by differentiating the solution algorithm, a different method must be used.

If no restrictions are placed on the algorithm, the *value* of the unknown temperature, denoted by $\hat{T}^k$ or $\hat{t}^k$, that satisfies Equation (3.17) or (3.18) can be determined by a simple search and interpolation procedure over the piecewise affine segments of the relevant composite curve. As shown in Figure 3-7, if the composite curves do not exactly align, the shorter curve can be extrapolated to the end of the longer one to find the corresponding temperature.

The LD-derivatives of this temperature at the value of the vector of unknowns, $\hat{\mathbf{x}}$, in the original directions $\mathbf{I}$, e.g., $T^{k\prime}(\hat{\mathbf{x}};\mathbf{I})$ or $t^{k\prime}(\hat{\mathbf{x}};\mathbf{I})$ must then be calculated independently. However, as this cannot be done directly, instead regard Equation (3.17) (and analogously Equation (3.18)) as $h(T^k,\mathbf{x},Q^k(\mathbf{x}))=0$, for the function $h:\mathbb{R}\times\mathbb{R}^{n_y}\times\mathbb{R}\to\mathbb{R}$ defined by the left-hand side of Equation (3.17). Therefore, there exists an implicit function $\eta:\mathbb{R}^{n_y}\times\mathbb{R}\to\mathbb{R}$ defined by the equation $h(T^k,\mathbf{x},Q^k(\mathbf{x}))=0$, such that $T^k(\mathbf{x})=\eta(\mathbf{x},Q^k(\mathbf{x}))$, for all $\mathbf{x}$ in a neighborhood of $\hat{\mathbf{x}}$. Note that the implicit function here depends on Q^k, which is (possibly) a function of the unknowns $\mathbf{x}$, so the chain rule is applied to obtain the desired LD-derivatives $T^{k\prime}(\hat{\mathbf{x}};\mathbf{I})=\eta'((\hat{\mathbf{x}},Q^k(\hat{\mathbf{x}}));\mathbf{M})$, where the directions $\mathbf{M}$ are determined by the LD-derivatives of the previous operations that calculated $Q^k(\hat{\mathbf{x}})$. To calculate the LD-derivatives of the implicitly-defined function $\eta'((\hat{\mathbf{x}},Q^k(\hat{\mathbf{x}}));\mathbf{M})$ correctly, Algorithm 2.4 can be invoked to searching through the essentially active selection functions $h^{(i)}$ that comprise the piecewise differentiable function h.

The computational complexity of this algorithm applied naïvely to this problem is exponential in the number of hot and cold streams because of the presence of $4^{|H|}$ selection functions due to the binary max terms in the definition of h, where each term has two possible differentiable functional forms that could be active at $h(\hat{T}^k,\hat{\mathbf{x}},Q^k(\hat{\mathbf{x}}))$. However, this unfavorable scaling can be mitigated by using the calculated value of the unknown temperature itself to reduce the number of possible

active selection functions. Once the value and the corresponding LD-derivative of the unknown temperature have been calculated, they are copied into the appropriate object type and used in the remainder of the procedure. Example 3.3 demonstrates the application of Algorithm 2.4 to calculate the LD-derivatives of the implicit function.

Example 3.3. Consider the triples from the previous example. A kink in the cold composite curve is located at $\hat{Q}^k = 44$ MW, corresponding to a cold stream temperature value of $\hat{t}^k = 180°\text{C}$. Notice that here, the hot temperature value is actually already known at this enthalpy value, so $\hat{T}^k = 200°\text{C}$ (if this were not the case, then an interpolation on the relevant affine segment would be used to find $\hat{T}^k$). Note that the LD-derivatives (and regular derivatives) of Q^k at this $\hat{\mathbf{x}}$ are all zero, since the value of the cold stream outlet variable $\hat{y}_2 = 205°\text{C}$ is strictly greater than all other cold stream temperatures, and so Equation (3.18) shows that only the zero selection function of the max term involving $x_2 \equiv t_2^{\text{out}}$ will be active when $\hat{t} = 180°\text{C}$. For notational simplicity, define $\mathbf{z}(\mathbf{y}) = (\mathbf{x},\, Q^k(\mathbf{x}))$, and then the direction matrix $\mathbf{M}$ is given by:

$$\mathbf{M} = \begin{bmatrix} \mathbf{I} \\ Q^{k\prime}(\hat{\mathbf{x}}; \mathbf{I}) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Then Equation (3.17) can be written for this example as follows:

$$\begin{aligned} h(T^k, \mathbf{z}) = z_3 - & F_{C_p,1} \left(\max\{0, T^k - T_1^{\text{out}}\} - \max\{0, T^k - T_1^{\text{in}}\} \right) \\ & - F_{C_p,2} \left(\max\{0, T^k - z_1\} - \max\{0, T^k - T_2^{\text{in}}\} \right) = 0. \end{aligned}$$

Naïvely, 16 different continuous selection functions would need to be considered here to account for the four max terms; however, note that three of the max terms will only have a single active selection function, since $\hat{T}^k > T_1^{\text{out}}$, $\hat{T}^k < T_1^{\text{in}}$, and $\hat{T}^k > \hat{z}_1$. In the final term however, $\hat{T}^k = T_2^{\text{in}}$, so there are two possible differentiable selection functions to consider: $h_i, h_2 \in I_h^{\text{ess}}(\hat{T}^k, \hat{\mathbf{z}})$:

$$h^{(1)}(T^k, \mathbf{z}) = z_3 - F_{C_p,1} \left(T^k - T_1^{\text{out}} \right) - F_{C_p,2} \left(T^k - z_1 \right),$$

$$h^{(2)}(T^k, \mathbf{z}) = z_3 - F_{C_p,1}\left(T^k - T_1^{\text{out}}\right) - F_{C_p,2}\left(T^k - z_1\right) + F_{C_p,2}\left(T^k - T_2^{\text{in}}\right).$$

Evaluating the required derivatives gives:

$$\begin{aligned}
&\frac{\partial h^{(1)}}{\partial \mathbf{z}}(\hat{T}^k, \hat{\mathbf{z}}) = [F_{C_p,2} \;\; 0 \;\; 1], && \frac{\partial h^{(2)}}{\partial \mathbf{z}}(\hat{T}^k, \hat{\mathbf{z}}) = [F_{C_p,2} \;\; 0 \;\; 1],\\
&\frac{\partial h^{(1)}}{\partial T^k}(\hat{T}^k, \hat{\mathbf{z}}) = -(F_{C_p,1} + F_{C_p,2}), && \frac{\partial h^{(2)}}{\partial T^k}(\hat{T}^k, \hat{\mathbf{z}}) = -F_{C_p,1}.
\end{aligned}$$

In practical implementation of this algorithm, these expressions can all be calculated using automatic differentiation. Solving the linear system $\frac{\partial h^{(1)}}{\partial T^k}(\hat{T}^k, \hat{\mathbf{z}})\mathbf{N}^{(1)} = -\frac{\partial h^{(1)}}{\partial \mathbf{z}}(\hat{T}^k, \hat{\mathbf{z}})\mathbf{M}$ for $\mathbf{N}^{(1)}$ then yields $\mathbf{N}^{(1)} = [\frac{F_{C_p,2}}{F_{C_p,1}+F_{C_p,2}} \;\; 0] = [0.625 \;\; 0]$. However, evaluation of the LD-derivative of h yields $h'((\hat{T}^k, \hat{\mathbf{z}}); (\mathbf{N}^{(1)}, \mathbf{M})) = [0.15625 \;\; 0]$, indicating that $\eta'(\hat{\mathbf{z}}; \mathbf{M}) \neq \mathbf{N}^{(1)}$. Solving the linear system involving $h^{(2)}$ gives $\mathbf{N}^{(2)} = [\frac{F_{C_p,2}}{F_{C_p,1}} \;\; 0] = [\frac{5}{3} \;\; 0]$. In this case, evaluation of the LD-derivative of h yields $h'((\hat{T}^k, \hat{\mathbf{z}}); (\mathbf{N}^{(2)}, \mathbf{M})) = [0 \;\; 0]$, and so $\eta'(\hat{\mathbf{z}}; \mathbf{M}) = \mathbf{N}^{(2)}$ Finally, by the chain rule for LD-derivatives:

$$T^{k\prime}(\hat{\mathbf{x}}; \mathbf{I}) = \eta'\left(\left(\hat{\mathbf{x}}, Q^k(\hat{\mathbf{x}})\right); \begin{bmatrix} \mathbf{I} \\ Q^{k\prime}(\hat{\mathbf{x}}; \mathbf{I}) \end{bmatrix}\right) = \eta'(\mathbf{z}(\hat{\mathbf{x}}); \mathbf{M}) = \mathbf{N}^{(2)}.$$

Applying this procedure for all Q^k completes the sorted list of triples at all nondifferentiable points on the composite curves. Now, the log-mean temperature difference between the endpoints of each enthalpy interval must be calculated. To do this, it is necessary to slightly alter the standard definition of the log-mean temperature difference so that evaluating the function never results in undefined behavior. The definition used for this work is as follows (from Zavala-Río *et al.*[150]):

$$\Delta T_{\text{LM}}^k(\Delta T^k, \Delta T^{k+1}) = \begin{cases} \frac{1}{2}(\Delta T^k + \Delta T^{k+1}) & \text{if } \Delta T^k = \Delta T^{k+1}, \\ \frac{\Delta T^{k+1} - \Delta T^k}{\ln(\Delta T^{k+1}) - \ln(\Delta T^k)} & \text{otherwise,} \end{cases} \tag{3.19}$$

where $\Delta T^k = \max\{\Delta T_{\min}, T^k - t^k\}$ is the temperature difference at the start of enthalpy interval k, and $\Delta T^{k+1} = \max\{\Delta T_{\min}, T^{k+1} - t^{k+1}\}$ is the temperature dif-

ference at the end of the interval. The max of this quantity and $\Delta T_{\min}$ is taken here so that this calculation of the temperature driving force is only based on feasible heat transfer. The if statement in the definition of the log-mean temperature difference is necessary to make the calculation defined for all possible inputs $\Delta T^k > 0$ and $\Delta T^{k+1} > 0$. Fortunately, since this function is continuously differentiable on the positive quadrant of $\mathbb{R}^2$,[150] the if statement in Equation (3.19) does not introduce any complications since the standard rules for automatic differentiation will produce correct derivatives. Note that the function obtained by composing the function $T^k(\mathbf{x})$ (that is itself already a composition of several nonsmooth functions) with the max functions defining ΔT^k and ΔT^{k+1}, and then the log-mean temperature difference function remains a nonsmooth function of the unknowns $\mathbf{x}$. As before, correct LD-derivatives of this composite function are computed through application of the chain rule for LD-derivatives.

In summary, the set of equations describing the multistream heat exchanger now consists of the following:

$$\sum_{i \in H} F_{C_p,i}(T_i^{\text{in}} - T_i^{\text{out}}) - \sum_{j \in C} f_{C_p,j}(t_j^{\text{out}} - t_j^{\text{in}}) = 0, \tag{3.4}$$

$$\min_{p \in \mathcal{P}} \{EBP_H^p - EBP_C^p\} = 0, \tag{3.13}$$

$$UA - \sum_{\substack{k \in K \\ k \neq |K|}} \frac{\Delta Q^k}{\Delta T_{\text{LM}}^k} = 0. \tag{3.14}$$

This is a nonsmooth equation system involving three equations in three unknowns that can solved using the LP-Newton method discussed previously. Additionally, it should be enforced that $T_i^{\text{out}} \leq T_i^{\text{in}}, \forall i \in H$ and $t_j^{\text{out}} \geq t_j^{\text{in}}, \forall j \in C$, which is most easily enforced by setting polyhedral bound constraints in the LP-Newton method. Note also that since it is necessary to calculate the temperature difference between the composite curves at each Q^k in order to evaluate Equation (3.14), one can use the following in place of Equation (3.13) in the previous equation system:

$$\min_{k \in K} \{T^k - (t^k + \Delta T_{\min})\} = 0. \tag{3.20}$$

This is a more expensive function to evaluate than Equation (3.13), although since most of the computational work must be done to evaluate Equation (3.14), using it in the system of three equations will be slightly cheaper computationally overall. The authors' testing has not been conclusive as to which of these two possible formulations results in faster convergence to the solution; it appears largely dependent on the problem at hand and the initial guess provided.

A larger illustrative example that makes use of the full multistream heat exchanger model is now given.

Example 3.4. Consider the following process data in Table 3.4 for five hot streams and five cold streams in a multistream heat exchanger (adapted from a heat exchanger network example in Chakraborty and Ghosh[23]). This MHEX is simulated under four different conditions to highlight the flexibility of the new model. In all cases, the CPLEX v12.5 callable library[55] is used to solve the linear program at each iteration. The problem is considered converged to a solution when the infinity norm of the residual functions is less than 10^{-9}.

Table 3.4: Stream data for Example 3.4.

Stream Name	T^{in} or t^{in} (°C)	T^{out} or t^{out} (°C)	F_{C_p} or f_{C_p} (kW°C^{-1})
H1	160.0	93.3	8.8
H2	248.9	137.8	10.6
H3	226.7	65.6	14.8
H4	271.1	148.9	12.6
H5	198.9	65.6	17.7
C1	60.0	160.0	7.6
C2	115.6	221.7	6.1
C3	37.8	221.1	8.4
C4	82.2	176.7	17.3
C5	93.3	204.4	13.9

Case I. For a first example, let $x_1 \equiv T_{H5}^{\text{out}}$, $x_2 \equiv t_{C5}^{\text{out}}$, and $x_3 \equiv UA$. Let all other temperatures be fixed at their values in Table 3.4 and let $\Delta T_{\min} = 10$°C. Solving the system of three equations (Equations (3.4), (3.13), and (3.14)) using the LP-Newton method yields $x_1 = 131.3$°C, $x_2 = 259.0$°C, and $x_3 = 314.7$ kW/K after 125 iterations starting from the solution with utilities present given by Chakraborty and Ghosh[23]

and $x_3^0 = 200$ kW/K. The composite curves for the multistream heat exchanger in this case are shown in Figure 3-8(a). Observe that the composite curves resemble those of a heat exchanger involving streams of nonconstant heat capacity, despite actually consisting of a number of affine segments. The same numerical result is obtained by applying only Equations (3.4) and (3.13) to resolve the composite curves and then calculating the overall conductance afterwards.

Case II. The strength of the new approach is more apparent when UA is specified, rather than calculated. Given a conductance (or area) value, a typical problem is to calculate $\Delta T_{\min}$ in the exchanger, which generally leads to better design than when $\Delta T_{\min}$ is specified[57] (as it was in Case I). For a second example, assume that an old heat exchanger with a $UA = 400$ kW/K is being re-purposed, let $x_1 \equiv T_{H5}^{\text{out}}$, $x_2 \equiv t_{C5}^{\text{out}}$ as before, and now let $x_3 \equiv \Delta T_{\min}$. Starting from the conditions at the solution of the previous case, the new solution is found in 13 iterations of the LP-Newton method with $x_1 = 126.2°$C, $x_2 = 265.5°$C, and $x_3 = 3.5°$C. The composite curves for the multistream heat exchanger in this case are shown in Figure 3-8(b). Here, the curves are more closely pinched together than in Case I as a result of the increased heat transfer potential afforded by the higher conductance value.

Case III. Now consider using the same variables as in Case II, but using a heat exchanger instead with $UA = 200$ kW/K. The solution is found in 49 iterations of the LP-Newton method with $x_1 = 162.4°$C, $x_2 = 219.4°$C, and $x_3 = 16.7°$C starting from the conditions at the solution of Case I. The composite curves for the multistream heat exchanger in this case are shown in Figure 3-8(c). In this case, the location of the pinch shifts to a significantly lower temperature than in Cases I and II. Additionally, T_{H5}^{out} increases and t_{C5}^{out} decreases substantially in response to the decrease in conductance, resulting in a larger temperature gap between the composite curves than in the previous cases.

Case IV. Finally, consider the problem where $x_1 \equiv T_{H5}^{\text{out}}$, $x_2 \equiv t_{C5}^{\text{out}}$, and x_3 is a third temperature, say $x_3 \equiv T_{H1}^{\text{out}}$, with $\Delta T_{\min} = 10°$C. A value of UA must also be specified for this MHEX, though note that not all values of UA will lead to a feasible solution due to the highly constrained nature of the problem. If $UA = 340$ kW/K, then the

solution is found in 26 iterations with $x_1 = 140.1°C$, $x_2 = 259.0°C$, and $x_3 = 75.6°C$ starting from the conditions at the solution of Case I. Increasing UA further, say to 345 kW/K results in an infeasible problem. Similarly, the area can be reduced down to $UA = 307$ kW/K to obtain a solution with $x_1 = 121.0°C$, $x_2 = 259.0°C$, and $x_3 = 114.0°C$ in 31 iterations starting from the conditions at the solution of Case I, but decreasing the conductance further again results in an infeasible problem. The composite curves for this case are shown in Figure 3-8(d) for $UA = 340$ kW/K and Figure 3-8(e) for $UA = 307$ kW/K. In both scenarios, the shape of the lower part of the hot composite curve shifts in response to the change in UA, with the pinch point remaining at the same location as in Case I.

The new nonsmooth model formulation for MHEXs can also be used as part of a rigorous process design strategy in a way that other existing models cannot. Current simulation-based models are over-constrained in the sense that they allow for only one unknown that can be adjusted to meet two requirements: the energy balance and the second law requirement that heat flows from hot streams to cold streams. In many such models, the adjustable temperature is set by the energy balance, so there is nothing left to adjust to satisfy the second law requirement; it is either satisfied or not based on the values given for the degrees of freedom in the problem, leading to temperature crossovers and other nonphysical solutions. The initial model proposed in the previous section consisting of Equations (3.4) and (3.13) addresses this issue by enabling the user to specify $\Delta T_{\min}$, thus freeing up two adjustable temperatures to meet the two requirements. It is much easier to specify degrees of freedom that have a feasible solution with this formulation. As noted in Jensen and Skogestad,[57] specifying $\Delta T_{\min}$ is somewhat artificial and can even be counterproductive; it is better thought of as an output of the model, not an input. The three equation model presented here consisting of Equations (3.4), (3.13) or (3.20), and (3.14) addresses this issue by enabling two temperatures and $\Delta T_{\min}$ to be adjustable. However, in order to make this work, an area (or conductance) must be specified as a degree of freedom. Again, it is much easier to specify degrees of freedom that have a feasible solution for this formulation, which enables to user to adjust the area to get desirable temperature

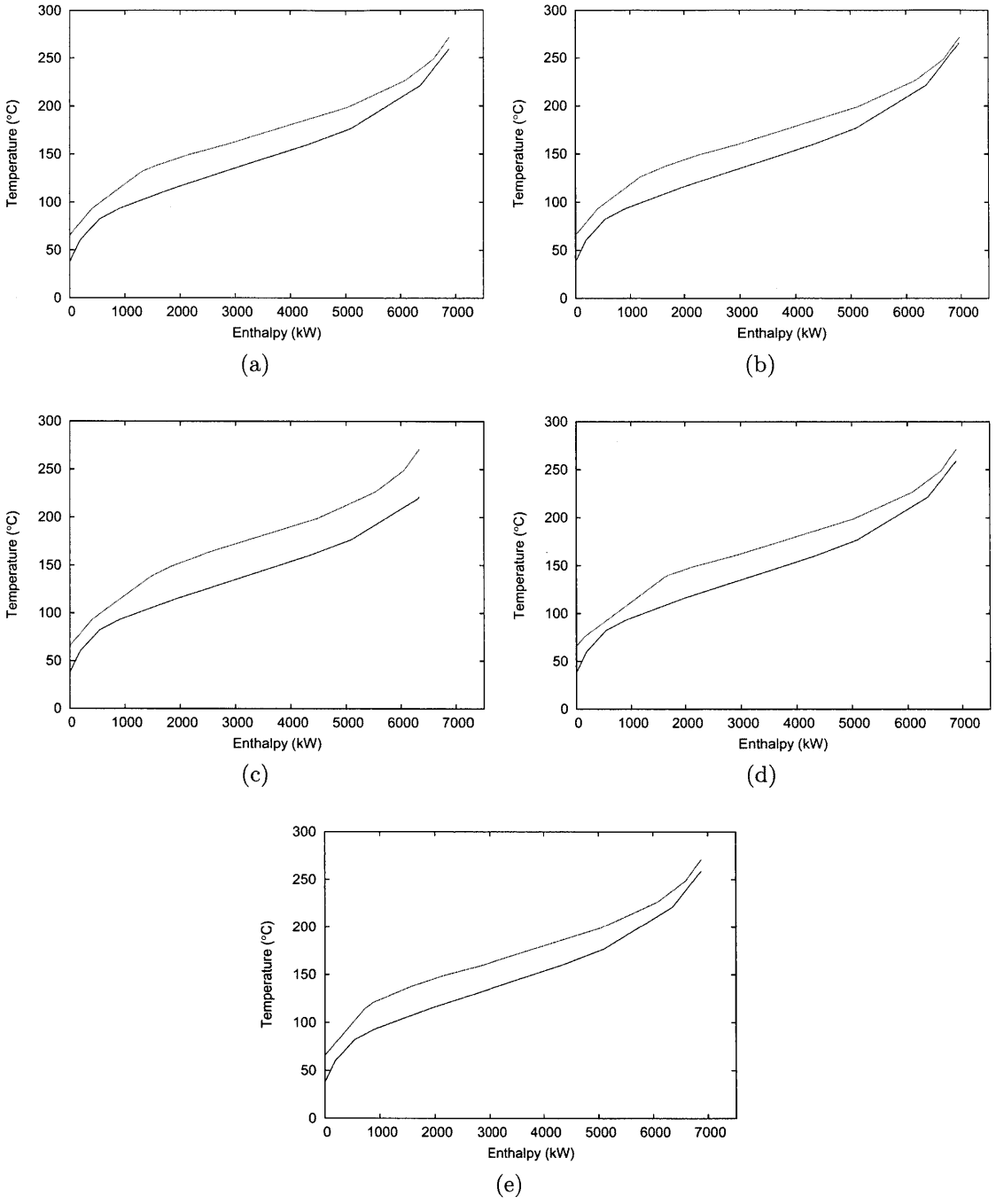


Figure 3-8: Composite curves of multistream heat exchangers simulated under conditions of (a) Case I, (b) Case II, (c) Case III, (d) Case IV with $UA = 340$ kW/K, (e) Case IV with $UA = 307$ kW/K in Example 3.4.

profiles in the MHEX. However, at the early stages of system design it may not be clear what is a reasonable area to specify. Therefore, the two equation model is very

useful at this preliminary stage because the user can specify a reasonable $\Delta T_{\min}$, obtain valid composite curves, and then calculate the corresponding MHEX area (or equivalently use the three equation model with the area as one of the unknowns, as in Case I of Example 3.4). Once this area is known, the user can use the three equation model with other quantities as unknowns while adjusting the area value around this base value.

3.5 LNG process case study

An application of the proposed method to the simulation of a complex LNG production process featuring compression and expansion of process streams as described in Wechsung *et al.*[141] is now presented.

Example 3.5. A flowsheet for an offshore LNG process is shown in Figure 3-9. Prior to considering heat integration, many of the physical process streams must be split into multiple independent substreams, each with constant heat capacity, to better model the true cooling curves. As in Wechsung *et al.*,[141] the natural gas process stream (NG-x) is split into three separate hot streams (H1-H3), the cold carbon dioxide stream (CO_2-x) is split into two separate cold streams (C1-C2), and the cold nitrogen stream (N_2-x) is split into three cold streams (C3-C5). The remaining process streams are not divided into substreams, resulting in a total of 4 hot streams and 7 cold streams that are considered from the perspective of heat integration in the model. HX-100 handles 3 hot streams and 6 cold streams while HX-101 handles 1 hot stream and 3 cold streams, as detailed in the first three columns of Table C.1.

This problem was originally designed as an optimization problem, so there are too many unknown variables in the formulation from Wechsung *et al.*[141] to simulate the process. Therefore, some of the variables (namely all of the pressures and flowrates, along with some of the temperatures) are fixed to their values from the solution given in Wechsung *et al.*[141] that involved no external utilities. Table C.1 gives the values of the parameters used in the model as well as the quantities left as unknown variables for this study. There are a total of nine unknowns in the simulation problem:

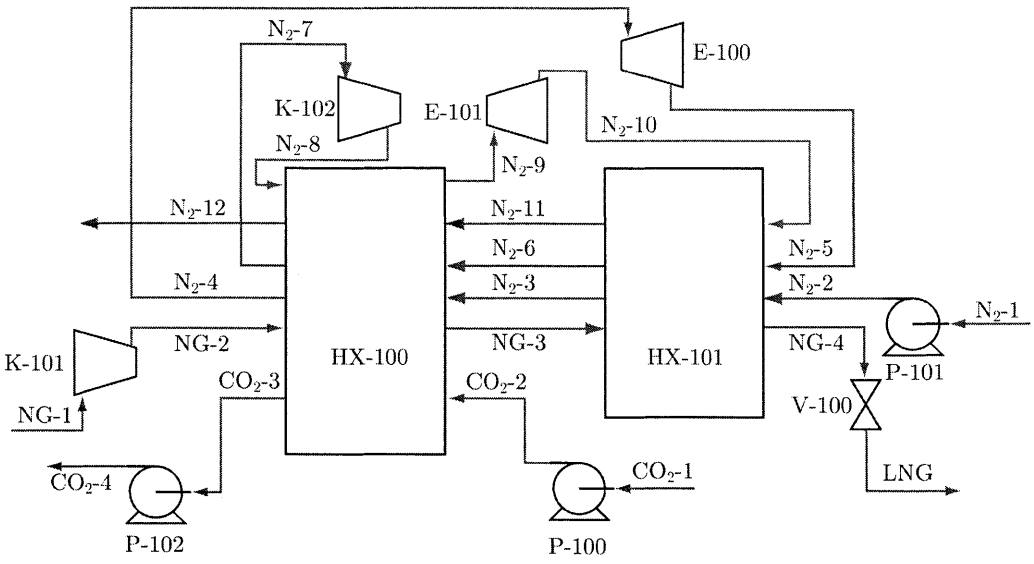


Figure 3-9: Flowsheet for the liquefaction process in Example 3.5 (from Wechsung *et al.*[141])

seven temperatures, of which two are solved for by each heat exchanger model and three are solved for by the equations describing the three compression/expansion operations, and two additional variables (one for each MHEX) that can be freely chosen as UA, ΔT_{min}, etc., as shown in the previous example. The compression and expansion operations are modeled as polytropic processes for ideal gases with polytropic exponent $\kappa = 1.352$, as in Wechsung *et al.*[141]

Several test cases are now explored, as in Example 3.4. As before, the CPLEX v12.5 callable library is used to solve the linear program at each iteration and the problem is determined to have converged to a solution when the infinity norm of the residual functions is less than 10^{-9}.

Case I. As a base case, the process is simulated with the variables x_1 through x_7 assigned as in Table C.1, $y_8 \equiv UA_{\text{HX-100}}$, and $y_9 \equiv UA_{\text{HX-101}}$. ΔT_{min} is specified as 4 K for both exchangers. The model converges to the solution shown in the Case I column of Table 3.6 after 128 iterations from the initial guess $\mathbf{x}^0 = [300\ 200\ 100\ 150\ 150\ 100\ 300\ 100\ 100]$. This solution differs slightly from the solution reported in Wechsung *et al.*,[141] however, note that the authors used the disjunctive formulation from Grossmann *et al.*,[46] whereas here the nonsmooth equations are solved directly,

Stream	Inlet	Outlet	F_{C_p}, $f_{C_p}(\frac{\text{kW}}{\text{K}})$	T^{in}, t^{in}(K)	T^{out}, t^{out}(K)	P (MPa)
H1	NG-2	NG-3	3.46	319.80	265.15	10.0
H2	NG-2	NG-3	5.14	265.15	197.35	10.0
H3	NG-3	NG-4	3.51	197.35	104.75	10.0
H4	N_2-8	N_2-9	1.03	x_1	x_2	2.7
C1	CO_2-2	CO_2-3	5.19	221.12	252.55	6.0
C2	CO_2-2	CO_2-3	6.10	252.55	293.15	6.0
C3	N_2-2	N_2-3	2.23	103.45	171.05	10.0
C4	N_2-3	N_2-4	1.62	171.05	218.75	10.0
C5	N_2-3	N_2-4	1.06	218.75	221.11	10.0
C6	N_2-6	N_2-7	0.96	x_3	221.15	0.4
C7	N_2-5	N_2-6	0.96	x_4	x_3	0.4
C8	N_2-11	N_2-12	0.93	x_5	x_6	0.1
C9	N_2-10	N_2-11	0.93	x_7	x_5	0.1

Table 3.5: Given data and unknown variables for the offshore LNG process case study.

so a small difference is not unexpected. The composite curves for the two MHEXs in this case are shown in Figure 3-10(a) and (b).

Case II. Now consider a case where the available $UA_{\text{HX-100}}$ is fixed at 120 kW/K, and $UA_{\text{HX-101}}$ is fixed at 30 kW/K. Variables x_1 through x_7 are as assigned as in Table C.1, $y_8 \equiv \Delta T_{\text{min,HX-100}}$, and $y_9 \equiv \Delta T_{\text{min,HX-101}}$. The model converges to the solution given in the Case II column of Table 3.6 after 28 iterations starting from the solution found in Case I. The composite curves for the two MHEXs in this case are shown in Figure 3-10(c) and (d). As can be seen, the curves are more closely pinched together throughout HX-100 than in Case I as a result of the increased conductance, with the pinch point location shifting to the high temperature extreme. Note that $\Delta T_{\text{min,HX-101}}$ also decreases relative to Case I in order to satisfy the overall process model (even though $UA_{\text{HX-101}}$ was specified as a lower value), so the other variable cold outlet temperatures decrease significantly to compensate.

Case III. Now consider the problem where x_1 through x_7 are as assigned as in Table C.1, $y_8 \equiv T_{H2}^{\text{out}}$, and $y_9 \equiv t_{C3}^{\text{out}}$. For this case, let $UA_{\text{HX-100}} = 85$ kW/K and $UA_{\text{HX-101}} = 35$ kW/K. ΔT_{min} is specified as 4 K for both exchangers. The model converges to the solution given in the Case III column of Table 3.6 after 48 iterations starting from the solution found in Case I, and the composite curves for this case are

Variable	Case I	Case II	Case III
x_1	365.07 K	365.07 K	365.07 K
x_2	225.44 K	217.66 K	231.10 K
x_3	193.35 K	196.09 K	195.06 K
x_4	95.08 K	95.08 K	95.08 K
x_5	180.85 K	174.75 K	194.58 K
x_6	357.14 K	362.46 K	352.12 K
x_7	95.14 K	91.85 K	97.52 K
y_8	97.54 kW/K	2.62 K	199.06 K
y_9	31.09 kW/K	1.26 K	168.25 K

Table 3.6: Results for the different cases of the LNG process case study.

shown in Figure 3-10(e) and (f). The reduction in $UA_{\text{HX-100}}$ as compared to Cases I and II leads to larger temperature differences throughout the exchanger than in those simulations, while the increase in $UA_{\text{HX-101}}$ leads to closer temperature approaches in this exchanger (though both exchangers maintain the same pinch points from Case I). However, as in Example 3.4, the feasibility of the problem is highly dependent on the specified conductance values. This again highlights the fact that for this particular designation of unknowns and degrees of freedom, there is only a small region in which all the constraints can be satisfied.

3.6 Conclusions

A new method for the simulation and design of processes with multistream heat exchangers has been presented, based on recent developments in nonsmooth analysis. While traditional models for multistream heat exchange operations can only be solved for a single unknown variable (using the energy balance), this new model allows for up to three unknown quantities to be calculated simultaneously. The model proposed here also allows for the specification of parameters such as the heat exchange area or the minimum approach temperature as inputs to the model, rather than simply calculating these quantities after the energy balance has already been solved. The nonsmooth equations in these formulations can be solved precisely and with a guar-

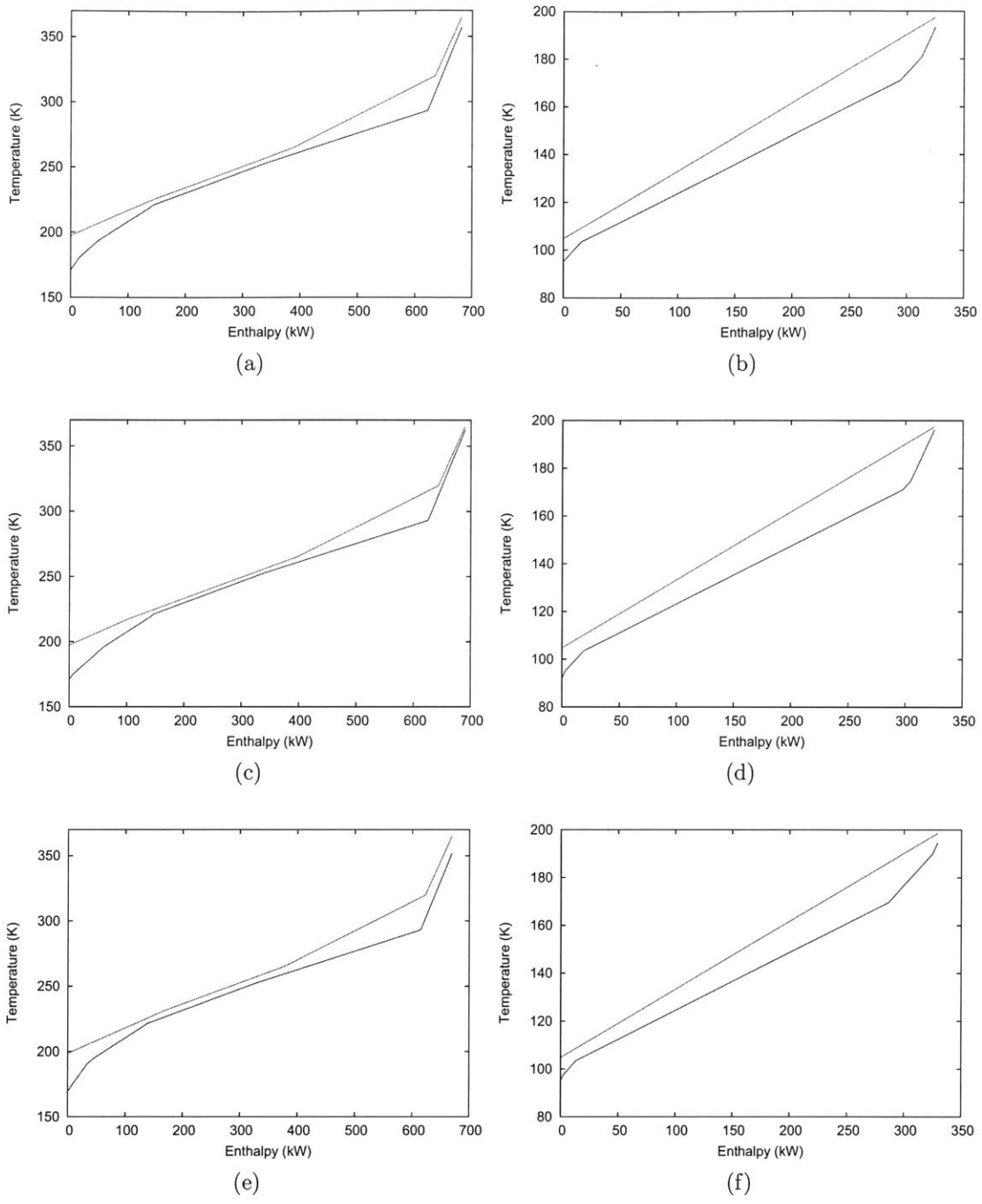


Figure 3-10: Composite curves of the simulated results of Example 3.5 under the conditions of (a) Case I for HX-100, (b) Case I for HX-101, (c) Case II for HX-100, (d) Case II for HX-101, (e) Case III for HX-100, (f) Case III for HX-101.

anteed local quadratic convergence rate, owing to the automatic calculation and use of B-subdifferential elements in the equation solving methods. The performance and

versatility of the solution procedure has been demonstrated in illustrative examples and on a LNG process flowsheet containing multiple MHEXs in addition to several other process units. In the next chapter, this basic MHEX model will be enhanced by including equations for the detection and simulation of phase changes, which are commonly encountered in cryogenic processes within multistream heat exchangers.

Chapter 4

Modeling of phase changes in multistream heat exchangers

In this chapter, a new method for modeling phase changes in multistream heat exchangers is presented. In many industrially relevant applications, streams in MHEXs will undergo phase changes between their inlet and outlet. In this model, nonsmooth equations are formulated that properly account for the existence or nonexistence of phases in heat integration, flash and physical property calculations in a MHEX. These new equations are used in conjunction with the nonsmooth model for MHEXs from the previous chapter to create a compact equation system that can be used for the simulation and design of complex processes. Notably, this formulation does not involve the solution of a difficult optimization problem, since it avoids the use of either disjunctive or complementarity constraints. The robustness and functionality of the new formulation is illustrated through several simulations of the PRICO process for liquefied natural gas production.

4.1 Introduction

Multistream heat exchangers are commonly found at the heart of many industrially relevant cryogenic processes, such as natural gas liquefaction processes. However, among the many difficulties of simulating MHEXs in cryogenic processes is the de-

tection and handling of phase changes. Taking LNG production as an example, the streams in the process MHEXs are usually both multicomponent and multiphase. Handling the nonlinear physical property variations associated with such streams during heat exchange creates a challenging simulation problem, especially when the phases traversed are not known *a priori*.

Methods for multiphase MHEX or heat exchanger network simulation have been reported by several authors in the literature. Among these, the focus is often placed on modeling pure component (isothermal) phase changes. An early approach along this theme is that of Grossmann *et al.*,[46] that adapts the earlier Duran and Grossmann[32] model for simultaneous process optimization and heat integration by accounting explicitly for streams that are known to be isothermal. This is done with the use of disjunctive constraints, which are reformulated with binary variables to yield a MINLP. Ponce-Ortega *et al.*[98] tackle isothermal streams in the context of heat exchanger network synthesis by applying a similar extension to the classic staged-superstructure approach from Yee and Grossmann.[149] However, these methods for isothermal phase changes do not extend to methods for the multicomponent case, which are more relevant for many practical applications.

Several approaches do exist in the literature for handling non-isothermal phase changes. Castier and Queiroz[20] describe a method based on solving a series of global optimization problems in successive temperature intervals to find pinch points and minimum energy targets in a HEN where the temperature-enthalpy relationship is possibly nonlinear. While more precise than methods that use piecewise-affine segments to approximate the composite curves, the approach requires significant computational effort.

Hasan *et al.*[48;49] use the superstructure concept as a basis for their work with mixed refrigerant processes by deriving a network of two-stream heat exchangers that is equivalent to an MHEX. This model handles phase changes in an MHEX by modeling the heat transfer in each phase as taking place in a separate two-stream heat exchanger in the superstructure bundle. The existence of the heat exchangers is determined by a disjunctive formulation and a set of propositional logic constraints to

formulate a very complex MINLP. The bubble and dew points are taken as constants in their model, so that stream pressures and compositions cannot change during optimization. Additionally, the temperature-enthalpy relationship in each phase is given by an empirical cubic correlation instead of a rigorous physical property model.

An alternate method for modeling phases changes in MHEX that is also based on the Duran and Grossmann[32] formulation is given by Kamath *et al.*[59] Here, the authors make the analogy between a heat exchanger network that requires no external utilities and an MHEX to derive an equation-oriented model. The authors use a simpler disjunctive representation of the phase detection problem than Hasan *et al.*,[49] which is subsequently handled by using complementarity constraints rather than binary variables. The model is also able to incorporate thermodynamics described by cubic equations of state. Applied to the PRICO process, the formulation results in a moderately-sized mathematical program with complementarity constraints (MPCC) (3,426 equations using Soave-Redlich-Kwong thermodynamics) that requires completing a rather involved initialization procedure to obtain a suitable initial guess from which to solve the problem and a solution method suitable for MPCCs.

Of particular note here is that all the approaches mentioned above require the solution of a hard optimization problem, with those methodologies that involve the solution of a nonconvex MINLP being particularly challenging. Among the approaches which avoid the use of binary variables, the use of smoothing approximations to remove the nonsmoothness caused by approximating the temperature-enthalpy relationship of streams as a piecewise-affine function is common. This is often done with the reformulation of the max operator given by Equation 3.10. In contrast, the model presented in this work is presented as an equation solving problem, rather than an optimization problem. Furthermore, the equations developed herein are substantially simpler than the mixed-integer or complementarity constraint formulations developed in previous works, at the expense of being nonsmooth. However, this is no longer a significant obstacle to practical implementation due to the recent development of automatic techniques to calculate generalized derivatives[64] and robust methods for nonsmooth equation solving.[99;33] The model size and problem complexity is thereby

substantially reduced compared to the models presented thus far in the literature.

4.2 Background

In this section, existing methods for the detection of phase regime in MHEXs and in flash calculations that do not rely on nonsmooth functions are discussed.

4.2.1 Equation-oriented approaches to phase detection in MHEXs

In the context of heat integration and MHEX simulation problems, the most significant challenge associated with a stream changing phase is modeling the change in the heat capacity flowrate. Heat integration and pinch analysis techniques are based on the assumption that every stream has a constant heat capacity flowrate, and therefore an affine temperature-enthalpy relationship. The classical algorithms for solving these problems cannot be applied readily when this assumption is violated. Since the MHEX model described in Chapter 3 relies on a modified pinch-locating strategy, it is also only applicable for such problems without the development in the present chapter.

Figure 4-1 shows an example of a cooling curve for a natural gas stream from ambient temperature to -160°C. The overall temperature-enthalpy relationship is clearly not affine, though the single phase (superheated and subcooled) regions show near-affine behavior, which is indicative of a near-constant heat capacity flowrate. Since natural gas is a mixture of hydrocarbons, the two phase region persists over a large temperature range and exhibits nonlinear temperature-enthalpy behavior. Clearly, naïvely assuming this stream has constant heat capacity in a heat integration calculation would introduce significant error and likely invalidate the solution, so in order to apply pinch analysis techniques reasonably, the cooling curve must be approximated by a series of affine segments. A good choice is to approximate each of the three individual phase regions in Figure 4-1 with one (or possibly more) affine seg-

ments. However, this is not so straightforward in the general case where the process stream inlet/outlet temperatures, pressures and compositions are possibly variables in the simulation. This requires the phase boundaries to move from iteration to iteration during the solution process (since the dew and bubble points are functions of pressure and composition) or even entire phases to disappear as the temperatures vary.

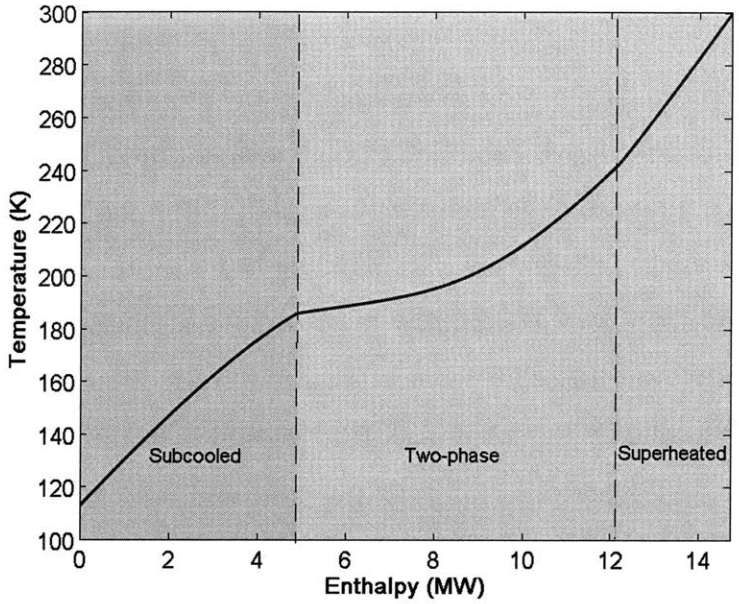


Figure 4-1: A typical cooling curve of a natural gas stream.

To address these issues, Kamath *et al.*[59] propose a model in which the physical streams in the process are subdivided into substreams corresponding to superheated (sup), two-phase (2p) and subcooled (sub) regions. The heat integration calculations for the MHEX are then performed using these substreams instead of the physical process streams. Each of the substreams therefore has an associated inlet temperature, outlet temperature and heat load. The inlet and outlet temperatures of these streams

are assigned by the following set of disjunctive constraints:

$$\begin{bmatrix} Y_{\text{sup}}^{\text{IN}} \\ T^{\text{IN}} \geq T_{\text{DP}} \\ T_{\text{sup}}^{\text{in}} = T^{\text{IN}} \\ T_{\text{2p}}^{\text{in}} = T_{\text{DP}} \\ T_{\text{sub}}^{\text{in}} = T_{\text{BP}} \end{bmatrix} \veebar \begin{bmatrix} Y_{\text{2p}}^{\text{IN}} \\ T_{\text{BP}} \leq T^{\text{IN}} \leq T_{\text{DP}} \\ T_{\text{sup}}^{\text{in}} = T_{\text{DP}} \\ T_{\text{2p}}^{\text{in}} = T^{\text{IN}} \\ T_{\text{sub}}^{\text{in}} = T_{\text{BP}} \end{bmatrix} \veebar \begin{bmatrix} Y_{\text{sub}}^{\text{IN}} \\ T^{\text{IN}} \leq T_{\text{BP}} \\ T_{\text{sup}}^{\text{in}} = T_{\text{DP}} \\ T_{\text{2p}}^{\text{in}} = T_{\text{BP}} \\ T_{\text{sub}}^{\text{in}} = T^{\text{IN}} \end{bmatrix}. \tag{4.1}$$

Here, $Y_{\text{sup}}^{\text{IN}}$, $Y_{\text{2p}}^{\text{IN}}$ and $Y_{\text{sub}}^{\text{IN}}$ are indicator binary variables for the phase of the inlet stream, T^{IN} is the temperature of the physical process stream entering the MHEX, T_{DP} is the dew point temperature, T_{BP} is the bubble point temperature and $T_{\text{sup/2p/sub}}^{\text{in}}$ are the inlet temperatures assigned to the substreams used in the actual heat integration. An exactly analogous disjunctive formulation exists for assigning the outlet temperatures, and an additional set of logic constraints governs the relationships between the inlet and outlet Y variables. In order to pick the correct set of constraints to enforce, Kamath *et al.*[59] formulate the following LP to be added to the MHEX formulation and solved for $Y_{\text{sup}}, Y_{\text{2p}}$ and Y_{sub} for both the inlet and the outlet of each physical process stream (IN/OUT superscripts omitted for clarity):

$$\begin{aligned} \underset{Y_{\text{sup}},Y_{\text{2p}},Y_{\text{sub}}}{\arg\min} \quad & -[Y_{\text{sup}}(T - T_{\text{DP}}) + Y_{\text{2p}}(T_{\text{DP}} - T)(T - T_{\text{BP}}) + Y_{\text{sub}}(T_{\text{BP}} - T)] \\ \text{s.t.} \quad & Y_{\text{sup}} + Y_{\text{2p}} + Y_{\text{sub}} = 1, \\ & Y_{\text{sup}}, Y_{\text{2p}}, Y_{\text{sub}} \geq 0. \end{aligned} \tag{4.2}$$

However, since their overall MHEX model uses the simultaneous optimization and heat integration formulation of Duran and Grossmann,[32] these LPs would have to be included as embedded subproblems in a larger optimization problem. Since embedding LPs in an outer optimization problem is usually undesirable, their equivalent optimality conditions are instead formulated as complementarity constraints, which are then added to the equation-oriented optimization formulation to create an MPCC. The full formulation is then solved using the penalty formulation,[14] in which a user-defined penalty parameter multiples the inner product of the vectors formed by the

variables on each side of the complementarity operators.

The possible appearance and disappearance of phases from iteration to iteration also causes issues in the solution of vapor-liquid equilibrium calculations, described in the following section.

4.2.2 Steady-state flash simulation

Consider a typical steady-state flash operation as shown schematically in Figure 4-2, in which a feed stream with molar flowrate F with n_c components at molar composition $\mathbf{z}_F$ separates into a liquid stream with molar flowrate L at molar composition $\mathbf{x}_L$ and a vapor stream with molar flowrate V at molar composition $\mathbf{y}_V$.

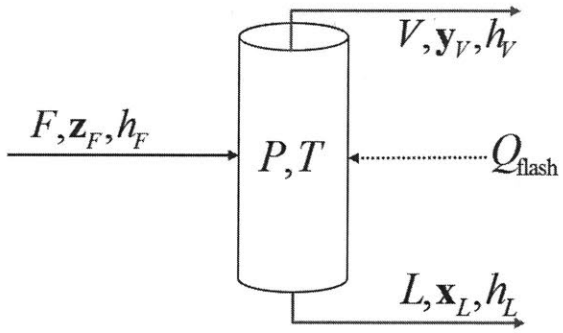


Figure 4-2: Schematic of a steady-state single-stage flash operation.

The standard model for vapor-liquid equilibrium calculations around such a flash unit is as follows:

$$F = V + L, \tag{4.3}$$

$$Fz_{F,i} = Lx_{L,i} + Vy_{V,i}, \quad i = 1, \ldots, n_c, \tag{4.4}$$

$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0, \tag{4.5}$$

$$y_{V,i} = k_i x_{L,i}, \quad i = 1, \ldots, n_c, \tag{4.6}$$

where k_i is the equilibrium ratio for component i, which is in general a function of temperature, pressure and composition of both phases. Additionally, an energy balance is required if either there is a specified heat duty, Q_{flash}, related to the flash

unit, or if there is a pressure drop from the feed stream to the flash unit. In both cases, the flash temperature is unknown and must be found by solving the following enthalpy balance together with Equations (4.3)-(4.6):

$$Vh_V + Lh_L - Fh_F = Q_{\text{flash}}, \tag{4.7}$$

with h_V, h_L and h_F as the molar enthalpies of the vapor, liquid and feed streams, respectively. Assuming the feed conditions are known, for an n_c component system, there are $2n_c + 3$ equations, but $2n_c + 5$ unknowns $(\mathbf{x}_L, \mathbf{y}_V, T, P, V, L, Q_{\text{flash}})$ in the previous model, so two quantities must be specified. In this chapter, the focus will be on flash calculations in which the specified quantities are the pressure and heat duty (usually referred to as a PQ-flash) and flash calculations in which the specified quantities are the pressure and temperature (usually referred to as a PT-flash). Respectively, these are the most difficult and least difficult of the typical flash calculations to solve, as well as being the most commonly encountered types in process simulation.

The system of equations given by (4.3)-(4.7) is rarely solved directly in this form. The following formulation due to Rachford and Rice[100] is often employed for its desirable convergence characteristics:

$$L = (1 - \alpha)F, \tag{4.8}$$

$$x_{L,i} = \frac{z_{F,i}}{1 + \alpha(k_i - 1)}, \quad i = 1, \ldots, n_c, \tag{4.9}$$

$$y_{V,i} = \frac{k_i z_{F,i}}{1 + \alpha(k_i - 1)}, \quad i = 1, \ldots, n_c, \tag{4.10}$$

$$\sum_{i=1}^{n_c} \frac{z_{F,i}(k_i - 1)}{1 + \alpha(k_i - 1)} = 0, \tag{4.11}$$

$$\alpha h_V + (1 - \alpha)h_L - h_F = Q_{\text{flash}}/F, \tag{4.12}$$

where $\alpha \equiv \frac{V}{F}$, the fraction of the feed that is vaporized. Note that for specified P and Q_{flash}, this is still a set of $2n_c+3$ equations in $2n_c+3$ variables $(\mathbf{x}_L, \mathbf{y}_V, T, \alpha, L)$. For an idealized system in which the k_i values have no composition dependence (e.g. when

assuming Raoult's Law holds), the mole fractions can be calculated from Equations (4.9) and (4.10) as a post-processing step after converging the other equations.

However, under conditions where only one outlet stream exists, the equilibrium constraints (Equations (4.5) and (4.6), or equivalently, Equation (4.11)) cannot be satisfied. A suggested extension of the Rachford-Rice formulation of the flash equations for finding single-phase solutions is the concept of the "negative flash".[144] In this approach, values for α calculated during the solution procedure are considered acceptable within a range from $\frac{1}{(1-k_{\max})} < 0$ to $\frac{1}{(1-k_{\min})} > 1$, where $k_{\min}$ and $k_{\max}$ are the minimum and maximum equilibrium ratios of the mixture at the solution, respectively. Values of α in this range will produce positive mole fractions for all components and therefore can be used to evaluate all thermophysical properties. Convergence to a solution with $\alpha < 0$ or $\alpha > 1$ indicates the presence of single-phase solution which can be post-processed to obtain a physically meaningful result.

An alternative procedure is suggested by Baumrucker *et al.*,[14] in which a new variable β is introduced to the formulation to relax Equation (4.6). A small linear program is then solved to determine the value of β. The vapor-liquid equilibrium for a flash operation is therefore calculated by solving Equations (4.4), (4.5), (4.7) and the following:

$$\begin{aligned} y_{V,i} &= \beta k_i x_{L,i}, \quad i = 1, \ldots, n_c, \\ (L, V) \in \arg\min_{L,V} \ & (1-\beta)(L-V) \\ \text{s.t.} \ & L + V = F, \\ & L, V \geq 0. \end{aligned} \tag{4.13}$$

Then, as before, instead of solving this embedded LP explicitly, its optimality conditions are used to generate complementarity constraints involving slack variables s_V and s_L that replace Equation (4.13) in the previous formulation:

$$\beta = 1 - s_L + s_V, \tag{4.14a}$$

$$F = L + V, \tag{4.14b}$$

$$0 \leq L \perp s_L \geq 0, \tag{4.14c}$$

$$0 \leq V \perp s_V \geq 0. \tag{4.14d}$$

This system of equations and inequalities can then be included in an optimization problem and solved using the penalty formulation or another appropriate solution method for MPCCs.

4.3 Nonsmooth models for phase phase detection in MHEXs

The models for handling phase changes described in the previous section give rise to either MINLPs or MPCCs when performing the heat integration calculations necessary for simulating multistream heat exchangers. However, this section will show how this significant increase in problem complexity can be avoided by the use of nonsmooth expressions in conjuction with the MHEX model from Chapter 3. However, in augmenting this model to simulate phase changes, care must be taken that any additional equations are also $\mathcal{PC}^1$ functions to preserve the desirable local convergence characteristics in equation-solving methods.

Consider the disjunctive model given in Equation (4.1) for assigning substream inlet temperatures. From inspection of the constraints, the behavior of these expressions for the substream inlet temperatures as a function of T^{IN} can be plotted as shown in Figure 4-3. The outlet temperatures follow an analogous trend. This suggests that the values of $T_{\text{sup}}^{\text{in/out}}$, $T_{\text{sub}}^{\text{in/out}}$ and $T_{\text{2p}}^{\text{in/out}}$ are in fact continuous nonsmooth functions of $T^{\text{IN/OUT}}$, T_{DP} and T_{BP}.

Therefore, instead of needing either binary variables or complementarity constraints to assign the temperatures $T_{\text{sup}}^{\text{in/out}}$, $T_{\text{sub}}^{\text{in/out}}$ and $T_{\text{2p}}^{\text{in/out}}$ correctly, the following nonsmooth equations may instead be used:

$$T_{\text{sup}}^{\text{in}} = \max\{T_{\text{DP}}, T^{\text{IN}}\}, \tag{4.15}$$

$$T_{\text{2p}}^{\text{in}} = \text{mid}\{T_{\text{DP}}, T^{\text{IN}}, T_{\text{BP}}\}, \tag{4.16}$$

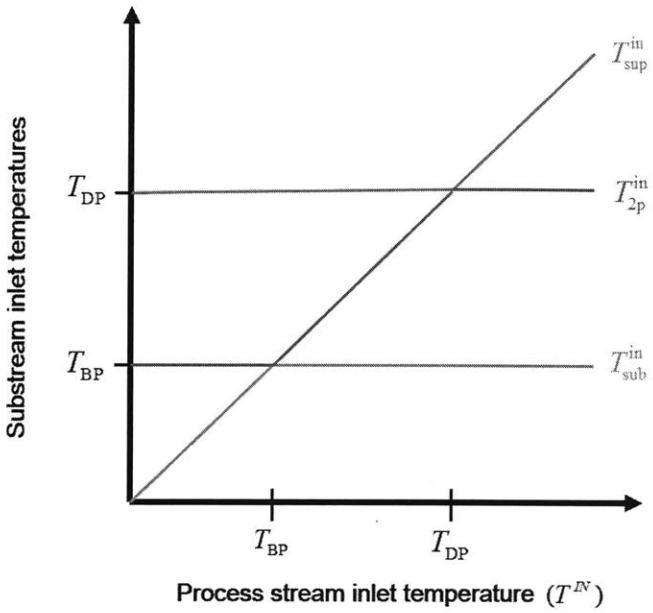


Figure 4-3: Inlet temperature of the three nonphysical substreams as a function of physical inlet stream temperature.

$$T_{\text{sub}}^{\text{in}} = \min\{T^{\text{IN}}, T_{\text{BP}}\}, \tag{4.17}$$

$$T_{\text{sup}}^{\text{out}} = \max\{T_{\text{DP}}, T^{\text{OUT}}\}, \tag{4.18}$$

$$T_{\text{2p}}^{\text{out}} = \text{mid}\{T_{\text{DP}}, T^{\text{OUT}}, T_{\text{BP}}\}, \tag{4.19}$$

$$T_{\text{sub}}^{\text{out}} = \min\{T^{\text{OUT}}, T_{\text{BP}}\}. \tag{4.20}$$

The function mid : $\mathbb{R}^3 \to \mathbb{R}$ maps to the median of its three arguments and is indeed a $\mathcal{PC}^1$ function, as can be seen from its equivalent representation in terms of the binary min and max functions:

$$\text{mid}\{a, b, c\} = \max\{\min\{a, b\}, \min\{\max\{a, b\}, c\}\}. \tag{4.21}$$

In cryogenic applications, it is not uncommon to find that some of the streams that change phase will do so over a very small temperature interval or isothermally. This includes streams that contain only a single pure component, such as in the propane precooling stage of the C3MR process for LNG production. In such systems, the pure component undergoes a phase change at a constant temperature, which is not possible

to model directly using a piecewise-affine approximation. However, the inclusion of nonsmooth functions in this work allows this behavior to be modeled more precisely than in approaches that rely on smoothing approximations, while avoiding the need for binary variables or disjunctions. For instance, if it is likely that a cold substream will undergo a very small or no temperature change in the two-phase region during iteration, Equations (4.16) and (4.19) can be replaced with the following expressions based on the discussion in Kamath *et al.*:[59]

$$t_{2\mathrm{p}}^{\mathrm{in}} = \min\left\{\mathrm{mid}\{t_{\mathrm{DP}}, t^{\mathrm{IN}}, t_{\mathrm{BP}}\}, \frac{\mathrm{mid}\{t_{\mathrm{DP}}, t^{\mathrm{IN}}, t_{\mathrm{BP}}\} + \mathrm{mid}\{t_{\mathrm{DP}}, t^{\mathrm{OUT}}, t_{\mathrm{BP}}\} - \varepsilon}{2}\right\}, \tag{4.22}$$

$$t_{2\mathrm{p}}^{\mathrm{out}} = \max\left\{\mathrm{mid}\{t_{\mathrm{DP}}, t^{\mathrm{OUT}}, t_{\mathrm{BP}}\}, \frac{\mathrm{mid}\{t_{\mathrm{DP}}, t^{\mathrm{IN}}, t_{\mathrm{BP}}\} + \mathrm{mid}\{t_{\mathrm{DP}}, t^{\mathrm{OUT}}, t_{\mathrm{BP}}\} + \varepsilon}{2}\right\}. \tag{4.23}$$

where ε is a user-defined but small fictitious temperature change. Conventionally, this approach is considered undesirable (hence the existence of methods that avoid needing to assign a fictitious temperature change). However, in the specific context of nonsmooth equations, nonsmooth equation solvers will not be affected by the usual ill-conditioning caused by making such a parameter too small. Analogous equations can also be written for a hot two-phase substream, as well as the substreams in the other regions of the phase diagram.

Another problematic aspect of simulating cryogenic processes is that certain light components, such as nitrogen or methane, may be present both above and below their pure component critical points. This affects the calculation of certain physical properties, such as the enthalpy of vaporization, Δh_{vap}, which is needed to obtain liquid enthalpy values, and is in general a function of temperature for a pure component. The form of this correlation used by default in Aspen Plus v8.4 is the Design Institute

for Physical Properties (DIPPR) Equation 106:

$$\Delta h_{\text{vap}}(T) = \begin{cases} A\left(1-\frac{T}{T_c}\right)^{B+C\left(\frac{T}{T_c}\right)+D\left(\frac{T}{T_c}\right)^2}, & T < T_c, \\ 0, & T \geq T_c, \end{cases} \tag{4.24}$$

where T_c is the critical temperature and A, B, C and D are species dependent parameters. The use of if-else logic in this statement makes it incompatible with the automatic generalized derivative evaluation procedure needed in this work. However, this expression can be recast using a nonsmooth expression as follows:

$$\Delta h_{\text{vap}}(T) = A\left(\max\left\{0, 1-\frac{T}{T_c}\right\}\right)^{B+C\left(\frac{T}{T_c}\right)+D\left(\frac{T}{T_c}\right)^2} \tag{4.25}$$

This modified correlation can now be used for simulation in the current computational framework. Example 4.1 illustrates the phase change model that has been developed thus far on a simple process.

Example 4.1. Consider the flowsheet shown in Figure 4-4, which is based on the motivating example from Kamath *et al.*,[59] in which three hot streams with constant heat capacity flowrate exchange heat with two cold streams of constant heat capacity flow rate and a nitrogen stream at two pressure levels. The stream data for the constant heat capacity streams is found in Table 4.1. The liquid nitrogen stream at 95 K and 6 bar is first pressurized and then enters the multistream heat exchanger as a subcooled liquid. On the first pass, the nitrogen stream is modeled using three phase segments since it will leave as a superheated vapor. After exiting the exchanger, it is isentropically expanded to ambient pressure through a turbine and enters the heat exchanger a second time as a superheated vapor. This second pass is modeled using a single phase segment since the nitrogen stream will not recondense.

Two simulations are performed that each converge to the solution given to the original optimization problem from Kamath *et al.*[59] To model the isothermal phase change precisely, a value of $\varepsilon = 10^{-9}$ is used in Equations (4.22) and (4.23) for the two-phase nitrogen substream on the first pass. The simulation is deemed to

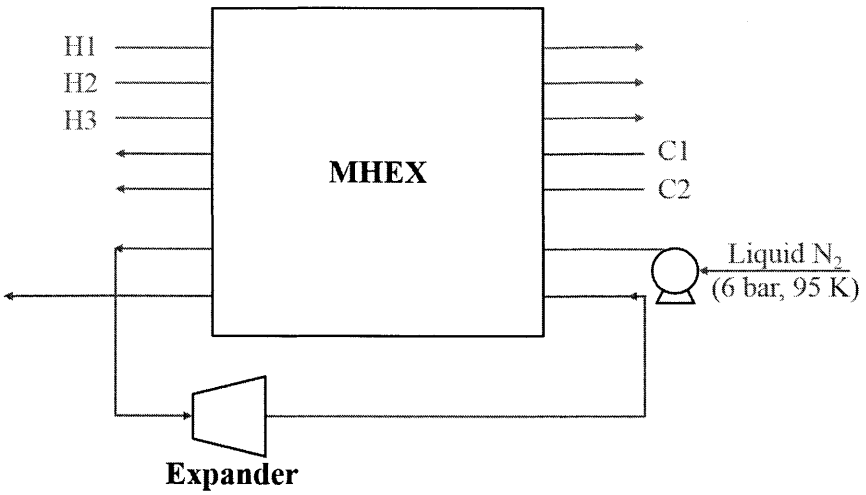


Figure 4-4: Process flowsheet for Example 4.1.

Table 4.1: Stream data for Example 4.1.

Stream Name	T^{in} or t^{in} (K)	T^{out} or t^{out} (K)	F_{C_p} or f_{C_p} (kW K^{-1})
H1	298	250	3.0
H2	265	180	4.0
H3	195	150	2.0
C1	220	245	3.0
C2	255	280	3.5

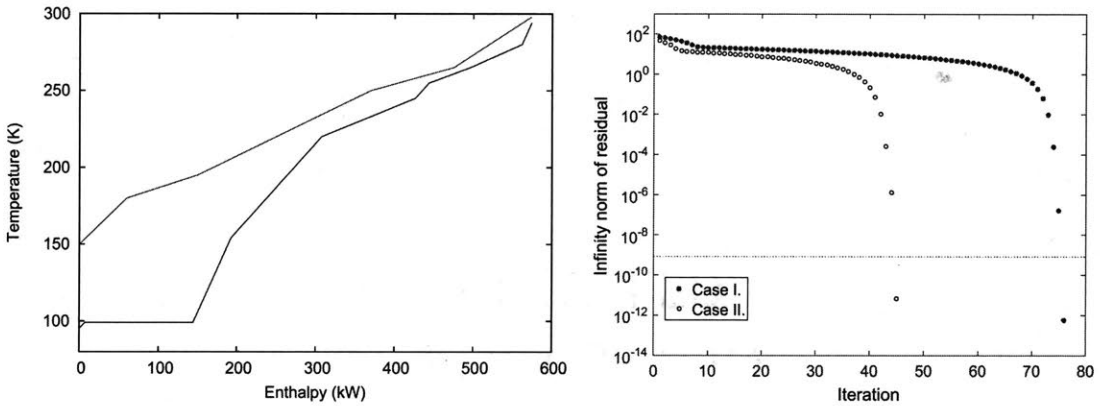
have converged when the infinity norm of the function residuals is below 10^{-9}. The ideal physical property model used in this example is given in Appendix B, and the necessary pure component model parameters for nitrogen were obtained from Aspen Plus v8.4.[5]

Case I. Let $x_1 \equiv t^{\text{OUT}}_{1^{\text{st}}\text{pass}}$, $x_2 \equiv t^{\text{OUT}}_{2^{\text{nd}}\text{pass}}$ and $x_3 \equiv UA$ be the unknown variables afforded by the base MHEX model consisting of Equations (3.4), (3.13) and (3.14). Then let $x_4 \equiv t^{\text{IN}}_{1^{\text{st}}\text{pass}}$ be given implicitly by Equation (B.15) and $x_5 \equiv t^{\text{IN}}_{2^{\text{st}}\text{pass}}$ be given by Equation (B.14). The minimum temperature difference in the exchanger is set as 4 K and the pump discharge pressure is set as 7.25 bar. This is therefore a simple case where the phase boundary will not change from iteration to iteration and the boiling temperatures of nitrogen can be calculated at 7.25 bar and 1 bar outside of the iterations. Given an initial guess of $\mathbf{x}^0 = (300, 300, 50, 95, 150)$, the solution $\mathbf{x}^* = (265.23, 294.00, 34.79, 95.11, 154.10)$ is found after 76 iterations of the

LP-Newton method taking 0.041 seconds.

Case II. With x_1, x_3, x_4 and x_5 as before, let x_2 be the discharge pressure of the pump. The minimum temperature difference is again set at 4K, and $t^{\text{OUT}}_{2^{\text{nd}}\text{pass}}$ is set as 294 K, the value from the solution of Case I. Unlike the previous case, here the boiling temperature of nitrogen needs to be recalculated at each iteration from the current value of x_2. Given an initial guess of $\mathbf{x}^0 = (300, 6, 50, 95, 150)$, the solution $\mathbf{x}^* = (265.23, 7.25, 34.79, 95.11, 154.10)$ is found after 45 iterations of the LP-Newton method taking 0.023 seconds.

The composite curves for the MHEX simulated in this example are shown in Figure 4-5(a.) and a convergence plot for both cases is shown in Figure 4-5(b.). Figure 4-5(a.) shows that the isothermal phase change is properly captured by the model, while Figure 4-5(b.) shows this model exhibits the well-known Newton-type method behavior of relatively slow convergence far away from the solution, followed by quadratic convergence in a local neighborhood of the solution point.



(a) (b)

Figure 4-5: (a.) Hot composite curve (red) and cold composite curve (blue) for the MHEX simulated in Example 4.1. (b.) Plot of convergence behavior for Cases I and II.

4.4 Nonsmooth models for vapor-liquid equilibrium calculations

As noted in Section 4.2.2, the appearance and disappearance of phases also makes equilibrium calculations challenging. However, in place of the complementarity formulation of Equation (4.14), a new nonsmooth model that accounts for the disappearance of phases while performing flash calculations is given by solving the following equation in place of Equation (4.11) in the Rachford-Rice form of the flash model:

$$\text{mid}\left\{\alpha,\ -\sum_{i=1}^{n_c}\frac{z_{F,i}(k_i-1)}{1+\alpha(k_i-1)},\ \alpha-1\right\}=0. \tag{4.26}$$

This formulation is an extension of the classical Rachford-Rice equation for solving flash calculations, where here the three arguments in the mid function correspond to finding an all liquid outlet, a two-phase outlet and an all vapor outlet respectively. Note that the second term in Equation (4.26) is just the negative of the standard Rachford-Rice expression (left-hand side of Equation (4.11)). The working mechanism of the equation is as follows. When the outlet is all vapor, $\alpha = 1$ and the Rachford-Rice expression is positive, so that here, the first term is equal to 1, the second term is negative and the third term is equal to zero. Thus, the mid expression picks the third term and evaluates to zero, satisfying Equation (4.26) with $\alpha = 1$. Similarly, when the solution is a two-phase mixture, $0 < \alpha < 1$ and the Rachford-Rice expression equals zero, so, the first term is greater than zero, the third term is less than zero and the second term is exactly zero. Thus, the mid expression picks the second term and evaluates to zero, satisfying Equation (4.26). The argument for an all liquid outlet is analogous. Equation (4.26) can also be equivalently written as follows:

$$\text{mid}\left(\alpha,\ \sum_{i=1}^{n_c}x_{L,i}-\sum_{i=1}^{n_c}y_{V,i},\ \alpha-1\right)=0, \tag{4.27}$$

and substituted in place of Equation (4.5) in the standard flash formulation. A detailed proof of the correctness of this formulation, which follows from the minimization

of the total molar Gibbs free energy of a mixture, is presented in the following section.

4.4.1 Proof of the nonsmooth flash formulation

The minimization of the Gibbs free energy at constant temperature and pressure is expressed with the following optimization problem:

$$\begin{aligned}
\min_{L,V,\mathbf{x}_L,\mathbf{y}_V} \quad & G = L\sum_{i=1}^{n_c} x_{L,i}\bar{G}_i^L + V\sum_{i=1}^{n_c} y_{V,i}\bar{G}_i^V \qquad (4.28)\\
\text{s.t.} \quad & F = L + V,\\
& z_{F,i}F = x_{L,i}L + y_{V,i}V, \quad i = 1,\ldots,n_c,\\
& \sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0,\\
& L \geq 0, \quad V \geq 0,\\
& x_{L,i} > 0, \quad y_{V,i} > 0, \quad i = 1,\ldots,n_c,
\end{aligned}$$

where G is the extensive Gibbs free energy; $\bar{G}_i^L$ and $\bar{G}_i^V$ are the partial molar Gibbs free energy of component i in the liquid and vapor phases, respectively. Note that since, in general, the partial molar Gibbs free energy depends on temperature, pressure and the (intensive) composition of the relevant phase, the optimization problem must be formulated with the mole fractions as explicit decision variables to prevent them from becoming undefined if either L or V is zero. This is necessary to avoid a common critical error in other published proofs of similar concepts, as described next.

The domain of definition for partial molar Gibbs free energy

At constant temperature and pressure, the molar Gibbs free energy of a mixture is given by a function $G : \mathbb{R}_+^{n_c} \to \mathbb{R}$, where $\mathbb{R}_+$ denotes the set of nonnegative real numbers. Note that the total molar Gibbs free energy is an extensive property of the mixture, and so G is a positively homogeneous function of degree 1, defined formally as follows.

Definition 4.1. Let $X \subset \mathbb{R}^n$ be a cone. A function $f : X \to \mathbb{R}$ is positively

homogeneous of integer degree k if for any real $\lambda > 0$ and any $\mathbf{x} \in X$,

$$f(\lambda \mathbf{x}) = \lambda^k f(\mathbf{x}). \tag{4.29}$$

The partial derivative $\left(\frac{\partial G}{\partial n_i}\right)_{n_{j \neq i}}$, denoted by $\bar{G}_i$, is the partial molar Gibbs free energy of component i. A well-known consequence of Euler's Homogeneous Function Theorem shows $\bar{G}_i$ is a positively homogeneous function of degree 0 for each i,[122] that is, $\bar{G}_i$ is an intensive property of the mixture. Importantly, since the domain of definition for G is $\mathbb{R}^{n_c}_+$, G is not classically differentiable on the boundary of $\mathbb{R}^{n_c}_+$. Therefore, the partial derivative function $\bar{G}_i$ is only defined on $\mathbb{R}^{n_c}_{++}$, where $\mathbb{R}_{++}$ denotes the set of strictly positive real numbers. Now consider the identity:

$$G(\mathbf{l}, \mathbf{v}) = \sum_{i=1}^{n_c} l_i \bar{G}_i^L(l_1, l_2, ..., l_{n_c}) + \sum_{i=1}^{n_c} v_i \bar{G}_i^V(v_1, v_2, ..., v_{n_c}), \tag{4.30}$$

where l_i and v_i are the liquid and vapor phase moles of component i, respectively, and note that this identity is *only* well-defined when $l_i > 0, \forall i$ and $v_i > 0, \forall i$ since $\bar{G}_i^L$ and $\bar{G}_i^V$ are only defined on $\mathbb{R}^{n_c}_{++}$. This restriction requires $L > 0$ and $V > 0$, and so the mixture must be in the two-phase region for Equation (4.30) to hold.

Additionally, the following general result establishes that a partial molar property function defined in terms of molar amounts, as in Equation (4.30), cannot be continuously extended to zero amount of substance (i.e. when $l_i = 0, \forall i$ or $v_i = 0, \forall i$) except in one case.

Lemma 4.1. Let $X \subset \mathbb{R}^n \setminus \{\mathbf{0}\}$ be a blunt cone. A continuous function $f : X \to \mathbb{R}$ that is positively homogeneous of degree 0 has a continuous extension $f(\mathbf{0})$ to the origin if and only if it is a constant function.

Proof. Suppose f is continuous, positively homogeneous of degree 0 and has a continuous extension to the origin. Then $\forall \varepsilon > 0, \exists \delta > 0$ such that for $\mathbf{d} \in X$, whenever $||\mathbf{d}|| < \delta$, $|f(\mathbf{d}) - f(\mathbf{0})| < \varepsilon$. Since f is positively homogeneous of degree 0, $f(\lambda \mathbf{d}) = f(\mathbf{d}), \forall \lambda > 0$, so that $|f(\lambda \mathbf{d}) - f(\mathbf{0})| < \varepsilon$ also. However, any $\mathbf{x} \in X$ can be expressed as $\mathbf{x} = \lambda \mathbf{d}$ for some choice of λ and $\mathbf{d}$ as defined previously, and so this

implies $|f(\mathbf{x}) - f(\mathbf{0})| < \varepsilon, \forall \mathbf{x} \in X$, so f must be a constant function with value $f(\mathbf{0})$.

Now suppose f is a constant function such that $f(\mathbf{x}) = c, \forall \mathbf{x} \in X$. Then f is clearly continuous on X, and also positively homogeneous of degree 0 since for any $\lambda > 0$, $f(\lambda \mathbf{x}) = c = f(\mathbf{x})$. Furthermore, if the definition of f is extended as follows:

$$f(\mathbf{x}) = \begin{cases} c, & \mathbf{x} \in X, \\ c, & \mathbf{x} = \mathbf{0}, \end{cases} \tag{4.31}$$

then f is continuous at $\mathbf{0}$, since $\forall \varepsilon > 0$, $\exists \delta > 0$ such that for $\mathbf{d} \in X$ with $||\mathbf{d}|| < \delta$, $|f(\mathbf{d}) - f(\mathbf{0})| = |c - c| = 0 < \varepsilon$. □

The constant function condition of the previous theorem would be met if each $\bar{G}_i$ had no concentration dependence, such as if each was equal to the pure component Gibbs free energy of component i at the system temperature and pressure. However, even for a mixture of ideal gases this is not the case, as $\bar{G}_i^{\text{ideal}} = G_i^0 + RT \ln(P_i/P^0)$, where P_i is the partial pressure of component i in the mixture. Therefore, any extension of $\bar{G}_i^L$ and $\bar{G}_i^V$ to zero substance will be discontinuous at the origin in any physical case. Analysis of the KKT conditions for the optimization problem minimizing G defined as in Equation (4.30) is therefore only applicable in the two-phase region, contrary to what has been claimed in many previous works.

Instead, this proof will make use of the identity:

$$G(\mathbf{x}_L, \mathbf{y}_V, L, V) = L \sum_{i=1}^{n_c} x_{L,i} \bar{G}_i^L(x_1, x_2, ..., x_{n_c}) + V \sum_{i=1}^{n_c} y_{V,i} \bar{G}_i^V(y_1, y_2, ..., y_{n_c}). \tag{4.32}$$

The same results regarding the domain of definition apply here; however, $L = 0$ does not imply $x_{L,i} = 0, \forall i$ (or vice-versa), and likewise $V = 0$ does not imply $y_{V,i} = 0, \forall i$ (or vice-versa). Thus, the partial molar properties defined in terms of mole fractions are well-defined even in the single-phase regimes, so long as $\mathbf{x}_L > \mathbf{0}$ and $\mathbf{y}_V > \mathbf{0}$. These constraints are enforced in the Gibbs free energy minimization problem used in the proof. Importantly, these constraints do not affect the physical solutions of the problem, as in the two-phase regime, $\mathbf{x}_L > \mathbf{0}$ and $\mathbf{y}_V > \mathbf{0}$ are already implied by the

other constraints; in the liquid regime, $\mathbf{x}_L > \mathbf{0}$ is implied by the other constraints, while the choice of $\mathbf{y}_V$ is arbitrary beyond satisfying the KKT conditions of Equation (4.28), as will be shown next, and similarly in the vapor regime, $\mathbf{y}_V > \mathbf{0}$ is implied by the other constraints, while the choice of $\mathbf{x}_L$ is also arbitrary aside from satisfying the KKT conditions of Equation (4.28).

Returning to Equation (4.28), the Lagrangian function of this equation is written as

$$\begin{aligned} L = {} & L\sum_{i=1}^{n_c} x_{L,i}\bar{G}_i^L + V\sum_{i=1}^{n_c} y_{V,i}\bar{G}_i^V \\ & + \gamma_F\left(F - L - V\right) + \sum_{i=1}^{n_c}\gamma_i(z_{F,i}F - x_{L,i}L - y_{V,i}V) \\ & + \gamma_S\left(\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i}\right) - \alpha_L L - \alpha_V V, \end{aligned} \tag{4.33}$$

with $\gamma_i, \gamma_F, \gamma_S \in \mathbb{R}$ and $\alpha_L, \alpha_V \geq 0$ being the Lagrange multipliers. Noting that $z_{F,i}F$ are constant, the Karush-Kuhn-Tucker (KKT) conditions read

$$\sum_{i=1}^{n_c} x_{L,i}\left(\bar{G}_i^L - \gamma_i\right) - \gamma_F - \alpha_L = 0, \tag{4.34a}$$

$$\sum_{i=1}^{n_c} y_{V,i}\left(\bar{G}_i^V - \gamma_i\right) - \gamma_F - \alpha_V = 0, \tag{4.34b}$$

$$L\left(\bar{G}_i^L + \sum_{j=1}^{n_c}\left(x_j\frac{\partial \bar{G}_j^L}{\partial x_{L,i}}\right) - \gamma_i\right) - \gamma_S = 0, \quad i = 1, \ldots, n_c, \tag{4.34c}$$

$$V\left(\bar{G}_i^V + \sum_{j=1}^{n_c}\left(y_j\frac{\partial \bar{G}_j^V}{\partial y_{V,i}}\right) - \gamma_i\right) + \gamma_S = 0, \quad i = 1, \ldots, n_c, \tag{4.34d}$$

$$F - L - V = 0,$$

$$z_{F,i}F - x_{L,i}L - y_{V,i}V = 0, \quad i = 1, \ldots, n_c,$$

$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0,$$

$$L \geq 0, \quad V \geq 0,$$

$$x_{L,i} > 0, \quad y_{V,i} > 0, \quad i = 1, \ldots, n_c,$$

$$0 \leq \alpha_L \perp L \geq 0,$$
$$0 \leq \alpha_V \perp V \geq 0,$$

From the Gibbs-Duhem relation, the summations in the LHS of Equations (4.34c) and (4.34d) are zero.[122] There are now three phase regimes to consider.

Liquid-vapor coexistence: In this regime, $L > 0$ and $V > 0$. The complementary slackness conditions therefore imply $\alpha_L = 0$ and $\alpha_V = 0$. The KKT conditions for this case then read

$$\sum_{i=1}^{n_c} x_{L,i} \left(\bar{G}_i^L - \gamma_i\right) - \gamma_F = 0, \tag{4.35a}$$
$$\sum_{i=1}^{n_c} y_{V,i} \left(\bar{G}_i^V - \gamma_i\right) - \gamma_F = 0, \tag{4.35b}$$
$$L \left(\bar{G}_i^L - \gamma_i\right) - \gamma_S = 0, \quad i = 1, \ldots, n_c, \tag{4.35c}$$
$$V \left(\bar{G}_i^V - \gamma_i\right) + \gamma_S = 0, \quad i = 1, \ldots, n_c, \tag{4.35d}$$
$$F - L - V = 0,$$
$$z_{F,i}F - x_{L,i}L - y_{V,i}V = 0, \quad i = 1, \ldots, n_c,$$
$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0,$$
$$L > 0, \quad V > 0,$$
$$x_{L,i} > 0, \quad y_{V,i} > 0, \quad i = 1, \ldots, n_c.$$

From Equations (4.35c) and (4.35d), it can be seen that

$$\gamma_i = \bar{G}_i^L - \gamma_S/L, \quad i = 1, \ldots, n_c, \tag{4.36a}$$
$$\gamma_i = \bar{G}_i^V + \gamma_S/V, \quad i = 1, \ldots, n_c. \tag{4.36b}$$

Substituting Equation (4.36a) into (4.35a) yields

$$\gamma_F = (\gamma_S/L) \sum_{i=1}^{n_c} x_{L,i}, \tag{4.37}$$

$$= (\gamma_S/L) \sum_{i=1}^{n_c} y_{V,i},$$

where the latter follows from equality of the sums of the mole fractions. Substituting both these results and Equation (4.36b) into Equation (4.35b) gives

$$(\gamma_S/L + \gamma_S/V) \sum_{i=1}^{n_c} y_{V,i} = 0, \tag{4.38}$$

and since $\sum_{i=1}^{n_c} y_{V,i} > 0$, it must be that $\gamma_S/L = -\gamma_S/V$. Therefore, Equations (4.36a) and (4.36b) imply that $\bar{G}_i^L = \bar{G}_i^V$ in this case. This equilibrium condition can be rewritten as

$$G_i^0 + RT \ln\left(\frac{\hat{f}_i^L}{f_i^0}\right) = G_i^0 + RT \ln\left(\frac{\hat{f}_i^V}{f_i^0}\right), \tag{4.39}$$

where R is the gas constant; $\hat{f}_i^L$ and $\hat{f}_i^V$ are the partial fugacities of component i in the liquid and vapor phases, respectively; G_i^0 is the pure component molar Gibbs free energy at an arbitrary standard state at the system temperature and f_i^0 is the fugacity of pure component i in this standard state. This expression reduces to equality of the partial fugacities, $\hat{f}_i^L = \hat{f}_i^V$. These partial fugacities can be written as $\hat{f}_{L,i} = x_{L,i}\hat{\phi}_i^L(T, P, \mathbf{x}_L)P$ and $\hat{f}_{V,i} = y_{V,i}\hat{\phi}_i^V(T, P, \mathbf{y}_V)P$, where $\hat{\phi}_i^L$ and $\hat{\phi}_i^V$ are fugacity coefficients. Noting $k_i := \frac{\hat{\phi}_i^L}{\hat{\phi}_i^V}$, the following results:

$$y_{V,i} = k_i x_{L,i}, \quad i = 1, \ldots, n_c, \tag{4.40}$$

which is the well-known equilibrium condition for two coexisting phases.

Liquid-only: In this regime, $L > 0$ and $V = 0$. The complementary slackness conditions therefore imply $\alpha_L = 0$ and $\alpha_V \geq 0$. The KKT conditions for this case then read

$$\sum_{i=1}^{n_c} x_{L,i} \left(\bar{G}_i^L - \gamma_i\right) - \gamma_F = 0, \tag{4.41a}$$

$$\sum_{i=1}^{n_c} y_{V,i} \left(\bar{G}_i^V - \gamma_i\right) - \alpha_V - \gamma_F = 0, \tag{4.41b}$$

$$L\left(\bar{G}_i^L - \gamma_i\right) - \gamma_S = 0, \quad i = 1, \ldots, n_c, \tag{4.41c}$$

$$V\left(\bar{G}_i^V - \gamma_i\right) + \gamma_S = 0, \quad i = 1, \ldots, n_c, \tag{4.41d}$$

$$F - L - V = 0,$$

$$z_{F,i}F - x_{L,i}L - y_{V,i}V = 0, \quad i = 1, \ldots, n_c,$$

$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0,$$

$$L > 0, \quad V = 0, \quad \alpha_V \geq 0,$$

$$x_{L,i} > 0, \quad y_{V,i} > 0, \quad i = 1, \ldots, n_c.$$

From Equation (4.41c), it can be seen that

$$\gamma_i = \bar{G}_i^L - \gamma_S/L, \quad i = 1, \ldots, n_c, \tag{4.42}$$

and as in the previous case, substituting Equation (4.42) into Equation (4.41a) yields

$$\begin{aligned}\gamma_F &= (\gamma_S/L)\sum_{i=1}^{n_c} x_{L,i} \\ &= (\gamma_S/L)\sum_{i=1}^{n_c} y_{V,i}.\end{aligned}$$

Substituting this along with Equation (4.42) into Equation (4.41b) gives the following inequality:

$$\sum_{i=1}^{n_c} y_{V,i}\left(\bar{G}_i^V - \bar{G}_i^L\right) \geq 0. \tag{4.43}$$

Introducing partial fugacities and equilibrium coefficients as before, this is equivalent to

$$\sum_{i=1}^{n_c} y_{V,i}\ln(y_{V,i}) \geq \sum_{i=1}^{n_c} y_{V,i}\ln(k_i x_{L,i}). \tag{4.44}$$

This inequality is satisfied for any constant $\beta \geq 1$ such that

$$y_{V,i} = \beta k_i x_{L,i}, \quad i = 1, \ldots, n_c. \tag{4.45}$$

Vapor-only: In this regime, $L = 0$ and $V > 0$. The complementary slackness conditions therefore imply $\alpha_V = 0$ and $\alpha_L \geq 0$. Following an analogous procedure to the previous case, the following inequality is obtained

$$\sum_{i=1}^{n_c} x_{L,i} \left(\bar{G}_i^L - \bar{G}_i^V \right) \geq 0, \tag{4.46}$$

which is equivalent to

$$\sum_{i=1}^{n_c} x_{L,i} \ln(k_i x_{L,i}) \geq \sum_{i=1}^{n_c} x_{L,i} \ln(y_{V,i}). \tag{4.47}$$

This inequality is satisfied for any constant $0 < \beta \leq 1$ such that

$$y_{V,i} = \beta k_i x_{L,i}, \quad i = 1, \ldots, n_c. \tag{4.48}$$

Taken together, the KKT conditions of the three regimes give that $y_{V,i} = \beta k_i x_{L,i}$, where $\beta = 1$ when both phases exist, $\beta \geq 1$ when no vapor phase exists and $\beta \leq 1$ when no liquid phase exists. This behavior can be captured by an expression in terms of the mid operator, and so the three equality constraints in the Gibbs free energy minimization plus the implications of its KKT conditions are equivalent to the nonsmooth model presented in the previous section:

$$F = L + V, \tag{4.49a}$$

$$z_{F,i} F = x_{L,i} L + y_{V,i} V, \quad i = 1, \ldots, n_c, \tag{4.49b}$$

$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0, \tag{4.49c}$$

$$y_{V,i} = \beta k_i x_{L,i}, \quad i = 1, \ldots, n_c, \tag{4.49d}$$

$$\text{mid}\,(\alpha,\ \beta - 1,\ \alpha - 1) = 0. \tag{4.49e}$$

To obtain a formulation where β is eliminated as a variable, the substitutions $y_{V,i} \equiv \frac{y'_{V,i}}{\sum_{i=1}^{n_c} y'_{V,i}}, \forall i$ with each $y'_{V,i} > 0$ and $x_{L,i} \equiv \frac{x'_{L,i}}{\sum_{i=1}^{n_c} x'_{L,i}}, \forall i$ with each $x'_{L,i} > 0$ are made.

An alternative formulation expressed in terms of $x'_{L,i}$ and $y'_{V,i}$ is as follows:

$$F = L + V, \tag{4.50a}$$

$$z_{F,i}F = \frac{x'_{L,i}}{\sum_{i=1}^{n_c} x'_{L,i}}L + \frac{y'_{V,i}}{\sum_{i=1}^{n_c} y'_{V,i}}V, \quad i = 1, \ldots, n_c, \tag{4.50b}$$

$$\sum_{i=1}^{n_c} \frac{y'_{V,i}}{\sum_{i=1}^{n_c} y'_{V,i}} - \sum_{i=1}^{n_c} \frac{x'_{L,i}}{\sum_{i=1}^{n_c} x'_{L,i}} = 0, \tag{4.50c}$$

$$\frac{y'_{V,i}}{\sum_{i=1}^{n_c} y'_{V,i}} = \beta k'_i \frac{x'_{L,i}}{\sum_{i=1}^{n_c} x'_{L,i}}, \quad i = 1, \ldots, n_c, \tag{4.50d}$$

$$\text{mid}\,(\alpha,\ \beta - 1,\ \alpha - 1) = 0, \tag{4.50e}$$

where k'_i is evaluated at $T, P, \mathbf{y}'_V$ and $\mathbf{x}'_L$. Note that in the case where the equilibrium coefficient is not a function of composition, $k'_i = k_i, \forall i$. Equation (4.50c) is clearly always satisfied and can be eliminated from the formulation. From Equation (4.50d), take $\beta \equiv \frac{\sum_{i=1}^{n_c} x'_{L,i}}{\sum_{i=1}^{n_c} y'_{V,i}}$, so that Equations (4.50d) and (4.50e) can be rewritten

$$y'_{V,i} = k'_i x'_{L,i}, \quad i = 1, \ldots, n_c, \tag{4.51}$$

$$\text{mid}\left(\alpha,\ \sum_{i=1}^{n_c} x'_{L,i} - \sum_{i=1}^{n_c} y'_{V,i},\ \alpha - 1\right) = 0. \tag{4.52}$$

Finally, noting that when $L = 0$, $y'_{V,i} = y_{V,i}, \forall i$ to preserve material balance, and likewise when $V = 0$, $x'_{L,i} = x_{L,i}, \forall i$, Equation (4.50b) can be rewritten without reference to the summation terms, which yields the alternate nonsmooth model from the previous section with $x'_{L,i}$, $y'_{V,i}$ and k'_i replacing $x_{L,i}$, $y_{V,i}$ and k_i everywhere. These new variables can be thought of as *pseudo* mole fractions and equilibrium coefficients that reduce to the physical mole fractions when their corresponding phase exists, and to the true equilibrium coefficient when two phases coexist. For notational simplicity, the superscript is omitted elsewhere. Moreover, the only difference between the formulation using Equation (4.49e) and the formulation using Equation (4.52) are the values assigned to the equilibrium coefficients and the composition of a missing phase in the single phase regimes. In the former, the mole fraction values are constrained to sum to unity even for a nonexistent phase, whereas in the latter, the pseudo values

will sum to less than unity (and the equilibrium coefficients will differ accordingly). The solution is therefore only changed mathematically; there is no physical difference between the results given by the two models.

Practical usage of Equation (4.26) is now demonstrated in a small example.

Example 4.2. A refrigerant mixture flowing at a rate of 1.0 mol/sec with the molar composition 5.82% N_2, 20.62% CH_4, 39.37% C_2H_6 and 34.19% $n-C_4H_{10}$ is initially at 298.15 K and 0.1 MPa. Assuming ideal thermodynamic behavior and Raoult's Law, a series of PQ-flash calculations are performed by solving Equations (4.7) and (4.26) for the flash temperature and outlet vapor fraction. The compositions of the resulting phases are then calculated with Equations (4.9) and (4.10). The details of the ideal physical property model used is found in Appendix B and the pure component model parameters for these correlations were obtained from Aspen Plus v8.4.[5]

The semismooth Newton method was used to solve the nonsmooth equation system in each case, with iterations stopping after the infinity norm of the function residuals was less than 10^{-9}. The specifications and results are detailed in Table 4.2. Figure 4-6 shows the nonsmooth behavior of the value of the left-hand side of Equation (4.26) as a function of the vapor fraction at the solution conditions for Case I. In all cases, the solutions were found in only a few iterations, even in Case III where the initial guess is in the two-phase region but the vapor outlet is not present at the solution, and similarly in Case IV, where the liquid outlet does not exist. Results were also verified against the results of identical simulations in Aspen Plus, which were found to predict virtually identical outlet conditions in all cases, with the exception that the mole fractions of phases not present are not normalized to sum to unity here.

4.5 Flowsheet simulation with multiphase MHEXs

The nonsmooth model components developed in Sections 4.3 and 4.4 can be combined with the base multistream heat exchanger model from Chapter 3 in order to simulate MHEXs in complex processes where the phase behavior of the streams is

Table 4.2: Flash specifications and results for the refrigerant mixture in Example 4.2.

	Case I	**Case II**	**Case III**	**Case IV**
Specification				
Pressure (MPa)	0.24	0.24	0.24	0.24
Heat duty (kW)	-10.0	-20.0	-30.0	10.0
Initial guess				
Temperature (K)	250	200	150	300
Vapor fraction	0.6	0.5	0.4	0.7
Results				
Iterations	7	7	6	7
Temperature (K)	244.02	188.40	104.42	434.55
Vapor fraction	0.698	0.319	0.0	1.0
Liquid composition (mol %)				
Nitrogen	0.01	1.34	5.82	3.33×10^{-5}
Methane	0.38	3.23	20.62	3.26×10^{-3}
Ethane	11.31	46.54	39.37	0.23
n-Butane	88.30	50.10	34.19	1.88
Vapor composition (mol %)				
Nitrogen	8.33	17.98	25.32	5.82
Methane	29.35	57.82	4.59	20.62
Ethane	51.48	24.04	4.42×10^{-3}	39.37
n-Butane	10.84	0.15	4.36×10^{-8}	34.19

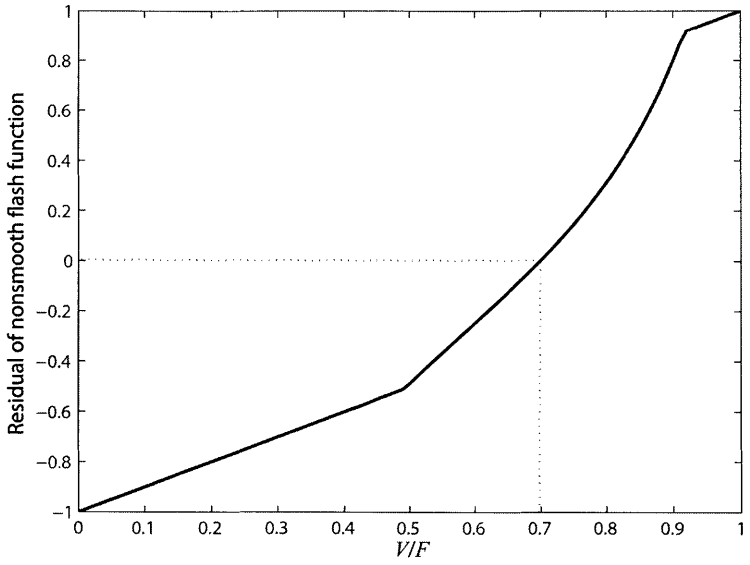


Figure 4-6: Plot of the value of the nonsmooth flash function as a function of $\frac{V}{F}$ at the solution conditions of Case I of Example 4.2.

not known *a priori.* In many such cases, splitting a physical stream into the three substreams described thus far will be insufficient to capture the true extent of the nonlinear temperature-enthalpy relationship. Therefore, as described in Kamath *et al.*,[59] the superheated, subcooled, and two-phase substreams are further discretized into $n_{\text{sup}}, n_{\text{sub}}$ and $n_{2\text{p}}$ affine segments, respectively, to further improve the approximation of the nonlinear behavior. The substreams are subdivided into segments of equal enthalpy difference. This means the inlet/outlet temperatures of each segment are implicitly given by solving energy balances in each phase. For example, the temperatures intervals along the superheated substream are defined by the following equations:

$$F\left(h^{\text{in}}_{\text{sup},i}(T^{\text{in}}_{\text{sup},i}) - h^{\text{out}}_{\text{sup},i}(T^{\text{out}}_{\text{sup},i})\right) = \frac{Q_{\text{sup}}}{n_{\text{sup}}}, \quad i = 1, \ldots, n_{\text{sup}} - 1, \tag{4.53}$$

with Q^{sup} as the total heat transferred by the superheated substream. The temperatures in the subcooled region are found analogously, while the temperatures (and corresponding vapor fractions) in the two-phase region are implicitly defined by Equations (4.7) and (4.26). Once the temperatures and heat loads of each segment are

known, a constant heat capacity flowrate can be determined for every stream, e.g. as follows for the superheated region of a hot stream:

$$F_{C_p,i} = F\frac{\left(h^{\text{in}}_{\text{sup},i}(T^{\text{in}}_{\text{sup},i}) - h^{\text{out}}_{\text{sup},i}(T^{\text{out}}_{\text{sup},i})\right)}{\left(T^{\text{in}}_{\text{sup},i} - T^{\text{out}}_{\text{sup},i}\right)}, \quad i = 1, \ldots, n_{\text{sup}}, \tag{4.54}$$

and analogously for the other phase regions and for cold streams. Note that the use of (4.22) and (4.23) will never allow this quotient to become undefined. These heat capacity flowrates are then used in evaluating Equations (3.4), (3.13) and (3.14). The total size of the MHEX model is then given by:

$$n_{\text{var}} = 3 + 2n_{\text{streams}} + n_{\text{streams}}\left[(n_{\text{sup}} - 1) + (n_{\text{sub}} - 1) + 2(n_{2\text{p}} - 1)\right], \tag{4.55}$$

where n_{var} is the number of variables/equations needed to model the MHEX, and n_{streams} is the number of physical process streams that enter the MHEX in the flowsheet. The first term of Equation (4.55) accounts for the base MHEX model consisting of Equations (3.4), (3.13) and (3.14), the second term accounts for calculating the dew and bubble points of each stream involved in the heat exchanger, and the last term accounts for all the temperatures that must be calculated for the piecewise-affine segments. The calculation of the dew and bubble points is necessary to track the point where each stream changes phase, which in the multicomponent case requires the solution of nonlinear equations.

Despite the reduction in complexity as compared to other approaches in the literature, the model remains difficult to solve. Most significantly, the LP-Newton method does not exhibit the same invariance to affine scaling as the classical and semismooth Newton methods. Scaling of the equations such that the residual values are bounded by ± 1 over the domain of interest results in substantially improved convergence behavior compared to solving the same system with poorly-scaled equations. Additionally, improved performance of the solution algorithm was observed by replacing all the infinity norms in Equation (2.26) with the 1-norm. The change of norm was motivated by the observation that the step size calculated by Equation (2.26) at each iteration

is bounded by unity if the largest residual value corresponds to an equation with a zero row in the generalized derivative. This can occur for Equation (3.13), as any temperature variables in the simulation only influence the minimum distance between the composite curves over a limited range of values, as shown in Chapter 3.

Finally, it is important that the initial guess provided be as near to the solution as possible to aid convergence. However, providing good initial guesses for the many unknown temperatures is challenging, and so a robust initialization subroutine was developed for such a purpose. In this procedure, only guesses for the three MHEX model variables and the bubble/dew point temperatures of each stream are required from the user. The bubble and dew point estimates are then refined by solving Equations (B.12) and (B.13). Initial guesses for the remaining temperatures are obtained by first assuming an affine relationship between temperature and enthalpy in each phase, and then improved by solving each of the energy balance equations independently to generate a better temperature estimate. This initial point is then passed to the main flowsheet simulation routine.

Example 4.3 shows the method developed in this chapter being applied to simulate the PRICO process under the assumption of ideal thermodynamic behavior and Raoults Law.

Example 4.3. Figure 1-1 shows the PRICO process for producing LNG. As described previously, the PRICO process is a single-stage single mixed refrigerant (SMR) process with the MR stream supplied at two pressure levels in the process by means of expansion and compression operations. The MR stream therefore serves as both a hot stream, high-pressure refrigerant (HPR), and as a cold stream, low-pressure refrigerant (LPR). The full ideal physical property model used in this example is given in Appendix B, and the pure component model parameters for these correlations were obtained from Aspen Plus.[5] Aspen Plus simulations were also used to validate the model solutions, as shown in the individual cases that follow.

Tables 4.3 and 4.4 give the data for the streams involved in the MHEX for the PRICO process under three different sets of simulation conditions. In each case, $n_{\text{sup}} = 4$, $n_{\text{2p}} = 8$ and $n_{\text{sub}} = 8$ for all three process streams entering the heat

Table 4.3: Natural gas stream data for Example 4.3.

Property	Natural gas
Flowrate (kmol/s)	1.00
Pressure (MPa)	5.500
Inlet temperature (K)	298.15
Outlet temperature (K)	118.15
Composition (mol %)	
Nitrogen	1.0
Methane	95.6
Ethane	3.1
Propane	0.2
n-Butane	0.1

exchanger (20 segments for each stream if all three phases exist). By Equation (4.55), the MHEX model contributes a total of 81 equations and variables to the problem.

Variable Set I: In this case, the pressures and compositions in the flowsheet are held fixed and $\Delta T_{\min}$ is specified as 1.2 K. Let $x_1 \equiv T_{\text{HPR}}^{\text{OUT}}$, $x_2 \equiv t_{\text{LPR}}^{\text{OUT}}$ and $x_3 \equiv UA$ be the unknown variables afforded by the base MHEX model consisting of Equations (3.4), (3.13) and (3.14), while $x_4 \equiv t_{\text{LPR}}^{\text{IN}}$ and $x_5 \equiv \alpha_{\text{LPR}}^{\text{IN}}$ are given by performing a fixed pressure/enthalpy flash calculation around the throttle valve. The remainder of the variable set comprises the internal variables of the MHEX model: unknown temperatures given by the energy balances of the form shown in Equation (4.53) in the superheated and subcooled regions, and unknown temperatures and vapor fractions given by Equations (4.7) and (4.26) in the two-phase region. Note that in this case with the pressure and composition held fixed, the bubble and dew points can be computed in a preprocessing step.

Using the CPLEX callable library v12.5[55] as the LP solver, the simulation converges to a solution with $\|\mathbf{f}(\mathbf{x}^*)\|_\infty < 10^{-9}$ after 107 iterations of the modified LP-Newton method. The initial point and the solution values for the key variables in this simulation are shown in Table 4.5. The remainder of the variables had initial points generated by the initialization subroutine. To validate these results, an Aspen Plus model was also built to run the PRICO process under ideal thermodynamics and Raoult's Law using the MHeatX block discretized into 20 zones, with the option

Table 4.4: Refrigerant stream and MHEX data for Example 4.3.

Property	Case I	Case II	Case III
Flowrate (kmol/s)	3.47	3.47	3.47
High pressure level (MPa)	3.695	3.695	3.695
Low pressure level (MPa)	0.165	0.165	x_1
HPR inlet temperature (K)	303.15	303.15	303.15
HPR outlet temperature (K)	x_1	107.22	118.15
LPR inlet temperature (K)	x_4	x_4	x_4
LPR outlet temperature (K)	x_2	x_2	x_2
Composition (mol %)			
Nitrogen	15.32	15.32	15.32
Methane	17.79	17.79	17.79
Ethane	40.85	40.85	40.85
Propane	0.41	$26.04 - 100x_1$	0.41
n-Butane	25.62	$100x_1$	25.62
UA (MW/K)	x_3	17.5	10
$\Delta T_{\min}$ (K)	1.2	x_3	x_3

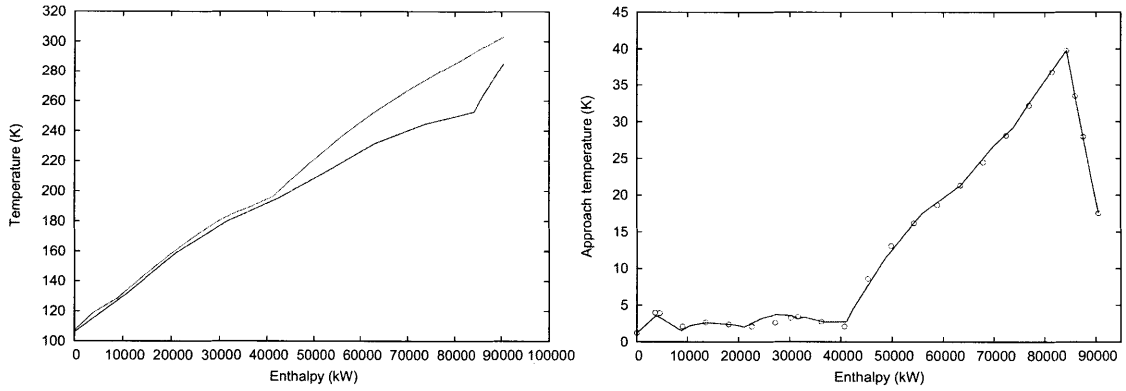
enabled to automatically add additional points at stream inlets and the bubble/dew points. The MHeatX block in Aspen Plus only allows for the outlet condition of one stream to be left as a variable, which was chosen as $T_{\text{LPR}}^{\text{OUT}}$, while $T_{\text{HPR}}^{\text{OUT}}$ was specified as the solution value from the nonsmooth model simulation. Note that if instead, $T_{\text{LPR}}^{\text{OUT}}$ is specified, the Aspen Plus simulation is unable to converge to a physical solution (despite the existence of such a solution). Additionally, the Aspen Plus simulation model has no way to enforce the $\Delta T_{\min}$ constraint, so it is simply a calculated output of the simulation. The numerical results of this simulation for the key variables in the process are shown in Table 4.5 and show good agreement with the nonsmooth model results.

Figure 4-7(a) shows the resulting composite curves for this simulation and Figure 4-7(b) shows the profile of the temperature difference between the hot and cold composite curves from both the nonsmooth model and the Aspen Plus simulation, which show good agreement. In spite of the piecewise affine approximations used, it is clear that the calculated composite curves capture much of the true profile curvature, particularly around the bubble point of the natural gas stream (196.5 K).

Variable Set II: In this case, the composition of the refrigerant mixture is allowed

Table 4.5: Numerical results for the process simulation in Case I. Quantities marked with a † were specified rather than calculated. Note that the set of specifications changes between the nonsmooth model and the Aspen Plus simulation.

Quantity	Initial Guess	Solution Value	Aspen Simulation
$T_{\text{HPR}}^{\text{OUT}}$ (K)	100	107.22	107.22†
$T_{\text{LPR}}^{\text{OUT}}$ (K)	300	285.58	285.59
UA (MW/K)	20	18.82	19.03
$T_{\text{LPR}}^{\text{IN}}$ (K)	98.8	116.02	106.02
$\alpha_{\text{LPR}}^{\text{IN}}$	0.10	0.025	0.025
ΔT_{min} (K)	1.2†	1.2†	1.21



(a) (b)

Figure 4-7: (a.) Hot composite curve (red) and cold composite curve (blue) for the MHEX in the PRICO process simulated in Case I. (b.) Approach temperature profile for the MHEX in the PRICO process simulated in Case I from the nonsmooth model (solid line) and the Aspen Plus simulation (open circles).

to vary and the conductance of the exchanger is fixed at 17.5 MW/K. Let x_1 be the mole fraction of n-butane in the refrigerant, $x_2 \equiv T_{\text{LPR}}^{\text{OUT}}$ and $x_3 \equiv \Delta T_{\text{min}}$ with x_4 and x_5 as before. The simulation converges to a solution with $\|\mathbf{f}(\mathbf{x}^*)\|_\infty < 10^{-9}$ after 151 iterations of the modified LP-Newton method. The initial point and the solution values for the key variables in this simulation are shown in Table 4.6. The initial point for the remainder of the variables was generated by the initialization subroutine. These results were verified as detailed previously; however in Aspen Plus, the refrigerant composition has to be specified as an input to the model and the conductance value is a calculated output, in contrast to the inverse problem that our

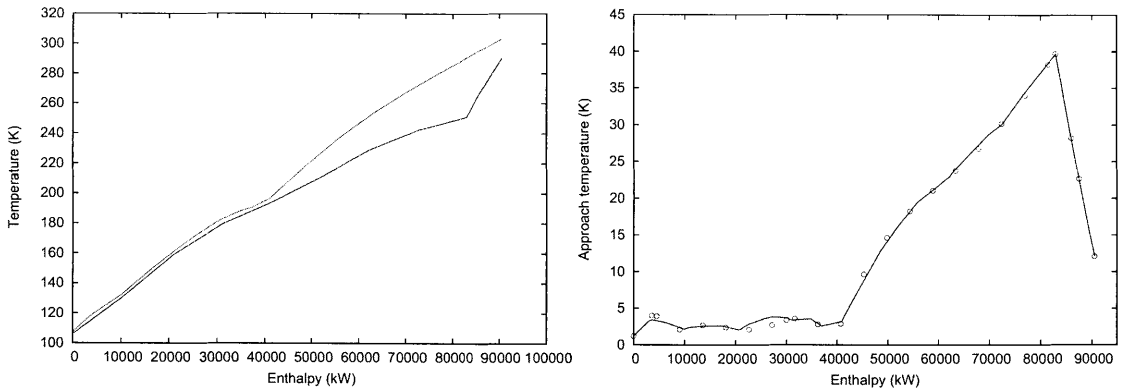
model is able to solve. The numerical results of the Aspen Plus simulation for the key variables in the process are shown in Table 4.6. The results once again mostly show good agreement with the proposed model results, with the exception of a slight over-prediction of the conductance by Aspen Plus, which can be attributed to its discretization missing some of the profile nonlinearity between 20,000 kW and 30,000 kW (see Figure 4-8(b.)).

Table 4.6: Numerical results for the process simulation in Case II. Quantities marked with a † were specified rather than calculated. Note that the set of specifications changes between the nonsmooth model and the Aspen Plus simulation.

Quantity	**Initial Guess**	**Solution Value**	**Aspen Simulation**
$T_{\text{HPR}}^{\text{OUT}}$ (K)	107.22†	107.22†	107.22†
$T_{\text{LPR}}^{\text{OUT}}$ (K)	280	291.06	291.06
UA (MW/K)	17.5†	17.5†	18.22
$T_{\text{LPR}}^{\text{IN}}$ (K)	106.02	106.01	106.01
$\alpha_{\text{LPR}}^{\text{IN}}$	0.10	0.025	0.025
ΔT_{min} (K)	1.2	1.21	1.21
n-Butane mole %	25.62	22.58	22.58†

Figure 4-8 shows (a.) the resulting composite curves and (b.) the approach temperature profiles from both the model proposed in this chapter and the Aspen Plus simulation. Due to the specification on the available heat transfer area (which is lower than the value calculated in Case I), the separation between the composite curves is slightly greater than in Case I.

Variable Set III: In this case, the pressure of the MR stream is allowed to vary while the conductance of the exchanger is fixed to only 10 MW/K. Let x_1 be the discharge pressure of the throttle valve, $x_2 \equiv T_{\text{LPR}}^{\text{OUT}}$ and $x_3 \equiv \Delta T_{\text{min}}$ with x_4 and x_5 as before. The simulation converges to a solution with $\|\mathbf{f}(\mathbf{x}^*)\|_\infty < 10^{-9}$ after 158 iterations of the modified LP-Newton method. The initial point and the solution values for the key variables in this simulation are shown in Table 4.6. The initial point for the remainder of the variables was generated by the initialization subroutine. These results were again verified using Aspen Plus, though in this case, the valve outlet pressure has to be specified as an input to the model while the



(a) (b)

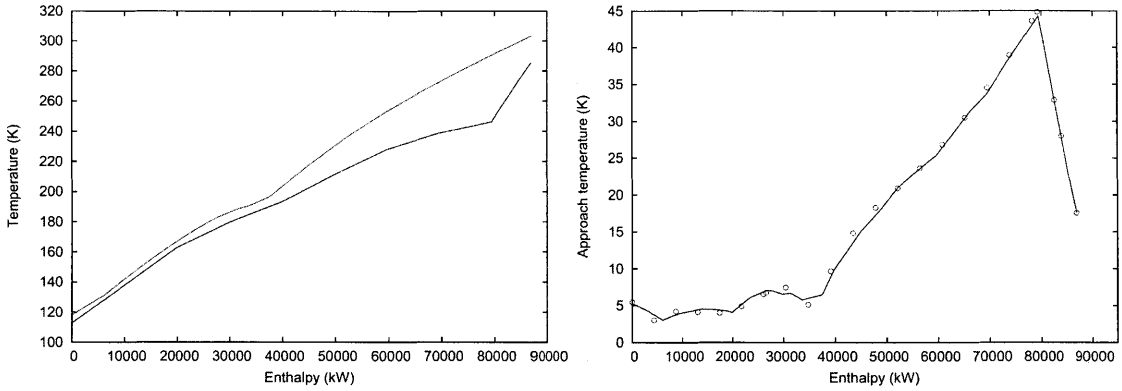
Figure 4-8: (a.) Hot composite curve (red) and cold composite curve (blue) for the MHEX in the PRICO process simulated in Case II. (b.) Approach temperature profile for the MHEX in the PRICO process simulated in Case II from the nonsmooth model (solid line) and the Aspen Plus simulation (open circles).

conductance is a calculated output. The numerical results of the Aspen simulation for the key variables in the process are shown in Table 4.7, which show excellent agreement with the nonsmooth model.

Table 4.7: Numerical results for the process simulation in Case III. Quantities marked with a † were specified rather than calculated. Note that the set of specifications changes between the nonsmooth model and the Aspen Plus simulation.

Quantity	**Initial Guess**	**Solution Value**	**Aspen Simulation**
$T_{\text{HPR}}^{\text{OUT}}$ (K)	118.15†	118.15†	118.15†
$T_{\text{LPR}}^{\text{OUT}}$ (K)	280	285.58	285.59
UA (MW/K)	10.0†	10.0†	10.02
$T_{\text{LPR}}^{\text{IN}}$ (K)	116.95	112.86	112.66
$\alpha_{\text{LPR}}^{\text{IN}}$	0.10	0.122	0.122
ΔT_{min} (K)	1.2	3.04	3.03
Low pressure level (MPa)	0.165	0.122	0.122†

Figure 4-9 shows (a.) the resulting composite curves and (b.) the approach temperature profiles from the model proposed in this chapter and the Aspen Plus simulation. Satisfying the conductance specification requires the pressure at the throttle valve exit to approach atmospheric pressure, leading to a larger temperature change across the valve and greater separation of the composite curves.



(a) (b)

Figure 4-9: (a.) Hot composite curve (red) and cold composite curve (blue) for the MHEX in the PRICO process simulated in Case III. (b.) Approach temperature profile for the MHEX in the PRICO process simulated in Case III from the nonsmooth model (solid line) and the Aspen Plus simulation (open circles).

In each case, the nonsmooth model is able to simulate the PRICO process successfully. Cases II and III particularly are challenging simulation problems since many of the equations in the model are sensitive to the composition and pressure of the refrigerant stream. However, as a result of the strategy developed in these previous sections, the results are found without requiring significant computational time or effort. Additionally, note that while Aspen Plus is a useful tool for validating the results of the nonsmooth model, it cannot solve the problems proposed in Cases I, II and III directly.

In the following, the results from these case studies are expanded upon to assess the true impact of the nonsmooth toolkit. Flowsheet simulations with each of the three variable sets were performed from a large number of initial guesses taken from a uniform grid in a region containing a solution. The performance of three methods for evaluating the generalized derivative information needed to solve the flowsheeting problems was compared:

- **LD**: B-subdifferential elements computed using LD-derivatives evaluated in the identity directions;
- **DD**: generalized derivative elements approxmated by concatenating directional

derivatives in the coordinate directions (i.e. the naïve construction from Equation (2.18)); and

- **FD**: generalized derivative elements approximated using finite differences.

The results are presented in Table 4.8. The finite differencing approach performs poorly, particularly for simulations with Variable Set II in which the method frequently fails to converge to the solution. Additionally, even in the cases where the finite differencing approach finds the solution, on average significantly more iterations are required to converge compared to the other methods.

Table 4.8: Iteration count statistics for simulating the PRICO process from a range of initial guesses with different variable sets and methods used to calculate generalized derivatives.

	Variable Set I			**Variable Set II**			**Variable Set III**		
x_1	[100, 160]			[0.2, 0.3]			[0.1, 0.2]		
x_2	[250, 300]			[275, 300]			[275, 300]		
x_3	20			[0.5, 3.0]			[0.5, 3.0]		
N	143			216			216		
Method	**LD**	**DD**	**FD**	**LD**	**DD**	**FD**	**LD**	**DD**	**FD**
Solve %*	100.0	100.0	97.2	100.0	100.0	20.4	82.8	82.8	79.6
Mean	217.7	218.2	225.8	139.8	140.7	159.7	129.3	130.3	191.5
Std. dev.	129.4	129.6	137.1	37.0	36.7	27.2	49.3	49.1	76.5
Median	206	207	217	138	138.5	155	130	130	178.5
Min	25	25	25	61	62	125	31	33	50
Max	455	455	483	206	206	218	221	223	471

[1]Percentage of simulations that converged in fewer than 500 iterations of the LP-Newton method. Statistics are only based on those instances that met this criterion.

In contrast, the LD and DD methods perform very similarly. However, the fact that a difference exists at all between the LD and DD methods implies that at least some of the simulations in each case must encounter points of nonsmoothness, i.e., that the set of points at which the model is nondifferentiable is reachable. Since the difference is not substantial, it is clear that the LP-Newton method is robust enough to be able to converge even with the slightly incorrect approximate generalized derivatives provided by the DD approach at these points. Based on the number

of simulations in which the iteration count differs between the two approaches, nonsmooth points are encountered in 12.6% (18/143) of Variable Set I simulations, 18.5% (40/216) of Variable Set II simulations and 45.4% (98/216) of Variable Set III simulations. To better test the extent to which this subset of simulations encountered nonsmooth points, these problems were rerun using the LD-derivative approach; this time however, at each iteration the approximate generalized derivative from DD was also constructed. These matrices only potentially differ exactly at points of nonsmoothness, so a calculation of the induced 1-norm of the difference of these matrices was used to detect nonsmoothness along the iterate sequence of the LD-derivative approach. Statistics on the number of nonsmooth points encountered in these simulations are presented in Table 4.9, and the corresponding histograms are shown in Figure 4-10.

Table 4.9: Nondifferentiable point count statistics from PRICO process simulations in which at least one such point was encountered.

	Variable Set I	**Variable Set II**	**Variable Set III**
Mean	7.9	3.9	7.6
Std. dev.	7.5	7.4	8.2
Median	5	2	4
Min	1	1	1
Max	27	46	37

As these numerical results demonstrate, it is entirely possible to visit multiple nondifferentiable points while solving an equation system with a high degree of nonsmoothness. This highlights the need for computing generalized derivative information accurately and automatically for such problems using the methods reviewed in this chapter.

4.6 Conclusions

A new method for simulating phase changes in multistream heat exchangers has been presented. The proposed method differs significantly from those presented thus far

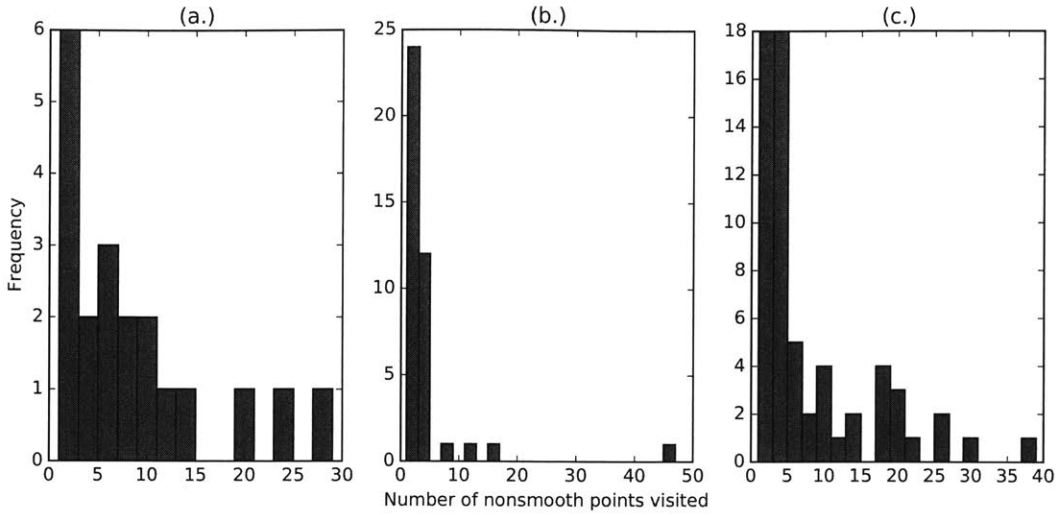


Figure 4-10: Histograms of the number of nonsmooth points encountered in MHEX simulations: (a.) Variable Set I (b.) Variable Set II (c.) Variable Set III.

in the literature in that it does not require solving a difficult optimization problem involving binary variables or complementarity constraints. Nonsmooth equations are used to define the variable inlet/outlet temperatures of substreams corresponding to different phase behavior, as well as to perform vapor-liquid equilibrium calculations robustly. This leads to a compact nonsmooth model that is solved purely through equation-solving methods. Accordingly, the model is significantly less complex and allows for realistic simulation of process flowsheets involving multiphase MHEXs outside of an optimization framework.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Nonsmooth inside-out algorithms for robust flash calculations

Dependable algorithms for nonideal vapor-liquid equilibrium calculations are essential for effective process design, simulation and optimization. Inside-out algorithms for flash calculations serve as the basis for many of the algorithms used by process simulation software due to their robustness with respect to initialization and inexpensive computational cost.[16] However, if the specified flash conditions imply a single-phase result, the conventional inside-out algorithms fail, as the solution is constrained to obey equilibrium relationships which are only valid in the two-phase region. These incorrect results can be post-processed to determine the true single-phase solution; however, such approaches either carry a high computational cost or are heuristic in nature and vulnerable to failure (or both). Such attributes are undesirable in a process simulation/optimization problem where many flash calculations must be performed for streams where the phase regime at the solution is not known *a-priori*. To address this issue, this chapter presents modifications of the classical inside-out algorithms using a nonsmooth equation system in the inner loop to relax equilibrium conditions when necessary, allowing reliable convergence to single-phase results. Numerical results for simulations involving several common flash types and property packages are shown, highlighting the capability of the new nonsmooth algorithms for handling both two-phase and single-phase behavior robustly and efficiently.

5.1 Introduction

The ability to solve vapor-liquid equilibrium (flash) calculations consistently and efficiently is critical for many process systems engineering applications. These calculations are necessary for accurate process simulation, whether for single-stage flash unit operations, for larger operations such as distillation columns, for performing energy balance calculations on mixed-phase streams, etc. The flash equations are well-known to be challenging to solve, particularly for non-ideal systems, and naïve solution methods will generally fail unless an extremely good initial guess of the result is provided.

The early work of Rachford and Rice[100] proposed an often-used formulation of the equations which was more amenable to numerical solution methods. In the following decades, many algorithms were suggested based on the Rachford-Rice formulation, with much attention given to nested-loop style algorithms. In these procedures, values of certain variables are fixed by an outer loop, while the remaining variables are used to converge a subset of the equations in an inner loop. The choice of inner versus outer loop variables is generally determined by whether the mixture at hand is wide-boiling or narrow-boiling.[36] An excellent summary of such methods can be found in King's perennial separations text.[70] However, these methods were largely superseded by the work of Boston and Britt,[16] who developed the "inside-out" class of flash algorithms, so-named because they use a unique nested-loop strategy which entirely separates the problem of converging the flash equations from the calculation of all rigorous thermophysical properties. Unlike the other approaches suggested at the time, the inside-out algorithms were broadly applicable, whether the problem dealt with wide-boiling or narrow-boiling mixtures, or ideal or highly non-ideal systems while remaining computationally inexpensive.

To this day, the Boston-Britt inside-out formulation (and modifications thereof) serves as the basis of the primary algorithms for flash calculations used by process simulation software such as Aspen Plus,[5] due to both its reliability and efficiency. The inside-out paradigm has also been extended to handle more complex behavior, such as

simultaneous chemical and phase equilibrium calculations.[147] However, these inside-out algorithms were developed assuming that the solution of the flash calculation would always lie in the two-phase vapor-liquid coexistence regime. In many situations, this is certainly the case; however, if it is unknown *a priori* whether the result of a flash calculation will actually be a two-phase mixture, this can be problematic. This can easily occur in the course of solving a process simulation or optimization problem where the convergence or optimization algorithm adjusts the flash parameters to values that lead to an all-vapor or all-liquid result. Methods which post-process the results to find the true single-phase answer are often used; however, these tend to rely on heuristics and/or increase the computational cost of the flash calculation and are vulnerable to failure.

Several authors have suggested reliable (non-heuristic) techniques for handling uncertainty in the number of equilibrium phases present in a mixture. Phase stability calculations, as developed by Michelsen,[81;82] check whether or not a postulated number of equilibrium phases is stable using the Gibbs tangent plane criterion. However, this ostensibly requires the solution of a global optimization problem for each set of trial phases until the set corresponding to a stable mixture is found, and then further requires the solution of the appropriate phase-split problem to determine the physical results. Mitsos and Barton recast this formulation by using the dual of the tangent plane criterion, which removes the guess-and-check nature of the solution process but still requires the solution of multiple global optimization problems.[84] In a similar vein to the guess-and-check approach, interval arithmetic based methods can also be applied to the flash equations to determine reliably whether or not a solution exists. An interval Newton/generalized bisection approach[110] can be used to reduce and partition an initial box which is thought to contain the solution(s) to the flash equations iteratively, such that each new box generated is also guaranteed to contain a solution, if it exists. If the method terminates having found no solution, then the user can be sure that there is indeed no equilibrium solution. However, such a method is both computationally expensive (particularly if not implemented using code generation) and prone to slow convergence or stalling before reducing boxes to

high accuracy, especially when there are points within the domain with singular Jacobians/generalized derivatives. Finally, fully equation-oriented approaches to the flash problem which are agnostic to the number of phases present have also been proposed. In such a formulation, the problem of determining if a single-phase mixture is present is generally recast using complementarity constraints.[40;41;60] However, this leads to a substantial increase in complexity (a single 3-component flash requires 86 variables and 93 equations/inequalities to model in the formulation of Kamath *et al.*[60]) and solutions must be obtained by solving an optimization problem in which complementarity constraints are modeled appropriately. Such a formulation also requires an initial guess for the values of all physical and non-physical variables, which can be potentially challenging to generate without *a priori* information about either the solution or the result of a similar flash calculation and can result in slow convergence when it is far from the true solution.[60]

Instead, this chapter proposes modified algorithms which retain all the benefits of the Boston-Britt inside-out algorithms without sacrificing reliability due to heuristics, therefore extending the celebrated robustness and efficiency of the original algorithms to finding single-phase solutions.

5.2 Classical inside-out algorithms

The formulations of the flash equations discussed in the previous chapter are often difficult and computationally expensive to solve when the equilibrium ratios and enthalpy departure functions are defined by complex models. To this end, Boston and Britt proposed a new class of solution algorithms for the single-stage flash problem, the inside-out algorithms.[16] As mentioned previously, this class of algorithms is almost universally applicable to different types of flash calculations and mixtures. Impressively, this versatility does not come at the price of high computational cost, as the method also reduces as much as possible the number of required calls to the thermophysical property system for fugacity and enthalpy evaluations.

The inside-out approach is an iterative nested loop procedure, and the two loops

are referred to as the inner loop and the outer loop throughout this work. In the outer loop, complex phase equilibrium and enthalpy models are used to generate parameters for simple models of the equilibrium ratios and vapor and liquid enthalpy departure functions. The constants and coefficients involved in these simple models become the independent variables in the flash problem and serve as proxies for the actual temperature, pressure, vapor/liquid composition and vapor fraction of the system. Then, in the inner loop, the simplified models are used to converge the flash equations. The nested procedure repeats until the values of the outer loop iteration variables do not change significantly from iteration to iteration, at which point the flash equations are solved. The outer loop variables were carefully designed to be independent of each other and not to depend strongly on multiple physical quantities, e.g., both temperature and composition. Due to this, the performance of the algorithm is not strongly dependent on good initial guesses, unlike many other such algorithms.

The inside-out formulation of the PQ-flash algorithm is now discussed. The main outer loop iteration variables are referred to as "volatility parameters", and defined as:

$$u_i = \ln(k_i/k_b), \tag{5.1}$$

where k_b is a reference equilibrium ratio defined by the following equations:

$$\ln k_b \equiv \sum_{i=1}^{n_c} w_i \ln k_i, \tag{5.2}$$

$$w_i \equiv \frac{t_i}{\sum_{i=1}^{n_c} t_i}, \tag{5.3}$$

$$t_i \equiv \frac{y_{V,i} \left(\frac{\partial \ln k_i}{\partial T}\right)_{P,\mathbf{x}_L,\mathbf{y}_V}}{1 + \alpha(k_i - 1)}, \tag{5.4}$$

where w_i and t_i are weighting factors derived by Boston and Britt. Accuracy of the partial derivative terms in Equation (5.4) is not essential as the weights themselves were derived from approximations of more complex conditions; therefore it is most conveniently approximated by finite differences since the procedure already requires

evaluation of the equilibrium ratios at two temperature levels, as will be explained next. The dependence of k_b on temperature is represented by the following model:

$$\ln\ k_b = A + B\left(\frac{1}{T} - \frac{1}{T^{\text{ref}}}\right), \tag{5.5}$$

where A and B are variables in the outer loop and T^{ref} is a user-defined reference temperature. In the algorithm, the variable B is calculated by evaluating a second reference equilibrium ratio, k_b', at a second temperature level, T', with the same pressure and composition as k_b, as follows:

$$B = \frac{\ln(k_b'/k_b)}{\left(\frac{1}{T'} - \frac{1}{T}\right)}, \tag{5.6}$$

and then A is determined from the following relationship:

$$A = \ln k_b - B\left(\frac{1}{T} - \frac{1}{T^{\text{ref}}}\right). \tag{5.7}$$

The inner loop iteration variable, R, is given by the following identity:

$$R \equiv \frac{k_b V}{k_b V + k_b^0 L}, \tag{5.8}$$

where k_b^0 is just a constant used to avoid numerical issues when k_b becomes very large or very small. Boston and Britt also introduced a vector $\mathbf{p}$ with each element defined as follows:

$$p_i \equiv \frac{l_i}{1-R} = \frac{f_i}{1 - R + k_b^0 R\mathrm{e}^{u_i}}, \tag{5.9}$$

where l_i are the individual component liquid phase flow rates. Only the second identity in Equation (5.9) is actually used in the implementation of the algorithm. The value of defining the quantities p_i in this way is that it allows k_b to be expressed entirely in terms of $\mathbf{p}$ and $\mathbf{u}$:

$$k_b = \frac{\sum_{i=1}^{n_c} p_i}{\sum_{i=1}^{n_c} \mathrm{e}^{u_i} p_i}. \tag{5.10}$$

The flash temperature, liquid phase flowrate and liquid/vapor phase compositions

can then be calculated from these quantities as follows:

$$T = \frac{1}{\frac{1}{T^{\text{ref}}} + \frac{\ln k_b - A}{B}}, \tag{5.11}$$

$$L = (1 - R) \sum_{i=1}^{n_c} p_i, \tag{5.12}$$

$$x_{L,i} = \frac{p_i}{\sum_{i=1}^{n_c} p_i}, \tag{5.13}$$

$$y_{V,i} = \frac{e^{u_i} p_i}{\sum_{i=1}^{n_c} e^{u_i} p_i}. \tag{5.14}$$

Knowing the temperature, pressure and compositions of each phase allows the vapor and liquid enthalpies to be calculated for the energy balance residual function in the inner loop:

$$\Psi = (L/F)(\Delta h_V - \Delta h_L) - h_F^{\text{id}} - \Delta h_V + h_F + Q_{\text{flash}}/F, \tag{5.15}$$

where Δh_L and Δh_V are the liquid and vapor enthalpy departures, respectively, and $h_F^{\text{id}} \equiv \sum_{i=1}^{n_c} z_{F,i} h_i^{\text{id}}$ is the ideal gas enthalpy of a mixture with the feed stream's composition at the flash temperature. The individual component ideal gas enthalpies, h_i^{id}, are functions of temperature only and can be evaluated using a suitable empirical correlation. Note that by definition:

$$\Delta h_V \equiv h_V - h_V^{\text{id}} = h_V - \sum_{i=1}^{n_c} y_{V,i} h_i^{\text{id}}, \tag{5.16}$$

$$\Delta h_L \equiv h_L - h_L^{\text{id}} = h_L - \sum_{i=1}^{n_c} x_{L,i} h_i^{\text{id}}. \tag{5.17}$$

Inspection of Equations (5.9) - (5.15) reveals that Ψ is actually only a function of R for a given feed condition and fixed values of the outer loop variables. Thus, in this algorithm, the inner loop consists of a single variable iteration procedure, varying R to satisfy Equation (5.15) following the calculation sequence just described. Additionally, when evaluating Ψ, the models for the liquid and vapor enthalpy departures

are also expressed in terms of simpler models (analogous to the simplified equilibrium ratio model) with only temperature dependence:

$$\Delta h_V = C + D(T - T^{\text{ref}}), \tag{5.18}$$

$$\Delta h_L = E + F(T - T^{\text{ref}}), \tag{5.19}$$

where C, D, E and F are also variables in the outer loop and are updated in an analogous manner to A and B by evaluating the real liquid and vapor enthalpy departure functions at a second temperature level (T^{ref} is an appropriate choice). Therefore, there are a total of $n_c + 6$ outer loop iteration variables in the inside-out PQ-flash formulation, which will be represented in shorthand by the vector $\boldsymbol{v}$:

$$\boldsymbol{v} \equiv (\mathbf{u}, A, B, C, D, E, F) \tag{5.20}$$

The inside-out algorithm terminates when the vector $\boldsymbol{v}$ does not change substantially from iteration to iteration; therefore, the outer loop error function, Ω, is defined as follows:

$$\Omega \equiv ||\hat{\boldsymbol{v}} - \boldsymbol{v}||_\infty, \tag{5.21}$$

where $\hat{\boldsymbol{v}}$ is the vector of calculated iteration variables after a pass through the outer loop with the vector $\boldsymbol{v}$ as a starting point.

As mentioned previously, owing to the use of these largely independent non-physical variables, the performance of this algorithm is quite insensitive to the quality of the initial guess. However, there is a substantial amount of initialization that must take place to begin calculations, even if it requires almost no input from the user. One such initialization strategy that has been used in this work is given by Algorithm 5.1. Alternatively, should the user have a better initial guess for the solution (e.g. from a previous, similar flash calculation), then this information can be used in place of that which is calculated in Steps 1, 2 and 5.

The full implementation of Boston and Britt's inside-out strategy is given in Algorithm 5.2. The inner loop convergence problem can be solved with a simple Newton-

Algorithm 5.1: An initialization subroutine for the PQ-flash inside-out algorithm.

1 Guess $T = 0.8 \sum_{i=1}^{n_c} z_{F,i} T_i^{\text{crit}}$ and $\alpha = 0.5$
2 Solve (4.7) and (4.26) for T, α assuming ideal thermophysical properties and Raoult's Law
3 Set $T^{\text{ref}} \leftarrow T - 1$
4 Set $T' \leftarrow T + 1$
5 Calculate $\mathbf{x}_L, \mathbf{y}_V$ from (4.9) and (4.10) assuming Raoult's Law
6 Calculate $\mathbf{k}(\mathbf{x}_L, \mathbf{y}_V, P, T)$ using real property models
7 Calculate $\mathbf{k}'(\mathbf{x}_L, \mathbf{y}_V, P, T')$ using real property models
8 Calculate k_b and k_b' using (5.2) - (5.4)
9 Set $k_b^0 \leftarrow k_b'$
10 Calculate $\mathbf{u}, A, B$ from (5.1), (5.6), (5.7)
11 Calculate $\Delta h_V(\mathbf{y}_V, P, T)$ and $\Delta h_L(\mathbf{x}_L, P, T)$ using real property models
12 Calculate $\Delta h_V^{\text{ref}}(\mathbf{y}_V, P, T^{\text{ref}})$ and $\Delta h_L^{\text{ref}}(\mathbf{x}_L, P, T^{\text{ref}})$ using real property models
13 Set $C \leftarrow \Delta h_V^{\text{ref}}$
14 Calculate D from (5.18)
15 Set $E \leftarrow \Delta h_L^{\text{ref}}$
16 Calculate F from (5.19)

type iteration. Outer loop convergence can be achieved through simple successive substitution, though can be accelerated through derivative-based procedures. As noted in the original paper, a full Newton-type iteration is more computationally expensive than necessary; however, the Broyden method works very well since the Jacobian matrix for the iteration variables generally does not differ significantly from the identity matrix. Additionally, since the variables B, D and F each require two calls to the real thermophysical property models to update, they are only updated once during the procedure for efficiency.

However, if the solution of the flash problem actually lies in a single-phase region, the iterates generated by Algorithm 5.2 will reach either the bubble or dew point and then fail to improve further, despite the residual error in the energy balance. To see this, observe that the value of R can only vary between 0 and 1 in any physical solution, as dictated by Equation (5.8). In the inner loop, only R is allowed to vary, so Equations (5.9) and (5.10) define a single value of k_b for a given R value. However, Equation (5.10) assumes that vapor-liquid equilibrium holds, so $R = 0$ gives a bubble point (BP) k_b value, and $R = 1$ gives a dew point (DP) k_b value. Therefore, outside

Algorithm 5.2: Boston and Britt's inside-out algorithm for a PQ-flash.

```
Initialize υ and set T', T^ref, k'_b, k^0_b, e.g. by using Algorithm 5.1
Calculate initial guess for R using (5.8)
Set Ω ← 2ε_out
while Ω > ε_out do
    while |Ψ| > ε_in do
        Calculate p, k_b, T, L from (5.9) - (5.11), (5.12)
        Calculate h_F^id
        Calculate Δh_V, Δh_L from (5.18)-(5.19)
        Calculate Ψ from (5.15)
        Assume new value of R
    end while
    Calculate α, x_L, y_V from (4.8), (5.13), (5.14)
    Calculate k, Δh_L, Δh_V (and k', Δh_L^ref, Δh_V^ref on first iteration only) using
    real property models
    Calculate υ̂ using (5.1), (5.6), (5.7), (5.18), (5.19)
    Calculate Ω from (5.21)
    Assume new values for u, A, C, E (and B, D, F on first iteration only)
end while
```

of the two-phase region, there is no way to satisfy the phase equilibrium relationships for all components while independently adjusting the temperature (which is found from k_b using Equation (5.11)) to satisfy the energy balance.

As this is a known short-coming of the method, several methods have been proposed to mitigate this behavior for PQ-flash types:

1. The (documented[92]) Aspen Plus[5] approach is to start from the non-converged all-liquid ($R = 0$) or all-vapor ($R = 1$) result returned by the PQ-flash inside-out algorithm and perform an iterative calculation that varies the temperature in order to satisfy the unconverged energy balance. Once a candidate temperature for the (possibly) single-phase solution is found, a PT-flash is performed. Based on the result of this calculation, the single-phase candidate solution is either accepted or rejected. If it is rejected, the PQ-flash is restarted using an improved initial guess provided by the PT-flash results.

2. The approach suggested by Parekh and Mathias[92] is similar to the Aspen approach, except that if an all-liquid) or an all-vapor solution is indicated on *any*

iteration of the PQ-flash procedure, the temperature iteration is performed.

3. A more rigorous approach, which relies on the work of Michelsen,[81;82] is to perform phase-stability analysis on any possibly single-phase solution returned by the PQ-flash algorithm ($R = 0$ or $R = 1$ with inner loop residual error). If this procedure confirms that a single-phase is present, then a temperature iteration is performed starting from the values suggested by the stability analysis. Otherwise, the PQ-flash is restarted using the initial estimate of the phase composition provided by the analysis.

The third option is computationally expensive and not a reasonable option when a large number of flash calculations need to be performed. The first two options tend to be inexpensive and show comparable behavior to each other (it is not entirely clear which approach is implemented in the modern versions of Aspen Plus), however, they are heuristic approaches which cannot provide a guarantee of success, as will be shown later in an example problem. Especially near bubble or dew points, all three approaches can potentially cycle back-and-forth between the single-phase and two-phase iterations, increasing solution time in the average case and not converging at all in the worst case. The need for a computationally inexpensive modification of this inside-out algorithm that will produce correct results for all flash conditions is therefore apparent.

5.3 Proposed Algorithms

To address the inability of the conventional Boston-Britt inside-out algorithms to converge automatically to single-phase solutions, a modification of the procedure is proposed in this section based on the ideas introduced in the simple nonsmooth flash formulation introduced in Chapter 4.

The central idea of the modification is to add an extra variable to the inner loop to avoid the problem discussed in the previous section of $R = 0$ always corresponding to a bubble point solution and $R = 1$ always corresponding to a dew point solution.

Since the objective is to relax the phase equilibrium constraint outside of the two-phase region, it is reasonable to use a variant of Equation (4.27) as the residual for the new variable. Replacing α with R in this equation yields the following:

$$\text{mid}\left(R,\ \sum_{i=1}^{n_c} x_{L,i} - \sum_{i=1}^{n_c} y_{V,i},\ R-1\right) = 0.$$

Now, the term involving the liquid and vapor compositions in this equation can be rewitten in terms of the inside-out formulation variables as follows:

$$\begin{aligned}
\text{mid}\left(R,\ \sum_{i=1}^{n_c} l_i/L - \sum_{i=1}^{n_c} v_i/V,\ R-1\right) &= 0,\\
\text{mid}\left(R,\ \sum_{i=1}^{n_c} l_i/L - \sum_{i=1}^{n_c} k_i l_i/L,\ R-1\right) &= 0,\\
\text{mid}\left(R,\ \sum_{i=1}^{n_c} l_i - \sum_{i=1}^{n_c} k_b e^{u_i} l_i,\ R-1\right) &= 0,\\
\text{mid}\left(R,\ (1-R)\sum_{i=1}^{n_c} p_i - k_b(1-R)\sum_{i=1}^{n_c} e^{u_i} p_i,\ R-1\right) &= 0,\\
\text{mid}\left(R,\ \sum_{i=1}^{n_c} p_i - k_b \sum_{i=1}^{n_c} e^{u_i} p_i,\ R-1\right) &= 0. \qquad (5.22)
\end{aligned}$$

From inspection of this last equation, k_b is clearly a reasonable choice for the additional variable in the new inner loop. Thus, the inner convergence problem is now to solve the following nonsmooth equation system:

$$\boldsymbol{\Psi}(R, k_b) \equiv \begin{bmatrix} \text{mid}\left(R,\ \sum_{i=1}^{n_c} p_i - k_b \sum_{i=1}^{n_c} \mathrm{e}^{u_i} p_i,\ R-1\right) \\ (L/F)(\Delta h_V - \Delta h_L) - h_F^{\text{id}} - \Delta h_V + h_F + Q_{\text{flash}}/F \end{bmatrix} = \mathbf{0}, \qquad (5.23)$$

wherein the original single variable problem based on satisfying the energy balance has been transformed into a two variable problem which also includes equilibrium and material balance considerations. Note that since k_b is treated as a variable in addition to R, it is no longer determined as a function of R by Equation (5.10) during the inner loop computational sequence. In the two-phase region, the solution of this equation system both satisfies Equation (5.10) and zeroes the residual of Equation

(5.15), exactly as in the original procedure. Outside the two-phase region, the energy balance is still always satisfied, while the first equation allows for relaxation of the equilibrium relationships using k_b as a new degree of freedom.

There are also a few subtler changes that must be made from Algorithm 5.2. The first is that it is no longer possible to ignore updating the variables B, D and F after just the first iteration. This is due to the fact that if the initial guess provided to or calculated by the algorithm suggests an incorrect number of phases, then B, D and F will need to be substantially adjusted to avoid extremely slow convergence, and there is no guarantee that the algorithm will predict the correct number of phases after just one iteration (especially if the solution is close to the mixture bubble/dew point). Therefore, all $n_c + 6$ outer loop variables must be calculated in each iteration. As in the classical case, a full-Newton type iteration is not recommended, as the cost of evaluating a single element of the generalized derivative is $3(n_c + 6) + 1 = 3n_c + 19$ times the cost of a function evaluation in the worst case.[64] As will be seen in the numerical examples, the derivative-free methods generally require substantially fewer than $3n_c + 19$ function evaluations to converge. In this work, the most effective method for rapidly converging the outer loop variables has been found to be Anderson acceleration,[2;134] a technique that computes the next iterate as a linear combination of the residuals from the past m iterations of the algorithm. Interestingly, the more widely known Wegstein acceleration technique was found to be largely ineffective for this application, showing very similar performance to basic successive substitution. As the numerical experiments later in this chapter suggest, the use of a version of Broyden's method appropriate for PC^1 functions[71] would not provide a significant advantage over Anderson accelerated successive substitution.

In addition, it has been observed that defining $\mathbf{u}$ using the following rule (in place of Equation (5.1)) is beneficial from a numerical standpoint:

$$u_i = \ln\left(\frac{k_i}{\text{mid}\left(\min_i k_i,\ k_b,\ \max_i k_i\right)}\right). \tag{5.24}$$

This modification is simply to prevent the values of each u_i from becoming very large or very small (and thereby e^{u_i} becoming extremely large or small) in the single-phase regimes when k_b is being used only to adjust the temperature to satisfy the energy balance. The new PQ-flash solution procedure is provided in full in Algorithm 5.3.

Algorithm 5.3: The proposed nonsmooth inside-out algorithm for a PQ-flash.

```
Initialize v and set T', T^ref, k_b', k_b^0 using Algorithm 5.1
Calculate initial guess for R using (5.8)
Set Ω ← 2ε_out
while Ω > ε_out do
    while ||Ψ(R, k_b)||_∞ > ε_in do
        Calculate p, T, L from (5.9), (5.11), (5.12)
        Calculate h_F^id
        Calculate Δh_V, Δh_L from (5.18)-(5.19)
        Calculate Ψ(R, k_b) from (5.23)
        Assume new values of R, k_b
    end while
    Calculate α, x_L, y_V from (4.8), (5.13), (5.14)
    Calculate k, k', Δh_L, Δh_V, Δh_L^ref, Δh_V^ref using real property models
    Calculate v̂ using (5.24), (5.6), (5.7), (5.18), (5.19)
    Calculate Ω from (5.21)
    Set v ← v̂
end while
```

Alternatively, the nonsmooth problem in the inner loop can be written using a nonphysical slack variable β as follows:

$$\boldsymbol{\Psi}(R,\beta) \equiv \begin{bmatrix} \text{mid}\,(R,\ \beta-1,\ R-1) \\ (L/F)(\Delta h_V - \Delta h_L) - h_F^{\text{id}} - \Delta h_V + h_F + Q_{\text{flash}}/F \end{bmatrix} = \mathbf{0}, \tag{5.25}$$

and k_b is then a calculated quantity in the inner loop, though given by the following equation in place of Equation (5.10):

$$k_b = \frac{\sum_{i=1}^{n_c} p_i}{\beta \sum_{i=1}^{n_c} \mathrm{e}^{u_i} p_i}. \tag{5.26}$$

This formulation accomplishes the same relaxation of the equilibrium relationships outside of the two-phase region as described above. This formulation is more

notationally consistent with work found in the equation-oriented simulation and optimization literature, such as in the work of Gopal and Biegler,[40;41] which has also been modified to handle single-phase flash conditions by Kamath *et al.*[60] However, the standard approach in these works is to formulate and solve such problems as large mathematical programs with complementarity constraints, which differs greatly from the highly efficient and reliable nonsmooth modular strategy suggested in this chapter.

Nonsmooth modifications of the inside-out algorithms for other specification sets are also possible. A PT-flash algorithm using the nonsmooth strategy is given as Algorithm 5.4. For the PT-flash, the outer loop and inner loop residual functions are as follows:

$$\Psi_{\mathrm{PT}}(R) \equiv \mathrm{mid}\left(R,\ \sum_{i=1}^{n_c} p_i - k_b \sum_{i=1}^{n_c} \mathrm{e}^{u_i} p_i,\ R-1\right), \tag{5.27}$$

$$\Omega_{\mathrm{PT}} \equiv ||\hat{\mathbf{u}} - \mathbf{u}||_\infty. \tag{5.28}$$

Nonsmooth algorithms for specified pressure-entropy and pressure-internal energy flash calculations can be simply adapted from the nonsmooth PQ-flash algorithm as described by Parekh and Mathias[92] for the original algorithm. The remaining flash types which fix the vapor fraction (or vapor flowrate) are not of interest here, as fixing this quantity automatically determines the phase regime and an appropriate method can be chosen *a priori* to solve the problem.

5.4 Example problems

A series of examples are now provided to show that the proposed algorithm indeed allows flash calculations to converge to single phase regimes while still performing as intended in the two-phase region. The examples described in this section were all coded and solved in the `C++` programming language on an Intel Xeon E5-1650 v2 workstation using six cores at 3.50 GHz and 12 GB RAM running Ubuntu v14.04. All feed streams in this section are assumed to be flowing at a rate of 1 kmol/sec and the

Algorithm 5.4: The proposed nonsmooth inside-out algorithm for a PT-flash.

```
Guess α = 0.5
Solve (4.26) for α assuming Raoult's Law
Calculate x_L, y_V from (4.9) and (4.10) assuming Raoult's Law
Calculate k(x_L, y_V, P, T) using real property models
Set k_b ← 1.0
Calculate u from (5.1)
Set R ← α
Set Ω_PT ← 2ε_out
while Ω_PT > ε_out do
    while |Ψ_PT| > ε_in do
        Calculate p, x_L, y_V, L from (5.9), (5.12) - (5.14)
        Calculate Ψ_PT from (5.27)
        Assume new value of R
    end while
    Calculate α from (4.8)
    Calculate k(x_L, y_V, P, T) using real property models
    Calculate û using (5.1)
    Calculate Ω_PT from (5.28)
    Set u ← û
end while
Calculate Δh_V(y_V, P, T) and Δh_L(x_L, P, T) using real property models
Set h_mix ← α(h_V^id + Δh_V) + (1 − α)(h_L^id + Δh_L)
```

following termination tolerances were used in all examples: $\varepsilon_{\text{in}} = 10^{-9}$, $\varepsilon_{\text{out}} = 10^{-8}$. The inner loop equation solving problem was converged using the semismooth Newton method unless otherwise noted. In each case, the outer loop convergence problem was converged using successive substitution both with and without Anderson acceleration (with $m = 3$) for the purpose of comparison. The least-squares problems associated with the Anderson updates are solved as recommended in the article by Walker and Ni[134] using the `C` interface to LAPACK v3.6.1.[3] All flash calculations were initialized using either Algorithm 5.1 or the first seven steps of Algorithm 5.4; the results of previous flash calculations were never used as initial guesses in order to highlight the robustness of the algorithm with respect to an *ab initio* starting point. All required pure component and binary interaction parameters were obtained from the Aspen Plus v8.4 databanks.[5] The results of the two-phase flash calculations and, when possible, the single-phase flash calculations were validated using Aspen Plus.

Example 5.1. This first example involves a PT-flash of the 5-component hydrocarbon system from Section 5.1.2 of Kamath *et al.*[60] The mixture is 2.5 mol% nitrogen, 65 mol% methane, 15 mol% ethane, 15 mol% propane and 2.5 mol% butane and is initially at 5.5 MPa and 300 K. Both the liquid and vapor phase are described by the Peng-Robinson (PR) cubic equation of state (this differs from the thermodynamic model used in the source article, for the sake of variety in these examples). As in the Kamath et al. article, a series of flash calculations were performed starting from this feed stream and parametrically varying the flash temperature. Results for temperatures in the range from 205 K (subcooled liquid) to 300 K (superheated vapor) were simulated in 0.1 K increments. Algorithm 5.4 was used for all problems, and the results are shown in Figure 5-1. For clarity, only the mole fraction of the most abundant component is shown in this figure and all others in this section. The 951 flash calculations take a total of 0.72 seconds to perform (average 0.76 ms per problem) using Anderson acceleration. Figure 5-2 shows histograms of the total number of outer loop iterations needed to converge the flash calculations, both with and without using Anderson acceleration, showing that the acceleration technique improves the convergence rate substantially. Overall, few outer loop iterations, and therefore few rigorous thermophysical property evaluations, are required to converge the flash calculations.

Another study was performed on the same mixture, this time fixing the temperature at 275 K and varying the pressure parametrically between 0.1 MPa and 12.0 MPa in increments of 0.01 MPa. The results are shown in Figure 5-3. Using Anderson acceleration, the 1,200 flash calculations take a total of 1.07 seconds to perform (average 0.89 ms per problem). This study is particularly interesting because it demonstrates correct prediction of retrograde condensation behavior by the nonsmooth algorithm, which is known to be challenging for simulation-based models using equation of state models.[119;143] The calculations also pass very close to the mixture critical point, as can be seen by the vapor and liquid mole fractions becoming almost indistinguishable from one another at high pressure; however, the values do remain distinct throughout this pressure range.

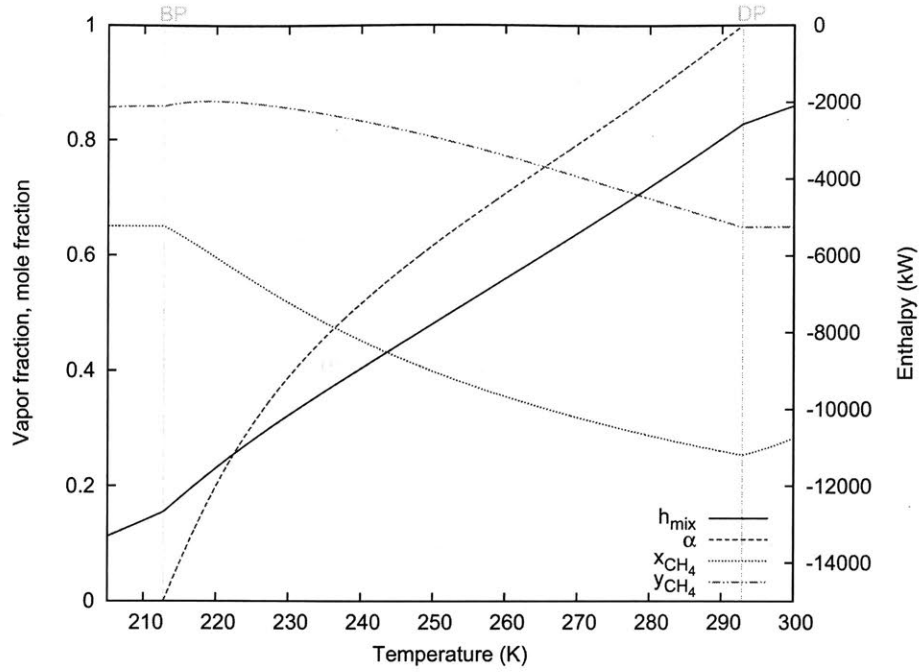


Figure 5-1: Results from parametrically varying the flash temperature in the hydrocarbon mixture problem described in Example 5.1.

The same parametric study was performed in Aspen Plus and the results are compared in Figure 5-4. From this, it appears that the heuristics used in Aspen Plus fail near the higher-pressure dew point, leading to the incorrect reporting of a liquid phase which is inconsistent with the true behavior of the mixture. Aspen Plus does not issue any warnings or errors for the cases where it reports that the vapor fraction is zero, indicating that it considers these results to be accurate. Similar poor performance of the Aspen Plus approach is observed when this same experiment is repeated at fixed temperatures of 260 K and 290 K, and also when holding the pressure constant at 9.5 MPa and varying the temperature (Figure 5-4). Note that if the flash calculation algorithm in Aspen Plus is changed from the inside-out algorithm to the Gibbs tangent plane method, then the results obtained by the nonsmooth inside-out algorithm are verified.

It is important to note that the algorithm presented here is not completely immune to numerical issues at extreme conditions (e.g. above critical temperature and pressure). These issues manifest through convergence to the trivial solution of the

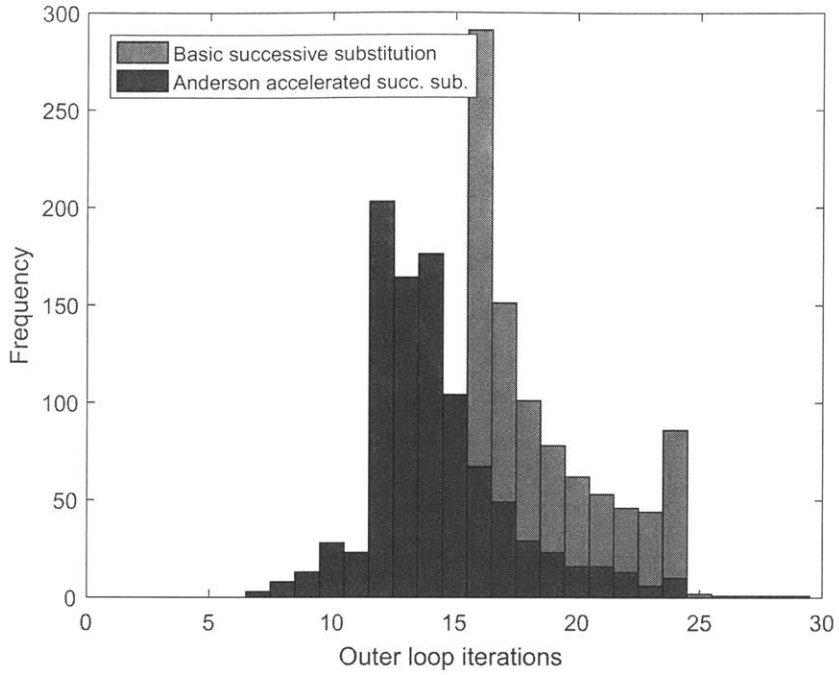


Figure 5-2: Histogram of total number of outer loop iterations needed to converge the flash calculations shown in Figure 5-1 using basic successive substitution (red; mean: 18.5 iterations) and Anderson acceleration (blue; mean: 14.3 iterations).

equilibrium relationships, $k_i = 1, \forall i$, and are caused by difficulties in the cubic root evaluation algorithm (Cardano's analytic method was used in all examples herein which required root finding). However, this is the only instance where the nonsmooth inside-out algorithms have numerical issues, although even in these cases, correct and continuous results for physical quantities (e.g. temperature, vapor fraction and composition of a single existing phase) are still obtained. A detailed discussion of this well-known issue can be found in a 1982 paper by Gundersen,[47] and several other empirical strategies for avoiding this behavior have been reported in the literature.[97;79;151;148] Further discussion of this topic, along with strategies for preventing discontinuous jumps in the solution when varying the flash parameters will be explored in Chapter 6.

Example 5.2. In this example, consider the non-ideal mixture described in Example 3 of Boston and Britt's original paper.[16] The mixture is 10 mol% ethanol, 45 mol% iso-octane and 45 mol% benzene, initially at 348.17 K and 0.101325 MPa. The liquid

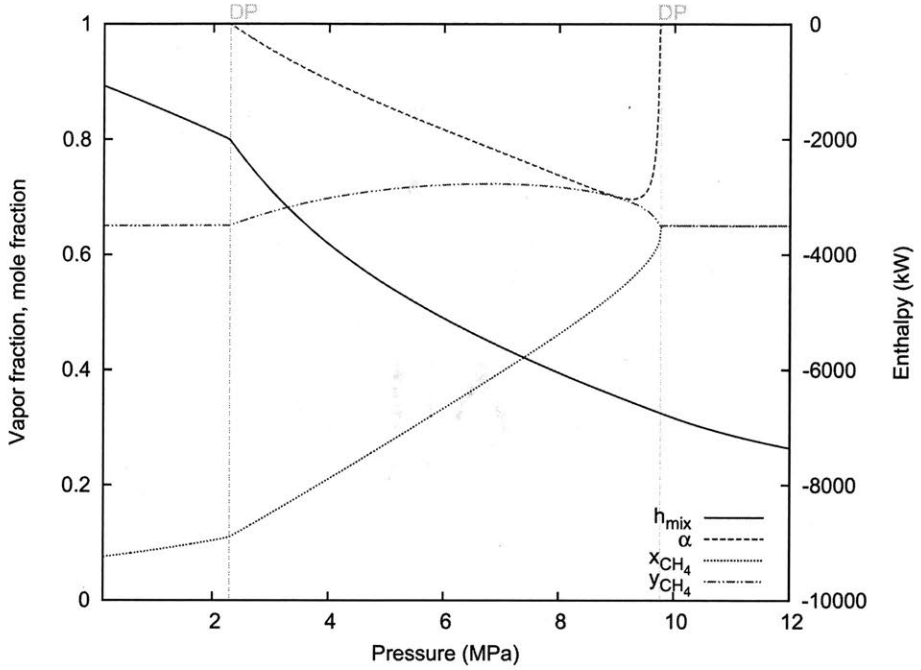


Figure 5-3: Results from parametrically varying the flash pressure in the hydrocarbon mixture problem described in Example 5.1.

phase is described by the Wilson activity coefficient model, while the vapor phase is described by the ideal gas equation of state. Flash calculations were performed starting from the initial point and parametrically varying the heat duty. A PT-flash was performed at the feed conditions to obtain the feed enthalpy. Results for heat loads in the range from -30,000 kW (all-liquid) to 50,000 kW (all-vapor) were simulated in 50 kW increments. The initialization for each problem was performed using Algorithm 5.1, and the results from Algorithm 5.3 are shown in Figure 5-5. In this case, singular or near-singular generalized derivative elements were generated frequently while converging the inner loop. To overcome this, whenever the semismooth Newton method encountered this behavior, the inner loop calculations were restarted and converged using the LP-Newton method instead. When necessary, the LPs given by Equation (2.26) were solved using the CPLEX[55] `C++` callable library v12.5. Using Anderson acceleration, the 1,601 PQ-flash calculations take a total of 6.44 seconds to perform (average 4.0 ms per problem), and Figure 5-6 shows histograms of the total number of outer loop iterations needed to converge the flash calculations in this example. From

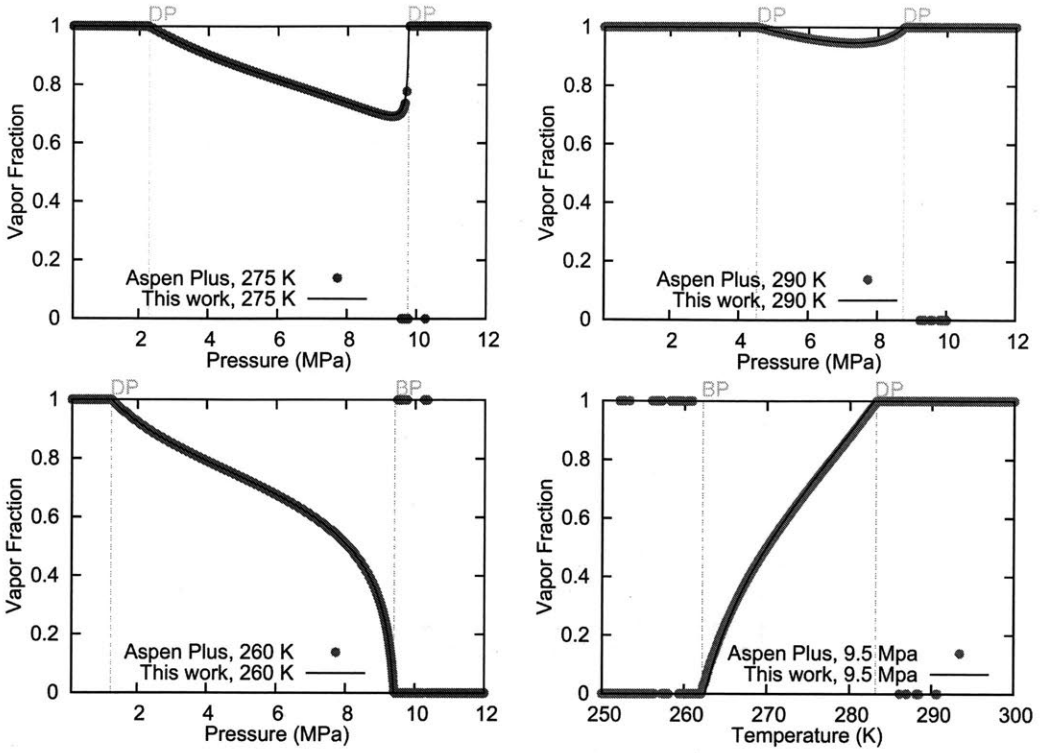


Figure 5-4: Comparison of vapor fraction prediction between Aspen Plus v8.4 and the algorithm described in this work for the hydrocarbon mixture problem described in Example5.1.

these results, it is clear that the nonsmooth inside-out algorithm does not experience difficulty in simulating the highly non-ideal two-phase region, indicating that the robustness of the original algorithm is preserved there, and transitions automatically to performing reliable single-phase simulations when needed.

Example 5.3. In this example, consider a mixture of 50 mol% water, 25 mol% isopropanol and 25 mol% n-butanol initially at 350 K and 0.5 MPa. The liquid phase is described by the non-random two liquid (NRTL) activity coefficient model, while the vapor phase is described by the Redlich-Kwong cubic equation of state. A series of PT-flash calculations were performed starting from the feed conditions and parametrically varying the temperature. Results for temperatures in the range from 350 K (all-liquid) to 470 K (all-vapor) were simulated in 0.1 K increments. The flash calculations were performed as described in Algorithm 5.4, and the results are shown in Figure 5-7. The 1,201 PT-flash calculations take a total of 0.72 seconds to perform (average 0.60

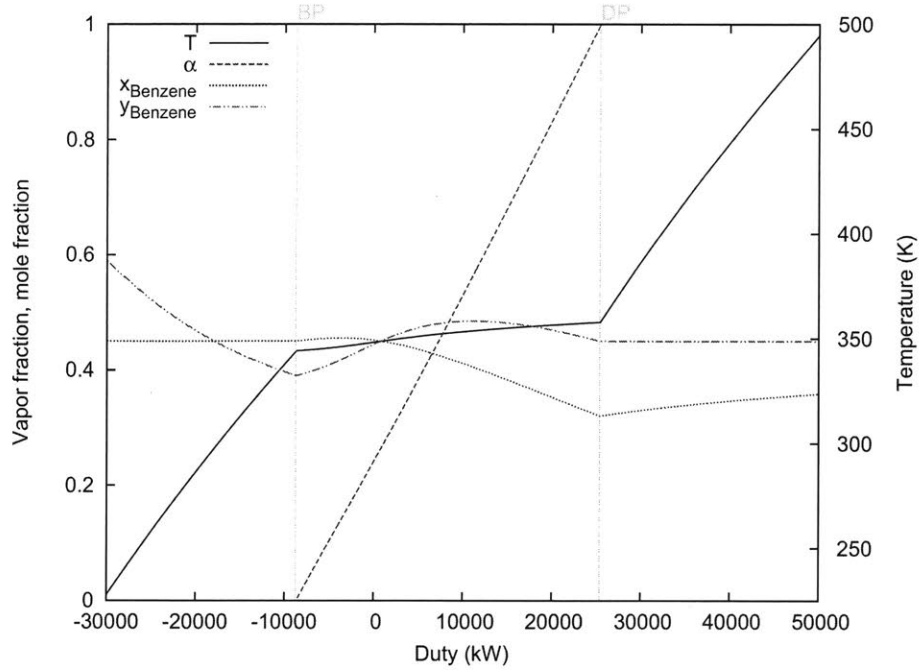


Figure 5-5: Results of the flash calculations performed in Example 5.2.

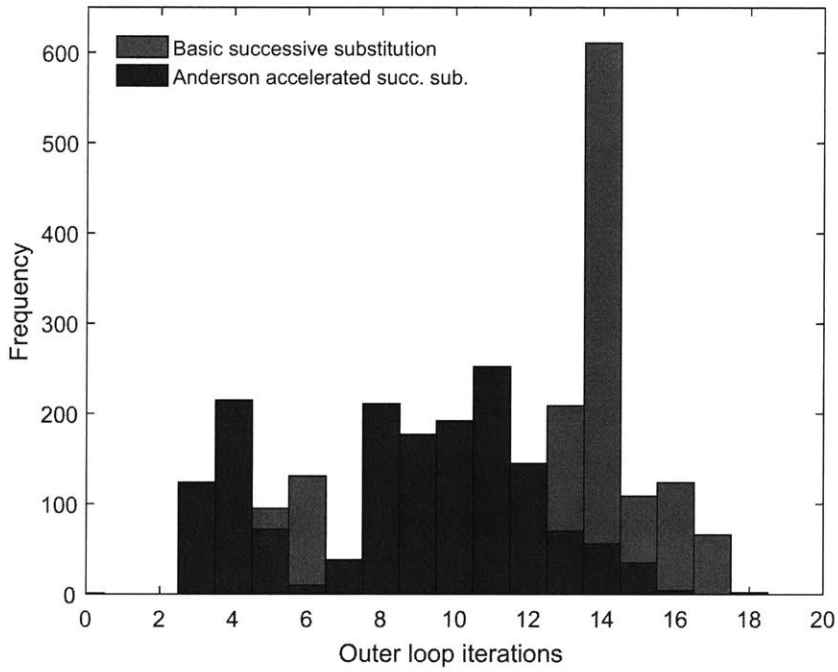


Figure 5-6: Histogram of total number of outer loop iterations needed to converge the flash calculations in Example 5.2 using basic successive substitution (red; mean: 11.7 iterations) and Anderson acceleration (blue; mean: 8.7 iterations).

ms per problem) using Anderson acceleration. Figure 5-8 shows histograms of the total number of outer loop iterations needed to converge the flash calculations in this example. Thus this nonsmooth inside-out algorithm has been demonstrated to work efficiently on the mixed cubic equation of state/activity coefficient model case and reliably performs simulations far into the single-phase regions.

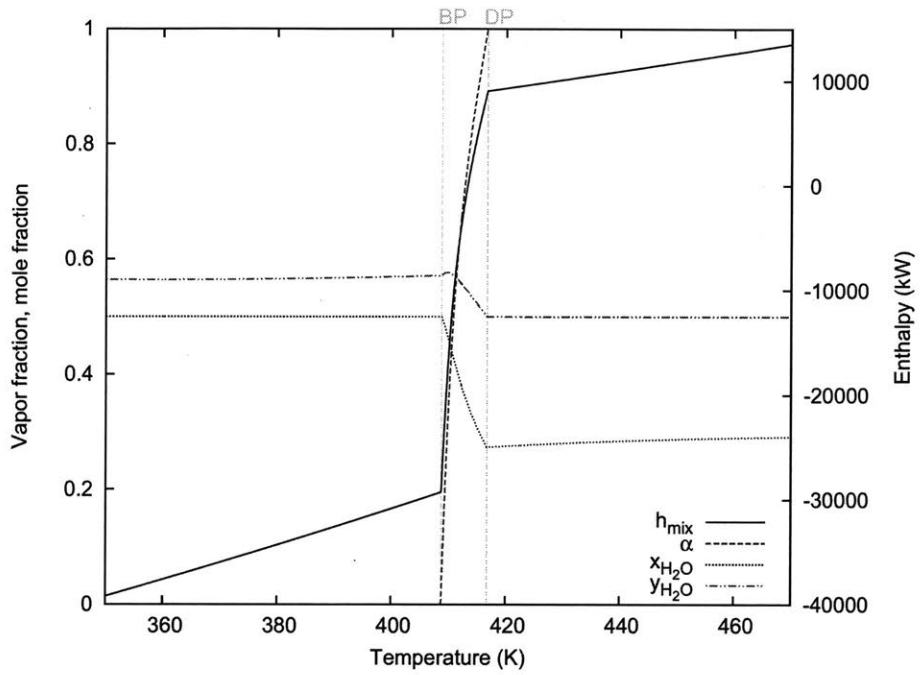


Figure 5-7: Results of the flash calculations performed in Example 5.3.

Example 5.4. This final example involves a PQ-flash of the 3-component air-like mixture from Section 5.1.1 of Kamath et al.[60] The mixture is 60 mol% nitrogen, 35 mol% oxygen and 5 mol% argon and is initially at 0.7 MPa and 95 K. Here, the thermodynamics of the two phases are described by the Soave-Redlich-Kwong (SRK) cubic equation of state. Flash calculations were performed starting from the initial point and parametrically varying the heat duty. A PT-flash was performed at the initial point to obtain the feed mixture enthalpy. Results for heat loads in the range from -1,000 kW (all-liquid) to 6,250 kW (all-vapor) were simulated in 5 kW increments. The initialization for each problem was performed using Algorithm 5.1, and the results of Algorithm 5.3 are shown in Figure 5-9. The 1,451 PQ-flash calculations take a total of 4.77 seconds to perform (average 3.3 ms per problem)

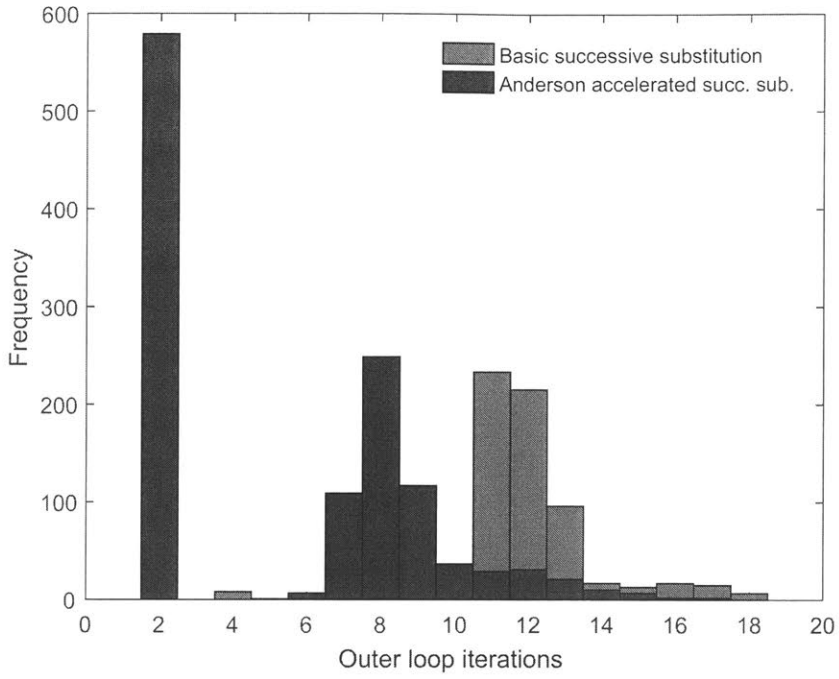


Figure 5-8: Histogram of total number of outer loop iterations needed to converge the flash calculations in Example 5.3 using basic successive substitution (red; mean: 7.2 iterations) and Anderson acceleration (blue; mean: 5.5 iterations).

using Anderson acceleration and Figure 5-10 shows histograms of the total number of outer loop iterations needed to converge these flash calculations. Note that the issue of the trivial solution discussed in the first example manifests in this example beyond $Q = 6300$ kW. At this point, the predicted molar composition of the fictitious liquid phase jumps to become equal to the molar composition of the physical vapor phase. The other physical variables (e.g., temperature, vapor fraction) however, still take correct values at the reported solution.

If the formulation involving β given by Equation (5.25) is used instead of Equation (5.23), then the same values of β reported in the Kamath et al. article are also obtained here, as shown in Figure 5-11. Our formulation involving β also does not avoid the problem of trivial solution convergence (the value of β becomes 1 at all Q beyond 6300 kW).

A study varying the flash pressure was also performed on the same mixture, this time initially at 1.75 MPa and 110 K, by expanding the feed adiabatically down to

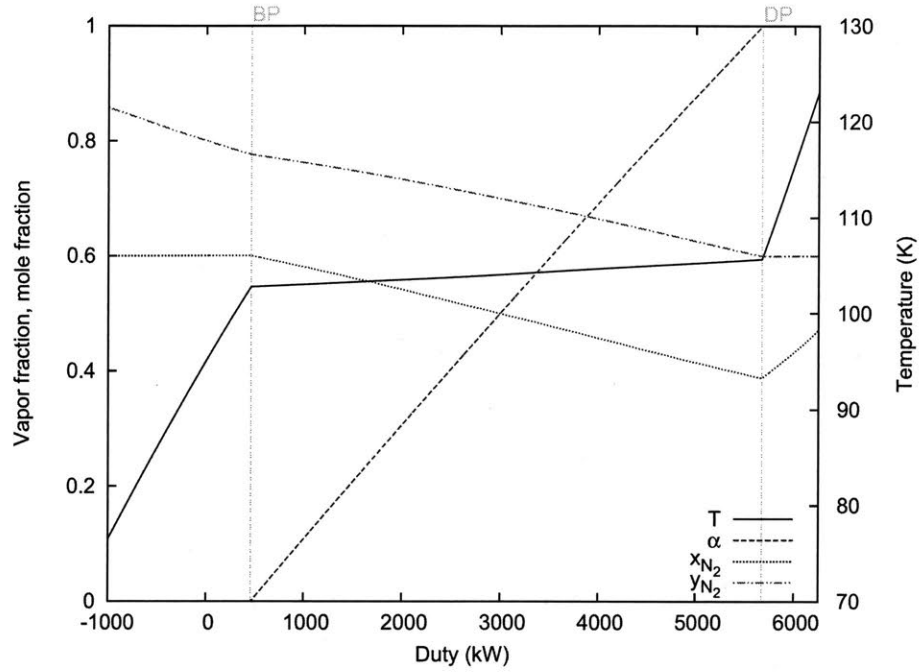


Figure 5-9: Results from parametrically varying the heat duty in the air-like mixture problem described in Example 5.4.

0.1 MPa in increments of 0.001 MPa leading to vapor formation. The results are shown in Figure 5-12. The 1,651 PQ-flash calculations take a total of 7.70 seconds to perform (average 4.7 ms per problem).

An overall comparison of the computational cost of the nonsmooth inside-out algorithms with the Boston-Britt inside-out algorithms is now given. From each of the previous examples, the two-phase region from each simulation was identified and discretized into 1000 points, and the times required to solve either the relevant PQ- or PT-flash calculations were compared. For additional comparison, the nonsmooth versions of the algorithms were tested using both basic successive substitution and Anderson-accelerated successive substitution, while the classical versions of the algorithms were tested using basic successive substitution, Anderson-accelerated successive substitution and Broyden's method for updating the inverse of the approximate Jacobian. The results are shown in Table 5.1.

As Table 5.1 shows, in the classical case, Anderson accelerated successive substitution is actually more efficient than Broyden's method for solving problems of this

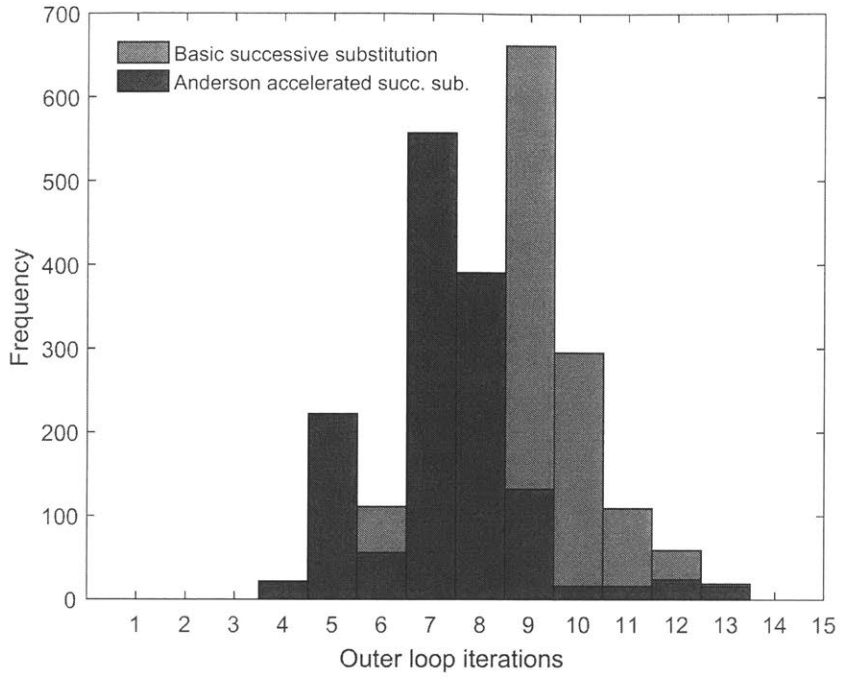


Figure 5-10: Histogram of total number of outer loop iterations needed to converge the flash calculations shown in Figure 5-9 using basic successive substitution (red; mean: 9.0 iterations) and Anderson acceleration (blue; mean: 7.2 iterations).

scale in all but one instance (where it is still comparable). It is expected that this trend will hold for the nonsmooth case as well. Also of note is that for PT-flash calculations (Examples 1 and 3), there is very little additional computational cost associated with using the nonsmooth version of the algorithm. In the case of PQ-flash calculations (Examples 2 and 4), the nonsmooth version of the algorithm is 30-40% slower on these problems than the classical version. This is an expected price for algorithmic robustness, since as previously discussed, the nonsmooth version must update all of the simple model parameters $(A - F)$ at every iteration, which requires additional thermophysical property evaluations and results in a problem size of $n_c + 6$ variables instead of just $n_c + 3$.

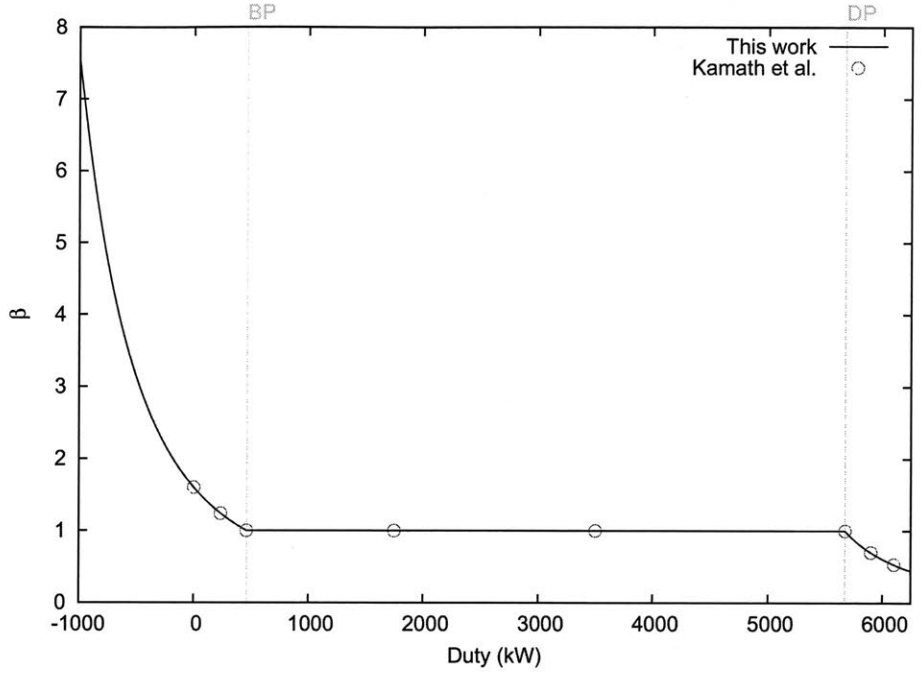


Figure 5-11: The parameter β as calculated in this work and in Kamath *et al.*[60] for the flash calculations in Example 5.4.

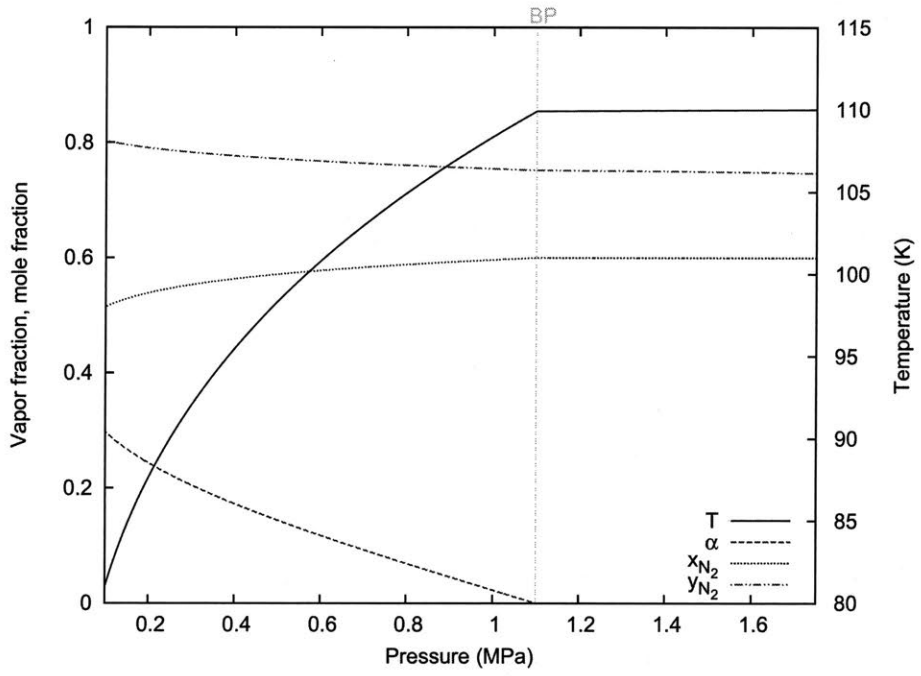


Figure 5-12: Results from parametrically varying the flash pressure in the air-like mixture problem described in Example 5.4.

Table 5.1: Computational cost comparison for several implementations of the inside-out algorithms. CPU times are presented as values normalized by the minimum time within each row.

	Nonsmooth		Classical		
Simulation	**Succ. Sub.**	**Anderson**	**Succ. Sub.**	**Anderson**	**Broyden**
Ex. 1 (vary T)	1.15	1.04	1.06	1.00	1.47
Ex. 1 (vary P)	1.37	1.03	1.21	1.00	1.34
Ex. 2	1.60	1.40	1.20	1.01	1.00
Ex. 3	1.20	1.04	1.12	1.00	1.56
Ex. 4 (vary Q)	1.63	1.36	1.18	1.00	1.16
Ex. 4 (vary P)	1.58	1.34	1.13	1.00	1.12

5.5 Conclusions

Modifications of the classical inside-out algorithms have been presented. These new algorithms use an additional nonsmooth equation in the inner loop to relax equilibrium conditions when necessary and feature a modified scheme for updating the outer loop variables. This new class of nonsmooth inside-out algorithms is capable of robustly solving the flash equations regardless of the number of phases present without the use of heuristics or significant computational burden. Numerical results for simulations involving different mixtures, flash types and property packages have been shown, highlighting the capability of the algorithms to handle both two-phase and single-phase behavior reliably and efficiently for any choice of inputs.

Chapter 6

A nonsmooth approach to density extrapolation and pseudoproperty evaluation

It is essential for reliable process simulation and optimization software to have extremely robust subroutines for thermophysical property evaluation and vapor-liquid equilibrium calculations. However, in the course of obtaining such values or solving such a problem, it is possible that the thermodynamic model will be queried for liquid or vapor properties of a mixture at conditions far from the physical solution or where one phase physically does not exist. Under these conditions, and in the common situation where both phases are described by the same equation of state, the model may return a liquid-like density for the vapor phase or vice-versa. It is well known that this behavior can cause a flash calculation algorithm to converge to the trivial solution of the equations (indistinguishable liquid and vapor phases) or simply fail. To mitigate this behavior, a number of articles have suggested methods for evaluating and post-processing the density calculated by the EOS to promote convergence of the flash calculations to physical solutions through creatively-defined extrapolations. In keeping with the efforts of the previous chapter, this chapter describes new nonsmooth algorithms for robustly evaluating appropriate density values for mixtures at conditions where use of the EOS alone yields unreasonable results. Unlike many

such proposals in the literature, this new approach requires only a simple evaluation procedure and may be augmented with accurate sensitivity analysis through the use of nonsmooth operators and automatic generalized derivative computation. Applications of this new strategy are highlighted for commonly studied equations of state in hydrocarbon systems.

6.1 Introduction

In many applications, use of the same equation of state to describe both the vapor and liquid phases of a mixture is either highly desirable or essential. This is due to such models requiring (generally) few parameters to describe fluid behavior reasonably over a wide range of temperatures and pressures, in addition to allowing for consistent prediction of properties close to vapor-liquid critical points. In most applications, the primary role of the EOS is to furnish a value for the mixture density (or volume) of each phase at a given pressure, temperature and composition. These liquid and vapor density values are then used in simulations to evaluate key thermodynamic properties such as fugacity coefficients and enthalpy or entropy departures.

However, the possible numerical issues associated with the dual-EOS approach in flash calculations are well-known and well-studied. In terms of properties easily derived from the EOS, the equal-fugacity criteria for phase equilibrium of an n_c component mixture can be written as:

$$x_{L,i}\phi_i^L(T,P,\mathbf{x}_L) = y_{V,i}\phi_i^V(T,P,\mathbf{y}_V), \quad \forall i = 1,...,n_c, \tag{6.1}$$

where ϕ_i is the fugacity coefficient of component i (with the corresponding phase denoted by the superscript), P is the system pressure, T is the system temperature, $\mathbf{x}_L$ (with components $x_{L,i}$) is the liquid phase molar composition vector and $\mathbf{y}_V$ (with components $y_{V,i}$) is the vapor phase molar composition vector. Note that the dependence of the fugacity coefficients on the density of the corresponding phase is implicit in the statement of Equation (6.1). Throughout this chapter, it is assumed

that the domains of pressure, temperature, and composition are physical, that is: $P \in \mathbb{R}_+ \setminus \{0\}$, $T \in \mathbb{R}_+ \setminus \{0\}$ and $\mathbf{x}_L, \mathbf{y}_V \in (0,1)^{n_c}$. Problematically, Equation (6.1) is always satisfied if $\mathbf{x}_L = \mathbf{y}_V$ component-wise when the same EOS model is used to describe both phases. This is known as the trivial solution of the flash equations and is only physically correct at the mixture critical point and in the supercritical region of the phase diagram. However, it is possible for flash calculations with a subcritical physical solution to converge to this solution incorrectly.

Many authors have proposed methods for circumventing or mitigating this problem. An early work by Asselineau *et al.*[6] details a modified Newton-type flash calculation algorithm that checks on each iteration whether calculated values are valid (e.g. the liquid density ρ_L is greater than the vapor density, ρ_V), and if not, iteratively dampens the Newton step until the result is physical. This can clearly become computationally expensive in regions where nonphysical specifications lie close to the current iterate, and, moreover, it is generally agreed in the literature that such approaches are inferior to methods in which the EOS subroutine, not the VLE solution algorithm, ascertains the suitability of e.g. density values.[83;97;78;116]

Along this line of thinking, Gundersen[47] proposed an algorithm for updating the parameters of an EOS to yield an appropriate solution for the compressibility factor of a given phase. This approach, while effective for certain cubic equations of state is not readily generalizable to other models. Poling *et al.*[97] established general heuristic criteria for accepting or rejecting the density value calculated from an EOS based on the value of the predicted isothermal compressibility, $\beta_j \equiv -\frac{1}{\nu}\left(\frac{\partial \nu}{\partial P}\right)_{T,\mathbf{z}}$, where $j = L$ and $j = V$ for the liquid and vapor phases, respectively, ν is the system volume and $\mathbf{z}$ is the appropriate phase composition. Note that $\mathbf{z}$ is used in this chapter to denote a composition whenever the formulation of an equation is identical for both phases or it is otherwise unnecessary to differentiate between phases, whereas $\mathbf{x}_L$ and $\mathbf{y}_V$ always specifically denote liquid and vapor phase compositions, respectively. The Poling *et al.*[97] criteria are that $\beta_L < 0.005\ \text{atm}^{-1}$ and $0.9/P < \beta_V < 3/P$. However, while often a useful heuristic, cases can be found in the literature where these rules do not adequately differentiate between liquid-like and vapor-like phases,

see e.g. Zhao and Saha.[151] A further iteration on this concept was described by Pasad and Venkatarathnam,[93] who proposed criteria based on the derivative of the predicted isothermal compressibility. Their criteria are stated as $\left(\frac{\partial \beta_L}{\partial T}\right)_{P,\mathbf{x}_L} < 0$ and $\left(\frac{\partial \beta_V}{\partial T}\right)_{P,\mathbf{y}_V} > 0$. However, the authors note that these criteria are not valid above the mechanical critical temperature, which is quite limiting as will be seen in the following section.

The authors of the previous approaches do not offer rigorous methods for obtaining new density values that comply with their criteria. Instead, the authors suggest modifying values of temperature, pressure and composition used in the density calculations until the criteria are met, and then restarting the flash calculations from these points. This can lead to expensive guess-and-check style iteration schemes and discontinuities in the values returned to the higher-level flash algorithm. A different school of thought that was proposed initially by Mathias *et al.*[79] involves constructing nonphysical extrapolations of density values into problematic regions based on generalizations about the $P-\rho$ behavior of mixtures. These extrapolations are intended to mimic the physical behavior and trends that are characteristic of each phase and to ensure continuity at the boundary points of non-equilibrium regions. This in turn allows for reasonable and continuous pseudoproperties (e.g. fictitious values for departures and fugacity coefficients) to be calculated from the extrapolations. This method has been implemented in the Aspen Plus[5] software and has evidently enjoyed much success. Their method forms the basis for the ideas presented in this chapter; however, the algorithms described in the original paper are designed for use with higher-level methods that do not require derivative information, leading to procedures involving several levels of nested iterative calculations and conditional statements. Moreover, there is potential for their extrapolations to be non-differentiable, which must be handled rigorously in a modeling environment that relies on exact sensitivity information. Since the present authors are interested in algorithms that will service higher-level methods that use exact derivatives and generalizations of the derivative at points of nondifferentiability, alternative algorithms must be formulated.

Several other authors have presented methods that either use similar concepts or

attempt to improve on the ideas of Mathias *et al.*[79] A recent patent authored by Xu *et al.*[148] describes an implementation of a conceptually similar method with different extrapolation functions and boundary definitions to assist in process control calculations. Stateva *et al.*[116] propose a method based on extrapolations of pressure-volume behavior, rather than pressure-density behavior. They propose dividing the $P - \nu$ space into several regions wherein either the EOS itself or a quadratic/cubic spline is used to describe the local fluid behavior. Their method could have been used as the basis for the method in the current chapter; however, due to the need for both numerical integration methods and the computation of many parameters to ensure continuity across region boundaries, the simpler method of Mathias *et al.*[79] was deemed a more appropriate starting point. In an article by Zhao and Saha,[151] a method is proposed that eliminates the iterative process for determining the density values at which to begin extrapolation in the special case of cubic equations of state. Instead, an auxiliary equation is solved to determine the pressure (at a given temperature and composition) at which the number of roots of the original cubic EOS changes from three to one, and this is taken as the boundary between physical and extrapolated values. Their final algorithm for density computation is however dependent on many conditional statements (enough that a sizable flowchart is used to describe the procedure in the original article) and therefore obtaining analytical sensitivity information about the extrapolations would not be possible directly. The method also generates potentially non-differentiable extrapolations and is only applicable to cubic equations of state. Additionally, as concluded in an article by Mathias and Benson,[78] the computational cost of evaluating the density root of an equation of state is minor compared to the cost of evaluating the mixture composition-dependent parameters. Therefore, reducing the cost of the density evaluation by eliminating iterative procedures is unlikely to have a substantial impact on the overall cost of the EOS computation. This also implies that the earlier methods that function by modifying variables that influence the EOS parameters are likely to be substantially more expensive than the extrapolation-based methods.

Yet another approach to the phase identification problem comes from a series of

articles focused on discriminating between the roots of cubic equations of state. Kamath *et al.*[60] proposed criteria for distinguishing vapor-like and liquid-like roots of a cubic EOS based on the derivatives of the equation with respect to the compressibility factor. Their mathematical analysis of cubic equations leads to the conclusion that the first and second derivatives must be nonnegative for a vapor-like root, whereas the first derivative must be nonnegative and the second derivatives must be nonpositive for a liquid-like root. If a phase disappears in the course of solving a flash problem, the second derivative criteria are relaxed to allow an optimizer to choose the single real root that remains. This approach was further refined by Dowling *et al.*[30] who observed several cases in which these criteria did not lead to correct phase classification, such as in the supercritical region of the phase space. These authors propose additional thermodynamically-motivated constraints to correct for these inconsistencies. Most recently, an article by Glass and Mitsos[38] gives alternative criteria to those suggested by Kamath *et al.*[60] for root discrimination and proposes a method based on relaxing the equality constraint implied by the cubic EOS itself (rather than the root discrimination criteria) when a phase disappears during flash calculations. These methods are designed to be implemented in an equation-oriented simulation or optimization environment. Indeed, the most rigorous of these formulations due to Glass and Mitsos necessitates global optimization techniques for its solution.[38] Accordingly, they incur a substantial cost in terms of both model complexity and computational burden in simulations where many such calculations are needed. These approaches also cannot exploit the power of tailored external subroutines for converging challenging VLE problems, in part because the formulations necessitate simultaneous solution of the root-finding problem and the flash equations, and also because the problem of obtaining sensitivity information about the solution of an optimization problem is still an area of active research.[117]

As noted earlier, this work also builds on the extrapolation concept of Mathias *et al.*[79] However, the approach detailed herein addresses some of the deficiencies of the aforementioned methods, particularly in the calculation of exact sensitivity information for the extrapolations. The new method is also applicable to general equations of

state and uses an easy-to-implement density evaluation procedure that only requires one (robust) iterative procedure beyond what is needed to solve the original EOS. The sensitivity analysis may be included at little additional cost by means of recent advances in generalized derivative computation for nonsmooth functions.

6.2 Background

This preliminary section gives a brief overview of the variation of mixture density with pressure and temperature, summarizes prior approaches to the unacceptable density problem and introduces relevant concepts in nonsmooth analysis and generalized derivative evaluation.

6.2.1 Behavior of mixture density

The majority of commonly used equations of state are explicit in pressure, that is, they are of the functional form: $P = f(T, \nu, \mathbf{z})$. These equations are often rewritten and evaluated in terms of the mixture density, ρ, or the mixture compressibility factor, $Z \equiv \frac{P\nu}{R_G T} = \frac{P}{\rho R_G T}$, where R_G is the universal gas constant. Since equations of state are primarily used when P, T and $\mathbf{z}$ are known, evaluating the corresponding volume, density or compressibility factor requires an equation solving procedure. The most widely-used equations of state are the cubic equations of state, e.g. the SRK and PR models, which may be solved either analytically or through iterative numerical methods. More complex models such as the Benedict-Webb-Rubin-Starling (BWRS) EOS necessitate iterative solution methods but may provide higher accuracy predictions of fluid behavior. For reference, the functional forms of the PR and BWRS equations and parameters used in this chapter are given in Appendix B.

As in several other works of similar focus, an example fluid that will be used for illustration purposes in this chapter is a equimolar mixture of ethane and n-heptane. Several pressure-density ($P - \rho$) isotherms generated by the Peng-Robinson cubic EOS for this mixture are shown in Figure 6-1. Superimposed on these isotherms are the mixture stability and coexistence boundaries. The outermost dashed dome is the

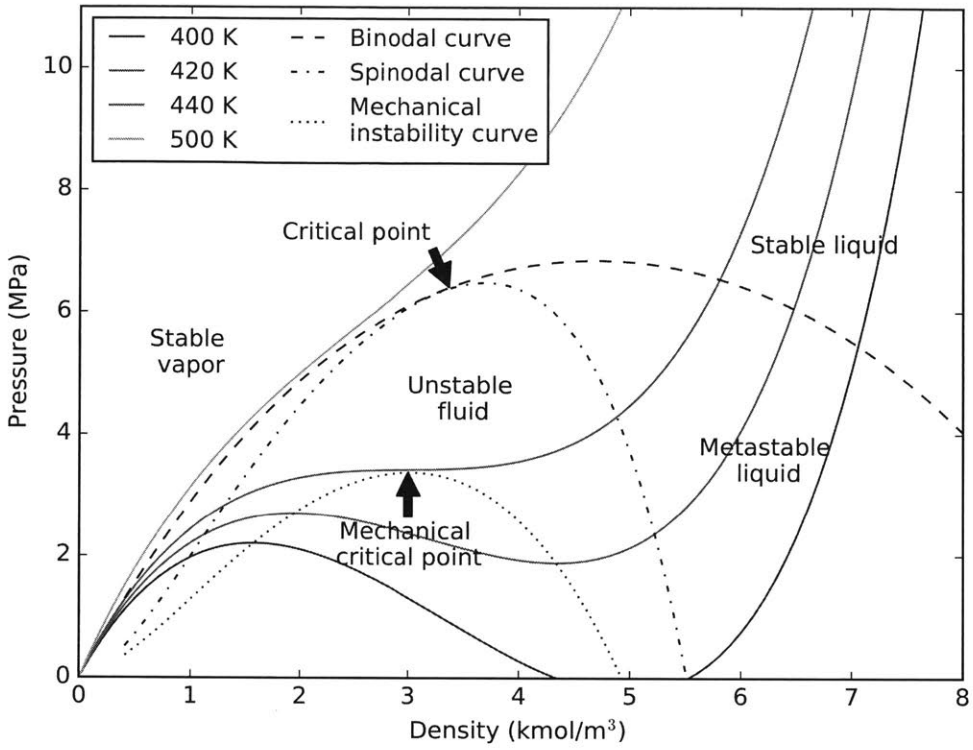


Figure 6-1: $P-\rho$ profiles of an equimolar ethane/n-heptane mixture and the stability regimes of the liquid and vapor phases, based on Figure 2 from Mathias *et al.*[79] Not labeled in the diagram is the narrow metastable vapor region between the binodal and spinodal curves to the left of the critical point. Note that non-physical negative pressures are predicted by the EOS at 400 K as a result of specifying a density value and calculating the corresponding pressure, rather than the usual case of calculating the density given a physical pressure.

binodal coexistence curve, defined by the densities of saturated liquid and saturated vapor in equilibrium mixtures. Inside this region lies the spinodal curve (dash-dot), which is the stability limit defined (for a binary mixture) by:

$$\det\left(\begin{bmatrix} A_{\nu,\nu} & A_{\nu,N} \\ A_{N,\nu} & A_{N,N} \end{bmatrix}\right) = 0. \tag{6.2}$$

where A is the Helmholtz free energy of the mixture, the subscript ν indicates a partial derivative with respect to volume and the subscript N indicates a partial derivative with respect to mole number of the first component.[51] In general for a mixture with

n_c components, this matrix is of size $n_c \times n_c$. Expressions for these derivatives may be obtained in terms of quantities more readily evaluated from a pressure-explicit EOS, for instance:

$$A_{V,V} = P_\rho \equiv \left(\frac{\partial P}{\partial \rho}\right)_{T,\mathbf{z}}. \tag{6.3}$$

Expressions for the other terms may be found in Beegle *et al.*[15] At conditions inside the binodal dome but outside the spinodal dome, a fluid will be metastable, that is, it will not spontaneously phase separate, though any perturbation will induce a split and lower the free energy of the mixture. At conditions inside the spinodal dome, a fluid will be unstable and spontaneously phase separate. As such, any specification of pressure, temperature or composition that lies within the binodal and spinodal regions cannot correspond to an equilibrium state. However, there is also no effective way to ensure that an iterative higher-level algorithm, e.g. for a flash calculation, will not stray into this region. Unmanaged, the liquid and vapor density values returned by the EOS here could potentially change discontinuously between iterations, which is likely to cause numerical issues in the VLE algorithm, or the values could coincide, which is likely to cause convergence to the trivial solution. Unfortunately, determination of the boundary of the spinodal dome is computationally expensive, as suggested by Equation (6.2).

However, a much easier to ascertain boundary lies further inside the spinodal dome. This is the curve corresponding to the limit of mechanical stability (dotted line), which is calculated by applying the spinodal condition for pure components directly to the mixture, i.e. $A_{\nu,\nu} = P_\rho = 0$. Note that for a mixture, this curve has no physical significance and will always lie inside the spinodal curve.[15] The peak of this curve, where the second partial derivative of pressure with respect to density is zero, is referred to as the mechanical critical point, and coincides with the true critical point only for pure substances.[105] As Figure 6-1 shows, the true critical point of the mixture occurs where the spinodal and binodal meet tangentially, and is almost always slightly higher in terms of density than the mechanical critical point[79] (in this instance, the mechanical critical density, ρ_{mc}, is 3.01 kmol/m^3 while the true critical

density, ρ_c, is 3.37 kmol/m^3). Fortunately, an EOS will generally return reasonable properties up until close to the boundary of the mechanical instability dome (dotted line in Figure 6-1), though as reported by Poling *et al.*[97] and Mathias *et al.*,[79] the values and derivatives of the physical properties derived from density can exhibit erratic behavior in the immediate vicinity of the boundary.

6.2.2 Density extrapolation models

The core concept behind the density extrapolation approach introduced in the literature review is that when an EOS is evaluated at conditions that correspond to unstable states or nonexistent phases, it is not necessary that the density returned by the subroutine be a solution of the EOS. Rather, it is more beneficial to ensure that the calculated density is "liquid-like" if a liquid phase density is requested or "vapor-like" if a vapor phase density is requested. Accordingly, extrapolation functions have been proposed that mimic the behavior of a given phase with respect to changes in pressure, temperature and composition and return reasonable density estimates for each phase. A key point to take away from the discussion in the previous section is that while it would be ideal to begin extrapolating density values at either the binodal or spinodal boundary, these points are far too expensive to calculate repeatedly. However, the boundary of the mechanical instability dome is far less expensive to compute, and the mechanical critical density is generally a tight underestimate of the critical density. Due to the numerical issues in the immediate vicinity of this boundary however, the common practice is to begin extrapolating density values slightly before reaching the mechanical instability dome, as measured by the derivative of pressure with respect to density attaining a small positive value. In this chapter, the value ρ_j^{Ω} (with j as a placeholder for the phase identifier) is defined as the density value at which the partial derivative of pressure with respect to density is equal to the limiting value. Mathias *et al.*[79] propose that the limiting density value for each phase be the solution of the equation:

$$P_{\rho_j}(\rho_j, T, \mathbf{z}) - 0.1 R_G T = 0. \tag{6.4}$$

for a given temperature and composition. In what follows, the value of P_{ρ_j} at $(\rho_j^\Omega, T, \mathbf{z})$ will be denoted by $P_{\rho_j}^\Omega$. This is not the only reasonable choice for the boundary of the acceptable region. Xu *et al.*[148] suggest ρ_L^Ω and ρ_V^Ω be the points at which:

$$P_{\rho_L}(\rho_L, T, \mathbf{x}_L) - \gamma\frac{P}{\rho_L} + \Gamma = 0, \tag{6.5}$$

$$\delta(P_{\rho_V}(\rho_V, T, \mathbf{y}_V) - R_G) + (1-\delta)P_{\rho_L}^\Omega + \Delta = 0, \tag{6.6}$$

for the liquid and vapor phase, respectively, where δ, γ, Γ and Δ are user-defined relaxation and offset constants. Note that the vapor phase criterion depends on the result of the liquid phase calculation unless $\delta = 1$. Zhao and Saha[151] take a different approach to locating the boundary of the acceptable region that is specific to cubic equations of state. These authors calculate the point at which the real part of the complex roots and the single real root of the cubic coincide by solving an auxiliary cubic equation, then use a series of conditionals to decide whether extrapolation is needed based on the result of this additional calculation.

As noted in the aforementioned articles, these criteria alone do not rule out the possibility of unacceptable liquid densities being returned at high temperatures where the isotherms exceed the mechanical critical temperature, T_{mc}, and yet never have near-zero derivatives (e.g. the isotherm at 500 K in Figure 6-1). For this case, they note that it suffices to make use of the previous observation that ρ_{mc} is a useful underestimate of the true critical density, and extrapolate liquid density whenever the calculated value from the EOS model would fall below this value.

In summary, these criteria address three cases in which the EOS model may return invalid properties for the phase requested. These are as follows:

1. A vapor density is requested but the EOS solution is liquid-like, e.g. at high pressure. This case is handled by extrapolating values for vapor density based on a derivative criterion such as Equation (6.4) with $j = V$ or Equation (6.6).

2. A liquid density is requested but the EOS solution is vapor-like and $T < T_{\text{mc}}$, e.g. at low pressure. This case is handled by extrapolating values for liquid

density based on a derivative criterion such as Equation (6.4) with $j = L$ or Equation (6.5).

3. A liquid density is requested but the EOS solution is vapor-like and $T \geq T_{\text{mc}}$. This case may be handled by extrapolating values for liquid density based on the $\rho_L > \rho_{\text{mc}}$ criterion.

In the cases where extrapolation is needed, the previous authors have proposed various models for extrapolating the density values. The Mathias *et al.*[79] liquid phase model is of the following functional form:

$$P = A_L + B_L \ln(\rho_L - 0.7\rho_{\text{mc}}), \tag{6.7}$$

where A_L and B_L are constants to be determined by equating P and P_{ρ_L} with values from the EOS model on the boundary of the acceptable region. This model is used for both the cases numbered 2 and 3 in the previous list. The mathematical form of this model will cause the pressure to tend to $-\infty$ as the liquid density decreases to $0.7\rho_{\text{mc}}$, which prevents the liquid phase density from ever becoming unrealistically small for any physical pressure specification. Zhao and Saha[151] use the same logarithmic model in their article, while Xu *et al.*[148] suggest a quadratic model:

$$P = A_L + B_L(\rho_L - \rho_L^{\Omega}) + C_L(\rho_L - \rho_L^{\Omega})^2, \tag{6.8}$$

where the constants are again determined by matching values and derivatives at the boundary point. In either case, once the boundary conditions are introduced, by design the pressure predicted by these extrapolating functions will be a strictly concave function of density that shares a value and tangent line with the pressure predicted by EOS model exactly at the boundary of the acceptable region. As any realistic EOS will predict pressure to be a strictly convex function of density between its highest density inflection point and $+\infty$ (see e.g. Figure 6-1), note that this common tangent line will underestimate the pressure calculated from the EOS and overestimate the pressure calculated from the extrapolating function at all densities

in this region containing the boundary point. Therefore, the pressure calculated from the EOS will always be greater than the pressure given by the extrapolating functions at any density in this region except for the boundary point where they are equal.

The Mathias *et al.*[79] vapor phase model is described textually as "a quadratic extrapolation of the reciprocal of pressure with density." Zhao and Saha[151] interpreted this to mean a model of the form:

$$\rho_V = A_V + \frac{B_V}{P} + \frac{C_V}{P^2}, \tag{6.9}$$

where A_V, B_V and C_V are constants obtained from equating P and $P_{\rho V}$ with values from the EOS model on the boundary of the allowed region and enforcing that $1/P = 0$ when the density becomes sufficiently large. Further numerical details are given in the following section. However, there is an alternative interpretation of the statement from Mathias *et al.*,[79] the use of which appears to better match the results of that article, as follows:

$$\frac{1}{P} = A_V + B_V \rho_V + C_V \rho_V^2, \tag{6.10}$$

where A_V, B_V and C_V are constants obtained from the same boundary conditions as just discussed. The extrapolation model proposed by Xu *et al.*[148] is more complex:

$$P = A_V + B_V \left(\frac{1 - C_V \rho_V^\Omega}{1 - C_V \rho_V} \right) (\rho_V - \rho_V^\Omega) + (e^R - 1) \left(\frac{1 - C_V \rho_V^\Omega}{1 - C_V \rho_V} \right) (\rho_V - \rho_V^\Omega)^2, \tag{6.11}$$

with the constants determined by the similar boundary conditions to that of the other authors. In each case, once the boundary conditions are introduced, by design the pressure predicted by these extrapolations will be a strictly convex function of density that shares a value and tangent line with that predicted by the EOS model, which will be a strictly concave function of density in a region containing the boundary point (between $-\infty$ and the lowest density inflection point of the model). Analogous to the earlier discussion, this implies that in this region, the pressure predicted by the EOS model will always be less than or equal to that predicted by the extrapolating functions, except at the boundary point where they are equal.

These extrapolating functions are generally used in subroutines which simultaneously compute the density root of the original equation of state and evaluate whether or not it is acceptable based on the discussed criteria. If not, the value obtained from the appropriate extrapolation is returned instead. Zhao and Saha[151] provide a flowchart in their article for deciding which density value should be returned that can be implemented as a reasonably complex structure of if/else statements. The approach described in Xu *et al.*[148] involves calculating the pressure value corresponding to the boundary points and comparing this to the value of the pressure at which the density has been requested. The Mathias *et al.*[79] algorithm is slightly more complex, though relies on a similar concept. At mechanically-subcritical conditions, their approach involves using several Newton-type iteration schemes to iteratively bound either a root of the equation of state or the boundary of the acceptable region. If the boundary is located, then the extrapolation is used to calculate a density; otherwise, the solution of the EOS is returned. At mechanically-supercritical conditions, the extrapolated liquid density is found using an iterative secant method. In all cases, the density value may then be used to evaluate the thermophysical properties as required by the higher-level algorithm.

6.3 Nonsmooth Algorithms for Density Extrapolation

A new nonsmooth strategy for implementing the extrapolations described in the previous section is now presented. Afterward, the minor modifications needed to allow for the calculation of accurate generalized derivative elements of the extrapolations will be discussed.

6.3.1 Calculation of extrapolated density values and pseudo-properties

First, the nonsmooth density evaluation functions for each phase will be developed. Note that the key idea here is that the final forms of the density models will be PC^1 functions that automatically return either the density from the EOS or a more acceptable extrapolated value as determined by the temperature, pressure and composition at which the density is queried. This will be accomplished using the mid function to compare the value returned by the direct EOS evaluation, a boundary value and the value returned by the extrapolation function.

Let $\rho_L^{\text{EOS}}(P, T, \mathbf{x}_L)$ and $\rho_V^{\text{EOS}}(P, T, \mathbf{y}_V)$ be the liquid and vapor density solutions obtained from solving the equation of state, respectively. As before, we denote by ρ_j^{Ω} as the density value of phase j at which the partial derivative of pressure with respect to density is equal to a limiting value, i.e.,

$$P_{\rho_j}^{\Omega}(\rho_j^{\Omega}, T, \mathbf{z}) = 0.1 R_G T, \tag{6.12}$$

for given T and $\mathbf{z}$ as suggested by Mathias *et al.*[79] This condition is chosen for the present work due to its simplicity and generality, though it is certainly possible to substitute the more complex conditions of e.g. Xu *et al.*[148] in place of Equation (6.12). However, the dependence of the vapor phase criterion on the result of the liquid phase calculation in Equation (6.6) is undesirable when only vapor phase properties are required.

Now, define the actual boundary functions for each phase as follows:

$$\rho_L^{\text{bound}} : (T, \mathbf{x}_L) \mapsto \text{mid}\left(\rho_{\text{mc}}(\mathbf{x}_L), \rho_L^{\Omega}(T, \mathbf{x}_L), \rho_{\text{hi}}(\mathbf{x}_L)\right), \tag{6.13}$$

$$\rho_V^{\text{bound}} : (T, \mathbf{y}_V) \mapsto \text{mid}\left(\rho_{\text{lo}}(\mathbf{y}_V), \rho_V^{\Omega}(T, \mathbf{y}_V), \kappa\rho_{\text{mc}}(\mathbf{y}_V)\right), \tag{6.14}$$

with $\kappa < 1$ as a user-chosen scalar, ρ_{mc} is implicitly defined as before, that is, the density at which:

$$\frac{\partial P}{\partial \rho}(\rho_{\text{mc}}, T_{\text{mc}}, \mathbf{z}) = \frac{\partial^2 P}{\partial \rho^2}(\rho_{\text{mc}}, T_{\text{mc}}, \mathbf{z}) = 0, \tag{6.15}$$

and ρ_{lo} and ρ_{hi} are lower and upper bounds, respectively, on the density value that may be furnished by the equation of state, e.g. density values at which $P \to +\infty$ such that $\rho_{\text{lo}}(\mathbf{y}_V) < \rho_{\text{mc}}(\mathbf{y}_V)$ (typically also ≤ 0) and $\rho_{\text{hi}}(\mathbf{x}_L) > \rho_{\text{mc}}(\mathbf{x}_L)$. For some equations of state (e.g. cubic), ρ_{mc}, ρ_{lo} and ρ_{hi} can be determined analytically as a function of the mixture parameters (and are therefore functions of composition of the phase under consideration). For notational simplicity, the values returned by the boundary functions for a given temperature and composition are denoted as follows: $\rho_L^* \equiv \rho_L^{\text{bound}}(T, \mathbf{x}_L)$ and $\rho_V^* \equiv \rho_V^{\text{bound}}(T, \mathbf{y}_V)$. The mid statement in Equation (6.13) will ensure that the extrapolation of density will always take place when $\rho_L^{\text{EOS}}(P, T, \mathbf{x}_L) < \rho_{\text{mc}}(\mathbf{x}_L)$, regardless of whether the derivative criterion is met. In this way, this nonsmooth expression is able to assert the conditions on the liquid phase density automatically without a separate check or algorithm. The mid statement in Equation (6.14) serves a different purpose, which is to keep the value of density sufficiently below ρ_{mc} so that the vapor-phase extrapolation is always well-defined. Mathias *et al.*[79] also note that it is useful to keep the vapor density below ρ_{mc} to avoid the calculation of negative pressures at low temperatures. As such, the exact value of κ is flexible, though it is suggested that $\kappa \in [0.7, 0.95]$ for reasonable results. κ has been assumed to be 0.9 throughout the examples in this work.

Let P^* be the pressure value calculated from the EOS corresponding to ρ_L^* or ρ_V^*, as appropriate, and the corresponding partial derivative of pressure with respect to density be denoted P_ρ^*. The extrapolation function for liquid density follows from a minor modification of Equation (6.7) (chosen over Equation (6.8) since it is a two rather than three parameter model) and the boundary conditions proposed by Mathias *et al.*,[79] which are that:

$$P^* = A_L + B_L \ln\left(\rho_L^* - 0.7\rho_{\text{mc}}\right), \tag{6.16}$$

$$P_\rho^* = \frac{B_L}{\left(\rho_L^* - 0.7\rho_{\text{mc}}\right)}. \tag{6.17}$$

Solving these equations for A_L and B_L yields:

$$A_L = P^* - B_L \ln\left(\rho_L^* - 0.7\rho_{\text{mc}}\right), \tag{6.18}$$

$$B_L = P_\rho^* \left(\rho_L^* - 0.7\rho_{\text{mc}}\right). \tag{6.19}$$

The extrapolation function for liquid density is then given by:

$$\rho_L^{\text{extrap}} : (P, T, \mathbf{x}_L) \mapsto \min\left(\exp\left(\frac{P - A_L(T, \mathbf{x}_L)}{B_L(T, \mathbf{x}_L)}\right) + 0.7\rho_{\text{mc}}(\mathbf{x}_L),\ \rho_{\text{hi}}(\mathbf{x}_L)\right), \tag{6.20}$$

The first term in the min statement of Equation (6.20) comes simply from rearranging Equation (6.7) to be explicit in density, while the latter term truncates the rapidly growing exponential function at a known upper bound. Since Equation (6.7) was designed to approach $0.7\rho_{\text{mc}}$ asymptotically as $P \to -\infty$, no additional term is needed to prevent large negative values at low pressures.

Finally, the function that calculates the liquid density that is to be used in physical property calculations is as follows:

$$\rho_L^{\text{mid}} : (P, T, \mathbf{x}_L) \mapsto \text{mid}\left(\rho_L^{\text{EOS}}(P, T, \mathbf{x}_L),\ \rho_L^{\text{bound}}(T, \mathbf{x}_L),\ \rho_L^{\text{extrap}}(P, T, \mathbf{x}_L)\right). \tag{6.21}$$

For notational simplicity and consistency, the value returned by this function for a given temperature, pressure and composition is denoted by $\rho_L \equiv \rho_L^{\text{mid}}(P, T, \mathbf{x}_L)$. The mid function is used here to exploit the knowledge of the relative ordering of its arguments to return an acceptable density value. The density value returned by the extrapolation function overestimates the EOS solution value except at the point of tangency, ρ_L^*. Below ρ_L^*, the liquid density predicted by the EOS is unacceptably low; therefore, the mid function correctly returns the extrapolated value since $\rho_L^{\text{bound}}(T, \mathbf{x}_L) > \rho_L^{\text{extrap}}(P, T, \mathbf{x}_L) > \rho_L^{\text{EOS}}(P, T, \mathbf{x}_L)$. Above ρ_L^*, the EOS solution is acceptable; therefore the mid function correctly chooses that value as the liquid density since $\rho_L^{\text{extrap}}(P, T, \mathbf{x}_L) > \rho_L^{\text{EOS}}(P, T, \mathbf{x}_L) > \rho_L^{\text{bound}}(T, \mathbf{x}_L)$.

The extrapolation of vapor density is based on Equation (6.10). The model of Zhao and Saha[151] leads to non-monotonic behavior of the vapor density with respect

to pressure, while the complex model of Xu *et al.*[148] is more expensive to use than Equation (6.10) for no readily apparent additional benefit. As in the liquid case, some modifications are required in the present approach. For the ease of applying the boundary conditions, the model is best expressed in the form:

$$\frac{1}{P} = A_V + B_V(\rho - \rho_V^*) + C_V(\rho - \rho_V^*)^2. \tag{6.22}$$

The constants obtained from solving boundary conditions from Mathias *et al.*[79] are as follows, where the previously mentioned condition that $1/P = 0$ at high density is enforced at the midpoint between ρ_{mc} and ρ_V^*:

$$A_V = \frac{1}{P^*}, \tag{6.23}$$

$$B_V = -\frac{P_\rho^*}{(P^*)^2}, \tag{6.24}$$

$$C_V = -\frac{\left|A_V + B_V\left(\frac{\rho_{\text{mc}} - \rho_V^*}{2}\right)\right|}{\left(\frac{\rho_{\text{mc}} - \rho_V^*}{2}\right)^2}. \tag{6.25}$$

The absolute value in the C_V term is almost always redundant as the term will generally be negative due to the relative magnitudes of A_V and B_V. However, it is needed to make certain mathematical guarantees, as will be seen next. A density can be calculated analytically from Equation (6.22) by application of the quadratic formula:

$$\rho = \rho_V^* + \frac{-B_V \pm \sqrt{B_V^2 - 4C_V(A_V - 1/P)}}{2C_V}. \tag{6.26}$$

There are several considerations here. The first is ensuring that the square root term is always well defined. B_V^2 is clearly positive and C_V is always nonpositive, therefore the discriminant is certain to be nonnegative if $A_V \geq 1/P$. Unlike in the algorithms of other authors where the vapor phase extrapolation would only performed when $P > P^*$, here the extrapolation must be evaluated regardless of the value of P for eventual use in a mid operator. Therefore, it is possible in the algorithm developed in this chapter for A_V to be strictly less than $1/P$. The $(A_V - 1/P)$ term in the

square root is therefore be modified to $\max(0,\ A_V - 1/P)$. Then, in the case that $P < P^*$ and the first argument of the max function is active, Equation (6.26) simply reduces to $\rho = \rho_V^*$ rather than leading to a complex result. Additionally, it must be decided whether to use the solution that adds the square root term or the solution that subtracts it. The final result should be a density greater than the boundary value to preserve the expected trend of density increasing with increasing pressure. Since $P_\rho^* \geq 0.1 R_G T$ by definition, $(-B_V) > 0$ at all times. By some algebraic manipulation, it can be shown that the square root term is always greater than or equal to $-B_V$, so that the numerator is a nonpositive number if the square root is subtracted from $(-B_V)$. When divided by C_V, this then yields a positive quantity. Therefore, the solution that subtracts the square root in Equation (6.26) is always taken.

Further modifications are needed beyond those in the analytic solution of Equation (6.26). As discussed, for $P < P^*$, the right-hand side of Equation (6.26) becomes constant at ρ_V^*, while at $P > P^*$, it is always greater than ρ_V^*. It is desirable for the low pressure behavior to be modified so that the extrapolated density value is less than ρ_V^* for $P < P^*$. This is accomplished adding on an additional term of the form $\min(0,\ P - P^*)/P_{\rho V}^{\Omega}$ that is always less than or equal to zero (and also under-approximates ρ_V^{EOS} in this region, by design) for $P < P^*$. Finally, an additional term is needed to account for the mechanically supercritical region in which $P_{\rho V}$ exceeds the value of $0.1 R_G T$. In this region, there is no need to extrapolate the vapor density; however, the value given by Equation (6.26) with the aforementioned modifications will lie above ρ_V^* and almost certainly below the density returned by the EOS. Therefore, it is actually desirable for the value of the extrapolation function to rapidly yet continuously grow to a value greater than $\rho_V^{\text{EOS}}(P, T, \mathbf{y}_V)$ once these conditions are met so that the EOS solution will be chosen by the mid operator instead of the extrapolated density. The final form of the vapor extrapolation used

in this work is then as follows:

$$\rho_V^{\text{extrap}} : (P,T,\mathbf{y}_V) \mapsto \text{mid}\Bigg(0,\ \ \rho_{\text{hi}}(\mathbf{y}_V),\ \ \rho_V^{\text{bound}}(T,\mathbf{y}_V) + \frac{\min\left(0, P - P^*(T,\mathbf{y}_V)\right)}{P_{\rho_V}^{\Omega}(T,\mathbf{y}_V)} + \frac{-B_V(T,\mathbf{y}_V) - \sqrt{B_V(T,\mathbf{y}_V)^2 - 4C_V(T,\mathbf{y}_V)\max(0, A_V(T,\mathbf{y}_V) - 1/P)}}{2C_V(T,\mathbf{y}_V)} + \max\left(0, T - T_{\text{mc}}(\mathbf{y}_V)\right)\max\left(0, P_\rho^*(T,\mathbf{y}_V) - P_{\rho_V}^{\Omega}(T,\mathbf{y}_V)\right) \times \left(\rho_V^{\text{EOS}}(P,T,\mathbf{y}_V) - \rho_V^{\text{bound}}(T,\mathbf{y}_V)\right)\Bigg). \tag{6.27}$$

The outermost mid function simply serves to keep the extrapolations bounded between 0 and ρ_{hi}, which are respectively lower and upper bounds on physical density values, and the third term comprises all the elements discussed previously.

Analogous to the liquid phase, the functions that calculates the final vapor density to be used in physical property calculations is as follows:

$$\rho_V^{\text{mid}} : (P,T,\mathbf{y}_V) \mapsto \text{mid}\left(\rho_V^{\text{EOS}}(P,T,\mathbf{y}_V),\ \rho_V^{\text{bound}}(T,\mathbf{y}_V),\ \rho_V^{\text{extrap}}(P,T,\mathbf{y}_V)\right). \tag{6.28}$$

As for the liquid, the value of this function at a given temperature, pressure and composition is denoted by $\rho_V \equiv \rho_V^{\text{mid}}(P,T,\mathbf{y}_V)$. As before, the mid function uses the relative ordering of its arguments to return an acceptable density value. The density value returned by the extrapolation function underestimates the EOS solution value except at the point of tangency, ρ_V^* except when $T > T_{\text{mc}}$ and $P_{\rho V} > P_\rho^{\Omega}$. Assuming mechanically subcritical conditions, below ρ_V^*, the vapor density predicted by the EOS is acceptable; therefore the mid function correctly chooses that value as the vapor density since $\rho_V^{\text{bound}}(T,\mathbf{y}_V) > \rho_V^{\text{EOS}}(P,T,\mathbf{y}_V) > \rho_V^{\text{extrap}}(P,T,\mathbf{y}_V)$. Above ρ_V^*, the EOS solution is unacceptable; therefore the mid function correctly returns the extrapolated value since $\rho_V^{\text{EOS}}(P,T,\mathbf{y}_V) > \rho_V^{\text{extrap}}(P,T,\mathbf{y}_V) > \rho_V^{\text{bound}}(T,\mathbf{y}_V)$. In the region where $T > T_{\text{mc}}$ and $P_\rho^* > P_{\rho V}^{\Omega}$, there is a narrow range in which the previous ordering holds. However, the value of the extrapolation function quickly exceeds that returned by the EOS model, allowing the EOS solution to be correctly chosen at

sufficiently high temperatures.

Once an acceptable density value has been calculated for the liquid or vapor phase, it may be used in the calculation of all physical properties, e.g. fugacity coefficients and enthalpy/entropy departure functions. As in Mathias *et al.*, the liquid fugacity coefficients are scaled by ratio of P^{calc} to P, that is:

$$\phi_i^L = \phi_i^{L,\text{calc}} \left(\frac{P^{\text{calc}}}{P} \right), \tag{6.29}$$

where P^{calc} is the pressure obtained by evaluating the EOS model at $(\rho_L, T, \mathbf{x}_L)$.[79] These equations form the basis of Algorithms 6.1 and 6.2.

Algorithm 6.1: Evaluate liquid density and scaling pressure.

Solve the EOS model for ρ_L^{EOS}.
Calculate ρ_{mc} and ρ_{hi}.
Solve Equation (6.4) for ρ_L^{Ω} in the interval $[\rho_{\text{mc}}, \rho_{\text{hi}}]$.
if no solution is found **then**
  $\rho_L^* \leftarrow \rho_{\text{mc}}$.
else
  $\rho_L^* \leftarrow \text{mid}(\rho_{\text{mc}}, \rho_L^{\Omega}, \rho_{\text{hi}})$.
end if
Calculate P^* and P_ρ^* at ρ_L^* from the EOS model.
$B_L \leftarrow P_\rho^* \left(\rho_L^* - 0.7\rho_{\text{mc}}\right)$.
$A_L \leftarrow P^* - B_L \ln\left(\rho_L^* - 0.7\rho_{\text{mc}}\right)$.
$\rho_L^{\text{extrap}} \leftarrow \min\left(\exp\left(\frac{P - A_L}{B_L}\right) + 0.7\rho_{\text{mc}},\ \rho_{\text{hi}}\right)$.
$\rho_L \leftarrow \text{mid}\left(\rho_L^{\text{EOS}}, \rho_L^*, \rho_L^{\text{extrap}}\right)$.
Calculate P^{calc} from evaluating the EOS model at ρ_L.
Return ρ_L, P^{calc}.

The solution methods used for the EOS model and ρ_L^{Ω} or ρ_V^{Ω} must be robust. For cubic equations of state, for instance, it is recommended to use Cardano's analytic method for cubic equations to solve the EOS, and a bisection method to search for ρ_L^{Ω} or ρ_V^{Ω} in the prescribed interval. The following subsection discusses the behavior of these quantities to show that the `if` statement in each of these algorithms does not introduce any potential discontinuities.

Algorithm 6.2: Evaluate vapor density.

Solve the EOS model for ρ_V^{EOS}.
Calculate ρ_{mc}, ρ_{lo} and ρ_{hi}.
Solve Equation (6.4) for ρ_V^{Ω} in the interval $[\rho_{\text{lo}}, \kappa\rho_{\text{mc}}]$.
if no solution is found **then**
  $\rho_V^* \leftarrow \kappa\rho_{\text{mc}}$.
else
  $\rho_V^* \leftarrow \text{mid}(\rho_{\text{lo}}, \rho_V^{\Omega}, \kappa\rho_{\text{mc}})$.
end if
Calculate P^* and P_ρ^* at ρ_V^* from the EOS model.
$A_V \leftarrow \frac{1}{P^*}$.
$B_V \leftarrow -P_\rho^*/(P^*)^2$.
$C_V \leftarrow -\left|A_V + \frac{1}{2}B_V\left(\rho_{\text{mc}} - \rho_V^*\right)\right| / \left(\frac{1}{2}(\rho_{\text{mc}} - \rho_V^*)\right)^2$.
Calculate ρ_V^{extrap} from Equation (6.27).
$\rho_V \leftarrow \text{mid}\left(\rho_V^{\text{EOS}}, \rho_V^*, \rho_V^{\text{extrap}}\right)$.
Return ρ_V.

Bifurcation analysis of ρ_L^{Ω} and ρ_V^{Ω}

Figure 6-2 shows several isotherms of the residual of Equation (6.4) generated by the Peng-Robinson EOS for the equimolar ethane/n-heptane mixture previously shown in Figure 6-1. Also indicated is the parametric variation in the turning point location of the isotherms with temperature (dotted line), which here is the point where the second partial derivative of pressure with respect to density is zero. The qualitative behavior of these curves is typical of the general behavior of the cubic equations of state for mixtures.

Of key importance here is the observation that the only situation in which there is no root of the residual function of Equation (6.4) is when the temperature is sufficiently high (or in an analogous situation, when the composition is sufficiently lean). Moreover, decreasing the temperature to create a solution will always cause the root(s) to appear very close to ρ_{mc} (the $-0.1R_GT$ term moves the bifurcation point very slightly below ρ_{mc}) and then bifurcate outwards from that point. This means that until the roots have spread far enough from this point to exceed ρ_{mc} in the liquid case or fall below $\kappa\rho_{\text{mc}}$ in the vapor case, the mid functions in Equations (6.13) and (6.14) will initially choose $\rho_L^* = \rho_{\text{mc}}$ and $\rho_V^* = \kappa\rho_{\text{mc}}$, which is consistent with the

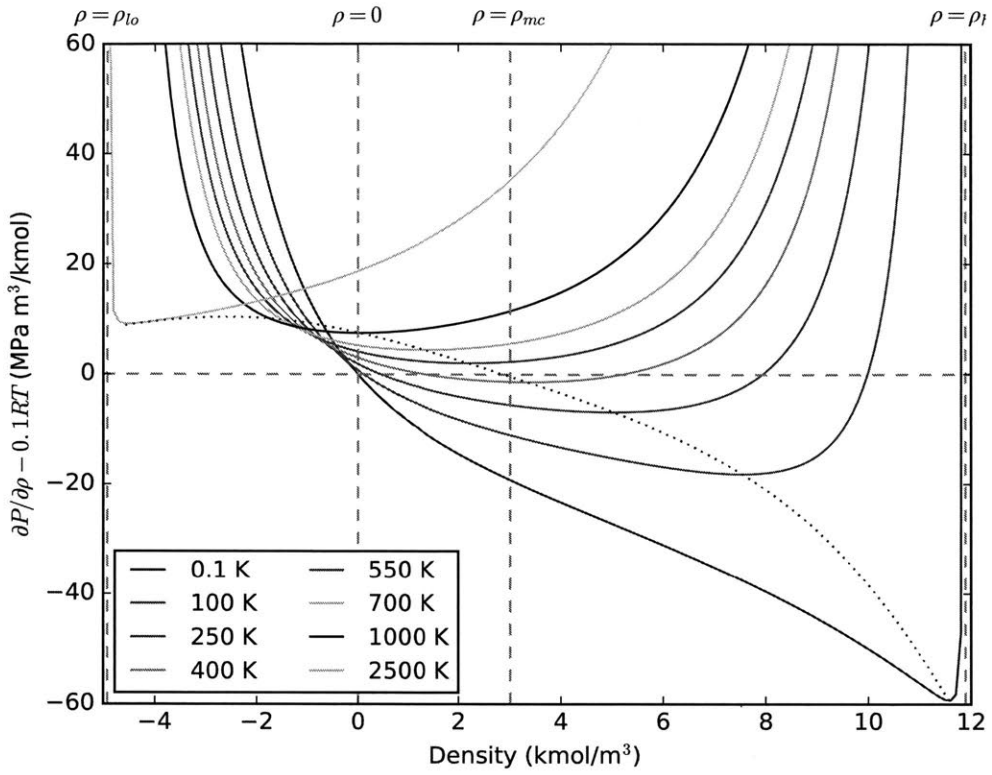


Figure 6-2: Behavior of the residual function of Equation (6.4) for an equimolar ethane/n-heptane mixture over a wide range of temperatures. The dotted line demarcates the location of the turning points of the curves as a function of temperature between 0.1K and 2500 K.

values enforced by the `if` statements in Algorithms 6.1 and 6.2 when no solution exists. Therefore, choosing ρ_{mc} (or $\kappa\rho_{\text{mc}}$) as the default solution when Equation (6.4) is not satisfied will lead to continuous behavior of ρ_L^* and ρ_V^* even when temperature (or composition) varies enough to cause roots to appear or disappear. Additionally, for the cubic equations of state, these roots can only exist between $\rho = \rho_{\text{lo}}$ and $\rho = \rho_{\text{hi}}$ (as shown in Figure 6-2) assuming that ρ_{lo} and ρ_{hi} are properly chosen as points of singularity for the equation of state (e.g. for PR: $\rho_{\text{lo}} \equiv \frac{1-\sqrt{2}}{b}$ and $\rho_{\text{hi}} \equiv \frac{1}{b}$; for SRK: $\rho_{\text{lo}} \equiv -\frac{1}{b}$ and $\rho_{\text{hi}} \equiv \frac{1}{b}$). This means that roots will not appear and bifurcate inwards from beyond ρ_{lo} or ρ_{hi}, so there will be no possibility of a discontinuity in the value of ρ_L^* (or ρ_V^*) with varying temperature and/or composition, i.e. it is impossible to jump from the default value ρ_{mc} (or $\kappa\rho_{\text{mc}}$) when there is no solution of Equation (6.4) to a value beyond ρ_{hi} (or ρ_{lo}) when a solution first appears.

For other equations of state where values of ρ_{lo} and ρ_{hi} cannot be calculated exactly, it is possible that roots of Equation (6.4) may exist in regions beyond the interval $[\rho_{\text{lo}}, \rho_{\text{hi}}]$. However, it is important to note that, assuming reasonable choices are made for these quantities, this will only happen in particularly extreme conditions as temperature approaches either zero or $+\infty$. In these rare cases, it may be necessary to increase the range of the bisection search in the appropriate direction iteratively (e.g. in the liquid case, the next search interval could be $[\rho_{\text{hi}}, k\rho_{\text{hi}}]$ for some scalar $k > 1$) until a value is found that can be used in the evaluation of Equations (6.13) or (6.14). Then, the mid functions in these equations will simply set $\rho_L^* = \rho_{\text{hi}}$ or $\rho_V^* = \rho_{\text{lo}}$ and so the continuity of these values is never compromised.

Example 6.1 illustrates the use of Algorithms 6.1 and 6.2 for evaluating the density of the mixture from Figure 6-1.

Example 6.1. Consider a mixture of ethane and n-heptane, as in Mathias *et al.*,[79] described by the Peng-Robinson cubic EOS. Pure component and binary interaction parameters for the components were obtained from Aspen Plus v8.4.[5] Expressions for the values of ρ_{mc} and T_{mc} may be obtained analytically from Equation (6.15) for the Peng-Robinson equation of state. Doing so gives $\rho_{\text{mc}} = \frac{1}{3b}\left(\sqrt[3]{2(4+3\sqrt{2})} - \frac{2}{\sqrt[3]{2(4+3\sqrt{2})}} - 1\right)$ $\approx \frac{0.25308}{b}$, and similarly, $T_{\text{mc}} \approx \frac{0.17014a}{Rb}$. Note that for mixtures described by this EOS, the parameter b is a function of composition and the parameter a is a function of temperature and composition (details given in Appendix B). Accordingly, mechanically supercritical temperatures are most easily detected by substituting the expression $\frac{0.17014a(T,\mathbf{z})}{Rb(\mathbf{z})}$ in place of T_{mc} in Equation (6.27), rather than solving for the value of T_{mc} *a priori*. As noted in the previous section, the lower bound on density values is given by $\rho_{\text{lo}} \equiv \frac{1-\sqrt{2}}{b}$ and the upper bound on density values is given by $\rho_{\text{hi}} \equiv \frac{1}{b}$ for this equation of state. The $P - \rho$ isotherm predicted by the Peng-Robinson equation of state, as well as the vapor and liquid densities obtained by solving the cubic at 420 K and $\mathbf{x}_L = \mathbf{y}_V = (0.5, 0.5)$ are shown with solid lines in Figure 6-3 for pressures between 0.1 and 10 MPa. Note that below $P = 1.88$ MPa and above $P = 2.67$ MPa, the densities of the two phases are indistinguishable, as the cubic equation of state only has one real root in these regions. In the liquid phase, a root of Equation

(6.4) is found at $\rho_L^{\Omega} = 4.66$ kmol/m^3 by a bisection algorithm, and for this mixture, $\rho_{\text{mc}} = 3.01$ kmol/m^3 and $\rho_{\text{hi}} = 11.89$ kmol/m^3. Since $\rho_{\text{mc}} < \rho_L^{\Omega} < \rho_{\text{hi}}$, the mid statement in Line 8 of Algorithm 6.1 will set the boundary value as ρ_L^{Ω}. Note that this value has no dependence on pressure, so it will be constant when constructing isotherms at constant composition. The value of the extrapolation function may then be calculated from Equation (6.20). For this temperature and composition, when $P < P^* = 1.94$ MPa, $\rho_L^{\text{bound}}(T, \mathbf{x}_L) > \rho_L^{\text{extrap}}(P, T, \mathbf{x}_L) > \rho_L^{\text{EOS}}(P, T, \mathbf{x}_L)$. Therefore, the mid function in Equation (6.21) will choose $\rho_L = \rho_L^{\text{extrap}}(P, T, \mathbf{x}_L)$ and the vapor-like density calculated from the equation of state is avoided. Similarly when $P > P^*$, $\rho_L^{\text{extrap}}(P, T, \mathbf{x}_L) > \rho_L^{\text{EOS}}(P, T, \mathbf{x}_L) > \rho_L^{\text{bound}}(T, \mathbf{x}_L)$ and the mid function in Equation (6.21) will correctly choose $\rho_L = \rho_L^{\text{EOS}}(P, T, \mathbf{x}_L)$. In the event that $P = P^*$, the value of all three functions in the mid statement are equal and a density corresponding to this common value is returned.

In the vapor phase, ρ_V^{Ω} is found to be 1.59 kmol/m^3 by a bisection algorithm, which is set by the mid function as the boundary of the allowed region since it is less than ρ_{mc} but greater than $\rho_{\text{lo}} = -4.92$ kmol/m^3. The value of the extrapolation is then calculated using Equation (6.27) and the mid function again selects the appropriate density values to avoid the nonphysical liquid-like roots given by the equation of state at high pressures.

For additional illustration, the auxiliary functions are also shown as functions of temperature for an equimolar mixture at 2.0 MPa (Figure 6-4) and as functions of composition at 420 K and 2.0 MPa (Figure 6-5). Note the non-constant dependence of the boundary values on temperature and composition.

Figure 6-6 (top row) shows the final $P - \rho$ isotherms with and without the density extrapolation algorithm at 420 K for the equimolar mixture. The same comparison is shown along a supercritical isotherm with $T = 500$ K in the bottom row. Note that the liquid and vapor density profiles coincide at high pressure, which prevents a spurious enthalpy of vaporization from being calculated at such conditions. See Figure 6-7 for examples of the pseudoproperties (enthalpy departures and fugacity coefficients) calculated through the use of the density values returned by Algorithms

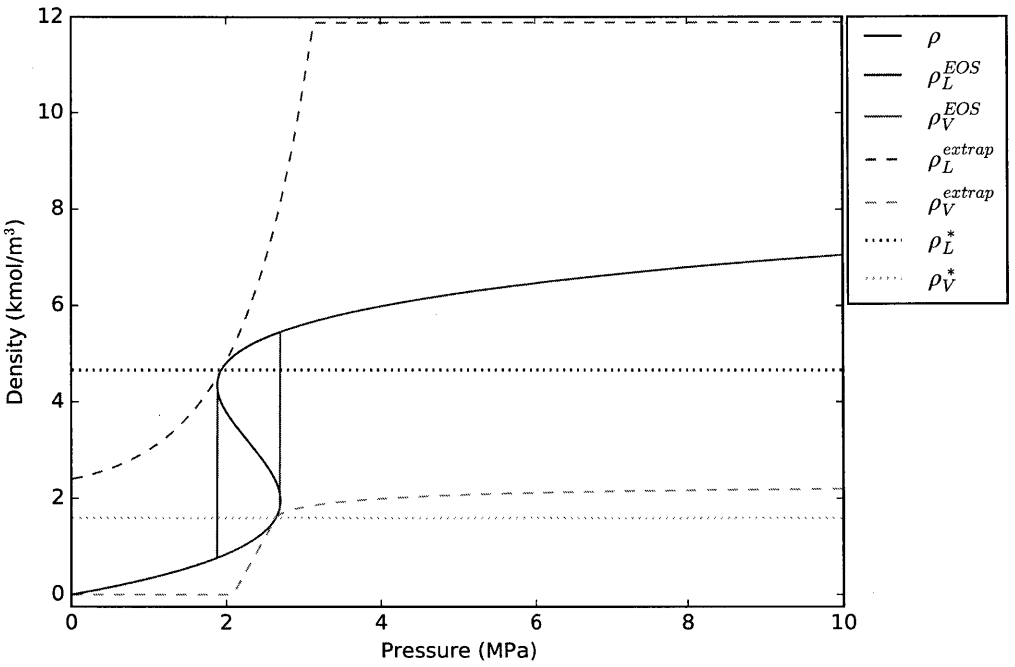


Figure 6-3: The auxiliary functions needed for density extrapolation at 420 K for an equimolar ethane/n-heptane mixture as functions of pressure using the nonsmooth method.

6.1 and 6.2. These results are qualitatively very similar to those presented by Mathias *et al.*,[79] *Xu et al.*[148] and Zhao and Saha,[151] suggesting that the nonsmooth algorithms are indeed calculating the same quantities as the previously published approaches.

6.3.2 Calculation of sensitivity information for extrapolated density values

A significant advantage of the nonsmooth formulations for density calculation proposed in the previous section is the ease with which sensitivity information about the extrapolated values may be calculated. In particular, the functions ρ_L^{mid} and ρ_V^{mid} are designed to be amenable to new automatic methods for calculating LD-derivatives to yield computationally-relevant generalized derivative elements.

The calculations of sensitivities proceeds almost identically to the execution of Algorithms 6.1 and 6.2. In addition to values of the P, T, and $\mathbf{z}$ at which the density and its LD-derivatives are to be calculated, a directions matrix $\mathbf{M} \in \mathbb{R}^{(n_c+2)\times k}$ must

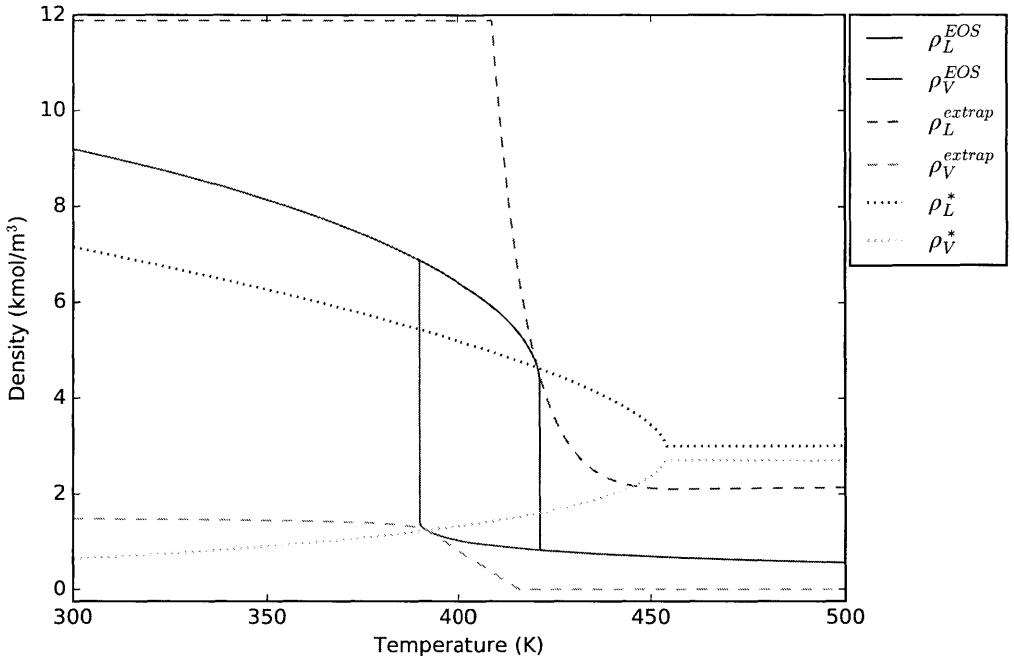


Figure 6-4: Illustration of the auxiliary functions needed for density extrapolation at 2.0 MPa for an equimolar ethane/n-heptane mixture as functions of temperature using the nonsmooth method.

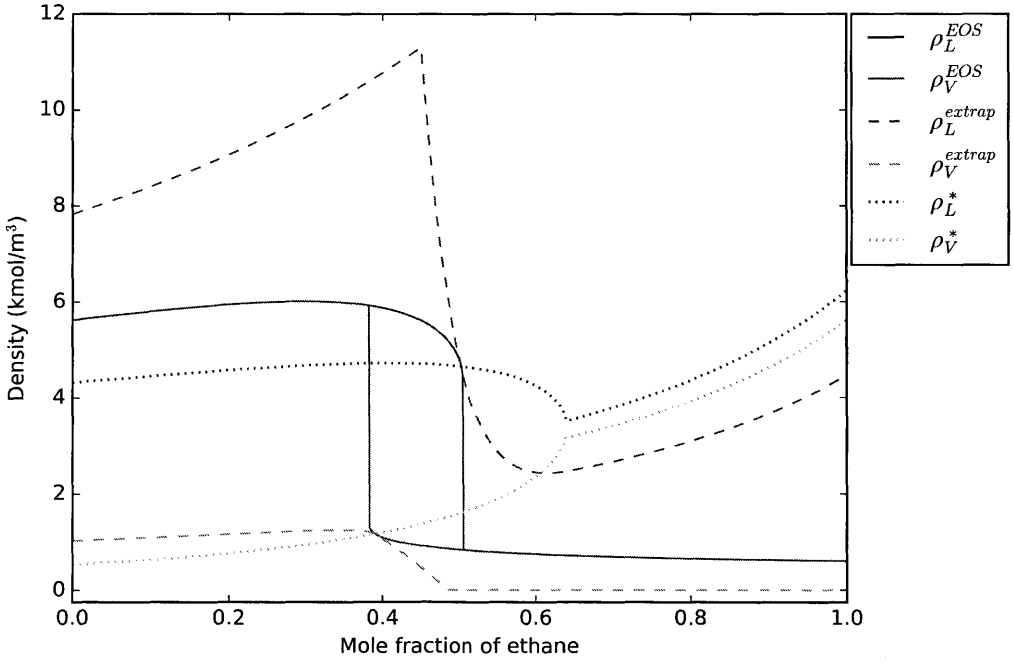


Figure 6-5: Illustration of the auxiliary functions needed for density extrapolation at 420 K and 2.0 MPa as functions of composition using the nonsmooth method.

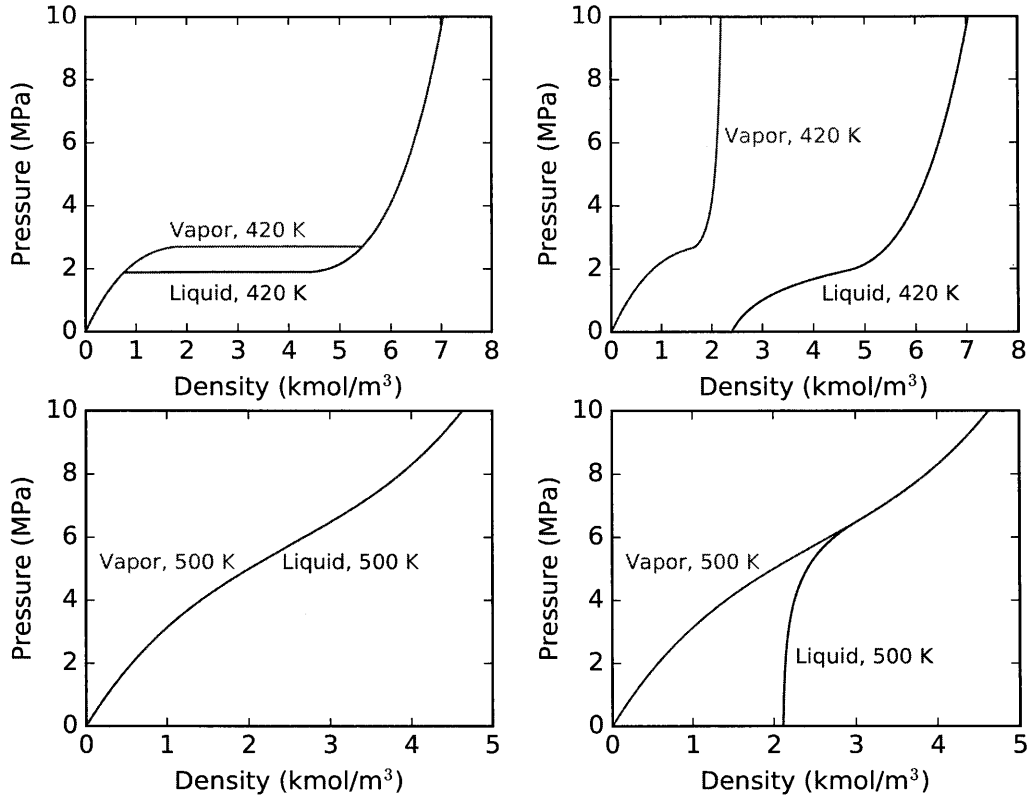


Figure 6-6: $P - \rho$ behavior of an equimolar ethane/n-heptane mixture. Top left: 420 K without density extrapolation. Top right: 420 K with density extrapolation. Bottom left: 500 K without density extrapolation. Bottom right: 500 K with density extrapolation.

also be provided that corresponds to the directions in which LD-derivatives with respect to P, T and $\mathbf{z}$ are needed. In general, this will be provided by the higher-level algorithm that is calling the density evaluation subroutine.

Analogous to Algorithms 6.1 and 6.2, the first step involves solving the EOS at the given P, T and $\mathbf{z}$. Next, the classical implicit function theorem is used to calculate the derivatives of ρ_j^{EOS} at the solution of the EOS in phase j with respect to P, T and $\mathbf{z}$. Consider an EOS in the form $h(T, P, \mathbf{z}, \rho) = 0$ by rearrangement of the standard pressure-explicit form with $\rho \in \mathbb{R}$. Let $\mathbf{p} = (T, P, \mathbf{z}) \in \mathbb{R}_+ \backslash \{0\} \times \mathbb{R}_+ \backslash \{0\} \times (0,1)^{n_c}$, so that the previous equation may be written as $h(\mathbf{p}, \rho) = 0$. Let $(\hat{\mathbf{p}}, \hat{\rho})$ be a solution of this equation. Assuming h is differentiable at $(\hat{\mathbf{p}}, \hat{\rho})$, as would be the case for all

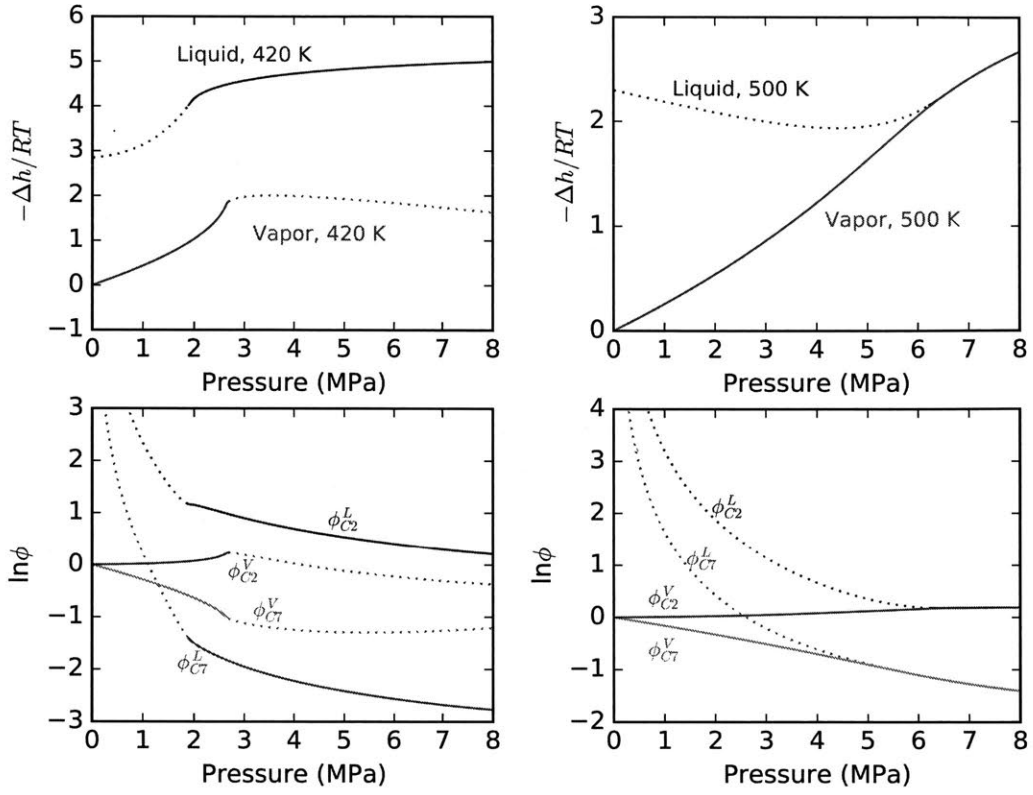


Figure 6-7: Pseudoproperties calculated from density extrapolations for an equimolar ethane/n-heptane mixture. Dotted lines show the extrapolated regions of each curve. Top left: enthalpy departures at 420 K. Top right: enthalpy departures at 500 K. Bottom left: fugacity coefficients at 420 K. Bottom right: fugacity coefficients at 500 K.

polynomial-based equations of state, and that $\frac{\partial h}{\partial \rho}(\hat{\mathbf{p}}, \hat{\rho}) \neq 0$, then:

$$\frac{\partial h}{\partial \rho}(\hat{\mathbf{p}}, \hat{\rho}) \frac{\partial \rho}{\partial \mathbf{p}}(\hat{\mathbf{p}}, \hat{\rho}) = -\frac{\partial h}{\partial \mathbf{p}}(\hat{\mathbf{p}}, \hat{\rho}), \tag{6.30}$$

by sensitivity analysis of the implicit function, and therefore:

$$\frac{\partial h}{\partial \rho}(\hat{\mathbf{p}}, \hat{\rho}) \left(\frac{\partial \rho}{\partial \mathbf{p}}(\hat{\mathbf{p}}, \hat{\rho}) \mathbf{M} \right) = -\frac{\partial h}{\partial \mathbf{p}}(\hat{\mathbf{p}}, \hat{\rho}) \mathbf{M}, \tag{6.31}$$

where $\mathbf{M}$ is the directions matrix provided by the higher-level algorithm. This equation may be easily solved for the LD-derivative $\rho'(\hat{\mathbf{p}}; \mathbf{M}) \equiv \frac{\partial \rho}{\partial \mathbf{p}}(\hat{\mathbf{p}}, \hat{\rho}) \mathbf{M}$ given $\frac{\partial h}{\partial \rho}(\hat{\mathbf{p}}, \hat{\rho})$ and $\frac{\partial h}{\partial \mathbf{p}}(\hat{\mathbf{p}}, \hat{\rho})$, both of which can be calculated either analytically or using automatic

differentiation. This calculation may be added as an additional step after Line 2 of Algorithms 6.1 and 6.2. In a computer implementation, the resulting LD-derivative $\rho'(\hat{\mathbf{p}}; \mathbf{M})$ is then associated in memory with the solution value $\hat{\rho}$, e.g. as an object compatible with AD subroutines. The calculation of LD-derivatives is not performed at local extrema of the EOS (where sensitivity analysis would fail since $\frac{\partial h}{\partial \rho}(\hat{\mathbf{p}}, \hat{\rho}) = 0$) as the branches of the mid functions in Equations (6.21) and (6.28) corresponding to the EOS solution will never be active at such points. Moreover, as a computational cost-saving measure, the calculation of EOS sensitivities can be entirely avoided in regions where they are certain not to be needed, i.e. for the liquid phase when $P < P^*$ and for the vapor phase when both $P > P^*$ and the last term in Equation (6.27) is inactive. Note that implementing these checks requires the EOS sensitivity calculation be performed later in Algorithms 6.1 and 6.2 once the other relevant quantities have been calculated.

Next, the value of ρ_{mc} and its derivatives must be found. If ρ_{mc} has a simple explicit definition, as it does for cubic equations of state, then this expression can be automatically differentiated in the required directions using the vector forward mode of AD. If this is not the case, then its derivatives can be found through another application of the classical implicit function sensitivity result to the solution of the two equation system given by Equation (6.15) to furnish appropriate LD-derivatives as in Equation (6.31). The LD-derivatives of ρ_{lo} and ρ_{hi} may also be obtained by automatic differentiation of the composition-dependent EOS-specific expressions that define these quantities, if available; otherwise, they may be calculated by appropriate sensitivity analysis of the procedure used to determine these bounds (or set to zero if the bounds are assumed to be constant with respect to composition). An analogous procedure to that used for the other implicit sensitivities can then be used to evaluate the sensitivities of ρ_j^{Ω} with respect to $\mathbf{p}$ from Equation (6.4), provided that a solution exists. This calculation can be added before Line 8 of Algorithms 6.1 and 6.2. If a solution does not exist, then this calculation is unnecessary for obtaining correct LD-derivatives of the final density, as noted earlier.

The remainder of the algorithms are written as explicit computations of differ-

entiable and piecewise differentiable functions (plus the `if` statement that does not impact the continuous nature of the algorithm, as discussed earlier), and as such, the sensitivity analysis for Algorithms 6.1 and 6.2 may be readily performed by application of modified vector-forward mode of AD for LD-derivative evaluation from Khan and Barton.[64] In an implementation of that method using operator overloading, the evaluation of the smooth and nonsmooth expressions in Algorithms 6.1 and 6.2 is automatically replaced with combined value and LD-derivative evaluation, requiring no additional changes to the algorithms beyond appropriate templating of the functions to operate on AD objects in place of floating point numbers. The LD-derivatives of the implicit functions found previously are propagated automatically as a result of the sharp chain rule (Equation (2.14)). The calculus rule for the LD-derivatives of the mid function may be found in Barton *et al.*[13] or obtained from the identity in Equation (2.9).

The LD-derivatives of the densities in the identity directions calculated by differentiation of Algorithms 6.1 and 6.2 for the mixture in Example 1 at 420 K are shown in Figure 6-8. These quantities are equivalent to classical partial derivatives wherever the functions are differentiable. Note that when viewed solely as functions of pressure, the functions ρ_L^{mid} and ρ_V^{mid} are in fact everywhere differentiable despite their definition in terms of nonsmooth functions. However, as can be seen later in Example 6.3, they are in general nonsmooth functions.

6.4 Examples

Three additional examples are now presented to highlight that the nonsmooth density extrapolation strategy is applicable to more complex equations of state (Example 6.2), more complex mixtures (Example 6.3) and flash calculations (Example 6.4).

Example 6.2. This example demonstrates that the method detailed in this chapter is also applicable to more general virial-type equations of state. The same equimolar ethane/n-heptane mixture is studied at 400 K using the BWRS EOS as given by Starling.[115] Pure component and binary interaction parameters for the components

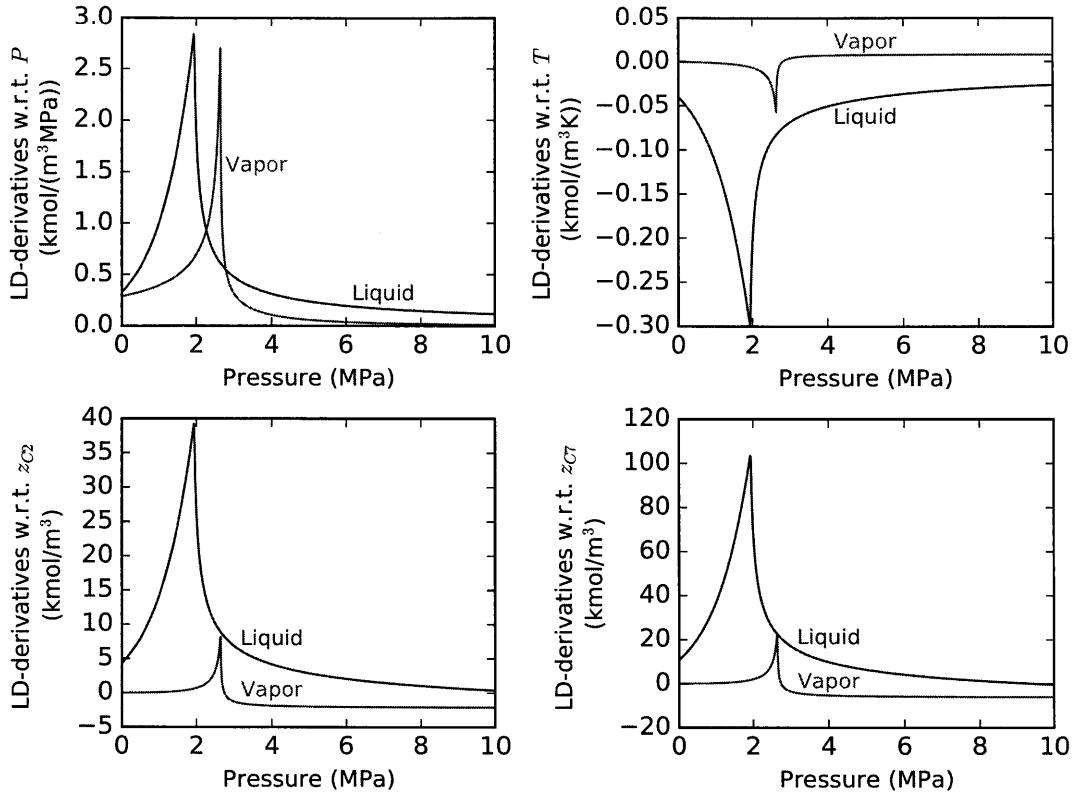


Figure 6-8: Pressure variation of the LD-derivatives of density in the directions $\mathbf{I}_{4\times4}$ with respect to the model parameters for an equimolar ethane/n-heptane mixture at 420 K.

were obtained from Aspen Plus v8.4.[5] The BWRS equation itself was solved using the method described by Mills *et al.*[83] One of the steps in their solution algorithm involves finding the highest density point at which $P_{\rho_L}(\rho_L, T, \mathbf{x}_L) = 0$ for the liquid phase and the lowest density point at which $P_{\rho_V}(\rho_V, T, \mathbf{y}_V) = 0$ for the vapor phase. These values are stored to use as the lower and upper endpoint, respectively, for the bisection search for the values of ρ_L^{Ω} and ρ_V^{Ω}. As the mechanical critical properties cannot be determined analytically for the BWRS EOS, the values for ρ_{mc} and T_{mc} were taken as the mole fraction weighted average of the pure component critical density and temperature, respectively. As an alternative, the highest and lowest density inflection points of the $P - \rho$ isotherm could be used as ρ_{mc} values for the liquid and vapor phases, respectively, though this is obviously more expensive and

requires additional iterative calculations. As suggested in Mills *et al.*,[83] the value of ρ_{hi} is set to 32.0 kmol/m^3 and the value of ρ_{lo} is set to zero. The results of applying Algorithms 6.1 and 6.2 to this mixture at these conditions with variable pressure are shown in Figure 6-9. Note that even though the predicted isotherm from the BWRS EOS is qualitatively different from that obtained by using the Peng-Robinson EOS (c.f. Figure 6-1), this does not affect the application of these new density evaluation algorithms.

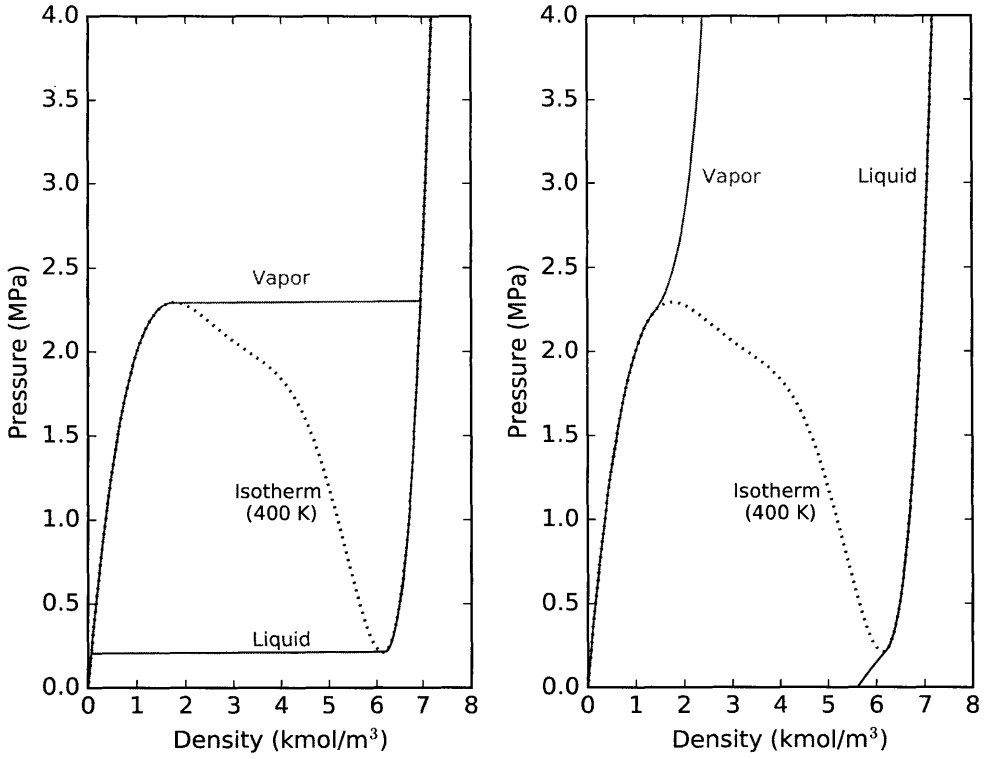


Figure 6-9: $P - \rho$ behavior of an equimolar ethane/n-heptane mixture described by the BWRS equation of state at 400 K. Left: without density extrapolation. Right: with density extrapolation.

Example 6.3. Consider the natural gas mixture entering the liquefaction process from the main example in Chapter 7. The molar composition of the mixture is 1.00% nitrogen, 91.60% methane, 4.93% ethane, 1.71% propane, 0.35% n-butane, 0.40% isobutane and 0.01% isopentane. The mixture is well-described by the Peng-

Robinson cubic EOS. When density values are requested in the superheated and subcooled regions of the phase-space, solving the EOS directly frequently leads to calculating the same density value for both phases. This in turn causes the flash algorithm to converge to the trivial solution frequently outside of a narrow region, which impacts the performance of the flowsheet simulations. Algorithms 6.1 and 6.2 are therefore applied to generate more reasonable density values for each phase. The construction of the extrapolating functions and the final results are shown in Figure 6-10 parametrized by pressure at 200 K (top row) and parametrized by temperature at 5.0 MPa (middle row). Note that the density returned by the algorithm is a continuous yet nondifferentiable function of temperature in this instance, as indicated by the discontinuities in the bottom right plot of the temperature LD-derivatives.

Example 6.4. As a final example, the differences in the calculated phase densities and the computational cost from using the nonsmooth extrapolation algorithm vs. no extrapolation are shown in the context of flash calculations. Three different mixtures are considered that will be relevant in the following chapter: the natural gas mixture from Example 6.3, the 16-component mixture from Cavett's flowsheeting problem[22] and a refrigerant stream with molar composition: 5.82% nitrogen, 20.62% methane, 39.37% ethane and 34.19% n-butane. Table 6.1 shows selected results of PT-flashes performed using the nonsmooth inside-out algorithm from Chapter 5 to an outer loop tolerance of 10^{-8} and inner loop tolerance of 10^{-9} using Anderson acceleration. For each mixture, the first pair of columns shows flash calculation results at mechanically supercritical conditions, the middle pair shows results in the middle of the two-phase region, and the last pair shows results at mechanically subcritical conditions. In each case, the calculations are performed assuming the feed streams are flowing at 1.0 kmol/s at the temperature and pressure specified in the table using the Peng-Robinson cubic EOS. The timing was performed on an Intel Xeon E5-1650 v2 workstation using six cores at 3.50 GHz and 12 GB RAM running Ubuntu v14.04.

The results clearly indicate the differences in the predicted densities behave as intended: the liquid phase density is always appreciably greater than that of the liquid. In addition, it is seen that when density extrapolation is not used, the trivial

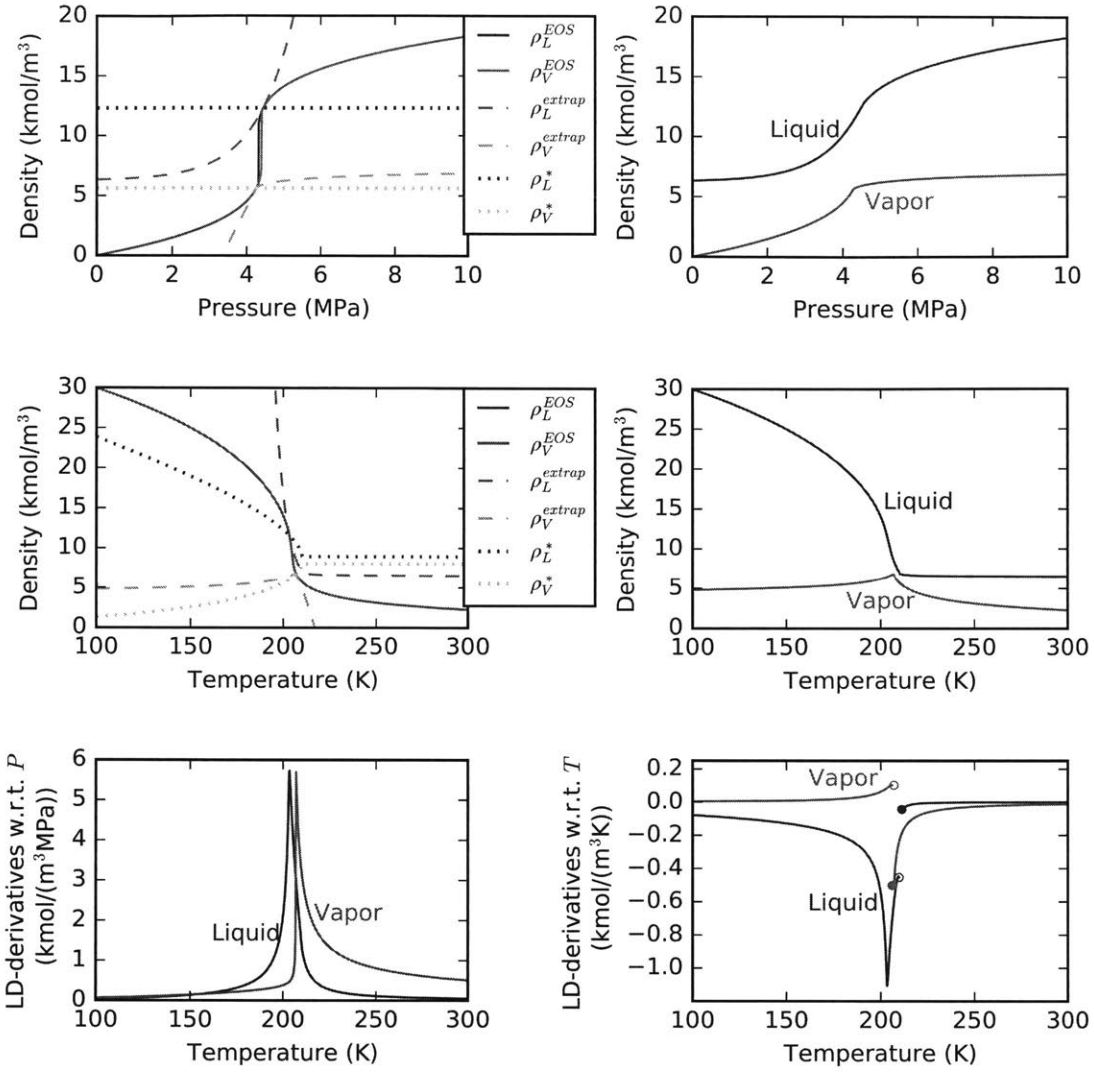


Figure 6-10: Top left: construction of the density extrapolation and boundary functions for a natural gas mixture at 200 K. Top right: density vs. pressure isotherm returned by the nonsmooth algorithm at 200 K. Middle left: construction of the density extrapolation and boundary functions for a natural gas mixture at 5.0 MPa. Middle right: density vs. temperature isobar returned by the nonsmooth algorithm at 5.0 MPa. Bottom left: LD-derivatives in $\mathbf{I}_{9\times9}$ directions of the calculated density w.r.t. pressure at 5.0 MPa. Bottom right: LD-derivatives in $\mathbf{I}_{9\times9}$ directions of the calculated density w.r.t. temperature at 5.0 MPa.

solution of the flash equations (indicated by equal liquid and vapor phase densities) is found by the solver in each of the six calculations with single-phase solutions. The trivial solution is never found when the nonsmooth density extrapolation method is active. Additionally, even in the cases where the result is a two-phase mixture, it is

Table 6.1: Results of example PT-flash calculations for mixtures both without density extrapolation ("None" columns) and with density extrapolation ("Nonsmooth" columns). The symbol α denotes the fraction of the feed stream that is vaporized. Iterations data refers to the number of passes through the outer-loop of the nonsmooth inside-out algorithm.

Natural Gas	None	Nonsmooth	None	Nonsmooth	None	Nonsmooth
P (MPa)	5.5	5.5	5.5	5.5	5.5	5.5
T (K)	295.15	295.15	204.71	204.71	110.15	110.15
α	1.0	1.0	0.5	0.5	0.0	0.0
ρ_L (kmol/m^3)	2.598	6.152	15.089	15.089	29.167	29.167
ρ_V (kmol/m^3)	2.598	2.598	7.493	7.493	29.167	5.337
Iterations	6	5	9	9	5	3
Time (ms)	4.5	3.6	5.2	5.9	4.7	3.7
Cavett's Mixture	None	Nonsmooth	None	Nonsmooth	None	Nonsmooth
P (MPa)	0.439	0.439	0.439	0.439	6.0	6.0
T (K)	600.0	600.0	308.92	308.92	200.0	200.0
α	1.0	1.0	0.5	0.5	0.0	0.0
ρ_L (kmol/m^3)	0.0894	1.422	6.632	6.632	10.692	10.692
ρ_V (kmol/m^3)	0.0894	0.0894	0.176	0.176	10.692	6.738
Iterations	3	5	4	4	14	8
Time (ms)	8.4	7.4	8.1	10.2	12.3	9.2
Refrigerant	None	Nonsmooth	None	Nonsmooth	None	Nonsmooth
P (MPa)	1.713	1.713	0.202	0.202	1.713	1.713
T (K)	370.0	370.0	206.25	206.25	110.15	110.15
α	1.0	1.0	0.5	0.5	0.0	0.0
ρ_L (kmol/m^3)	0.619	3.148	13.682	13.682	19.227	19.227
ρ_V (kmol/m^3)	0.619	0.619	0.121	0.121	19.227	3.254
Iterations	43	12	4	4	12	4
Time (ms)	13.0	5.3	3.7	4.3	6.1	4.0

seen that the use of the extrapolation method does not significantly affect the cost per iteration of the flash calculation (as expected from the conclusions of Mathias and Benson[78]). In the cases where the calculations with and without density extrapolation arrive at distinct results, using the nonsmooth extrapolations generally results in fewer outer-loop iterations of the inside-out algorithm and therefore lower computational cost, in addition to returning a physically correct answer.

Next, more challenging PQ-flash calculations are performed on the same mixtures. In each case, an adiabatic flash is performed on a feed stream at the same pressure and temperature as the corresponding stream in Table 6.1. This implies that, if the

methods are working correctly, the same temperature, vapor fraction and densities should be calculated as in the previous case at the given pressure. The results are shown in Table 6.2. The differences between using and not using the extrapolation methods are far more pronounced here, as four of the nine example calculations fail to converge to a solution (even a trivial one) after 100 outer loop iterations when no extrapolation technique is used. Of the remaining five calculations, two converge to the trivial solution and one converges to an incorrect solution altogether without density extrapolation. Note that again, the overall computation time is not significantly negatively affected (in fact, sometimes it is even improved) by the use of Algorithms 6.1 and 6.2. Moreover, the impact of these density extrapolation algorithms in calculations such as these had an enabling impact on the simulations of a natural gas liquefaction process performed in this work. Such simulations were unable to converge without the nonsmooth density extrapolation methods, as both the outright failures in PQ-flashes and the discontinuities introduced by converging to trivial solution points in both PQ- and PT-flashes led to failures in the equation-solving algorithm in almost all cases. However, with the density extrapolation algorithms implemented, these simulations were extremely robust and efficient, as shown in Chapter 7.

6.5 Conclusions

New nonsmooth algorithms have been presented for the extrapolation of liquid and vapor phase density when evaluation of an EOS alone yields unacceptable values. In addition to the benefit of the algorithms being less convoluted to implement than many other approaches, accurate generalized derivative information may be readily obtained with minimal extra calculations. The method is not specific to cubic equations of state and may also used for more general and complex virial-type models. The nonsmooth strategy carries very little additional computational cost beyond that of simply evaluating density from the EOS itself and can successfully avoid convergence to the trivial solution of the flash equations when embedded in a VLE solution algorithm. This strategy has been successfully employed in large-scale complex flowsheet

Table 6.2: Results of adiabatic PQ-flash calculations performed on the streams initially at the temperatures and pressures given in Table 1, both without density extrapolation ("None" columns) and with density extrapolation ("Nonsmooth" columns). Iterations data refers to the number of passes through the outer-loop of the nonsmooth inside-out algorithm and dashes "–" represent failed calculations.

Natural Gas	None	Nonsmooth	None	Nonsmooth	None	Nonsmooth
P (MPa)	5.5	5.5	5.5	5.5	5.5	5.5
T (K)	–	295.15	206.29	204.71	110.15	110.15
α	–	1.0	0.0	0.5	0.0	0.0
ρ_L (kmol/m^3)	–	6.152	11.304	15.089	29.167	29.167
ρ_V (kmol/m^3)	–	2.598	11.304	7.493	29.167	5.337
Iterations	–	8	20	11	6	3
Time (ms)	–	7.7	24.6	24.5	70.3	4.9
Cavett's Mixture	None	Nonsmooth	None	Nonsmooth	None	Nonsmooth
P (MPa)	0.439	0.439	0.439	0.439	6	6
T (K)	–	600	308.92	308.92	–	200
α	–	1.0	0.5	0.5	–	0.0
ρ_L (kmol/m^3)	–	1.422	6.632	6.632	–	10.692
ρ_V (kmol/m^3)	–	0.0894	0.176	0.176	–	6.738
Iterations	–	5	7	6	–	8
Time (ms)	–	50.2	22.3	18.6	–	43.9
Refrigerant	None	Nonsmooth	None	Nonsmooth	None	Nonsmooth
P (MPa)	1.713	1.713	0.202	0.202	1.713	1.713
T (K)	–	370	206.25	206.25	110.15	110.15
α	–	1.0	0.5	0.5	0.0	0.0
ρ_L (kmol/m^3)	–	3.148	13.682	13.682	19.227	19.227
ρ_V (kmol/m^3)	–	0.619	0.121	0.121	19.227	3.254
Iterations	–	13	5	5	5	5
Time (ms)	–	21.2	20.2	24.6	5.3	5.8

calculations, as seen in the following chapter.

Chapter 7

Process flowsheeting with nonsmooth models and generalized derivatives

This chapter presents new methods for robustly simulating process flowsheets containing nondifferentiable models using exact sensitivity analysis methods for nonsmooth functions. Among other benefits, this allows flowsheeting problems to be equipped with the nonsmooth inside-out algorithms for non-ideal vapor-liquid equilibrium calculations developed in Chapter 5. Furthermore, process models for inherently nonsmooth unit operations may be seamlessly integrated into process flowsheets, so long as computationally-relevant generalized derivative information is computed correctly and communicated to the flowsheet convergence algorithm. These techniques may be used in either sequential-modular simulations or simulations in which the most challenging modules are solved using tailored external procedures while the remaining flowsheet equations are solved simultaneously. This new nonsmooth flowsheeting strategy is capable of solving process simulation problems involving nonsmooth models more reliably and efficiently than the algorithms implemented in existing software, and, in some cases, allows for the solution of problems that are beyond the capabilities of classical approaches. As examples of the latter, it will be shown that the nonsmooth approach is particularly well-suited for highly accurate simulation of natural

gas liquefaction processes, in which many nonsmooth modeling elements are present in combination with non-ideal thermodynamic behavior and complex heat transfer considerations.

7.1 Introduction

The use of analytical or exact derivative information in process simulation and optimization problems is known to be beneficial for achieving reliability, rapid convergence and high accuracy. In spite of this, methods for calculating exact derivatives are not commonly implemented in process flowsheet calculations. Instead, when derivative information is required, it is common that simple forward finite difference approximations are used. However, even when the perturbations for the difference approximations are chosen optimally, derivative evaluation will only be accurate to at most two-thirds of the precision of a function evaluation in terms of significant digits, and usually closer to half.[42] This loss of precision is unacceptable for many applications as it can lead to a loss of guaranteed convergence properties in equation-solving methods, e.g. local quadratic convergence in Newton-type methods, or even failure.

The benefits of instead using exact derivatives in process systems engineering applications have been noted by some authors for several decades. Chan and Prince[24] presented early evidence that application of the chain rule to propagate derivatives around flowsheets can be more efficient than perturbation-based differencing methods. They noted particularly significant computational cost improvement for flowsheets containing many copies of the same unit operation in local optimization studies. Chen and Stadtherr[26] also noted that the use of analytical derivatives was the most suitable technique for furnishing sensitivity information in their early simultaneous-modular simulator, though they noted that (at the time) the computational cost was often prohibitive. Wolbert *et al.*[146] demonstrated how the use of inexact derivatives can lead to failure to compute correct Newton steps in process optimization problems. They also described an implementation of analytical derivative evaluation using the chain rule and implicit function sensitivity analysis in a sequential modular process

simulator and gave examples of its efficacy.

However, in the absence of fully automatic techniques for exact numerical differentiation, the use of analytical derivative evaluation was generally considered laborious, error-prone and inefficient compared to implementations of simpler differencing approximations. However, the advent of AD provided researchers with a completely automatic tool for the calculation of exact numerical derivatives. For the unfamiliar reader, an excellent introduction to the subject of AD for classical derivative evaluation may be found in the text by Griewank and Walther.[42] Tolsma and Barton[123] compared multiple approaches for numerical derivative evaluation, including finite difference approximations, symbolic differentiation and AD for problems involving models commonly found in chemical processes. They concluded that AD (particularly the reverse mode implemented via code generation) is superior to all other approaches considered in terms of accuracy and the computational cost of evaluating Jacobian matrices. These same authors went on to develop DAEPACK, software that is able to extract and subsequently compile the information needed to perform sensitivity analysis from legacy models written in FORTRAN using AD.[124] Particularly in the dynamic case, not having to treat legacy models as simple black boxes that can only produce derivative information through perturbation of inputs has proved to be a highly useful numerical tool. This methodology was further improved by the work of Tolsma *et al.*,[125] who describe how an equation-oriented modeling environment may be seamlessly extended to include external procedures, e.g. tailored subroutines for specific models. The sensitivity analysis of the external subroutine is performed automatically and communicated back to the primary equation-based solver. This idea of separating challenging submodels from the upper-level solution algorithm but still communicating back values and automatically-calculated exact sensitivity information is used to great effect for nonsmooth models in the present chapter.

This chapter presents a methodology for extending the benefits of exact numerical sensitivity analysis in flowsheets described by differentiable models to flowsheets containing nonsmooth models and subroutines. In this approach, computationally-

relevant generalized derivative elements can be calculated automatically and exactly both for model equations written explicitly and for models whose outputs are defined implicitly and solved by external tailored procedures. This allows for process models such as flash drums, throttle valves, compressors and turbines to be solved with the nonsmooth inside-out algorithms for flash calculations developed in Chapter 5. These algorithms are designed to provide greater reliability for flash calculations than the classical inside-out algorithms when the phase regimes at the results of these calculations are not known or fixed *a priori*. The use of the nonsmooth toolkit described in Chapter 2 also allows for the inclusion of process models of inherently nonsmooth unit operations, such as multistream heat exchangers, into flowsheets. The nonsmooth flowsheeting strategy described in this chapter is primarily intended to service modular flowsheeting problems in which there are multiple recycle streams in addition to either design specifications or particular submodels for which it is advantageous to converge their model equations simultaneously with the overall flowsheet. Of course, it is not essential that a problem has either for the method to be used; however, the most significant advantages of the nonsmooth strategy will be seen for problems with these complicating factors.

7.2 The nonsmooth flowsheeting strategy

This section describes how the concepts from nonsmooth analysis reviewed in Chapter 2 find use in the context of process flowsheeting applications. A brief overview of the various approaches to process simulation are first given, then it is shown how the chain rule (Theorem 2.1) and implicit function theorem (Theorem 2.2) for L-smooth functions may be used in a process simulation context. Finally, specifics for embedding the nonsmooth inside-out algorithms for flash calculations into a simulation problem are detailed.

7.2.1 Approaches to process simulation

In the most fundamental terms, a process simulation is the problem of solving the (usually nonlinear) equation system $\mathbf{f}(\mathbf{p}, \mathbf{z}) = \mathbf{0}_m$ for $\mathbf{z} \in \mathbb{R}^m$ given the fixed parameter values $\mathbf{p} \in \mathbb{R}^{n_p}$, where $\mathbf{f} : W \subset \mathbb{R}^{n_p} \times \mathbb{R}^m \to \mathbb{R}^m$ is classically assumed to be a continuously-differentiable function on its domain. For the purposes of this chapter, this assumption is relaxed and $\mathbf{f}$ should instead be taken to be an L-smooth function on its domain. Simulation of process flowsheets purely by solving the equation system generated by combining the equations describing all the unit operations is known as the equation-oriented (EO) approach and generally results in large and challenging equation-solving problems. The common alternative method is the sequential-modular (SM) approach. The SM approach breaks a chemical process model down into interconnected, constituent modules that describe unit operations or stream manipulations. Each module contains an internal algorithm that takes parameters from the user and information from the inlet streams to solve the associated unit operation model and (if requested) obtain input-output sensitivities. The final calculated output stream variable values (and sensitivities) are then sent to the next module in the flowsheet. In the common case of recycle structures, the model must be solved iteratively after tearing streams to create an acyclic flowsheet. The thermodynamic state of each tear stream is fully described by $n_c + 2$ independently variable quantities (e.g. component flowrate of each species, pressure and temperature), which must be guessed and then iteratively reconciled with the calculated values returned to the stream after a flowsheet pass. In the fully SM case, the flowsheeting problem is therefore described by a model of the form:

$$\begin{aligned} \mathbf{y}_{(1)} - \mathbf{g}_{(1)}(\mathbf{p}, \mathbf{y}_{(1)}, \ldots, \mathbf{y}_{(n_t)}) &= \mathbf{0}_{n_c+2}, \\ &\vdots \\ \mathbf{y}_{(n_t)} - \mathbf{g}_{(n_t)}(\mathbf{p}, \mathbf{y}_{(1)}, \ldots, \mathbf{y}_{(n_t)}) &= \mathbf{0}_{n_c+2}, \end{aligned}$$

where $\mathbf{y}_{(i)}, i = 1, \ldots n_t$, with each $\mathbf{y}_{(i)} \in \mathbb{R}^{n_c+2}$ are the vectors of tear stream variables and where $\mathbf{g}_{(i)}, i = 1, \ldots n_t$, $\mathbf{g}_{(i)} : \mathbb{R}^{n_p} \times \mathbb{R}^{n_t(n_c+2)} \to \mathbb{R}^{n_c+2}$, are the tear stream func-

tions, i.e. the functions defined by a pass through the acyclic flowsheet. Note that the functions $\mathbf{g}_{(i)}$ are rarely known in closed form, and, for the purposes of this chapter, may be L-smooth functions on their domains. Additionally, the functions $\mathbf{g}_i$ will often consist of a composition of the functions describing the various unit operations in the process flowsheet, e.g. $\mathbf{g}_{(i)} \equiv \mathbf{u}_1(\mathbf{u}_2(\mathbf{u}_3(...(\mathbf{u}_{n_u}(\mathbf{p}, \mathbf{y}_{(1)}, \ldots, \mathbf{y}_{(n_t)})$ for the L-smooth functions $\mathbf{u}_j, j = 1, ..., n_u$, describing the unit operations in the calculation sequence. However, due to the unique structure of the tear equations, derivative-free methods for fixed-point iteration (e.g. successive substitution, Wegstein's method, Anderson acceleration, etc.) are often the most efficient and widely-used algorithms for the SM approach. However, if additional constraints or design specifications are made on the process, these algorithms may be poorly suited. Additionally, it may often be advantageous to converge some, but not all, of the process model equations simultaneously with the tear equations as opposed to during the unit-by-unit calculation sequence dictated by flowsheet connectivity. In these instances, (generalized) derivative-based methods become desirable once again. Therefore, the general problem for which the approach in this chapter is best suited may be written as follows:

$$\begin{aligned}
\mathbf{y}_{(1)} - \hat{\mathbf{g}}_{(1)}(\mathbf{p}, \mathbf{y}_{(1)}, \ldots, \mathbf{y}_{(n_t)}, \zeta) &= \mathbf{0}_{n_c+2}, \\
&\vdots \\
\mathbf{y}_{(n_t)} - \hat{\mathbf{g}}_{(n_t)}(\mathbf{p}, \mathbf{y}_{(1)}, \ldots, \mathbf{y}_{(n_t)}, \zeta) &= \mathbf{0}_{n_c+2}, \\
\mathbf{h}(\mathbf{p}, \mathbf{y}_{(1)}, \ldots, \mathbf{y}_{(n_t)}, \zeta) &= \mathbf{0}_{n_d},
\end{aligned}$$

where $\hat{\mathbf{g}}_{(i)}, i = 1, \ldots n_t$, $\hat{\mathbf{g}}_{(i)} : \mathbb{R}^{n_p} \times \mathbb{R}^{n_t(n_c+2)} \times \mathbb{R}^{n_d} \to \mathbb{R}^{n_c+2}$, are the appropriately modified L-smooth tear stream functions, the L-smooth function $\mathbf{h} : \mathbb{R}^{n_p} \times \mathbb{R}^{n_t(n_c+2)} \times \mathbb{R}^{n_d} \to \mathbb{R}^{n_d}$ includes the design specifications and/or complicating equations and $\zeta \in \mathbb{R}^{n_d}$ are the unknown variables afforded by these equations that may include a subset of the original model variables $\mathbf{z}$.

7.2.2 Propagation of sensitivity information

In general, there are two types of unit operation models that may be encountered in a process flowsheet: explicit and implicit. In explicit models, the direct relationship mapping the inputs to the outputs is known to the modeler. In implicit models, the outputs of the model are determined by solving an equation system involving both the inputs and outputs to the unit operation. In either case, both the values of the model outputs and the exact sensitivities of the outputs with respect to to the inputs must be calculated in the present approach.

For explicit models, the case is straightforward. Assume that both the model inputs, $\mathbf{p}^* \in \mathbb{R}^{n_p}$ (which for the sake of notational simplicity may generally include outputs from previous models, tear variables and unit operating parameters), and a directions matrix, $\mathbf{M} \in \mathbb{R}^{n_p \times k}$, that represents the set of k direction vectors in which LD-derivatives at $\mathbf{p}^*$ are needed, are known. Given an explicitly known function $\mathbf{u} : \mathbb{R}^{n_p} \to \mathbb{R}^m$ describing the unit operation that is L-smooth at $\mathbf{p}^*$, the values of the model outputs are given by $\mathbf{u}(\mathbf{p}^*)$ and the LD-derivatives at $\mathbf{p}^*$ in the directions $\mathbf{M}$ by $\mathbf{u}'(\mathbf{p}^*; \mathbf{M})$. The LD-derivatives may be evaluated using the modified vector-forward mode of AD described previously.

For an implicit model, as before denote the accumulated known model inputs by $\mathbf{p}^* \in \mathbb{R}^{n_p}$ and the directions matrix by $\mathbf{M} \in \mathbb{R}^{n_p \times k}$. Given the implicit unit operation model $\mathbf{u}_{\text{impl}}(\mathbf{p}, \mathbf{x}) = \mathbf{0}$, the model outputs are determined by solving this equation system for the value of the implicit function at $\mathbf{p}^*$, $\mathbf{x}(\mathbf{p}^*)$. Note that this may be performed by any suitable algorithm, often a procedure specifically tailored for the unit operation model at hand. If $\mathbf{u}_{\text{impl}}$ and the solution $(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ satisfy the hypotheses of Theorem 2.2, then the sensitivity analysis result implied by this theorem can be applied and Algorithm 2.3 may be used to obtain the LD-derivatives of the model outputs with respect to the model inputs in the $\mathbf{M}$ directions, $\mathbf{x}'(\mathbf{p}^*; \mathbf{M})$. If $\mathbf{u}_{\text{impl}}$ instead satisfies the hypotheses of the PC^1 implicit function theorem at $(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ with a modest number of known essentially active selection functions, then it may be desirable to use Algorithm 2.4 instead to calculate $\mathbf{x}'(\mathbf{p}^*; \mathbf{M})$.

Connectivity in a flowsheet is described by the composition of the individual unit operation functions, so that the outputs of one module (e.g. $\mathbf{u}'(\mathbf{p}^*;\mathbf{M})$ in the explicit case or $\mathbf{x}'(\mathbf{p}^*;\mathbf{M})$ in the implicit case) are among the inputs to subsequent units in the sequence. Fortunately, the sharp chain rule for LD-derivatives (Equation (2.14)) provides the means of propagating exact sensitivity information through intermediate calculations and compositions of functions. The sensitivity propagation is therefore accomplished automatically through use of the vector-forward mode of AD for LD-derivatives.

Figure 7-1 summarizes these calculations for explicit models, implicit models, and the connectivity thereof. In general, these calculations may all occur in a lengthy sequence of mixed implicit and explicit modules in a flowsheet; however, evaluating the constituent unit operations as detailed here will ensure that correct LD-derivatives are propagated through the flowsheet with respect to the initial directions matrix.

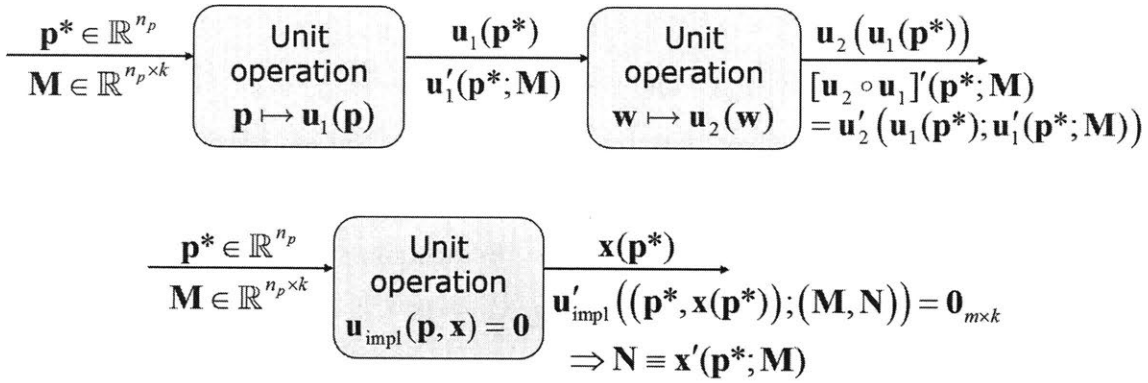


Figure 7-1: Framework for modular process calculations and sensitivity analysis with unit operations described by (possibly) nondifferentiable models. Top: explicit models. Bottom: implicit models.

7.2.3 Sensitivity analysis for nonsmooth flash calculations

The problem of obtaining exact sensitivities from the results of the nonsmooth inside-out algorithms from Chapter 5 is now considered. In a typical steady-state flash operation, there is a feed stream with molar flowrate F with n_c components at molar composition $\mathbf{z}_F$ that separates into a liquid stream with molar flowrate L at molar

composition $\mathbf{x}_L$ and a vapor stream with molar flowrate V at molar composition $\mathbf{y}_V$. The distribution of each component i between the vapor and liquid phases can be quantified by its equilibrium ratio k_i. The fraction of the feed that is vaporized in the flash operation is denoted $\alpha \equiv \frac{V}{F}$. When the temperature of the flash is unknown, an energy balance must also be included in the model involving h_V, h_L and h_F as the molar enthalpies of the vapor, liquid and feed streams, respectively and Q_{flash} as an additional heat duty term. Assuming the feed conditions are known, for an n_c component system, there are $2n_c+3$ equations in the classical flash model, but $2n_c+5$ unknowns $(\mathbf{x}_L, \mathbf{y}_V, T, P, V, L, Q_{\text{flash}})$ in this model, so two quantities must be specified. The most common pairs of fixed parameters are PQ and PT; however, this work also makes use of the PV specification for bubble and dew point calculations and the set PS, pressure-entropy, for compressor calculations, in which case the temperature is implied through an equation involving entropies rather than enthalpies of the relevant process streams.

When the flash equations are solved with the nonsmooth inside-out algorithms, sensitivity analysis of the classical flash model yields correct derivatives only within the two-phase region. Therefore, it is necessary to know a set of nonsmooth equations that these algorithms satisfy explicitly and that will provide correct sensitivity information no matter the outlet phase behavior. There are several equivalent equation systems that may be used to generate the required sensitivity information. One such equation system that includes a nonphysical variable β to represent relaxing the equilibrium constraints is as follows (for a PQ-flash):

$$L - (1-\alpha)F = 0, \tag{7.1}$$

$$\alpha y_{V,i} + (1-\alpha)x_{L,i} - z_{F,i} = 0, \quad i = 1, \ldots, n_c, \tag{7.2}$$

$$y_{V,i} - \beta k_i x_{L,i} = 0, \quad i = 1, \ldots, n_c, \tag{7.3}$$

$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0, \tag{7.4}$$

$$\alpha h_V + (1-\alpha)h_L - h_F - Q_{\text{flash}}/F = 0, \tag{7.5}$$

$$\text{mid}\,(\alpha,\ \beta - 1,\ \alpha - 1) = 0. \tag{7.6}$$

Note that the residual function of this equation system is a PC^1 function, as expected, due to the presence of the mid function in Equation (7.6) that is nondifferentiable at the bubble and dew point conditions of the mixture under consideration. There are several ways in which the variable β can be eliminated to yield a system that is entirely in terms of physical quantities. Perhaps the most obvious reformulation makes use of the following identity from Part 1: $\beta \equiv \frac{\sum_{i=1}^{n_c} y_{V,i}}{\sum_{i=1}^{n_c} k_i x_{L,i}}$, in addition to the fact that the nonsmooth inside-out algorithms always return a solution satisfying $\sum_{i=1}^{n_c} y_{V,i} = \sum_{i=1}^{n_c} x_{L,i} = 1$ in order to eliminate Equation (7.4). However, the resulting equation system will be either poorly-conditioned or singular at most solution points. Instead, the formulation used in this work (and it is clear that there are other reasonable alternatives) is as follows:

$$L - (1-\alpha)F = 0, \tag{7.7}$$

$$\alpha y_{V,i} + (1-\alpha)x_{L,i} - z_{F,i} = 0, \quad i = 1, \ldots, n_c, \tag{7.8}$$

$$y_{V,i} \sum_{j=1}^{n_c} k_j x_{L,j} - k_i x_{L,i} \sum_{j=1}^{n_c} y_{V,j} = 0, \quad i = 1, \ldots, n_c - 1, \tag{7.9}$$

$$\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} x_{L,i} = 0, \tag{7.10}$$

$$\alpha h_V + (1-\alpha)h_L - h_F - Q_{\text{flash}}/F = 0, \tag{7.11}$$

$$\text{mid}\left(\alpha, \sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} k_i x_{L,i}, \alpha - 1\right) = 0. \tag{7.12}$$

The residual function is again PC^1 due to the presence of the mid function. Note that the scaled equilibrium constraints in Equation (7.9) are enforced for all but one component to yield the correct number of equations. In the two-phase region, the second argument of the mid function is active and enforces the final equilibrium relationship not covered by Equation (7.9). Note that it is not essential which one of the equilibrium constraints is eliminated, though the present authors have observed the best conditioning of the equation system occurs when the equation corresponding to the largest k_i value is chosen.

Note that the equivalence of these formulations indicates that $\beta \equiv \frac{\sum_{i=1}^{n_c} y_{V,i}}{\sum_{i=1}^{n_c} k_i x_{L,i}}$. In

the two-phase region, the solution of either of these equation systems coincides with the solution of the classical flash equations (since $\sum_{i=1}^{n_c} k_i x_{L,i} = 1$, $\sum_{i=1}^{n_c} x_{L,i} = 1$ and $\sum_{i=1}^{n_c} y_{V,i} = 1$ in this regime), and is also identical to that obtained from solving the nonsmooth system of Equations (4.8)-(4.10), (4.12) and (4.26). Conversely, the nonsmooth inside-out algorithms will return different values for the mole fractions in nonexistent phases and the equilibrium ratios outside of the two-phase region as compared to those at the solution of Equations (4.8)-(4.10), (4.12) and (4.26); however, this is a purely numerical difference since in either case the solution values correspond to the same physical realization of the system. Additionally, both solutions show continuous dependence on the flash parameters, and therefore both are acceptable for simulation purposes.

In the notation of the previous sections, denote the flash model variables by $\mathbf{x} \equiv (T, \alpha, L, \mathbf{x}_L, \mathbf{y}_V)$ and the flash model parameters by $\mathbf{p} \equiv (P, Q_{\text{flash}}, \mathbf{f}_F, h_F)$, where $\mathbf{f}_F$ is the vector of component flowrates in the feed stream, i.e. the vector with components $f_{F,i} = z_{F,i}F$. In the case of the flash equations, even though there are three branches of the mid function, at any point $|I_{\mathbf{g}}^{\text{ess}}(\mathbf{x}^*, \mathbf{x}(\mathbf{p}^*))| \leq 2$. A check of the value of α can immediately eliminate either one or two of the branches, and so it is highly recommended that Algorithm 2.4 be used for the calculation of LD-derivatives rather than Algorithm 2.3 for computational efficiency. The selection functions for the application of the nonsmooth implicit function theorem are given by taking a single branch of the mid function at a time, i.e. the two-phase selection function is given by Equations (7.7)-(7.11) and $\sum_{i=1}^{n_c} y_{V,i} - \sum_{i=1}^{n_c} k_i x_{L,i} = 0$. Likewise, the all-liquid selection function is given by Equations (7.7)- (7.11) and $\alpha = 0$ and the all-vapor selection function is given by Equations (7.7)-(7.11) and $\alpha - 1 = 0$.

The formulations for the other flash types are analogous. For the PT case, Equation (7.11) is eliminated from the formulation with $\mathbf{x} \equiv (\alpha, L, \mathbf{x}_L, \mathbf{y}_V)$ and $\mathbf{p} \equiv (P, T, \mathbf{f}_F)$. In the PV case, the classical two-phase model may be used directly in conjunction with the sensitivity analysis from the classical implicit function theorem with $\mathbf{x} \equiv (T, L, \mathbf{x}_L, \mathbf{y}_V)$ and $\mathbf{p} \equiv (P, V, \mathbf{f}_F)$. For the PS specification, Equation (7.11) is replaced by $\alpha s_V + (1-\alpha)s_L - s_F = \Delta S_{\text{flash}}/F$, and the model variables and param-

eters become $\mathbf{x} \equiv (T, \alpha, L, \mathbf{x}_L, \mathbf{y}_V)$ and $\mathbf{p} \equiv (P, \Delta S_{\text{flash}}, \mathbf{f}_F, s_F)$, where s_V, s_L and s_F are the molar entropies of the vapor, liquid and feed streams, respectively and ΔS_{flash} is the total entropy increase of the flash operation.

In the course of applying Algorithm 2.4 to the nonsmooth flash equations, the authors have observed that due to the lack of error control associated with using direct methods (i.e. Gauss elimination) for solving Equation (2.31) in Line 6, the computed matrix $\mathbf{N}$ can erroneously fail to satisfy the condition $\mathbf{g}'(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*); (\mathbf{M}, \mathbf{N})) = \mathbf{0}_{m \times k}$ to within tolerance in Line 7. These failures are due to the matrix $\frac{\partial \mathbf{g}_{(i)}}{\partial \mathbf{x}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ having a high condition number, usually when the values of k_i for the various components range over many orders of magnitude. One potential solution is to use iterative linear solution methods; however, due to the frequency with which Equation (2.31) must be solved in a complex simulation problem, this is not the most computationally efficient method. Instead, the approach that has been used successfully in the examples of this chapter is the method of iterative refinement.[145] Following the solution of Equation (2.31) in Line 6 of Algorithm 2.4, Algorithm 7.1 is employed to improve the accuracy of the calculated sensitivity matrix. Note that in this procedure, $\varepsilon_{\text{tol}}^{\text{sens}}$ is the same as in Algorithm 2.4. Once this algorithm terminates, the updated matrix $\mathbf{N}$ is then used

Algorithm 7.1: Iterative refinement

1 $k \leftarrow 1$, $\mathbf{R} \leftarrow [\varepsilon_{\text{tol}}^{\text{sens}}]$.
2 **while** $\left(||\mathbf{R}||_1 > \frac{1}{2}\varepsilon_{\text{tol}}^{\text{sens}} \text{ and } k < k_{\max}\right)$ **do**
3 $\quad \mathbf{R} \leftarrow -\frac{\partial \mathbf{g}_{(i)}}{\partial \mathbf{p}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\mathbf{M} - \frac{\partial \mathbf{g}_{(i)}}{\partial \mathbf{x}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\mathbf{N}$.
4 $\quad$ Solve $\frac{\partial \mathbf{g}_{(i)}}{\partial \mathbf{x}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))\mathbf{D} = \mathbf{R}$ for $\mathbf{D}$.
5 $\quad \mathbf{N} \leftarrow \mathbf{N} + \mathbf{D}$.
6 $\quad k \leftarrow k + 1$.
7 **end while**
8 Return $\mathbf{N}$.

as the sensitivity matrix in the remainder of Algorithm 2.4. Note that this procedure is extremely inexpensive to perform because the LU factors of $\frac{\partial \mathbf{g}_{(i)}}{\partial \mathbf{x}}(\mathbf{p}^*, \mathbf{x}(\mathbf{p}^*))$ are obtained in the original solution step and may be reused in the iterative refinement procedure. In the examples in this chapter, sufficient improvement in the accuracy of the sensitivity matrix calculation was observed in fewer than five iterations of the

procedure in all cases.

At the critical point and in the supercritical regime of the phase space, the trivial solution to the flash equations ($\mathbf{z}_F = \mathbf{x}_L = \mathbf{y}_V$) is physically correct and the Jacobian of the model with respect to the unknown variables is guaranteed to be singular because the value of α is nonunique. When the nonsmooth density extrapolation algorithms from Chapter 6 are used in the flash calculations, the trivial solution will be found correctly in supercritical regimes. When the trivial solution is found, the sensitivity analysis will fail unless the nonuniqueness is removed. The strategy for addressing this situation is as follows: an equation is added to the model to fix the value of α according to some heuristic, then the sensitivity analysis proceeds as before except that now the exact solution to the resulting overdetermined linear system in Equation (2.31) is found using the method of least squares. A reasonable and simple heuristic for the vapor fraction constraint is to enforce that $\alpha = 1$. In rare cases, this may lead to parametric discontinuities in the predicted vapor fraction; however, developing a heuristic that always ensures a continuous transition is not straightforward and is an area for future investigation.

7.3 Example problems

Two complex flowsheeting problems that rely on the nonsmooth inside-out algorithms for flash calculations are now presented to highlight the key advantages of the new flowsheeting strategy outlined in the previous sections. The examples are written in a combination of C++ and the Julia programming language (v0.6.0), and executed on an Intel Xeon E5-1650 v2 workstation using six cores at 3.50 GHz and 12 GB RAM running Ubuntu v14.04. For both examples, nonideal thermophysical properties and their sensitivities for vapor and liquid phases are furnished by the Peng-Robinson cubic equation of state, augmented with the nonsmooth density extrapolation algorithm described in Chapter 6. Pure component and binary interaction parameters for the simulations take the values found in the Aspen Plus v8.4 databanks.[5]

Example 7.1. Figure 7-2 shows the flowsheet of the well-studied Cavett problem.[22]

The process consists of 4 PT-flash units (Flash 1-4 in the flowsheet) as well as two adiabatic mixing operations that can be modeled as PQ-flash units. The feed mixture to the process is a 16-component mixture of nitrogen, carbon dioxide, hydrogen sulfide and C1-C11 hydrocarbons, as detailed in Table 7.1. At these conditions, the feed is a two-phase mixture with $\alpha = 0.536$, as calculated by a PT-flash (not shown explicitly in the flowsheet). The base case process data is given in Table 7.2. In the base case,

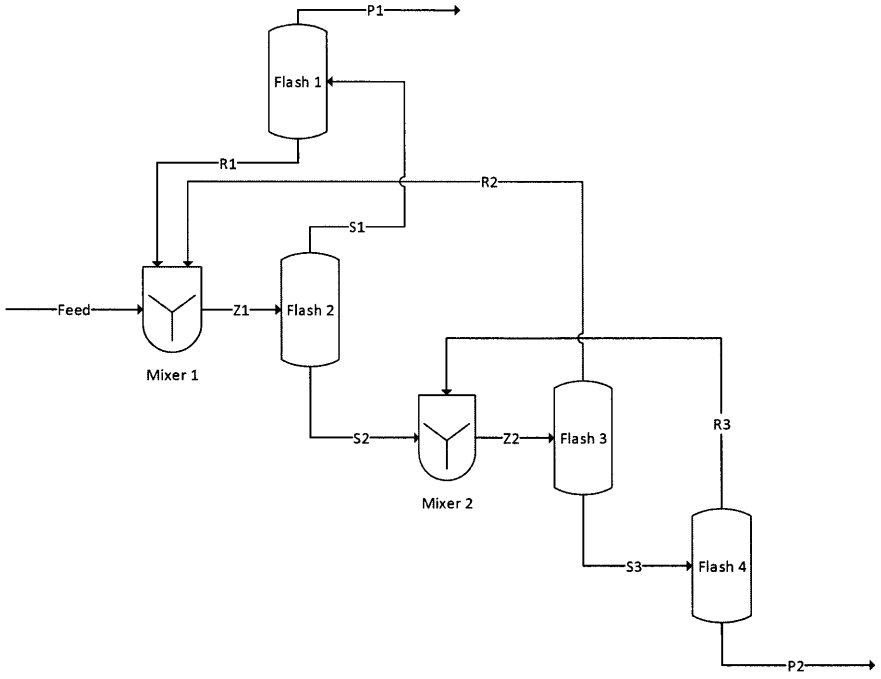


Figure 7-2: Flowsheet for Cavett's flowsheeting problem.[22]

Table 7.1: Feed stream data for Cavett's flowsheeting problem.

Component	N_2	CO_2	H_2S	Methane	Ethane
Mole Fraction	0.0131	0.1816	0.0124	0.1096	0.0876
Component	Propane	n-Butane	iso-Butane	n-Pentane	iso-Pentane
Mole Fraction	0.0838	0.0563	0.0221	0.0413	0.0289
Component	n-Hexane	n-Heptane	n-Octane	n-Nonane	n-Decane
Mole Fraction	0.0645	0.0953	0.0675	0.0610	0.0304
Component	n-Undecane		Flowrate (kmol/s)		
Mole Fraction	0.0444		3.445		

Table 7.2: Process data for the Cavett's flowsheeting problem.

	Temperature (K)	Pressure (Mpa)
Feed	322.04	0.4392
Flash 1	310.93	5.6171
Flash 2	322.04	1.9629
Flash 3	308.71	0.4392
Flash 4	302.59	0.1910

each flash unit operates in the two-phase regime. However, if the pressures and/or temperatures of the flash units change, this may no longer be the case. For the purposes of this example, design specifications on the product streams will be added to the flowsheet simulation that parametrically move some of the flash units out of vapor-liquid coexistence conditions.

The tear streams are chosen as streams Z1 and Z2 (see Figure 7-2) and the simulation problem therefore consists of $2(n_c + 2) = 36$ variables and equations. The $n_c + 2$ tear variables for each tear stream were chosen as the component flowrate of each of the n_c components, pressure and enthalpy, in order to match the default choices in Aspen Plus. The PT and PQ flash operations are all solved as implicit modules using the appropriate nonsmooth inside-out algorithms from Chapter 5 to an outer loop tolerance of 10^{-8} and inner loop tolerance of 10^{-9} using Anderson acceleration. The sensitivity analysis for these blocks is performed using Algorithms 2.4 and 7.1 with $\varepsilon_{\text{tol}}^{\text{sens}} := 10^{-8}$. Accordingly, these unit operations do not contribute any additional variables or equations to the flowsheet simulation, so the total size of the problem remains 36 equations and variables. The tear streams are converged using Algorithm 2.2, first using LD-derivatives calculated in the $\mathbf{I}_{36\times 36}$ directions (to yield B-subdifferential elements of the residual functions), and then again using forward finite differences to estimate generalized derivative elements (Equation 2.19) instead of the exact sensitivity calculations. Convergence is measured by the infinity norm of the tear stream residual functions falling below 10^{-8}.

First, the base case process is simulated to verify that the nonsmooth machinery still performs as intended and provides the same derivative information as expected

from classical sensitivity analysis. The process is also simulated in Aspen Plus using the built-in Newton convergence algorithm with the same tolerances on the flash calculations and overall simulation as described above. In both cases, the initial guess is given that specifies that both tear streams Z1 and Z2 are identical to the feed stream. Note that even though the tear variables, tear equations and initial guesses are the same, the error reported after the first flowsheet evaluation differs slightly between the Aspen Plus simulation and the others, which appears to be due to internal scaling of the variables and residuals within the software. Nevertheless, the three simulations arrive at the same result to within the specified tolerance. Figure 7-3 (left) shows the convergence rate of the three simulation methods. Even in the smooth case, the use of the nonsmooth strategy with exact sensitivity analysis results in more rapid convergence than when using either finite differences or the Newton method embedded in Aspen Plus that also relies on perturbation of the flowsheet to provide derivative information.

A design specification is now added to the simulation to create a more complicated problem that benefits more from the use of a (generalized) derivative-based method. The specification is that the product P1 must leave Flash 1 with 10 degrees of superheat (10 K above the mixture dew point) by allowing the temperature of the vessel to vary. This will result in a simulation in which the outlet from this flash operation is only vapor and the recycle stream R1 has zero flow at the solution, and adds one additional equation and variable to the flowsheeting problem, for a new total of 37. This case is simulated by the three methods as before (all with the same initial guess and tolerances) and the convergence rates to the solution are shown in Figure 7-3 (right). A solution is found with the temperature of the Flash 1 increased to 341.7 K. The nonsmooth inside-out algorithm for the PT-flash automatically calculates the single-phase result and exact sensitivity information is furnished by Algorithm 2.4, leading to more rapid (locally quadratic) convergence than in the other simulations.

For a final example, the previous design specification is removed and a new design specification on the value of the mass density of the liquid leaving Flash 4 in product stream P2 is enforced by varying the temperature of this flash drum. For illustration,

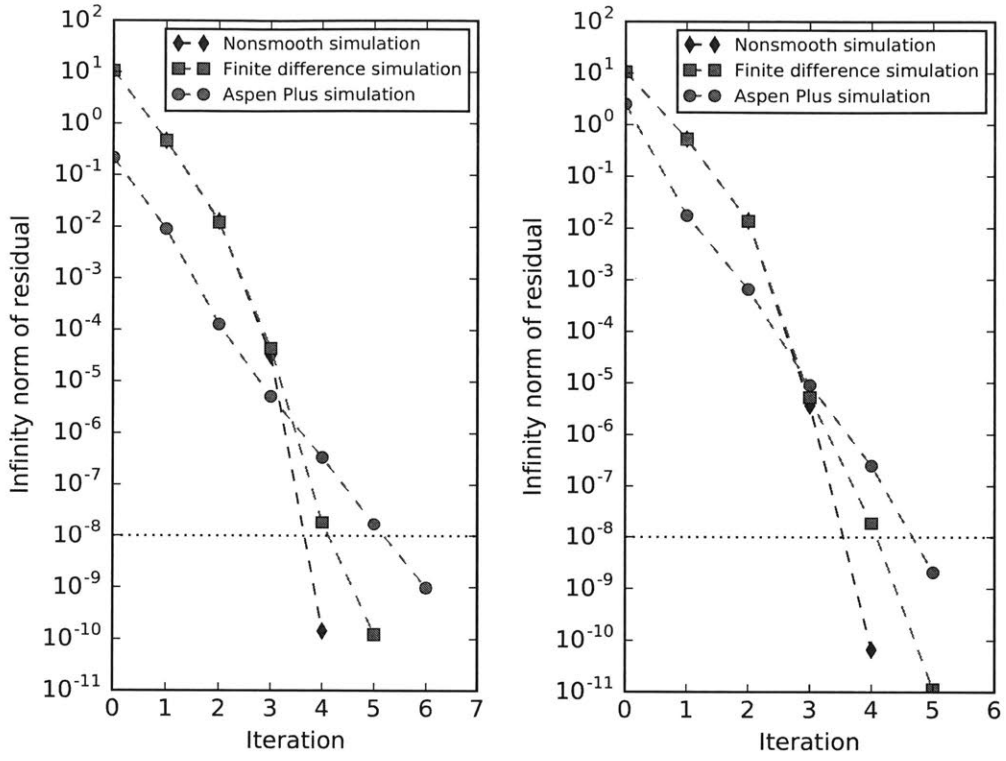


Figure 7-3: Convergence rate for simulations of the base case Cavett flowsheet simulation (left) and modified simulation with no liquid flow from Flash 1 (right).

the value of this specification is varied over a range that causes the flash unit to enter the single-phase liquid regime. The initial guess for the temperature of Flash 4 in each case was the base case value from Table S2. The results of this experiment are shown in Figure 7-4. In the top plot, it is clear that the flash unit is indeed driven into the liquid-only regime when high density is required, which drops the recycle flow of R3 to zero. The middle and bottom plots show the differences in the number of solver iterations and flowsheet evaluations needed to converge these problems between the three strategies. There are several notable observations. Firstly, the nonsmooth simulations require the least number of flowhseet evaluations (one per Newton iteration) because the sensitivity analysis is performed simultaneously with the residual evaluation by calculating LD-derivatives in the $\mathbf{I}_{37\times37}$ directions via AD and operator overloading. The simple forward finite-differencing scheme requires the most flow-

sheet evaluations because each solver iteration requires 38 flowsheet evaluations (one to obtain the residual value and 37 to obtain the approximate Jacobian). Aspen Plus, however, appears to perform a limited number of flowsheet perturbations per solver iteration to estimate partial Jacobians, likely due to built-in heuristics, and generally requires more iterations but less flowsheet evaluations overall than naïve finite differencing. In Aspen Plus, the number of flowsheet evaluations needed also tends to increase as the design specification requires Flash 4 to enter a single-phase operating regime, before reaching a point where the simulation can no longer be solved from the provided initial guess in more than 8600 flowsheet evaluations (as reported by the software, corresponding to 500 Newton iterations). Note that providing an improved initial guess that indicates a single-phase solution from Flash 4 will allow the Aspen Plus simulations to converge. The use of Aspen's built-in implementation of Broyden's method in place of Newton's method also does not provide any benefit, as it is also unable to converge any of the simulations with a single-phase solution from the original initial guess.

However, it is not possible to conclude from this alone that the nonsmooth approach is actually more efficient, as the flowsheet evaluations are necessarily more expensive when LD-derivatives are employed even though there are fewer of them required. As Aspen Plus has a highly developed and optimized codebase that the author's implementation cannot rival for a problem of this size (in addition to not reporting computation times), the costs of both single flowsheet evaluations and entire simulations were instead compared between naïve implementations of both finite differencing and LD-derivative evaluation through AD via operator overloading. The results are given in Table 7.3 averaged over the 100 simulations shown in Figure 7-4. Note that all simulations were successfully converged starting from the same initial guess when using the nonsmooth inside-out algorithms for the flash calculations, both when using exact LD-derivatives and finite difference approximations.

These results support the earlier claim that the complexity bound on the LD-derivative evaluation can indeed be quite weak. The cost of the flowsheet evaluation with exact sensitivity analysis is only around three times higher than a standard

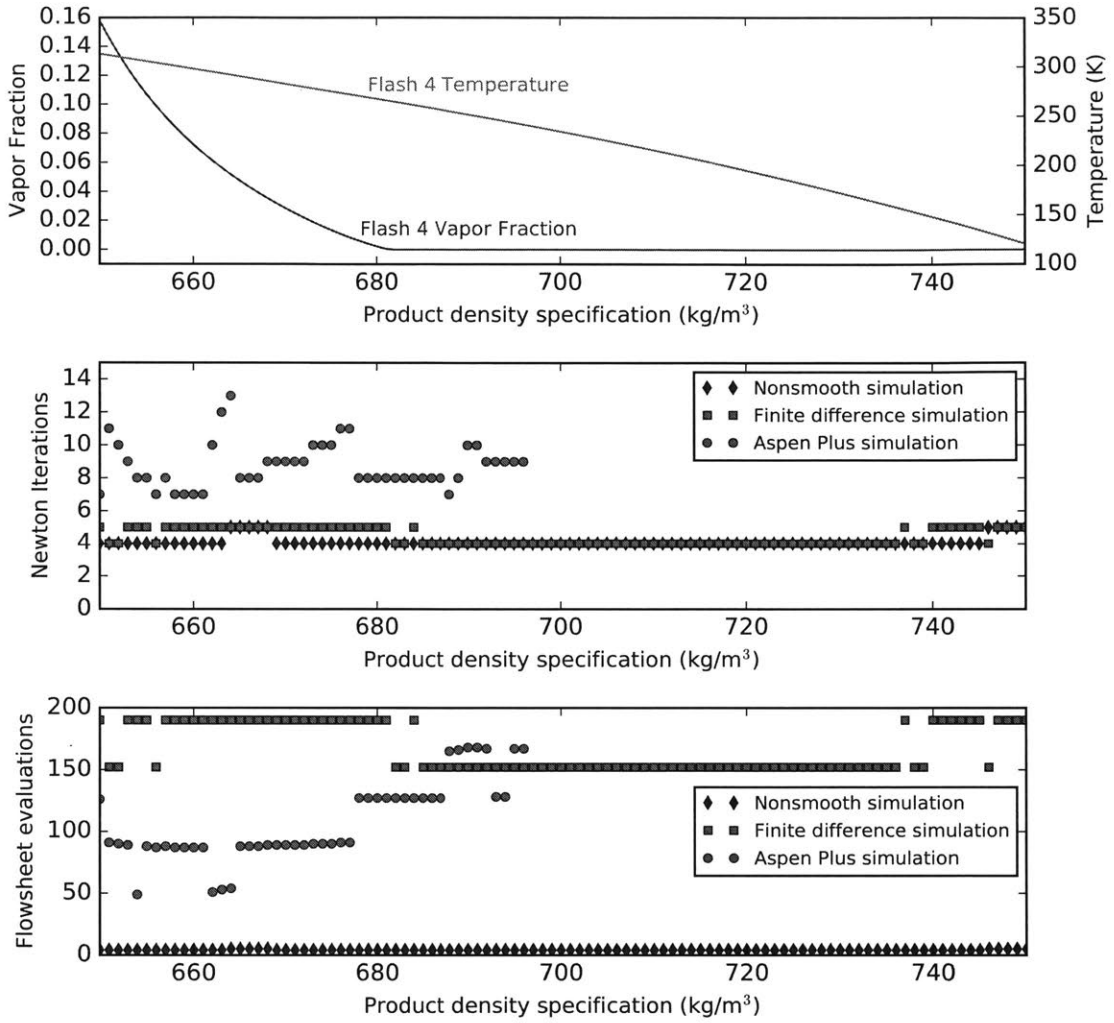


Figure 7-4: Results (top) and comparison between required number of solver iterations (middle) and number of flowsheet evaluations (bottom) for different simulation methods in the density design specification problem, as a function of the specified density value. Note that the Aspen Plus calculations fail to converge for density specifications above 696 kg/m^3 from the provided initial guess.

flowsheet evaluation on average in this case. As the basic finite difference approach requires 38 flowsheet evaluations on each iteration, this approach ends up being more costly by a factor of 13.1 to perform all the necessary flowsheet evaluations on each iteration, and on average 12.8 times slower per full iteration (including the Newton step calculation and other setup and allocations). Added to the fact that the inexact derivatives often require more iterations to converge, the finite-differencing approach

Table 7.3: Comparison between naïve implementations of LD-derivatives and finite differences for sensitivity analysis in the density design specification problems. Data reported are overall averages.

	LD-derivatives	Finite differences
Number of iterations	4.10	4.41
Number of flowsheet evaluations	4.10	167.43
Time per flowsheet evaluation (s)	0.222	0.077
Total solution time (s)	1.068	14.689

ends up being around 13.8 times more expensive on average than the exact sensitivity approach using LD-derivatives and the methods of the previous sections.

The cases presented in this example are illustrative of the significant advantages that the nonsmooth toolkit can offer process simulation. For more complex flowsheets, the high number of flowsheet evaluations required to furnish sensitivity information using a finite differencing approach will become prohibitively expensive, in addition to being less robust than the exact method using LD-derivatives.

The cases presented in this example are illustrative of the significant advantages that the nonsmooth toolkit can offer process simulation. For more complex flowsheets, the high number of flowsheet evaluations required to furnish sensitivity information using a finite differencing approach will become prohibitively expensive, in addition to being less robust than the exact method using LD-derivatives.

Example 7.2. Figure 7-5 shows the PRICO process configuration that will be studied in the simulation studies in this chapter and the optimization studies in Chapter 8. The MHEX unit operation is described by the PC^1 model given in Chapter 3 with the additional considerations for streams that change phase given in Chapter 4. As in the simulations from Chapter 4, the process streams are subdivided into the three phase regimes mentioned previously (wherein some streams may never enter a given phase regime, but this is handled automatically and does not need to be specified). The total heat load of each of these substreams is then further divided into affine segments each representing an equal portion of the total enthalpy change in the corresponding phase. This results in the temperatures at the end points of these

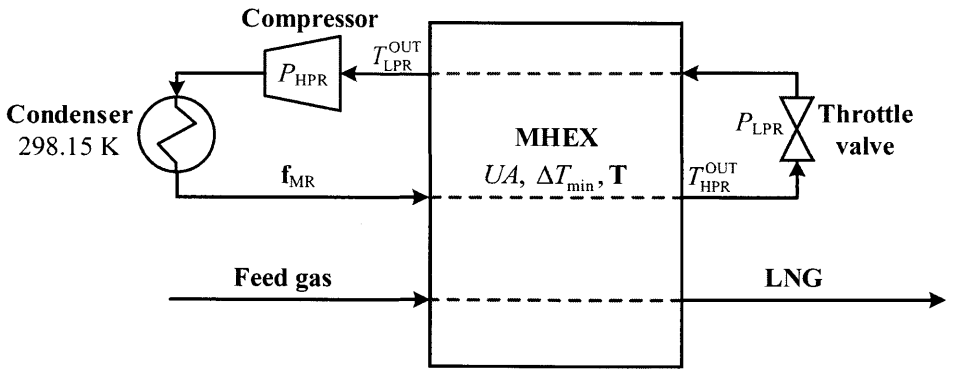


Figure 7-5: Flowsheet of the PRICO liquefaction process for natural gas.

segments being implicitly defined by energy balances and flash calculations.In this example, unless noted otherwise, the subcooled liquid and superheated vapor regions are discretized into five affine segments each, and the much more nonlinear two-phase region is discretized into twenty affine segments. The rationale behind this choice will be explored further later.

The PRICO process simulations in Example 4.3 were limited to using ideal thermodynamic models; however, with the advent of the nonsmooth inside-out algorithms, this restriction is lifted. These algorithms are both necessary and well-suited for this flowsheeting problem when nonideal thermodyamics are involved, as the phase regime of many of the streams at the solution will change based on the choices and values of the simulation parameters. Compounding this challenge is the fact that if the pressure and composition of the streams are variables in the problem (as they are in these examples), bubble and dew points will shift from iteration to iteration, adding additional uncertainty to the phase characterization of each stream on each flowsheet evaluation. The unit operations other than the MHEX are also almost entirely dependent on the nonsmooth inside-out algorithms. In each case, the outlet phase behavior is not fixed *a priori* and is instead determined automatically by the appropriate nonsmooth flash algorithm: The seawater cooler is modeled as a PT-flash operation. The outlet state of the throttle valve is determined with a PQ-flash calculation with $Q_{\text{flash}} = 0$. The compressor is assumed to be isentropic and so the outlet

state is modeled with a PS-flash calculation.

Since the MHEX model is by far the most difficult unit operation to converge in the flowsheet, it is not advisable to perform this simulation in an entirely modular fashion. Therefore, the three equations describing the MHEX model, as well as the energy balances linking the subcooled and superheated segments of each stream are handled simultaneously by an equation-solving algorithm: This means the MHEX model is represented by the solution of 27 equations: three from the base MHEX model, four per physical stream (of which there are three) for the energy balances needed to discretize the superheated vapor regime into five affine segments and four per physical stream for the energy balances needed to discretize the subcooled liquid regime into five affine segments.The two-phase regime of each physical stream is modeled as a chain of PQ-flash calculations with each handling an equal portion of the total enthalpy change in this regime. The solution and sensitivity analysis of these flash calculations are performed modularly so that they do not contribute any equations to the overall flowsheeting problem, unlike in Example 4.3 where such calculations were visible to the top-level solver.

Owing to the unique structure of the flowsheet, the MR loop must be torn in two locations, e.g. after the seawater cooler and after the throttle valve. However, since the material flow in the refrigerant loop is unchanging, there is no need to reconcile component flows in the tear streams. The remaining tear equations are generally trivial and in most cases can be enforced automatically by direct assignment in the code, i.e. the pressure and temperature of the high pressure refrigerant stream entering the MHEX are set as the temperature and pressure at which the seawater cooler operates. As a result, for many useful problem specifications, the flowsheet follows a unidirectional calculation sequence. As an example of an exception to this, if $T_{\mathrm{HPR}}^{\mathrm{OUT}}$ is a variable in the simulation, then an explicit tear equation is needed to reconcile the temperatures on the cold side of the MHEX. A modeler may exploit this knowledge to reduce the size of the overall problem; however, even if the problem is approached naïvely as in a standard SM approach, flowsheet simulations may still be converged with little additional effort due to the simplicity of satisfying the tear

Table 7.4: Natural gas stream data for Example 7.2.

Property	Natural gas
Flowrate (kmol/s)	1.00
Pressure (MPa)	5.500
Inlet temperature (K)	295.15
Outlet temperature (K)	110.15
Composition (mol %)	
Nitrogen	1.00
Methane	91.60
Ethane	4.93
Propane	1.71
n-Butane	0.35
iso-Butane	0.40
iso-Pentane	0.01

equations, as shown in the examples. All flash calculations within the MHEX, in addition to the other unit operation models in the flowsheet are treated as implicit modules that are solved by an appropriate nonsmooth inside-out algorithm and have their sensitivities evaluated as described in the previous section with the sensitivity tolerance in Algorithms 2.4 and 7.1 set to 10^{-8}. In this way, the flash calculations are hidden entirely from the flowsheet simulation algorithm and converged robustly with tailored algorithms. The flash calculations themselves are converged to a tolerance of 10^{-8} throughout the flowsheet in the sense of the outer loop tolerance (with inner loop tolerance of 10^{-9}).

Table 7.4 gives the data for a representative natural gas stream in such a process, which is assumed to be fixed throughout this example.

As indicated in Figure 7-5, the following notation is used for the variables related to the MR stream:

- P_{LPR}: pressure level of the low pressure refrigerant,
- P_{HPR}: pressure level of the high pressure refrigerant,
- $T_{\text{LPR}}^{\text{OUT}}$: outlet temperature of the low pressure refrigerant,
- $T_{\text{HPR}}^{\text{OUT}}$: outlet temperature of the high pressure refrigerant,

- $\mathbf{f}_{\mathrm{MR}}$: vector of individual molar flowrates of each component in the refrigerant. The flowrate of each component i is denoted $f_{\mathrm{MR},i}$ and is the product of the MR stream flowrate (F_{MR}) and the mole fraction of component i in the MR stream ($z_{\mathrm{MR},i}$),

- $\mathbf{T}$: vector with components corresponding to the temperatures needed to satisfy the energy balances in the discretized superheated and subcooled regimes for each process stream (note that these will always be unknowns in the model in the present solution strategy).

The three sets of unknown MHEX variables considered for the purposes of this example are:

- **Set I** : P_{LPR}, $T_{\mathrm{LPR}}^{\mathrm{OUT}}$, ΔT_{min};

- **Set II** : $f_{\mathrm{MR,nC4}}$, $T_{\mathrm{LPR}}^{\mathrm{OUT}}$, ΔT_{min}; and

- **Set III** : P_{LPR}, $T_{\mathrm{LPR}}^{\mathrm{OUT}}$, P_{HPR}.

Note that in Set II, variation in the individual component flowrate of n-butane will affect both the total flowrate and the overall composition of the MR stream as $F_{\mathrm{MR}} = \sum_{i=1}^{n_c} f_{\mathrm{MR},i}$ and $z_{\mathrm{MR},i} = f_{\mathrm{MR},i}/F_{\mathrm{MR}}$. Table 7.5 gives the values for the initial values for the unknown MHEX variables and the values of the known parameters for each of these simulations. The numerical values for the nominal base case are taken from an example in Kamath *et al.*[59]

The problem is initialized only with the data given in Table 7.5. The initial guess values for the remaining variables in the problem (the set of temperatures implicitly defined by energy balances in the MHEX, $\mathbf{T}$) are generated automatically by assuming a linear relationship between temperature and enthalpy exists in the superheated and subcooled regions. This means that the user is only responsible for providing initial guesses for three variables and the calculations are highly insensitive to the values chosen. Initial guesses for the flash calculations performed in the first iteration are calculated as described in Chapter 5. For subsequent iterations, convergence is

Table 7.5: Refrigerant stream and MHEX data for Example 7.2. Values in square brackets are initial guesses rather than fixed parameters.

Property	**Set I**	**Set II**	**Set III**
Flowrate (kmol/s)	2.928	[2.928]	2.928
P_{HPR} (MPa)	1.713	1.713	[1.713]
P_{LPR} (MPa)	[0.202]	0.202	[0.202]
$T_{\text{HPR}}^{\text{IN}}$ (K)	298.15	298.15	298.15
$T_{\text{HPR}}^{\text{OUT}}$ (K)	110.15	110.15	110.15
$T_{\text{LPR}}^{\text{OUT}}$ (K)	[298.15]	[298.15]	[298.15]
$\mathbf{z}_{\text{MR}}$ (mol %):			
Nitrogen	5.82	[5.82]	5.82
Methane	20.62	[20.62]	20.62
Ethane	39.37	[39.37]	39.37
n-Butane	34.19	[34.19]	34.19
UA (MW/K)	12	20	12
ΔT_{min} (K)	[1.2]	[1.2]	0.95

accelerated by exploiting the restart capabilities of the nonsmooth inside-out algorithms. The results from a given flash calculation on one call are fed forward to the same flash calculation on the next call as an initial guess. This leads to increasingly rapid and efficient solution of the flash subproblems as the overall problem converges.

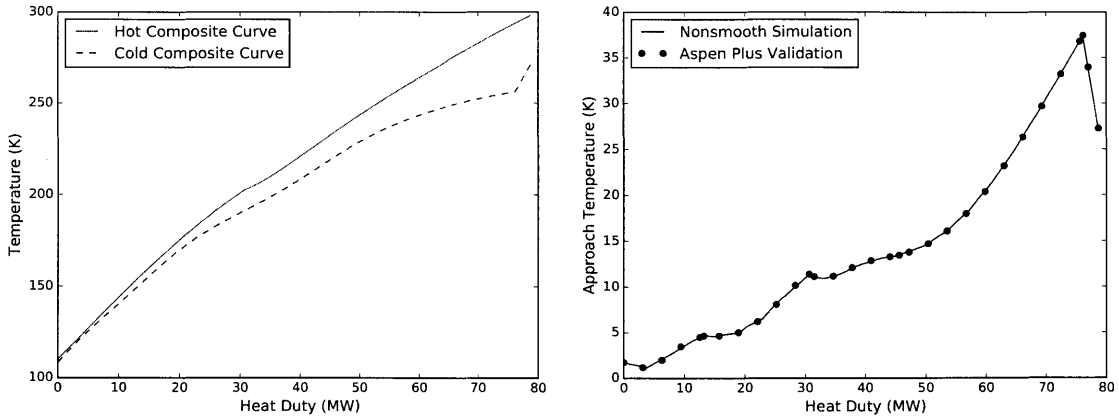
For each variable set, following the simulation with the nonsmooth strategy, a validation of the result was performed in Aspen Plus. It is important to note that this does not mean that Aspen Plus could have performed the simulation; on the contrary, Aspen Plus fails to solve any of the following cases when using a combination of design specifications and the MHeatX block (which will only solve the energy balance around the MHEX for a single piece of unknown information) to create an equivalent problem. Instead, the values of all pressures and compositions from the solution of the nonsmooth simulation are given to Aspen Plus with the exception of the outlet temperature of the low pressure refrigerant (which it will obtain through energy balance), and the resulting temperature profile in the MHEX is compared with the result of the nonsmooth simulation to assess the accuracy and physicality. Aspen Plus will also calculate and validate the values of ΔT_{min} and UA for a given set of composite curves, though it cannot accept this information as inputs to the block. The MHeatX block also cannot be discretized in exactly the same way as the

nonsmooth MHEX model, so instead the Aspen Plus model is divided into 25 zones with the option enabled for adding extra points for phase change, stream entry and stream exit to achieve a similar level of fidelity.

Some results and analysis for each of the variable sets is now given. In each case, the simulation was converged to a tolerance of 10^{-6} in terms of the infinity norm of the residual function using the hybrid semismooth/LP-Newton solution method described in the earlier section.

Variable Set I: In this first case, the conductance value of the MHEX is fixed while the pressure and outlet temperature of the LPR stream and the minimum approach temperature vary. This can be viewed as the problem of finding new process conditions to make use of an existing heat exchanger of fixed size. A solution is found with $P_{\text{LPR}} = 0.151$ MPa, $T_{\text{LPR}}^{\text{OUT}} = 270.99$ K and $\Delta T_{\min} = 1.165$ K. The isentropic compression power required is 18.02 MW. Performing the simulation with the tear equations enforced implicitly as described previously takes 5.2 seconds including the automated initialization procedure. When the tear equations are instead added explicitly to the flowsheet model, the same solution is reached after 6.4 seconds, which shows that the inclusion of these trivial-to-solve equations does not have a significant impact on performance. Figure 7-6(a) shows the resulting hot and cold composite curves in the MHEX for this simulation and Figure 7-6(b) shows the profile of the temperature difference between the hot and cold composite curves from both the nonsmooth model and the Aspen Plus validation, showing excellent agreement. The Aspen Plus validation also gives that $\Delta T_{\min} = 1.174$ K, $UA = 11.96$ MW/K and the compression power requirement as 17.99 MW, all differences of less than 0.8% from the nonsmooth model results.

It is important to know whether or not the discretized composite curves are in fact accurate enough. To assess the true number of affine segments needed, a series of additional simulations were performed with P_{LPR}, $T_{\text{LPR}}^{\text{OUT}}$ and either UA or $\Delta T_{\min}$ as unknowns wherein the discretization of the highly non-linear two-phase region was varied from 3 to 50 affine segments. The results for the predicted compression power in each simulation are shown in Figure 7-7. This study suggests that with 20



(a) (b)

Figure 7-6: (a.) Hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process simulated in Case I. (b.) Approach temperature profile for the MHEX in the PRICO process simulated in Case I from the nonsmooth simulation (solid line) and the Aspen Plus validation (points).

segments, the power is likely to be accurate to within about 1% of the value given by 50 segments, which is assumed to be essentially the true value. Interestingly, it also shows that with fewer than ~15 segments, the predicted compression power can be highly incorrect, tending to be far too low. Note that the number of equations in the process model does not change with increasingly fine discretization of the two-phase region, both because the flash calculations are solved as implicit modules and because Equation (3.13) for pinch point location remains a single equation regardless of the number of streams integrated in the MHEX. Accordingly, despite the increased number of flash calculations, the solution time for the simulations in Figure 7-7 only increases by a factor of 7 (on average) between solving the problems with five segments (average: 1.96 seconds) compared to the problems with fifty segments (average: 13.72 seconds).

Variable Set II: In this case, the composition of the refrigerant mixture is allowed to vary and the conductance of the exchanger is fixed at 20.0 MW/K. A solution is found with $f_{\mathrm{MR,nC4}} = 0.846$ kmol/s, $T_{\mathrm{LPR}}^{\mathrm{OUT}} = 292.87$ K and $\Delta T_{\mathrm{min}} = 0.365$ K. The isentropic compression power required is 15.96 MW. This value of the component

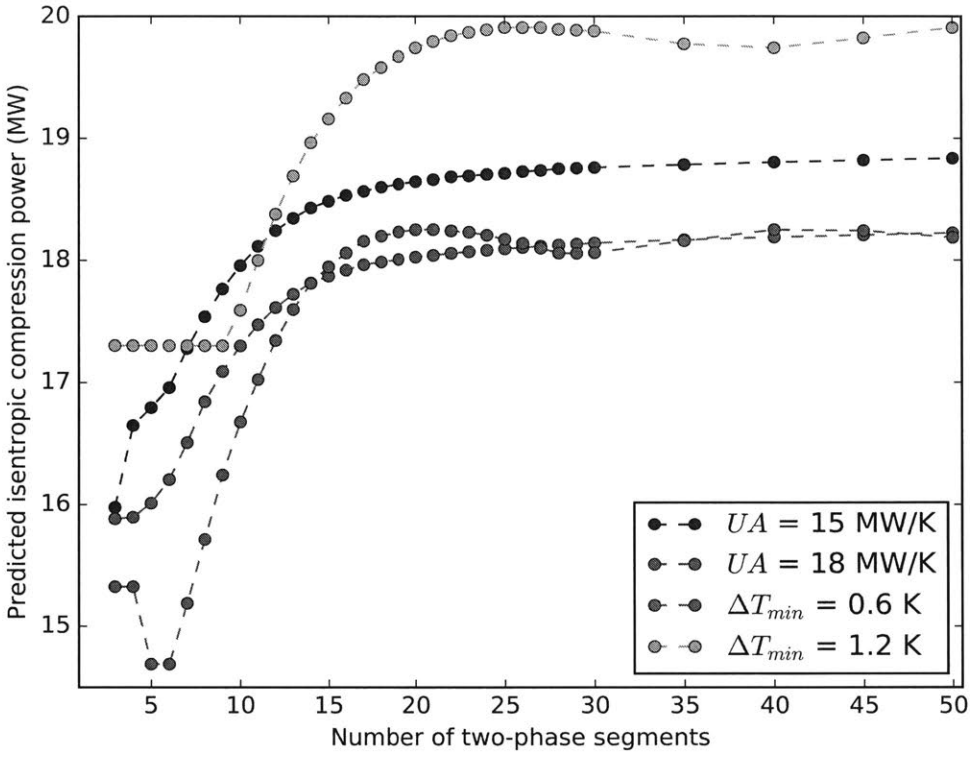
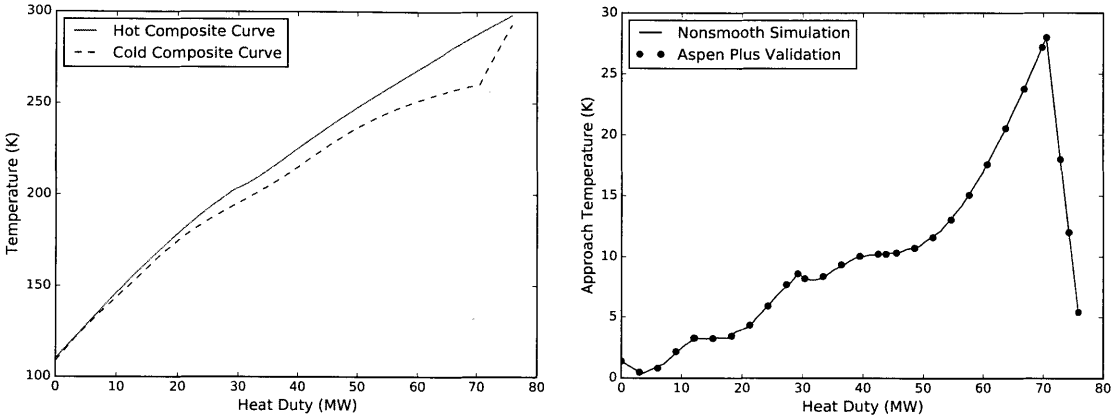


Figure 7-7: Value of the isentropic compression power obtained by simulating the PRICO process with increasing discretization of the two-phase region. Each set of data points corresponds to a specification for either UA or $\Delta T_{\min}$ in the simulation. In each case, if the conductance was specified, $\Delta T_{\min}$ was solved for, and vice-versa, with the other variables solved for as $T_{\text{LPR}}^{\text{OUT}}$ and P_{LPR}.

flowrate of n-butane changes the overall MR flowrate to 2.772 kmol/s with molar composition: 6.15% nitrogen, 21.77% methane, 41.57% ethane and 30.51% n-butane. When the simple tear equations are reconciled automatically as described earlier, the simulation takes 7.9 seconds to converge including initialization. With the tear equations modeled explicitly, the same solution is reached after 10.9 seconds (one additional iteration is needed). Figure 7-8 shows (a.) the resulting composite curves and (b.) the approach temperature profiles from both the model proposed in this chapter and the Aspen Plus simulation. The Aspen Plus validation gives that $\Delta T_{\min} =$ 0.481 K, $UA = 19.32$ MW/K and the compression power requirement as 15.95 MW, showing excellent agreement in the compressor but slightly different results in the

MHEX. However, increasing the number of zones in the Aspen block and the number of two-phase segments in the nonsmooth simulation by 10 causes the results to match to within 5% in ΔTmin value and <1% in conductance value, indicating that the difference in discretization schemes is mostly responsible for the discrepancy.

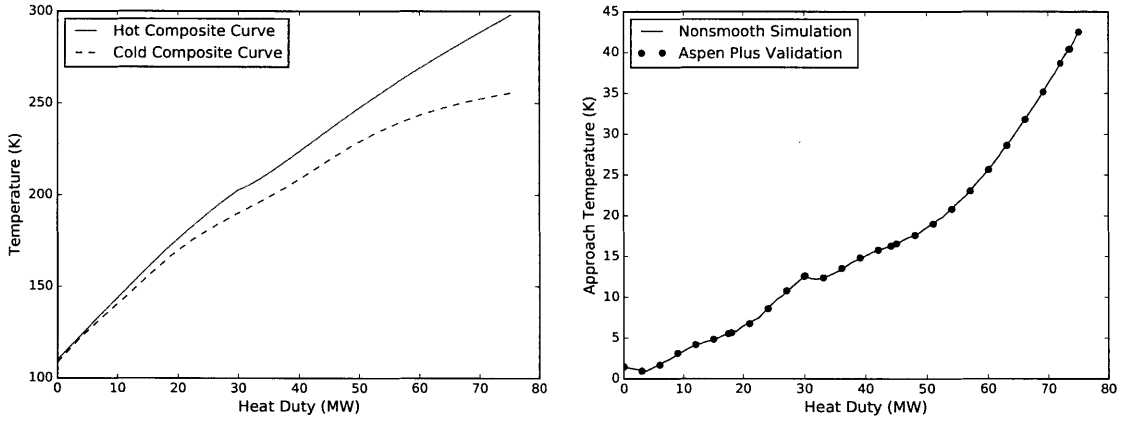


(a) (b)

Figure 7-8: (a.) Hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process simulated in Case II. (b.) Approach temperature profile for the MHEX in the PRICO process simulated in Case II from the nonsmooth simulation (solid line) and the Aspen Plus validation (points).

Variable Set III: In this case, both pressure levels are allowed to vary while the design criteria of the MHEX are fixed. A solution is found with $P_{\text{LPR}} = 0.150$ MPa, $T_{\text{LPR}}^{\text{OUT}} = 255.62$ K and $P_{\text{HPR}} = 2.251$ MPa. The isentropic compression power required is 18.79 MW. Note that at this solution, the LPR stream exits the MHEX in the two-phase region with a vapor fraction of 0.988, indicating that the iterates successfully traversed a phase boundary after starting from the initial guess of 298.15 K in the superheated vapor regime. The simulation takes 9.1 seconds (including initialization) when the simplicity of the tear equation is exploited as described earlier. With a naïve SM approach that models the tear equations explicitly, the same solution is reached after 10.7 seconds. Figure 7-9 shows (a.) the resulting composite curves and (b.) the approach temperature profiles from the model proposed in this chapter and the Aspen Plus simulation. The Aspen Plus validation also gives that $\Delta T_{\text{min}} = 0.955$

K, $UA = 11.95$ MW/K and the compression power requirement as 18.77 MW, again all only minor differences from the nonsmooth model results. It is also possible in this case to add an extra design specification on the degrees of superheat of the LPR stream leaving the MHEX in case the compressor is not well-equipped to handle any liquid droplets. This is accomplished by including $\Delta T_{\min}$ as an additional manipulated variable to meet the specification. For instance, specifying that this stream should exit at its dew point (0 degrees superheat) yields a solution with $\Delta T_{\min} = 1.00$ K, $P_{\text{LPR}} = 0.151$ MPa, $T_{\text{LPR}}^{\text{OUT}} = 256.41$ K and $P_{\text{HPR}} = 2.087$ MPa.



(a) (b)

Figure 7-9: (a.) Hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process simulated in Case III. (b.) Approach temperature profile for the MHEX in the PRICO process simulated in Case III from the nonsmooth simulation (solid line) and the Aspen Plus validation (points).

Figure 7-10 shows the convergence rate of the simulations described above using three different methods for calculating or approximating an element of the generalized derivative. The first is the method detailed in this chapter using LD-derivatives in the identity directions, which provides elements of the B-subdifferential and leads to the hybrid nonsmooth Newton method converging quadratically in the neighborhood of the solution. The second method is to calculate an approximate B-subdifferential element by concatenating directional derivatives in each of the coordinate directions (i.e. the right-hand side of Equation (2.18)). For Variable Sets I and III, this method

takes similar steps to the exact method, though the convergence is not as rapid as the solution is approached, indicating that the numerical method visits points of nonsmoothness. For Variable Set II, this method fails to converge altogether. The third method is forward finite differences, which fails to converge the simulations involving Variable Set II or III, and converges slowly for Variable Set I. It is clear from these results that exact generalized derivatives are in fact necessary to achieve robust and rapid convergence of complex simulations involving nonsmooth models.

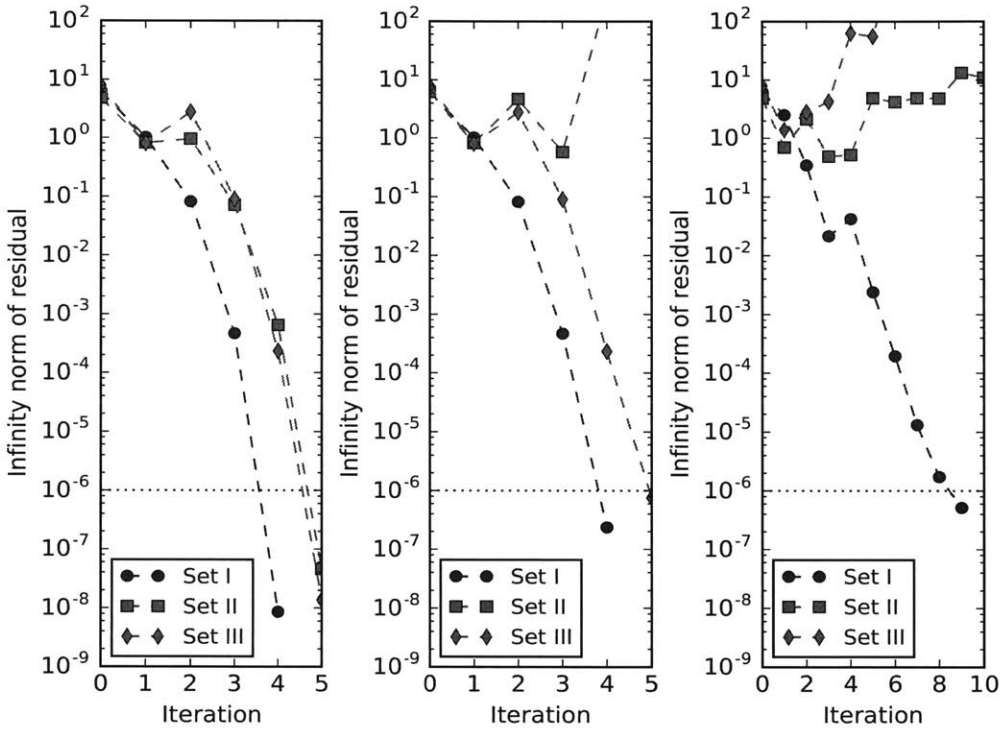


Figure 7-10: Convergence rate of the previous simulations when sensitivity information is computed using: LD-derivatives in the identity directions (left); concatenated directional derivatives in each coordinate direction (middle); forward finite differences (right).

7.4 Conclusions

A new paradigm for process flowsheet calculations using nonsmooth models and generalized derivatives has been presented. At the heart of this new approach are the

nonsmooth inside-out algorithms from Chapter 5, which allow for reliable flash calculations with complex thermodynamic models and, when augmented with accurate sensitivity analysis procedures, provide useful generalized derivatives that may be propagated to flowsheet convergence algorithms. It has been demonstrated that this nonsmooth strategy generates quadratically convergent iterates in process flowsheet calculations, outperforming other known methods for calculating sensitivity information (or an approximation thereof), including those embedded in the Aspen Plus software.

Chapter 8

An optimization strategy for liquefied natural gas production processes

A new strategy for the optimization of natural gas liquefaction processes is presented, in which flowsheets formulated using nondifferentiable process models are efficiently and robustly optimized using an interior-point algorithm. The constraints in the optimization formulation lead to solutions that ensure optimal usage of the area of multistream heat exchangers in the processes in order to minimize irreversibilities. The process optimization problems are solved reliably without the need for a complex initialization procedure even when highly accurate descriptions of the process stream cooling curves are requested. In addition to the well-studied PRICO liquefaction process, two significantly more complex single mixed-refrigerant processes are successfully optimized and results are reported for each process subject to constraints imposed by several different operating scenarios.

8.1 Introduction

Numerous authors have proposed methodologies for the problem of optimizing natural gas liquefaction processes. Optimization strategies for liquefaction processes

found in the literature typically fall into two broad categories. The first is the use of an optimization algorithm that calls process simulation software (such as Aspen HYSYS or Aspen Plus) for the flowsheet evaluation. A number of deterministic local optimization and heuristic global optimization codes have been used in such studies, including: SQP,[131;113;9] interior-point algorithms,[102] Box's complex method,[50;68] genetic algorithms (GA),[29;54] artificial neural networks,[67] simulated annealing,[11] particle swarm,[69] mesh search[73] and tabu search.[4] The other general approach is to use an equation-oriented strategy in which the full process model is described explicitly in a modeling language and then solved with an appropriate optimization algorithm. This is often performed in the GAMS modeling environment, paired with either a local solver[59] or a global solver on a simplified model or a superstructure,[49;141;66;102] though the use of other software such as gPROMS for this application is also reported in the literature.[56;57;95] Many additional examples of liquefaction process optimization studies are documented and classified for ease of reference in an extensive literature review by Austbø *et al.*[10]

The flowsheet models used for function evaluations in the process simulator case generally consist of traditional unit operation models solved in a sequential-modular fashion. Sensitivity information is obtained by finite differencing through perturbation of the flowsheet inputs. In the second case, the models and required solution algorithms take more exotic forms, including complementarity-constrained nonlinear programs,[59] mixed-integer nonlinear programs[49;141;102;66] and differential-algebraic equation systems.[95] Such methodologies typically result in models with very high variable and constraint counts that are challenging to initialize and solve robustly, even for simple liquefaction processes. The complexity of these formulations largely results from different methodologies for modeling the MHEXs in such processes. However, the nonsmooth approach of the previous chapters provides a more convenient approach to describing the MHEX process model. One significant advantage of using the nonsmooth model is the ability to specify fixed information about the MHEX area or conductance value. As will be seen in the following section, the ability to incorporate this information into the optimization formulation is vital for finding process

conditions that minimize irreversibilities and thereby the cost of operation. Some authors have also reported using more detailed models for MHEX units that consider the internal geometry and hydrodynamics[113] in order to move beyond the early-stage design based on UA values. However, this tends to increase the complexity of the optimization significantly even for small-scale processes and solutions have only been published for problems with a limited set of decision variables. Such detailed MHEX models are however beyond the scope of the present work.

Many of the different modeling and optimization strategies mentioned previously have been applied to the PRICO process and a wealth of different solutions for optimal design and optimal operation problems have been reported. Table 8.1 gives several of these proposed optimal operating conditions for minimizing the required compression power in the process.

In each case study summarized in Table 8.1, the natural gas stream was supplied at 5.5 MPa at 1.0 kmol/s and a minimum allowable temperature difference between the hot and cold streams in the MHEX of 1.2 K was enforced. The target temperature for the natural gas was set to 110.15 K in the articles by Lee *et al.*[72] and Del Nogal *et al.*,[29] whereas it was taken as 118.15 K in the other four articles. The Peng-Robinson EOS was used in the articles by Lee *et al.*[72] and Del Nogal *et al.*,[29] while the remaining articles use the Soave-Redlich-Kwong EOS. The articles by Jensen and Skogestad,[56] Pattison *et al.*[95] and Rao and Karimi[102] each use a feed gas composition of 2.8% nitrogen, 89.7% methane, 5.5% ethane, 1.8% propane and 0.1% n-butane, while the other papers do not report the composition of the feed gas. The power requirements reported are all based on an assumed isentropic efficiency of 80% for the compressor. The UA value of the MHEX corresponding to the optimal solution is not reported in any of these articles.

The purpose of showing Table 8.1 is not to claim that some authors have found comparatively better or worse solutions than others. For one, the variability in the process conditions mentioned previously precludes direct comparison between all these studies. Beyond these obvious distinctions, it is still largely meaningless to attempt to compare these solutions to one another, as seemingly benign details such as values

Table 8.1: Locally optimal solutions for operating the PRICO process found in the literature.

Authors	Lee *et al.*[72]	Del Nogal *et al.*[29]	Jensen & Skogestad[56]
Solution strategy	Mesh Search	GA	n/a
Flowsheeting/modeling environment	WORK[128]	STAR[128]	gPROMS
Power (MW)	26.60	24.53	17.40
High pressure (MPa)	4.00	4.387	1.912
Low pressure (MPa)	0.37	0.484	0.322
MR flowrate (kmol/s)	3.2	3.53	3.12
Composition (mol %)			
Nitrogen	11.0	10.08	7.72
Methane	27.3	27.12	23.65
Ethane	35.6	37.21	39.49
Propane	5.20	0.27	0.00
n-butane	20.9	25.31	29.14
Authors	Kamath *et al.*[59]	Pattison *et al.*[95]	Rao & Karimi[102]
Solution strategy	CONOPT[31]	DAE solver	KNITRO[19]
Flowsheeting/modeling environment	GAMS	gPROMS	Aspen HYSYS
Power (MW)	21.51	20.00	18.97
High pressure (MPa)	1.713	2.655	2.337
Low pressure (MPa)	0.202	0.338	0.301
MR flowrate (kmol/s)	2.93	2.89	3.00
Composition (mol %)			
Nitrogen	5.82	8.81	7.85
Methane	20.62	32.29	24.87
Ethane	39.37	32.79	37.99
Propane	0.00	0.63	0.01
n-butane	34.19	25.48	29.27

of physical property parameters are generally unreported yet can significantly impact feasibility and optimality. An example of this is shown in Vikse *et al.*,[132] in which it is noted that just the difference in the default ideal gas heat capacity calculation method between Aspen Plus and Aspen HYSYS produces quantitatively different results for liquefaction process simulations. These differences are further compounded

by the lack of attention given to MHEX UA values in the literature studies. The specification of $\Delta T_{\min}$ does not uniquely describe a process design, especially in the absence of UA information, and so it is unsurprising that many different optimal values for the compression work could be found, i.e. smaller values corresponding to larger heat exchangers and vice-versa. The guiding principle must therefore be that an optimal set of decision variable values for a liquefaction process should only be discussed in reference to a specific instance of the process, e.g. with respect to feed gas composition, pressure and target temperature, heat exchanger area, heat-sink temperatures, etc., and with thermophysical property methods and parameters tuned to describe the working fluids as accurately as possible, ideally matching with experimental or plant data if available. This level of specificity means that no single optimization study of a liquefaction process can truly claim to find a universal best solution for all realizations of the process. Accordingly, this goal of this chapter is to develop an implementation of a flexible optimization method in which complex liquefaction process models can be efficiently, accurately and robustly optimized, then easily modified and re-optimized given either changes in process conditions or the changing demands of the user.

8.2 Optimization Strategy

This section first describes the process optimization formulation for the natural gas liquefaction processes studied in this chapter, then details the configuration of the optimization algorithm used to solve the example problems of the following section.

8.2.1 Problem formulation

A recent article by Austbø and Gundersen[9] compared several different problem formulations for liquefaction process optimization. Through studies of the PRICO process, they determined that the optimal utilization of the area of the multistream heat exchanger in the process can only be obtained when the heat exchanger conductance is constrained to be less than some preset value $UA_{\max}$. Unlike in much of the litera-

ture, the minimum temperature difference, $\Delta T_{\min}$ (also called the minimum approach temperature), is not constrained to being greater than a preset value. Constraining the heat exchanger conductance is tantamount to making the optimizer find the optimal distribution of temperature differences throughout the entire exchanger instead of just prescribing the pinch point. This is especially important in liquefaction processes, where reducing temperature differences at subambient temperatures is key to minimizing exergy losses, as thermodynamic irreversibilities increase with both increasing temperature driving force and decreasing operating temperature.[7;9] The $UA_{\max}$ constraint also forces the optimizer to consider the trade-off between the total heat duty of the MHEX and the temperature driving forces, generally leading to lower refrigerant flow rates and thereby lower power requirements than when constraints are placed on the driving forces alone (e.g. by specifying $\Delta T_{\min}$).[9] Earlier work by Jensen and Skogestad[57] also asserts that the $\Delta T_{\min}$ approach leads to suboptimal utilization of multistream heat exchangers. In terms of objective function, minimizing the required compressor power ($\dot{W}_{\text{comp}}$) in the process is common and recommended as it is equivalent to minimization of the irreversibilities in the process. The ideal optimization formulation for liquefaction processes suggested by the literature is therefore as follows:

$$\begin{aligned} \min_{\mathbf{x}} \ & \dot{W}_{\text{comp}}(\mathbf{x}) \\ \text{s.t.} \ & \mathbf{h}(\mathbf{x}) = \mathbf{0}, \\ & UA(\mathbf{x}) \leq UA_{\max}, \\ & \Delta T_{\text{sup}}(\mathbf{x}) \geq \Delta T_{\text{sup,min}}, \\ & \mathbf{x}^{\text{LB}} \leq \mathbf{x} \leq \mathbf{x}^{\text{UB}}, \end{aligned} \tag{8.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the vector of decision variables with lower bounds $\mathbf{x}^{\text{LB}}$ and upper bounds $\mathbf{x}^{\text{UB}}$, $\mathbf{h}$ is function describing the process model and ΔT_{sup} is the degree of super-heating in the stream entering the compressor (i.e. the difference between this stream's temperature and dew point), which is constrained to be greater than some minimum value, $\Delta T_{\text{sup,min}}$. The value of $\Delta T_{\text{sup,min}}$ is 10 K in all examples

of this chapter. This constraint on the degree of superheating is a practical safety consideration to prevent the formation of liquid droplets in the compressor.

In order to provide exact sensitivity information to the optimizer instead of potentially highly inaccurate finite difference approximations, the optimization problems herein do not rely on a commercial process simulation engine for flowsheet evaluations. Instead, the models are built following the nonsmooth flowsheeting strategy outlined in Chapter 7. Throughout this chapter, the LD-derivatives of the objective and constraint functions with respect to the decision variables are always computed with respect to the identity directions matrix in order to obtain B-subdifferential elements of these functions. The simulation strategy for the PRICO process was discussed in detail in Example 7.2, while the same modeling strategy was used to simulate more complex SMR processes successfully in an article by Vikse *et al.*[132]

8.2.2 Optimization algorithm

The optimization problem in Equation (8.1) is a nonconvex, nonsmooth and constrained nonlinear program. Consultation of the nonsmooth optimization literature indicates that the algorithms ostensibly best suited for this class of problem are bundle methods.[62] However, attempts to use the solver MPBNGC v2.0,[77] which is an implementation of the proximal bundle method for constrained nonsmooth problems, were unsuccessful. Given an initial feasible point, the algorithm was never able to find an objective-improving feasible point, both when the constraints were included explicitly and when they were included in an exact penalty term to produce an unconstrained nonsmooth problem. It is conceivable that there exists some combination of option values for which this solver would be able to make progress on the problems presented in this chapter. However, a method for determining these values is not clear, and the use of a different solution method was deemed a more viable strategy.

Instead, the primal-dual interior-point optimizer, IPOPT,[133] proved to be an excellent choice for solving the optimization problem given in Equation (8.1) for the examples in this chapter. This is perhaps surprising, given that a core assumption of this interior-point method for nonlinear programs is that the objective and constraint

functions are at least twice continuously differentiable. That IPOPT performs well on the highly nonsmooth problems posed herein is both a testament to the robustness of the software and its heuristics as well as to the ability of the nonsmooth modeling framework to produce a compact representation of the process flowsheet while providing highly accurate sensitivity information. Nevertheless, the use of nonsmooth process models is (potentially) problematic in the context of the dual feasibility calculations in IPOPT. Dual feasibility in IPOPT is determined by the infinity-norm of the (internally-scaled) residual of the following equation:

$$\nabla f(\mathbf{x}) + \nabla \mathbf{c}(\mathbf{x})\boldsymbol{\lambda} - \mathbf{z}_L + \mathbf{z}_U = \mathbf{0}, \tag{8.2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $\mathbf{c} : \mathbb{R}^n \to \mathbb{R}^m$ are the equality constraints (including the inequality constraints reformulated using slack variables), $\mathbf{x} \in \mathbb{R}^n$ are the decision variables (including slack variables as needed), $\boldsymbol{\lambda} \in \mathbb{R}^m$ are the Lagrange multipliers for the equality constraints, $\mathbf{z}_L \in \mathbb{R}^n$ are the Lagrange multipliers for the lower bound constraints and $\mathbf{z}_U \in \mathbb{R}^n$ are the Lagrange multipliers for the upper bound constraints.[133] Clearly, the use of Equation (8.2) to assess the dual feasibility of a nondifferentiable optimal point is flawed; the correct statement for this case is as follows:

$$\mathbf{0} \in (\partial f(\mathbf{x}))^\intercal + (\partial \mathbf{c}(\mathbf{x}))^\intercal \boldsymbol{\lambda} - \mathbf{z}_L + \mathbf{z}_U, \tag{8.3}$$

where $(\partial f(\mathbf{x}))^\intercal$ is the Clarke generalized gradient of f at $\mathbf{x}$ and $\partial \mathbf{c}(\mathbf{x})$ denotes the Clarke Jacobian of $\mathbf{c}$ at $\mathbf{x}$. If f and $\mathbf{c}$ are continuous differentiable at $\mathbf{x}$, then Equation (8.3) reduces to Equation (8.2); however, at general nonsmooth points it is not tractable to calculate the full Clarke generalized gradient/Jacobian and check if Equation (8.3) is satisfied, and thus all known algorithms for nonsmooth local optimization rely on alternative termination criteria. For instance, MPBNGC uses the norm of a vector that represents an aggregation of Clarke generalized gradients calculated in recent iterations (along with information about the accuracy of the constraint linearization for problems with nonlinear constraints) to determine when to terminate, a condition that is not readily implemented in the context of another optimization

algorithm. Another nonsmooth local optimization algorithm, SolvOpt,[61] uses simple relative error metrics for the argument and function value iterates as termination criteria, which are only reasonable in conjunction with the specific step-size strategy employed by the algorithm. As a result, there is not a clear way to include such termination criteria for a nonsmooth problem in the existing IPOPT framework. By Rademacher's Theorem, the set of points at which a locally-Lipschitz continuous function is nondifferentiable is of Lebesgue measure zero; however, in finite precision arithmetic, this set is indeed reachable, as shown by e.g. the studies in Barton *et al.*[13] As will be shown in the following examples, the behavior of IPOPT suggests that the optimal solutions found in Examples 8.1 to 8.3 occur at differentiable points, while the optimal solutions in Example 8.4 occur at points of nondifferentiability, the latter case requiring modifications to the solver options in order to achieve convergence. Unless otherwise specified, the options used in IPOPT in the remainder of the chapter are as shown in Table 8.2. Any options not explicitly listed in Table 8.2 are left at their default values. The rationale for the choices made in Table 8.2

Table 8.2: Summary of the non-default IPOPT options specified for this work (unless otherwise noted).

IPOPT option	**Value**
`constr_viol_tol`	10^{-6}
`tol`	10^{-4}
`bound_push`	10^{-9}
`bound_frac`	10^{-9}
`mu_strategy`	`adaptive`
`hessian_approximation`	`limited-memory`
`limited_memory_max_history`	number of decision variables
`recalc_y_feas_tol`	10

is as follows. The low constraint violation tolerance simply requires high accuracy of the solution. With the constraint violation (primal) tolerance set to 10^{-6} and the complementarity tolerance left at the default value of 1×10^{-11}, the value of "tol" is then equivalent to the dual feasibility tolerance (as "tol" is defined as the maximum of the primal, dual and complementarity tolerances). As noted earlier, for the examples herein that appear to converge to differentiable points, this value

is as given in Table 8.2. In Example 8.4, this value is modified, as described later. The options `bound_push` and `bound_frac`, which move the initial point away from any active lower or upper bounds, are decreased substantially from their defaults. This is helpful in this work as the solver is provided with an initial feasible point in which some decision variables may be at their bounds, and it is undesirable for these values to be perturbed substantially prior to commencing solver iterations. The `mu_strategy` option, which governs the update behavior for the interior-point barrier value, was changed from the default (`monotone`) to `adaptive` based on empirical observations of improved performance. The option `hessian_approximation` is chosen so that the algorithm uses a limited-memory quasi-Newton update to approximate the "second derivatives" of the nonsmooth system since there is no method for supplying these directly for nonsmooth functions (nor are they even defined at nondifferentiable points). As a means of mitigating the consequences of approximating these quantities, the option `limited_memory_max_history` is increased from its default value of 6 to equal the number of decision variables in the problem, which determines the number of recent iterations that are considered when updating Hessian approximations. Increasing this value has a regularizing effect on the computed approximate Hessian, which can otherwise vary dramatically between iterations due to the nonsmoothness of the process models. This option value has been observed to work well for the problems in this chapter; it should not be considered a general heuristic for nonsmooth functions without substantially more investigation. It is likely that there should be an upper bound for this value that is independent of the problem size for larger problems, though it is not clear what value this bound should take. Finally, the option `recalc_y_feas_tol` tells the solver to recompute the constraint multipliers explicitly whenever the constraint violation is less than the value chosen (so long as the `recalc_y` option is enabled, which it is by default when the Hessian matrix is approximated). The IPOPT options reference notes that this can be helpful, although the multiplier recalculation requires additional factorization of the linear KKT system. However, since the function evaluations in this work are substantially more expensive than the solution algorithm itself, the cost of this additional linear algebra

is of little concern. In the context of the problems solved in this chapter, increasing this option from the default value of 10^{-6} generally substantially improves the quality of the subsequent steps taken by the algorithm whenever it moves to a point with significant constraint violation. It is important to note that there are almost certainly other IPOPT option sets that would produce equivalent or even better convergence behavior for the optimization problems considered in this chapter. However, overall good performance for the ensuing examples has been observed with only the minimal tuning described above, despite the nonsmoothness of the process models considered.

8.3 Liquefaction process optimization studies

Three liquefaction processes of increasing complexity are now presented and optimized with the strategy outlined in the previous section. The flowsheet models are written in the Julia programming language (v0.6.0) interfaced with IPOPT v3.12.1, and the optimization is performed on an Intel Xeon E5-1650 v2 workstation using six cores at 3.50 GHz and 12 GB RAM running Ubuntu v14.04. Nonideal thermophysical properties are calculated using the Peng-Robinson cubic equation of state with pure component and binary interaction parameter values as found in the Aspen Plus v8.4 databanks.[5] Ideal gas enthalpy and entropy values are also obtained using the default methods in Aspen Plus, namely via the use of the Aly-Lee model[1] for the ideal gas heat capacity, again with component parameters taken from the software's database. All vapor-liquid equilibrium calculations in the flowsheets are performed using nonsmooth inside-out algorithms from Chapter 5 augmented with nonsmooth density extrapolation procedures from Chapter 6. This includes calculations both within the MHEX units and in the other unit operations, i.e. compressors, throttle valves, mixers, condensers and phase separators. The equations and variables in these calculations are not visible to the optimization algorithm as they are performed with external subroutines. As a result, none of these other unit operations contribute to the size of the process model, and their outputs, e.g. work or temperatures after throttling/mixing, are calculated during the course of simulating these units in a

modular fashion as functions of their inputs and the decision variables.

The results and input-output sensitivities of these submodels are communicated to other models in the flowsheet and the optimizer as described in the previous section. All such calculations and associated sensitivity evaluations are converged to a tolerance of 10^{-8}. The MHEXs in each flowsheet are modeled using the nonsmooth method of Chapter 3 using the necessary provisions for handling internal phase change in a MHEX are described in Chapter 4 As in the previous chapter, the cooling curves of each stream in the nearly affine superheated and subcooled liquid regimes are discretized into five affine segments. In the far more nonlinear two-phase regime, the substreams are discretized into 20 affine segments each unless otherwise noted. The number of segments used to model the two-phase regime is denoted n_{2p}. As has been shown in Chapter 7 and will be revisited in Example 8.1, this is sufficient for a faithful representation of the true temperature-enthalpy relationship of the process streams.

Initialization of the following examples is performed rapidly and automatically with little input required from the user. The procedure involves first solving a simulation problem given a set of decision variables within the problem bounds. The set of unknown variables in the simulation problem generally consists of three variables afforded by the base nonsmooth MHEX model plus the set of temperatures needed to satisfy the energy balances in the superheated vapor and subcooled liquid regimes for each process stream entering the heat exchanger. The simulation problem itself has an automatic initialization procedure for choosing initial guesses for this set of temperatures, as described previously, leaving the user to only choose three variables and respective initial guesses. It is recommended that the UA value for MHEXs in the simulation be consistent with the value chosen for $UA_{\max}$ for the optimization. All other variables that will become decision variables in the optimization problem are held constant. The initial flowsheet simulation is then performed as described in previous work, e.g. Chapter 7 for the PRICO process and the article by Vikse *et al.* for the more complex liquefaction processes.[132]

This simple and robust procedure stands in contrast to that of many of the equation-based approaches outlined in the literature, which generally involve ardu-

ous and time-consuming calculations to produce an initial point. Additionally, while providing an initial feasible point to IPOPT tends to result in favorable convergence behavior, it is by no means always necessary. It is entirely possible to converge the problems in the following examples from infeasible starting points in most cases, although the computation time spent in additional IPOPT iterations is substantially higher than required to obtain a feasible initial guess from solving a simulation problem.

8.3.1 The PRICO process

The configuration of the PRICO process for this example is the same as shown in the previous chapter in Figure 7-5. The same nomenclature for the unknown variables in the process is also used here. For this model of the PRICO process, the decision variable vector is as follows:

$$\mathbf{x} \equiv (P_{\text{LPR}}, P_{\text{HPR}}, \mathbf{f}_{\text{MR}}, \Delta T_{\text{min}}, T_{\text{LPR}}^{\text{OUT}}, \mathbf{T}).$$

Given the nearly affine temperature-enthalpy behavior of these phase regimes, these equations add little complexity to the model when deferred to the optimizer instead of being converged on every iteration. Note that the outlet temperature of the high-pressure refrigerant stream, $T_{\text{HPR}}^{\text{OUT}}$, does not appear in the decision variable vector as it is simply set equal to the target temperature of the natural gas stream in each case. In total, this problem has 33 variables, 26 equality constraints and 2 inequality constraints. The breakdown of these numbers is as follows: length($\mathbf{T}$) = (number of streams entering MHEX)$\times$[(number of segments in superheated region - 1) + (number of segments in subcooled region - 1)] = $3 \times [(5-1)+(5-1)] = 24$. Each component of $\mathbf{T}$ is associated with an equation, and, in addition to the energy balance and the pinch operator in the nonsmooth MHEX model, this gives a total of 26 equality constraints. The two inequality constraints are those given in Equation (8.1). The refrigerant is allowed to consist of nitrogen, methane, ethane, propane and n-butane, so length($\mathbf{f}_{\text{MR}}$) = 5, which means length($\mathbf{x}$) = 33. The lower and upper bounds ($\mathbf{x}^{\text{LB}}$

and $\mathbf{x}^{\mathrm{UB}}$) for the decision variables are given in Table 8.3. As suggested by Austbø and Gundersen,[9] the lower bound on the component flowrate of ethane (which is always present in the solution) is nonzero to prevent the optimizer finding a trivial solution with zero refrigerant flowrate.

Table 8.3: Optimization variables and bounds for the PRICO process case studies.

Variable	Lower bound	Upper bound
P_{LPR} (MPa)	0.1	0.5
P_{HPR} (MPa)	1.0	5.0
f_{N2} (kmol/s)	0.0	1.0
f_{C1} (kmol/s)	0.0	2.0
f_{C2} (kmol/s)	0.1	3.0
f_{C3} (kmol/s)	0.0	1.0
f_{nC4} (kmol/s)	0.0	3.0
ΔT_{min} (K)	0.1	10.0
$T_{\mathrm{LPR}}^{\mathrm{OUT}}$ (K)	200.0	400.0
Components of $\mathbf{T}$ (K)	80.0	400.0

Example 8.1. The first example involving the PRICO process will be to study the effect that changing the UA_{max} value has on the optimal solution. A similar study was performed by Austbø and Gundersen[9] using the SQP algorithm NLPQLP[109] in conjunction with Aspen HYSYS, and similar trends in the optimal solutions are expected to be seen using the methodology of this chapter. An attempt at a direct comparison between the solutions given in Table 8.1 with the nonsmooth optimization results is also performed. Table 8.4 gives the data for the natural gas feed stream in the process for these first test cases.

In the first study, the process is optimized subject to different values of UA_{max}, with all other process parameters held equal. The compressor is assumed to have an isentropic efficiency of 80%. Numerical results for four such points are given in Table 8.5 for selected UA_{max} values. In Table 8.5, the iteration count, CPU time and improvement statistics are reported based on an initial point generated by solving a simulation problem as described in Example 7.2 (Variable Set I), the only difference being the change in UA_{max} values. Optimal composite curves for each case are shown in the top pane of Figures 8-1 to 8-4. In the bottom pane of each figure, the

Table 8.4: Natural gas stream data for Example 8.1.

Property	UA study	Literature comparison
Flowrate (kmol/s)	1.00	1.00
Pressure (MPa)	5.50	5.50
Inlet temperature (K)	295.15	298.15
Outlet temperature (K)	110.15	118.15
Composition (mol %)		
Nitrogen	1.00	2.80
Methane	91.60	89.70
Ethane	4.93	5.50
Propane	1.71	1.80
n-Butane	0.35	0.10
iso-Butane	0.40	0.00
iso-Pentane	0.01	0.00

temperature driving force profile is given for the initial feasible point (simulated with $n_{2p} = 20$) as well as for the solutions obtained by performing the optimization with different values of n_{2p}. Notice that while the optimal profiles in the $n_{2p} = 20$ and $n_{2p} = 50$ cases are nearly identical, the profiles for the $n_{2p} = 5$ cases are noticeably different in each instance. Optimizing with the coarse discretization indicates different pinch points in the optimal solution and the resulting temperature profiles clearly miss much of the nonlinearity of the cooling curves. In each case, while the optimal objective value in the $n_{2p} = 5$ case is consistently 2-5% lower than the optimal objective value in the other two cases (which themselves differ by at most 0.4% across all cases), the $n_{2p} = 5$ solutions are not even feasible in the constraints of the more finely discretized models. This underscores the need for highly accurate descriptions of the composite curves within the MHEX process model. Fortunately, in the flowsheeting framework used here, the model size does not increase with n_{2p}, only the cost of the function evaluation. This scaling is quite modest however, and comparing the CPU time spent optimizing the $n_{2p} = 50$ case to the time spent optimizing the $n_{2p} = 5$ case shows between an 8 and 11-fold increase across these cases.

The numerical trends in the results indeed conform with the observations of Austbø and Gundersen.[9] The optimal compressor power requirement is a trade-off between the pressure ratio and the refrigerant flowrate, and tends towards high ra-

Table 8.5: Key process metrics for locally-optimal solutions obtained from instances of the PRICO process with varying $UA_{\max}$ value ($n_{2p} = 20$).

$UA_{\max}$ (MW/K)	5.0	12.0	20.0	25.0
Power (MW)	24.43	19.25	17.55	16.93
Pressure ratio	37.34	14.84	7.87	6.47
P_{LPR} (MPa)	0.100	0.122	0.234	0.279
P_{HPR} (MPa)	3.734	1.805	1.842	1.804
F_{MR} (kmol/s)	1.83	2.05	2.55	2.75
$\mathbf{z}_{\text{MR}}$ (mol %):				
Nitrogen	6.33	5.70	7.93	8.32
Methane	29.59	22.90	23.75	24.02
Ethane	27.82	34.80	36.46	36.88
Propane	8.80	0.00	0.00	0.00
n-Butane	27.47	26.60	31.86	30.77
$\Delta T_{\min}$ (K)	2.73	1.44	1.07	0.95
IPOPT iterations	32	39	32	44
CPU time (s)	126	137	125	171
Improvement from initial feasible point	25.8%	17.5%	13.1%	13.8%

tios and low flowrates for low $UA_{\max}$ values. Note that these pressure ratios may be unrealistic for single-stage compression and in reality would require multistage compression with interstage cooling to be achieved. The optimal compressor train design is not considered at present. The large pressure difference in the exchanger causes the hot and cold streams to not pinch tightly, especially at low temperature levels, increasing the exergy losses but not requiring large heat exchanger conductance. Propane is also present in the optimal refrigerant mixture only for low $UA_{\max}$. For higher $UA_{\max}$ values, the optimal solution tends towards higher refrigerant flowrates and lower pressure ratios, leading to designs with small temperature differences at low temperatures and lower overall power consumption. While the constraint on $UA_{\max}$ is always active at the optimal solution, the constraint on the degree of superheat is notably not active in the optimal solution for any of these cases. This is consistent with the observations of Jensen and Skogestad[56] and Kamath *et al.*,[59] who note that a degree of superheating can sometimes be optimal for a simple refrigeration cycle with internal heat exchange such as the PRICO process.

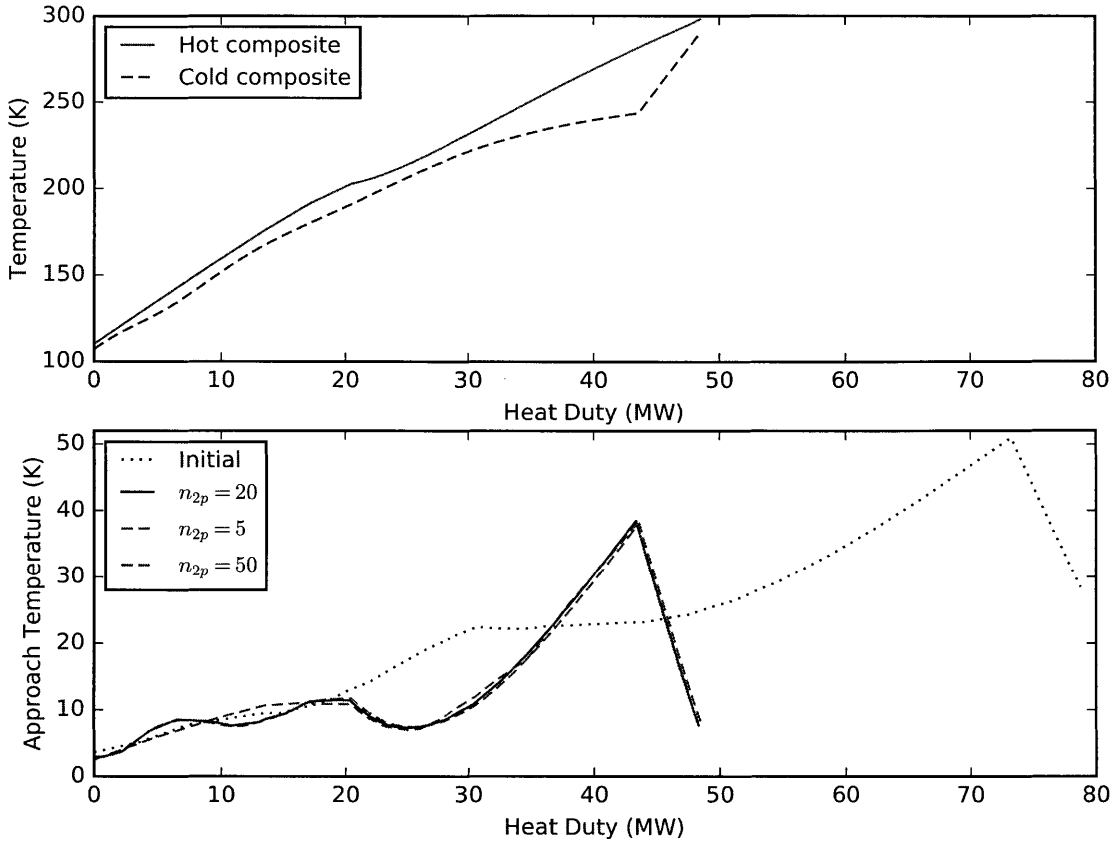


Figure 8-1: Top: hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process with $UA = 5.0$ MW/K optimized with $n_{2p} = 20$. Bottom: approach temperature profile for the MHEX in the PRICO process at the initial feasible point as well as the optimal solution computed with varying levels of discretization.

In terms of performance of the optimization strategy, Austbø and Gundersen[9] report that for the problem formulation used here, NLPQLP generally required between 1,200 and 1,900 flowsheet evaluations from Aspen HYSYS. In the present work, the worst case in Table 8.5 requires 44 iterations of IPOPT, with each iteration usually requiring only a single flowsheet evaluation to obtain function values and exact sensitivity information through operator overloading. In rare instances, a single iteration would require between two and ten evaluations of the objective function value (without sensitivity calculations) to satisfy the optimization algorithm's line search criteria. Note also that the computing time reported in Table 8.5 (and elsewhere in the chapter) includes the time needed to initialize and solve the simulation problem

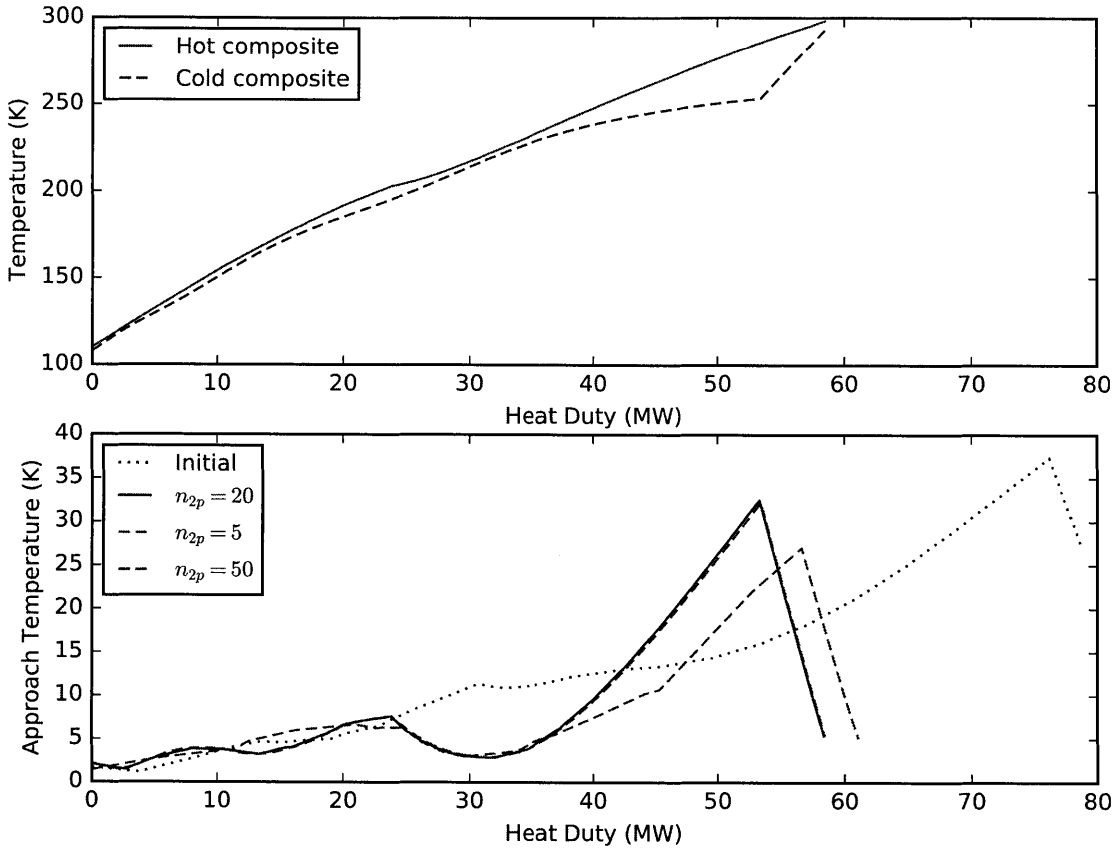


Figure 8-2: Top: hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process with $UA = 12.0$ MW/K optimized with $n_{2p} = 20$. Bottom: approach temperature profile for the MHEX in the PRICO process at the initial feasible point as well as the optimal solution computed with varying levels of discretization.

used to generate the initial feasible point for the optimizer, meaning that the entire optimization procedure can be completed in a short span of time and in an almost completely automated fashion.

A curve showing the trends in optimal power requirement and minimum approach temperature over a range of $UA_{\max}$ values between 5 MW/K and 50 MW/K was constructed from the results of solving a number of additional optimization problems, incrementing $UA_{\max}$ by 0.2 MW/K each time. These results are shown in Figure 8-5. The nonsmoothness of the curve corresponding to optimal $\Delta T_{\min}$ value suggests a large qualitative change in the overall solution behavior around $UA_{\max} = 8.0$ MW/K. The clear monotonicity and lack of noise in the curves also suggests that suboptimal

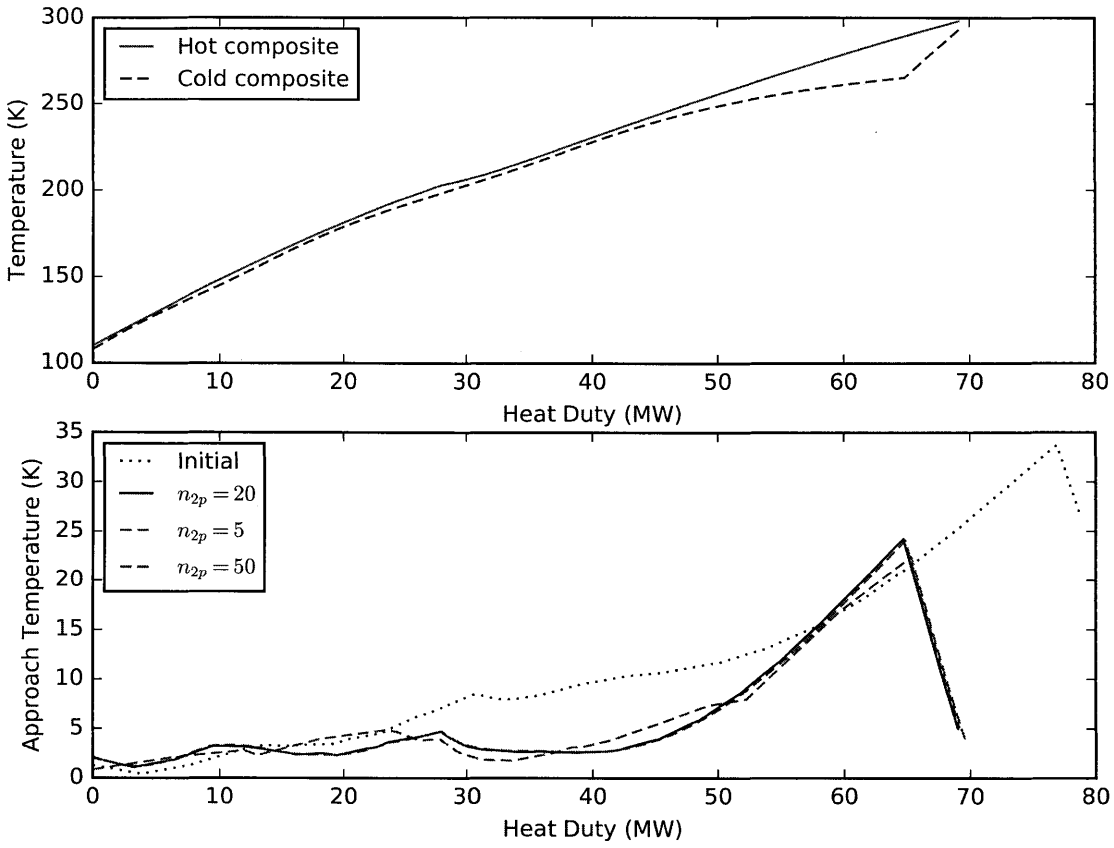


Figure 8-3: Top: hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process with $UA = 20.0$ MW/K optimized with $n_{2p} = 20$. Bottom: approach temperature profile for the MHEX in the PRICO process at the initial feasible point as well as the optimal solution computed with varying levels of discretization.

local solutions are likely not being found by the optimization procedure.

However, to test for better locally-optimal solutions, the four optimization problems represented in Table 8.5 were each solved from a set of distinct starting points each corresponding to one of the six solutions given in Table 8.1. Owing to residual differences in physical property methods and parameter values, it was not possible to solve for a feasible point in four of the six cases given the limited degrees of freedom available in the simulation environment. In those cases, the solution shown in Table 8.1 was passed directly to the optimizer with the remaining variables set to the midpoint between their bounds. Nevertheless, in each case including those with infeasible initial guesses, the solutions given in Table 8.5 were found (or indistinguishable solu-

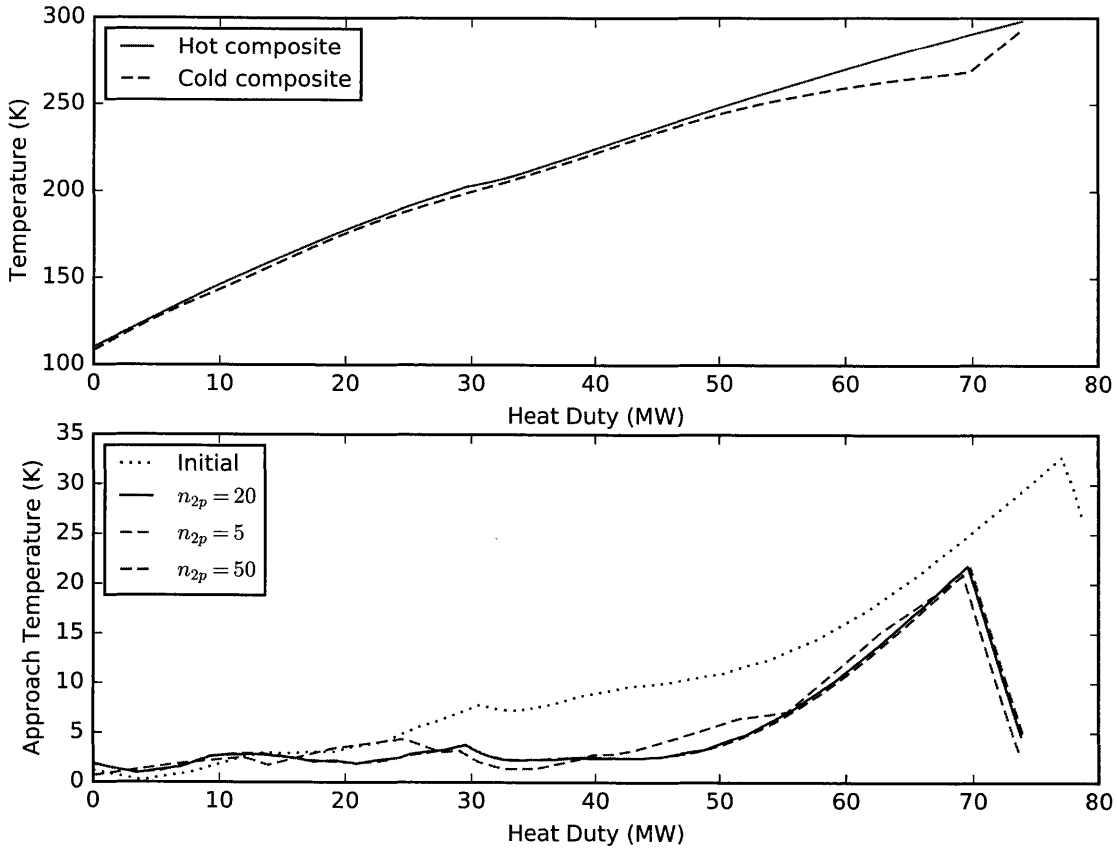


Figure 8-4: Top: hot composite curve (solid) and cold composite curve (dashed) for the MHEX in the PRICO process with $UA = 25.0$ MW/K optimized with $n_{2p} = 20$. Bottom: approach temperature profile for the MHEX in the PRICO process at the initial feasible point as well as the optimal solution computed with varying levels of discretization.

tions within the given tolerances). In contrast, Austbø and Gundersen[9] report that NLPQLP/Aspen HYSYS only returns their process' best-known solution 10-20% of the time when the constraint on $UA_{\max}$ is included in their multistart experiments.

Next, for the sake of comparison and subject to the many caveats discussed earlier, another instance of the PRICO process is optimized with specifications set as close to those found in the articles summarized in Table 8.1 as possible. First, the optimization is performed without the constraint on $UA_{\max}$ in Equation (8.1), instead replacing it with the more commonly-used constraint that the approach temperature is at least 1.2 K everywhere in the MHEX. In the absence of the $UA_{\max}$ constraint, the tendency of the optimizer is to find MHEXs with very high conductance values.

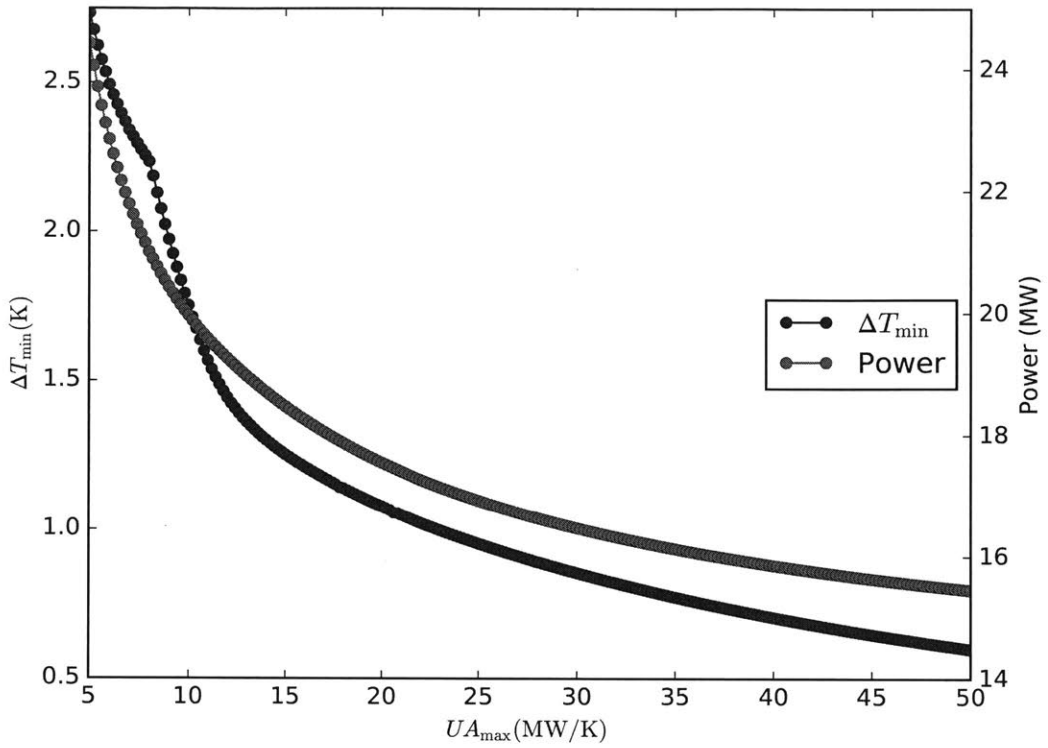


Figure 8-5: Optimal values for $\Delta T_{\min}$ and compression power in the PRICO process as the value of $UA_{\max}$ is parametrically varied in the optimization formulation of Equation (8.1).

With $n_{2p} = 50$ to ensure extremely high accuracy, the optimizer finds a solution with $UA = 30.30$ MW/K with a power requirement of 14.90 MW, assuming 80% isentropic efficiency. More details of the solution are given in Table 8.6. However, this MHEX can be further improved by using the formulation from Equation (8.1) and fixing $UA_{\max} = 30.30$ MW/K. This new optimization finds a slightly improved solution with a power requirement of 14.85 MW that has closer approach temperatures (i.e. less than 1.2 K) in the cold end of the exchanger and larger ones at the warm end, in addition to a lower overall cooling load. This is exactly as expected for a more optimal utilization of the MHEX area. Numerical values for this solution are also found in Table 8.6, and the approach temperature profiles are compared in the left pane of Figure 8-6.

Noting that the optimal value of $\Delta T_{\min}$ decreases monotonically with increasing

$UA_{\max}$ value, a solution at which $\Delta T_{\min} = 1.20$ K using the optimization formulation in Equation (8.1) was determined by performing a number of optimizations parametrically varying $UA_{\max}$, as shown in the right pane of Figure 8-6. The value of $UA_{\max}$ that yields a solution with $\Delta T_{\min} = 1.20$ K is 16.85 MW/K, and the process power requirement at this point is 16.16 MW. Therefore, given that it was obtained by the $UA_{\max}$ formulation, this is the optimal configuration for the process with a $\Delta T_{\min}$ value of 1.2 K. The solution is detailed in Table 8.6. Note that this solution is a feasible point in the formulation with the $\Delta T_{\min}$ constraint; however, it is not locally optimal in this formulation (the infinity-norm residual of Equation (8.2) at this point is 6.57×10^{-1}) since the objective value can be reduced by moving to significantly higher conductance values in the absence of a constraint on $UA_{\max}$.

Table 8.6: Results of using different optimization formulations to compare with PRICO process results in the literature.

	Initial solution using $\Delta T_{\min}$ formulation with $\Delta T_{\min} := 1.20$ K	Improved sol. using $UA_{\max}$ formulation with $UA_{\max} := 30.30$ MW/K	$UA_{\max}$ formulation such that $\Delta T_{\min} = 1.20$ K with $UA_{\max} := 16.85$ MW/K
$\Delta T_{\min}$ (K)	1.20	0.83	1.20
UA (MW/K)	30.30	30.30	16.85
Power (MW)	14.90	14.85	16.16
Pressure ratio	5.06	5.28	8.34
P_{LPR} (MPa)	0.316	0.302	0.209
P_{HPR} (MPa)	1.601	1.594	1.740
F_{MR} (kmol/s)	2.82	2.74	2.27
$\mathbf{z}_{\text{MR}}$ (mol %):			
Nitrogen	7.01	6.32	5.55
Methane	23.51	23.65	23.11
Ethane	38.62	38.63	37.67
Propane	0.00	0.00	0.00
n-Butane	30.85	31.39	33.67

To ensure that the solution in the last column of Table 8.6 is indeed a high-quality solution and to give a sense of the overall robustness of the optimization procedure, a multistart strategy was also employed in search for better solutions. One hundred randomly chosen initial guesses (with components chosen from uniform distributions

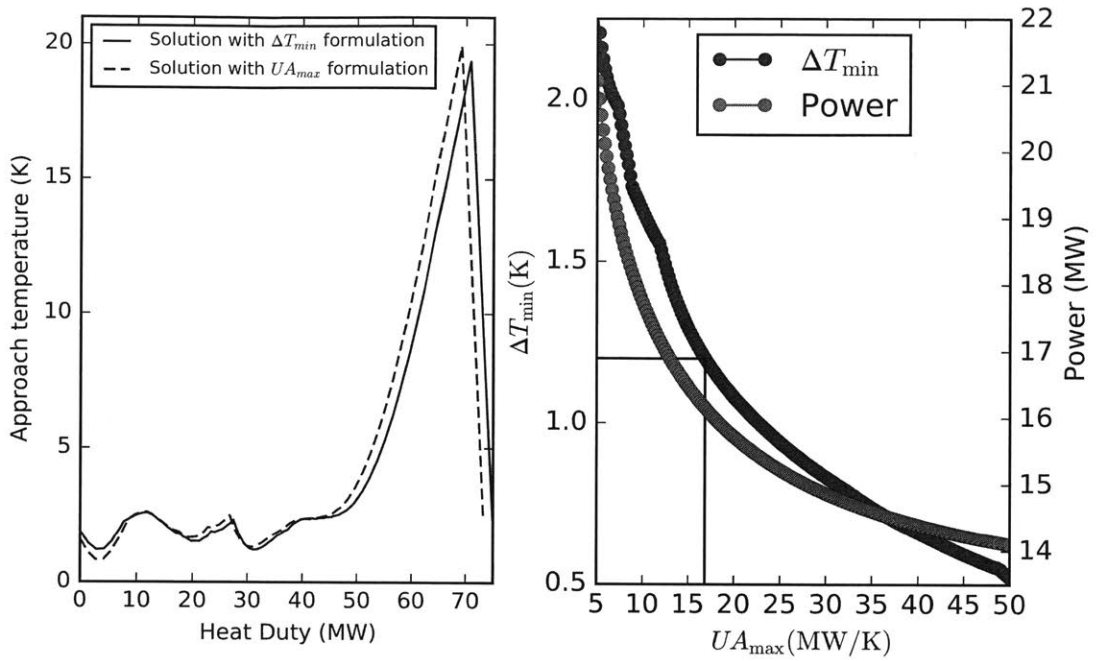


Figure 8-6: Left: approach temperature profiles for the initial and improved solutions detailed in the first two columns of Table 8.6. Right: optimal values for $\Delta T_{\min}$ and compression power for the PRICO process (literature configuration) as the value of $UA_{\max}$ is parametrically varied in the optimization formulation of Equation (8.1).

within each variable's bounds) were used to solve the optimization problem. Note that it is extremely unlikely that initial points generated in this manner will be feasible in the problem constraints. From the 100 initial guesses, 83 runs converged to the solution in Table 8.6 (or an indistinguishable solution within the overall tolerance of 10^{-4}), 1 run converged to a barely-suboptimal point (0.027% increase in objective function value), 9 runs exceeded the maximum iteration limit of 200 and 7 runs aborted due to numerical difficulties in the IPOPT feasibility restoration phase. For the 83 runs in which the best known solution was found, a histogram of the number of IPOPT iterations required to converge each instance is shown in Figure 8-7. This demonstrates that the optimization strategy is very robust even for this challenging problem formulation and in the absence of a feasible initial guess (c.f. the 10-20% success rate reported in Austbø and Gundersen[9]).

Figure 8-8 shows the composite curves and approach temperature profile for this

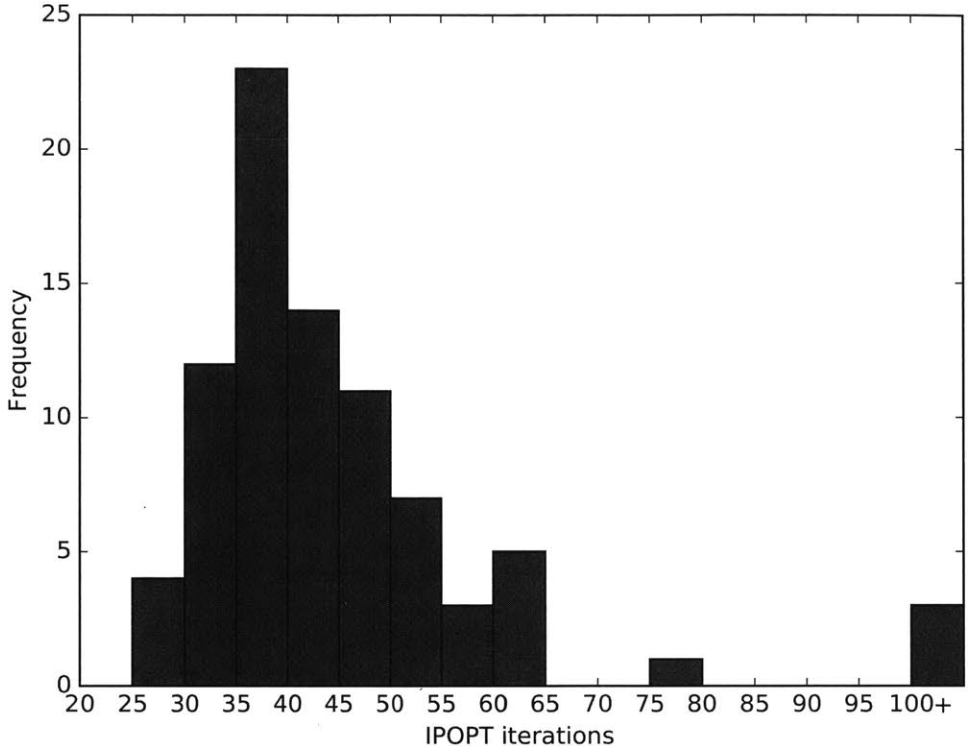


Figure 8-7: IPOPT iteration count histogram for PRICO optimization using a multistart strategy.

best solution. The approach temperature profile predicted by Aspen Plus v8.4[5] is overlaid onto the optimization result (obtained by providing the entire solution with the exception of the value of $T_{\mathrm{LPR}}^{\mathrm{OUT}}$ to the software) to validate that the solution is indeed physical, assuming that these thermodynamic models and parameter values (from Aspen Plus) provide a valid description of the real fluid behavior. The validation simulation returns a power requirement of 16.15 MW, which is in excellent agreement with the optimization result of 16.16 MW.

Example 8.2. In this second example involving the PRICO process, a case that has been less explicitly studied in the literature is considered, which is that of variable natural gas composition entering the process. The problem formulation and process conditions are identical to that described in the UA study of Example 8.1, except that now the value of UA_{max} for the MHEX is fixed at 15.0 MW/K and four different

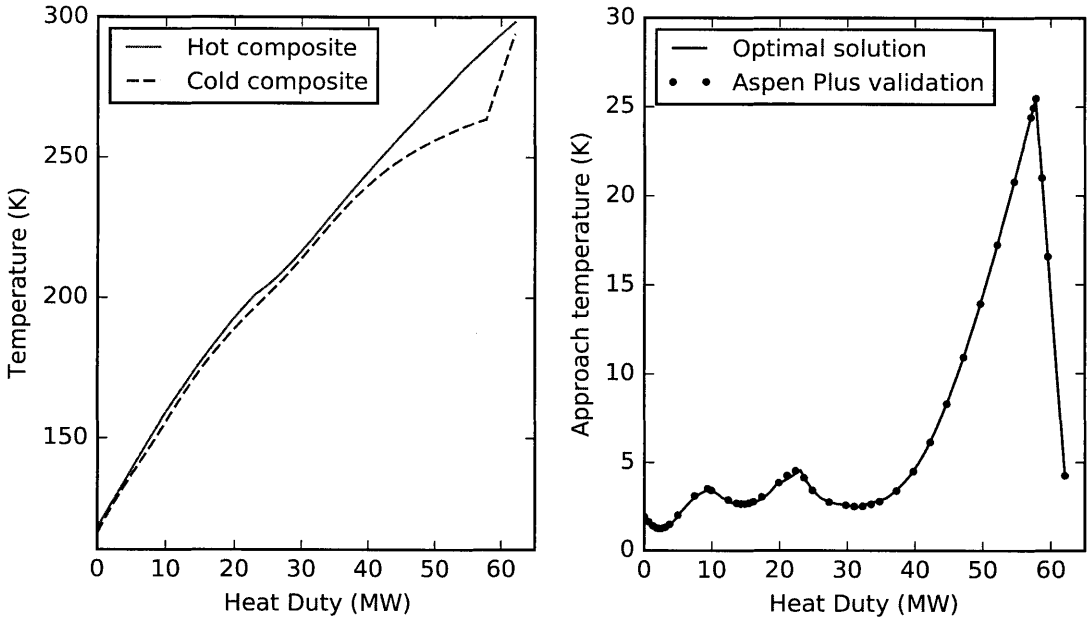


Figure 8-8: Optimal composite curves and approach temperature profile of a PRICO process modeled using specifications frequently found in the literature.

Table 8.7: Rich and lean natural gas compositions considered in Example 8.2.

Composition (mol %)	Lean gas	Rich gas	Very rich gas
Nitrogen	1.0	1.0	1.0
Methane	95.6	87.6	83.6
Ethane	3.1	6.5	8.5
Propane	0.2	3.9	4.9
n-Butane	0.1	1.0	2.0

natural gas streams are supplied to the process. One case assumes the natural gas is the same as in the first column of Table 8.4, while the other three compositions considered are given in Table 8.7, two of which have lower methane content than the base case (richer natural gas) and one of which has higher methane content than the base case (leaner natural gas). The natural gas streams are supplied at 5.5 MPa, 295.15 K and 1.0 kmol/s in each case with a target temperature of 110.15 K.

Each case is optimized using IPOPT with the options given in Table 8.2. Numerical results for each feed stream are given in Table 8.8 and the temperature driving force profiles in the MHEX for all four cases are compared in Figure 8-9. As in the previous example, the iteration count, CPU time and improvement statistics are

reported based on an initial point generated by solving a simulation problem as described in Example 7.2 (Variable Set I) except with different natural gas feed streams and a UA value of 15.0 MW/K.

Table 8.8: Key process metrics for locally-optimal solutions obtained from instances of the PRICO process with varying natural gas composition ($n_{2p} = 20$, $UA_{max} = 15.0$ MW/K).

NG methane mol%	83.6%	87.6%	91.6%	95.6%
Power (MW)	18.68	18.56	18.46	18.35
Pressure ratio	11.70	11.29	10.80	11.10
P_{LPR} (MPa)	0.142	0.155	0.171	0.168
P_{HPR} (MPa)	1.664	1.748	1.850	1.863
F_{MR} (kmol/s)	2.22	2.24	2.27	2.22
$\mathbf{z}_{MR}$ (mol %):				
Nitrogen	6.02	6.48	6.97	6.90
Methane	22.98	22.99	23.30	23.69
Ethane	33.05	34.44	35.72	36.11
Propane	0.00	0.00	0.00	0.00
n-Butane	37.95	36.08	34.01	33.29
ΔT_{min} (K)	1.22	1.24	1.25	1.32
IPOPT iterations	29	89	33	29
CPU time (s)	107	359	140	133
Improvement from initial feasible point	14.8%	14.0%	13.3%	13.1%

A very similar process in terms of the power requirement, pressure ratio, refrigerant flowrate and minimum approach temperature is found for each set of optimal operating conditions in Table 8.8. However, the optimal refrigerant composition and pressure levels do vary from case to case, as do the temperature driving force profiles in the region where the natural gas stream traverses the two-phase region. As a result, the optimal conditions corresponding to any of these natural gas streams are not optimal for any of the others and can even lead to infeasible operation as a result of temperature crossovers. Table 8.9 shows the changes in process power requirement as a result of operating the PRICO process with each solution shown in Table 8.8 for each natural gas feed composition, pairwise. In the context of Table 8.8, "infeasible" means that some target parameter of the process (e.g. the product LNG tempera-

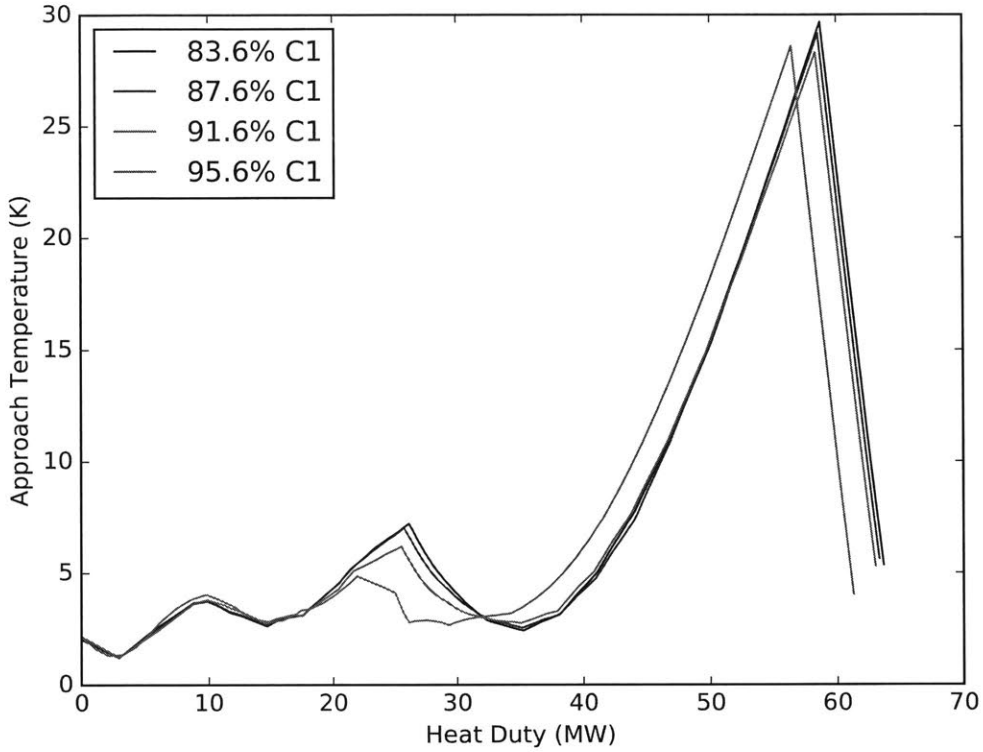


Figure 8-9: Optimal approach temperature profiles in the MHEX of a PRICO process with $UA = 15.0$ MW/K with varying natural gas composition.

ture) would have to be relaxed for the flowsheet to have a physical solution. These data underscore the importance of being able to determine new optimal conditions efficiently in the event of changes in feed gas composition to ensure process reliability.

8.3.2 Complex SMR processes

Two more advanced SMR liquefaction processes that were previously simulated successfully using the nonsmooth flowsheeting strategy by Vikse *et al.*[132] are now studied. The first process (Figure 8-10) is a more advanced version of the PRICO process in which the refrigerant mixture is phase-separated prior to entering the MHEX. The resulting liquid phase MR stream containing the heavier hydrocarbons only participates in heat exchange at the warmer end of the MHEX as a safeguard against freezing and plugging of the exchanger tubes. The second process (Figure 8-13) is a natural gas

Table 8.9: Change in power consumption for the PRICO process when each solution from Table 8.9 (rows) is used to operate the process for each of the four natural gas feed streams (columns).

Solution \ Feed	Very rich gas	Rich gas	Base case	Lean gas
Very rich gas	–	+0.19%	+0.68%	+1.08%
Rich gas	+0.33%	–	+0.18%	+0.49%
Base case	Infeasible	+0.37%	–	+0.18%
Lean gas	Infeasible	Infeasible	+0.51%	–

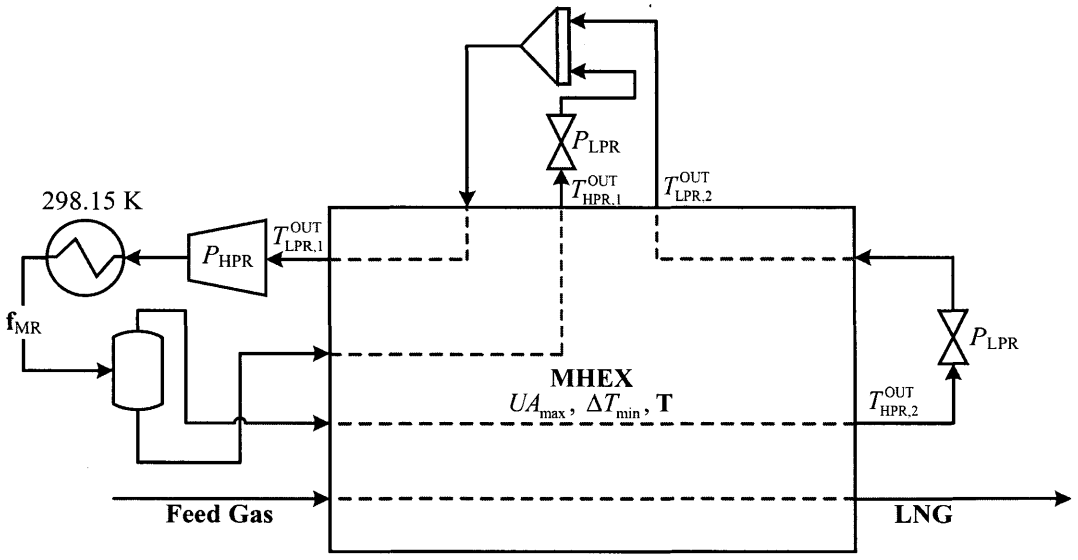


Figure 8-10: Flowsheet of an SMR process with phase separation of the refrigerant.

liquids extraction process with two MHEXs in which the heavier components of the natural gas stream are removed by flash separation between the two heat exchangers. This process also features phase separation of the refrigerant stream. As in the previous examples, all compressors are assumed to have an isentropic efficiency of 80% and condensers are able to exchange heat with a large enough sink to cool hot mixed-refrigerant streams to 298.15 K.

Example 8.3. The flowsheet of this more advanced SMR process with a phase separator is shown in Figure 8-10 with key decision variables and parameters indicated. Most of these variables are the same as detailed in Example 8.1, with some additional

variables included to model the additional exit temperatures of the refrigerant mixture streams. In particular, $T^{\text{OUT}}_{\text{HPR},1}$ and $T^{\text{OUT}}_{\text{LPR},1}$ refer to the hot and cold MR stream leaving the warmer side of the MHEX, respectively, while $T^{\text{OUT}}_{\text{HPR},2}$ and $T^{\text{OUT}}_{\text{LPR},2}$ denote the hot and cold MR stream leaving the cold end of the MHEX, respectively. As before, $T^{\text{OUT}}_{\text{HPR},2}$ is fixed to the target temperature of the LNG product stream, i.e. 110.15 K in this instance of the process, as this is well-known to be optimal. $T^{\text{OUT}}_{\text{HPR},1}$ is included as an additional decision variable with a lower bound of 180 K and an upper bound of 400 K. $T^{\text{OUT}}_{\text{LPR},2}$ is also included as a decision variable with lower and upper bounds of 150 K and 400 K, respectively. All other variables are still constrained within the ranges shown in Table 8.3. The flash vessel in the refrigerant stream operates adiabatically at the high-pressure level, while the stream mixer operates adiabatically at the low-pressure level. Neither of these additional unit operations have decision variables associated with them. Thus, the decision variable vector for this process optimization problem is as follows:

$$\mathbf{x} \equiv (P_{\text{LPR}}, P_{\text{HPR}}, \mathbf{f}_{\text{MR}}, \Delta T_{\text{min}}, T^{\text{OUT}}_{\text{LPR},1}, T^{\text{OUT}}_{\text{HPR},1}, T^{\text{OUT}}_{\text{LPR},2}, \mathbf{T}),$$

with $\mathbf{T}$ again representing the vector of unknown temperatures in the superheated and subcooled region of each of the five process streams in the MHEX. With this variable set, this process optimization problem has 51 variables, 42 equality constraints and 2 inequality constraints.

The process is now optimized subject to four different values of UA_{max}. As noted by Vikse *et al.*,[132] the use of different refrigerant mixture compositions (as a result of the phase separation) to provide cooling at different temperature levels results in less refrigerant needing to be circulated at the cold end of the heat exchanger where irreversibilities are most significant. This means that less power is required to achieve the same liquefaction as the basic PRICO process for the same heat exchanger conductance, or equivalently that a significantly smaller heat exchanger can be installed to achieve the same power requirement. For this example, the options set in IPOPT are exactly the same as shown in Table 8.2. The numerical results for optimization of

the process subject to $UA_{\max} = 6.0$, 8.0, 10.0 and 12.0 MW/K are given in Table 8.10 and the approach temperature profiles for the MHEX in each case are shown in Figure 8-11. In Table 8.10, the iteration count, CPU time and improvement statistics are reported based on an initial point generated by solving a simulation problem under conditions given in Case II of Example 1 from Vikse *et al.*,[132] though with different fixed values of the MHEX conductance.

Table 8.10: Key process metrics for locally-optimal solutions obtained from instances of the advanced SMR process in Figure 8-10 with varying $UA_{\max}$ value ($n_{2p} = 20$).

$UA_{\max}$ (MW/K)	6.0	8.0	10.0	12.0
Power (MW)	19.61	18.66	18.05	17.59
Pressure ratio	23.23	17.92	13.96	10.75
P_{LPR} (MPa)	0.100	0.121	0.153	0.199
P_{HPR} (MPa)	2.323	2.167	2.148	2.136
F_{MR} (kmol/s)	1.99	2.07	2.19	2.37
$\mathbf{z}_{\text{MR}}$ (mol %):				
Nitrogen	2.69	3.38	4.35	5.46
Methane	19.78	19.19	19.31	19.78
Ethane	37.03	38.21	39.65	40.50
Propane	1.80	0.84	0.00	0.00
n-Butane	38.69	38.38	36.69	34.26
$\Delta T_{\min}$ (K)	2.09	1.72	1.39	1.21
IPOPT iterations	41	46	54	41
CPU time (s)	267	281	326	214
Improvement from initial feasible point	12.9%	7.8%	5.0%	3.8%

The overall trend in the results is very similar to what was observed in Example 8.1 for the PRICO process. As $UA_{\max}$ is increased, the refrigerant flowrate increases, the pressure ratio decreases and propane disappears from the optimal refrigerant mixture. Unlike the PRICO process, however, the superheating constraint is active in each of the four cases, implying that the power requirement could be further decreased if a less conservative value of $\Delta T_{\text{sup,min}}$ were chosen. The overall power requirement for the process is also substantially reduced compared to the demands of the basic PRICO process. Indeed, similar process power requirements are observed for this process and the basic PRICO process when the MHEX conductance value is approximately

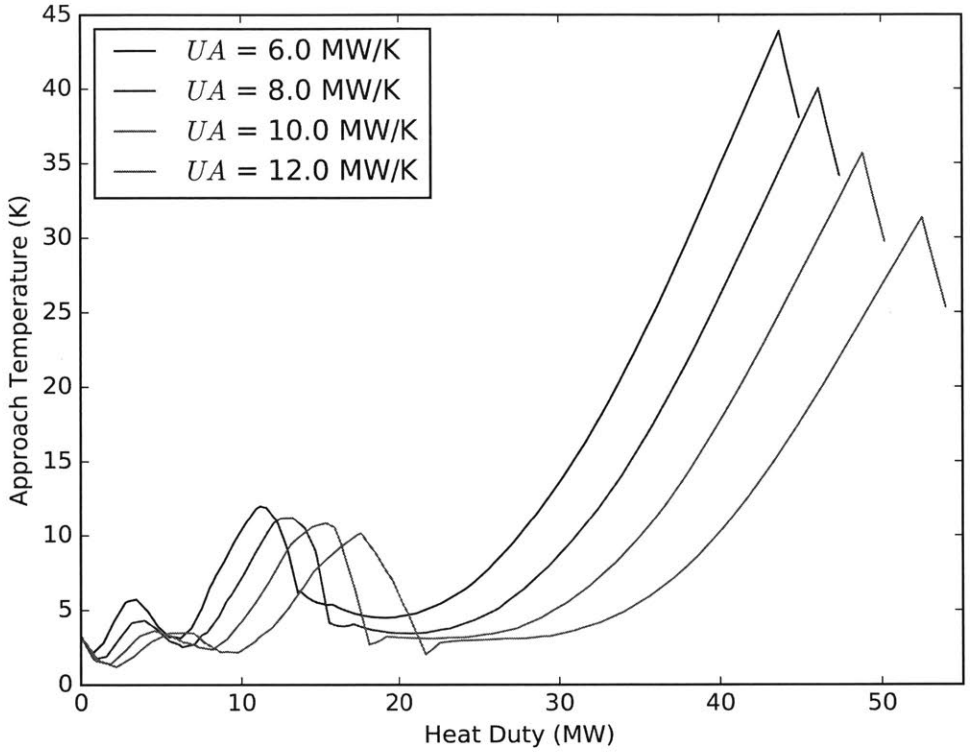


Figure 8-11: Optimal approach temperature profiles in the MHEX of an advanced SMR process with varying UA value.

halved, indicating significant benefits to performing phase separation of the mixed refrigerant for liquefaction trains that have stringent space or capital expenditure limits.

As in Example 8.1, additional optimization problems were solved to give a better sense of the parametric variation of the power requirement and $\Delta T_{\min}$ over a range of $UA_{\max}$ values even larger than that shown in Table 8.10. The results of these optimization problems are shown in Figure 8-12. Once again, the clear trends and monotonicity of these profiles suggest that high-quality local solutions are being found reliably by this optimization strategy.

Example 8.4. The flowsheet of this complex liquefaction process with intermediate NGL extraction and phase separation of the mixed refrigerant is shown in Figure 8-13 with the decision variables indicated. In this case, the outlet temperatures as-

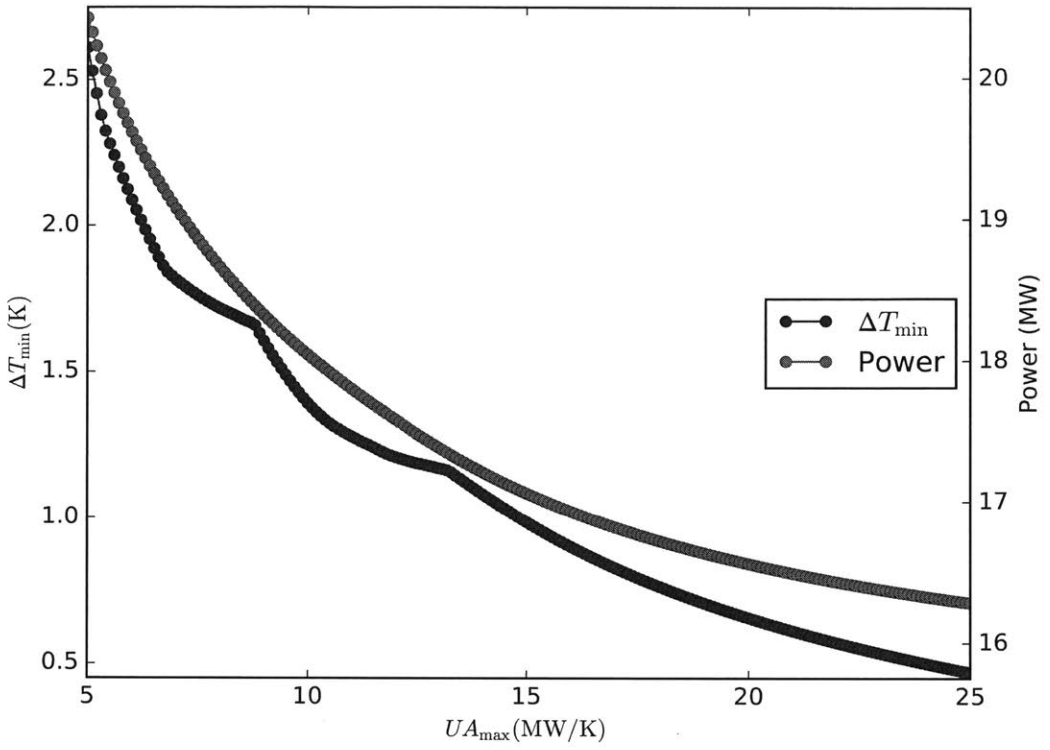


Figure 8-12: Optimal values for $\Delta T_{\min}$ and compression power in the advanced SMR process as the value of $UA_{\max}$ is parametrically varied in the optimization formulation of Equation (8.1).

sociated with MHEX 1 (the precooling heat exchanger) are denoted by $T^{\text{OUT}}_{\text{HPR},1}$ and $T^{\text{OUT}}_{\text{LPR},1}$ and the outlet temperatures associated with MHEX 2 (the main cryogenic heat exchanger) are denoted by $T^{\text{OUT}}_{\text{HPR},2}$ and $T^{\text{OUT}}_{\text{LPR},2}$. In total, three streams will exit at temperature $T^{\text{OUT}}_{\text{HPR},1}$, both hot MR streams and the natural gas stream entering the phase separator, though in this process, this variable value is not fixed *a priori*. As before, $T^{\text{OUT}}_{\text{HPR},2}$ is fixed to the target temperature of the LNG product, which in this case is 120.15 K. Additionally, each heat exchanger will have a conductance, minimum temperature difference and unknown temperature vector associated with it, as indicated by appropriate subscripts. For this process, instead of individually specifying $UA_{\max}$ for each MHEX, a $UA_{\max}$ value is chosen for the overall maximum heat exchanger conductance (i.e. a value based on the total allowable capital investment or physical space restrictions), and the optimizer will decide how to apportion the

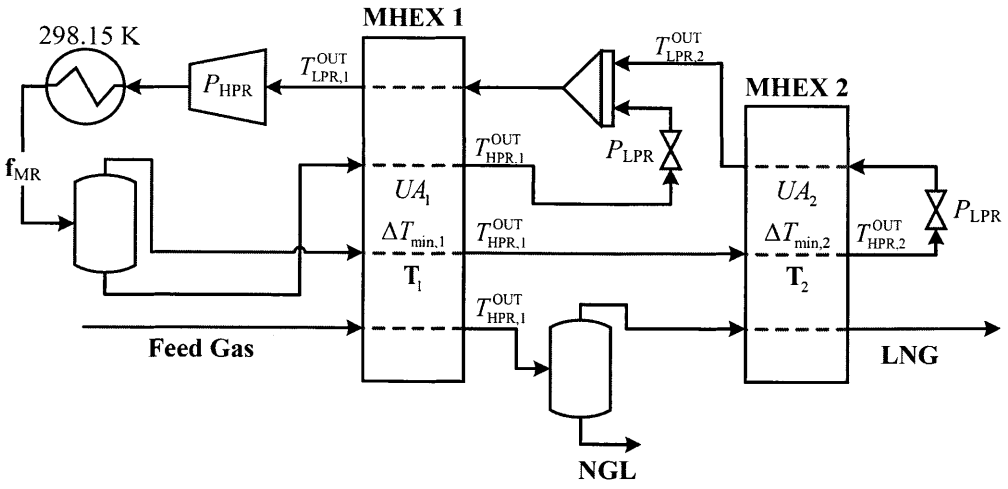


Figure 8-13: Flowsheet of an advanced SMR liquefaction process with intermediate NGL extraction.

total amount between MHEX 1 and MHEX 2. Accordingly, the $UA_{\max}$ constraint in Equation (8.1) is replaced by $UA_1(\mathbf{x}) + UA_2(\mathbf{x}) \leq UA_{\max}$. In summary, the decision variable vector is as follows for this NGL extraction process:

$$\mathbf{x} \equiv (P_{\text{LPR}}, P_{\text{HPR}}, \mathbf{f}_{\text{MR}}, \Delta T_{\text{min},1}, \Delta T_{\text{min},2}, T^{\text{OUT}}_{\text{LPR},1}, T^{\text{OUT}}_{\text{HPR},1}, T^{\text{OUT}}_{\text{LPR},2}, \mathbf{T}_1, \mathbf{T}_2),$$

and so the optimization problem has 68 variables, 60 equality constraints and 2 inequality constraints in total. The bounds on the pressure levels, component flowrates and outlet temperatures are either as shown in Table 8.3 or as described in Example 8.3, as appropriate. Both $\Delta T_{\min}$ values are constrained to the interval [0.1, 10.0] and the each component of $\mathbf{T}_1$ and $\mathbf{T}_2$ is constrained to the interval [80.0, 400.0]. Both the flash vessel in the refrigerant stream and the NGL extraction vessel operate adiabatically, at the high pressure level and at the natural gas feed pressure, respectively. The stream mixer operates adiabatically at the low pressure level, as in the previous example. As in the simulations performed by Vikse *et al.*,[132] the feed natural gas stream is richer and enters at lower pressure than in the previous examples to ensure adequate separation of the NGL product. For this instance of the process, the natural

gas initial composition is given by the "very rich gas" column of Table 8.7, and the stream enters the process at 3.5 MPa with a flowrate of 1.0 kmol/s. For the purposes of this example, assume that the low methane content of very rich feed stream makes the final heating value of the product unacceptable. Therefore, the quality of this stream is upgraded by splitting the feed into a NGL product consisting primarily of heavier hydrocarbons and a lean LNG product. The three target methane molar percentages for the LNG considered in this example are 87.6%, 91.6% and 95.6%, and three optimization problems are solved subject to the addition of an inequality constraint on these minimum methane content values for the LNG product. For this process, the options in IPOPT are chosen as shown in Table 8.2 with the exception of the overall tolerance, "tol", which is increased to a value of 0.1, which, as described earlier, is equivalent to loosening the dual feasibility tolerance. This is done because it was observed that in each of the three cases, the optimization algorithm would quickly make significant improvement from the initial point to new primal feasible points and then iterate around such points, unsuccessfully attempting to reduce the dual infeasibility until the iteration limit was reached. As noted earlier however, Equation (8.2) need not hold for nonsmooth points, and so this behavior suggests that the solver is likely attempting to converge to an optimal but nonsmooth point. Accordingly, the dual tolerance is reduced to a level where the algorithm terminates at such points.

The numerical results for these studies with $UA_{\max} = 10.0$ MW/K for the three different levels of product upgrading are given in Table 8.11 and the approach temperature profiles in the MHEXs in each case are shown in Figure 8-11. The reported iteration counts and CPU times are reported based on an initial point generated by solving a simulation problem analogous to those described in Example 3 of Vikse *et al.*[132] In this example, the initial point generated in each case was not feasible in the quality constraint; however this evidently did not impact the rapid convergence to optimal solutions.

As higher purity natural gas is required, the temperature at which the NGL extraction process occurs increases to remove more of the C2+ hydrocarbons, as ex-

Table 8.11: Key process metrics for locally-optimal solutions obtained from instances of the NGL extraction process in Figure 8-13 with $UA_{\max} = 10.0$ MW/K ($n_{2p} = 20$).

LNG methane mol %	87.6%	91.6%	95.6%
Power (MW)	17.04	15.56	13.69
Pressure ratio	9.14	10.25	14.41
P_{LPR} (MPa)	0.161	0.158	0.101
P_{HPR} (MPa)	1.474	1.624	1.460
$T^{\text{OUT}}_{\text{HPR},1}$ (K)	196.29	218.71	239.54
LNG flowrate (kmol/s)	0.931	0.840	0.573
F_{MR} (kmol/s)	2.52	2.17	1.56
$\mathbf{z}_{\text{MR}}$ (mol %):			
Nitrogen	2.72	2.48	1.17
Methane	16.21	16.04	12.52
Ethane	39.18	41.76	41.83
Propane	4.78	0.00	0.00
n-Butane	37.10	39.72	44.48
MHEX 1 $\Delta T_{\min}$ (K)	6.76	3.87	1.50
MHEX 2 $\Delta T_{\min}$ (K)	1.68	1.49	0.962
MHEX 1 UA value (MW/K)	1.95	4.02	6.47
MHEX 2 UA value (MW/K)	8.05	5.98	3.53
IPOPT iterations	42	75	53
CPU time (s)	281	458	325

pected. This also means that the LNG yield of the process decreases as the feed is progressively upgraded, and so the size of the cold-end MHEX and the refrigerant flowrate are reduced proportionally. Simultaneously, the pressure ratio in the process increases as the cold-end MHEX increasingly resembles the low conductance exchangers from previous examples, with the net effect of a decrease in the compressor power requirement when producing higher quality LNG at a reduced flowrate.

8.4 Conclusions

An optimization strategy for complex natural gas liquefaction processes has been presented that uses the interior-point local optimization method of IPOPT paired with a compact yet highly accurate description of process flowsheets given by nonsmooth models. Despite the nonsmoothness, IPOPT proved to be a robust solver for these problems, converging in fewer than fifty iterations for the majority of the examples considered in this chapter. The nonsmooth process model for the MHEXs at the core of each of the cases described in this work was readily amenable to the problem formulation given in Equation (8.1) that has been shown to produce the most optimal process operating conditions in terms of reducing thermodynamic losses and minimizing compressor power requirements. The processes considered in Examples 8.3 and 8.4 have never been optimized subject to such a formulation in the literature and the solutions reported here indicate that optimal operation of these processes can lead to significant cost savings over basic SMR processes such as PRICO.

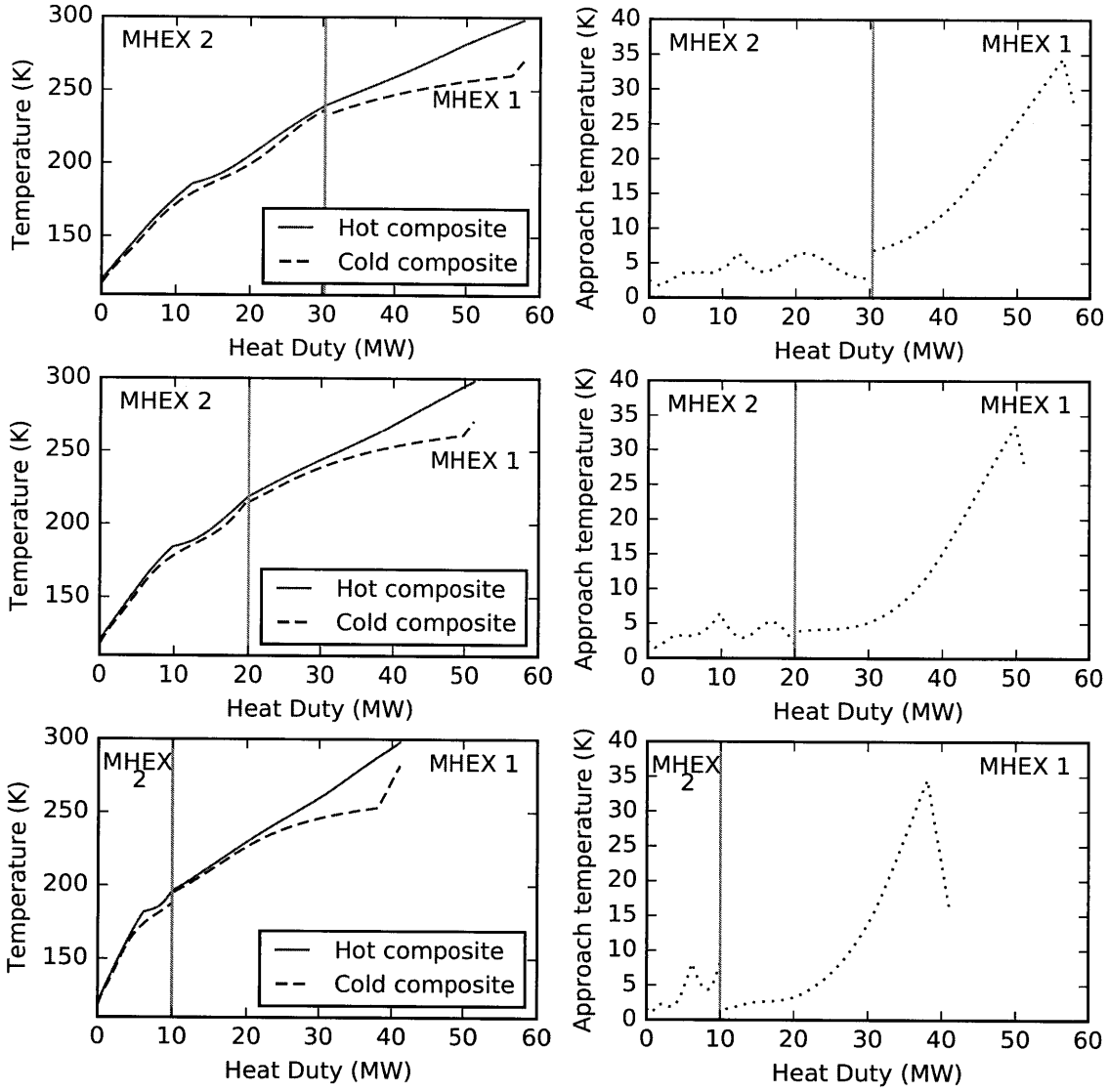


Figure 8-14: Top row: composite curves (left) and approach temperature profile (right) for the NGL extraction process upgrading the feed gas to 87.6% methane. Middle row: composite curves (left) and approach temperature profile (right) for the NGL extraction process upgrading the feed gas to 91.6% methane. Bottom row: composite curves (left) and approach temperature profile (right) for the NGL extraction process upgrading the feed gas to 95.6% methane. $UA_{\max} = 10.0$ MW/K and $n_{2p} = 20$ in all cases.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 9

Conclusions and future research directions

In this thesis, a new paradigm for simulating and optimizing natural gas liquefaction processes has been developed using models and methods grounded in recent advances in sensitivity analysis for nondifferentiable functions. In this final chapter, a summary of the work completed in the thesis project is presented and avenues for possible future work based on these contributions are proposed.

9.1 Project summary and conclusions

As the starting point of this project, simulation models that were amenable to rigorous equation solving and optimization methods needed to be developed for key unit operations in liquefaction processes. It was quickly observed that even in state-of-the-art process simulators, the unit models for MHEXs were inflexible and unreliable, frequently converging to nonphysical solutions due to temperature crossovers. Therefore, a nonsmooth modeling strategy for MHEXs was developed. This nonsmooth model represents a significant advance over other methods found in the literature and commercial software that generally only solve the MHEX energy balance, do not guarantee satisfaction the second law requirement of feasible heat transfer and do not incorporate information about the physical unit, such as the available heat

exchange area. In contrast, the new model consists of an energy balance in addition to two nonsmooth equations, one representing an extension of the classical pinch analysis algorithm for heat integration, the other an explicit dependence on the heat exchange area. Respectively, these new equations ensure that the model output is both thermodynamically and physically feasible. Furthermore, these equations allow for the specification of two additional parameters, such as the heat exchange area or the minimum temperature difference, as inputs to the model. This flexibility means that the model can be used not only in traditional rating calculations but also in process design problems. The recent development of AD and equation-solving techniques for nonsmooth functions means this model can be solved just as reliably and efficiently as a differentiable process model. A number of case studies were performed which demonstrated the effectiveness of the new approach, and this idea of using nonsmooth equations as a natural means of modeling complex thermodynamic and transport phenomena became a theme that was repeated to great effect throughout the project.

In the LNG processing applications relevant to this project, streams in MHEXs often undergo phase changes between their inlet and outlet. This represented a major challenge for heat integration calculations (such as those embedded in the MHEX model), since the assumption of constant heat-capacity streams is violated. To address this, nonsmooth models were formulated to account for the existence or nonexistence of phases in heat integration and physical property calculations. In contrast to many other approaches found in the literature, this formulation does not involve the solution of a difficult optimization problem since it avoids the use of either disjunctive or complementarity constraints. A nonsmooth model for phase equilibrium calculations was also developed. Such a model is necessary in simulation/optimization problems where many flash calculations must be performed for streams where the phase regime at the solution is not known *a priori*, which is often the case for complex LNG production processes. It was proven that this flash formulation follows directly from local minimization of the total molar Gibbs free energy of a mixture, and is therefore thermodynamically sound. The use of this flash model and the enhancements to

the base nonsmooth MHEX formulation allowed for the simulation and design of liquefaction processes such as the PRICO process, assuming the validity of simple thermodynamic models.

The nonsmooth method for flash calculations was effective for systems with simple thermodynamics (i.e. ideal gas EOS/Raoult's Law), but convergence difficulties were observed when attempting to solve systems with more complex behavior. Notably, accurate simulations of processes involving hydrocarbon mixtures, such as natural gas, required the use of cubic equations of state, and initial efforts to use the Peng-Robinson EOS in the existing PRICO process simulations were met with very limited success. In most commercial process simulators, flash calculations are handled using inside-out algorithms due to their robustness with respect to initialization and computational efficiency. However, these conventional inside-out algorithms fail if specified flash conditions imply a single-phase result, because the solution is constrained to obey equilibrium relationships which are only valid in the two-phase region. Therefore, modified inside-out algorithms were developed that use a nested nonsmooth equation system to relax equilibrium conditions when necessary, allowing reliable convergence to single-phase results while still benefiting from the reliability and efficiency of the original methods. These modified algorithms were found to be very effective for solving phase equilibrium problems for highly non-ideal mixtures in both single-phase and two-phase regimes, and even succeeded where commercial simulation software fails in predicting unusual thermodynamic phenomena such as retrograde condensation.

In most cases, the nonsmooth inside-out algorithms performed flawlessly; however, in certain instances that arose when trying to simulate process streams in liquefaction processes, flash calculations involving volatile mixtures at high pressures and/or temperatures often either converged to nonphysical solutions or failed. It was observed that the primary cause for this was the underlying thermodynamic model being queried for liquid or vapor properties of a mixture at conditions where one or both of the phases physically did not exist. To mitigate this, it was noted that ideas had been suggested in the literature for evaluating and post-processing the density calculated

by the EOS to promote convergence of the flash calculations to physical solutions through nonphysical extrapolations. Starting from these methods, new nonsmooth algorithms were developed for evaluating appropriate density values for mixtures at conditions where use of the EOS alone yielded unreasonable results. Unlike the other proposals in the literature, this new approach required only a reasonably simple algorithmic procedure and could be augmented with accurate sensitivity analysis through the use of nonsmooth operators and automatic generalized derivative computation. This strategy was initially tested for several commonly studied equations of state in hydrocarbon systems and later used successfully in liquefaction process simulation and optimization studies.

The key elements of the preceding work were then synthesized into a new framework for robustly simulating process flowsheets containing nondifferentiable models using exact sensitivity analysis for nonsmooth functions. Notably, this allowed for the inclusion of the nonsmooth inside-out algorithms and the nonsmooth density extrapolation techniques as external subroutines in flowsheeting problems, resulting in extremely reliable embedded flash calculations. This new nonsmooth flowsheeting strategy was capable of solving process simulation problems involving nonsmooth models more reliably and efficiently than the algorithms implemented in existing software such as Aspen Plus® or Aspen HYSYS®, and even allowed for the solution of problems that were beyond the capabilities of classical approaches. The nonsmooth flowsheet models generated by this strategy yielded extremely compact descriptions of complex processes that could be solved efficiently using nonsmooth equation-solving methods (even from poor initial guesses) to find solutions that were guaranteed to be thermodynamically feasible and physically realizable. Highly accurate simulations of the PRICO process using the Peng-Robinson EOS were then performed using these methods.

Finally, a methodology for the optimization of natural gas liquefaction processes was developed, in which the flowsheets formulated using the aforementioned nonsmooth framework could be efficiently and robustly optimized using an interior-point algorithm. The form of the MHEX models at the core of these flowhseets allowed for

the inclusion of constraints in the optimization formulation that led to solutions ensuring optimal MHEX conductance utilization in order to minimize process irreversibilities. Liquefaction process optimization problems could be solved reliably without the need for tedious initialization procedures even when highly accurate descriptions of the process stream cooling curves were requested. As examples of the efficacy of this strategy, the PRICO process, as well as two significantly more complex SMR liquefaction processes, were successfully optimized in the context of industrially-relevant operating scenarios.

The final product of these contributions is a collection of algorithms and models that may be used to design, simulate and optimize liquefaction processes efficiently and with extremely high accuracy and reliability. These models are highly compact in terms of the number of equations and variables seen by the convergence or optimization algorithm, less than 1% of the reported size of some of the discrete-continuous models found in the literature. Accordingly, they require far less complex machinery to solve, with particularly striking improvements in the region of convergence. Simple initialization procedures are all that are required, even for complex process optimization problems, and in many cases the initial guesses do not even need to be feasible in all of the problem constraints for a solution to be found successfully. This stands in contrast to the involved initialization procedures described in the literature and often conveniently neglected from run time reports. The automatic calculation of exact sensitivity information about the nonsmooth functions participating is integral to the robustness and efficiency of the algorithms developed in this thesis, allowing nonsmooth problems that would otherwise have been unmanageable to be solved essentially as readily as their smooth counterparts. In total, this work represents a significant advance in the state of the art for simulation and optimization of processes such as liquefaction operations, and is promising evidence that a new generation of process simulation and optimization technology based around these advances in nonsmooth modeling and sensitivity analysis would be of great value to the PSE community and industry.

The algorithms for robust flash calculations and density extrapolation were also

enormously successful in the context of this work. These algorithms are also not at all specific to natural gas liquefaction processes; they are readily applied to general VLE problems in any given process. As shown in Chapter 5 the algorithms handle activity coefficient models just as robustly as cubic equations of state. These algorithms are direct generalizations of the traditional two-phase algorithms and could be implemented into existing or newly developed process simulation or optimization software to great effect.

9.2 Opportunities for further research

As the successful optimization studies of Chapter 8 indicate, the methods developed in this thesis have not yet been pushed to their limit in terms of the complexity of the liquefaction processes that can be simulated and optimized. Therefore, the most obvious extension of this work is to continue exploring this space, working with even more complex liquefaction cycles featuring, for example, multiple mixed refrigerant streams and more MHEX units. More realistic treatment of the compressor trains could also be included, explicitly modeling multistage compression with interstage cooling to get a truer sense of the process efficiency. Optimization of processes such as the Air Products® Dual Mixed Refrigerant process would be of much industrial interest and relevance. Beyond this, more complex models of the MHEX internals could also be considered in future, replacing the specifications on UA with specifications on quantities more relevant to detailed design.

The development of a reliable local optimization algorithm for noncovex, constrained nonsmooth functions that can effectively utilize exact generalized derivative information would also be of substantial benefit to the present work and should be a focus of future research efforts in the area. The necessity of relaxing the dual tolerance of the optimizer in Example 8.4 suggests that the weakest link in the current strategy is the reliance on an optimization algorithm intended for use with twice-differentiable functions. However, as discussed, algorithms specifically designed for nonsmooth optimization are paradoxically far less robust when used in the studies shown herein.

Available codes for nonsmooth local optimization algorithms are almost exclusively based around the use of Clarke generalized gradient information, which until recently have been challenging to compute for all but toy problems due to the lack of sharp calculus rules for nondifferentiable functions. These algorithms that purport to work for nonsmooth nonconvex functions then attempt to use aggregated Clarke gradient information to converge to local minima with limited success. However, the ability to calculate useful sensitivity information for general nonsmooth functions could replace the current reliance on the Clarke gradient information and open new avenues for exploration in the field of nonsmooth optimization. As alluded to in Chapter 8 however, there are likely to be certain challenges for practical implementations of such algorithms, such as determining termination criteria that can be tractably verified.

Preliminary attempts to apply deterministic global optimization techniques to liquefaction process optimization have been met with limited success (see Appendix C). A positive observation is that the compactness of the MHEX model compared to other heat integration-based formulations makes it a natural choice for algorithms that scale exponentially with the problem size. However, the inclusion of realistic thermodynamic models appears problematic. The bounding and relaxation methods needed to underapproximate the objective and constraint functions in a branch-and-bound algorithm struggle or fail to generate useful information for even individual flash calculations, and the flowsheet models of Chapter 8 require hundreds of nonideal phase equilibrium calculations, which themselves rely on additional subroutines for density calculation and possible extrapolation. While the methods of Scott *et al.*[112] and Stuber *et al.*[121] theoretically prescribe methods for calculating and propagating relaxations for implicit functions, these methods have never been applied to problems with the level of complexity of those under consideration herein. It therefore seems that improved numerical strategies for generating bounding and relaxation information for implicit functions, especially in instances with deeply nested submodels, need to be proposed and tested. The need for robust local optimization techniques to generate strong upper bounding information for deterministic global optimization methods also necessitates the further development of efficient nonsmooth optimization codes,

as described previously.

In terms of further developing the robust models for the single-stage flash problem described in Chapters 5 and 6, a clear extension is the derivation of algorithms for distillation column models using the nonsmooth inside-out approach. Such models would allow for convergence to solutions with either dry or flooded trays as a result of (suboptimal) column operating conditions. The articles by e.g. Boston and Sullivan[17] and Russell[106] describe inside-out strategies for multistage separations that could serve as the starting point for this investigation. It would also be interesting to see if the new approach to density extrapolation can be applied either directly or with minor modifications to more exotic equations of state such as the statistical associating fluid theory (SAFT) model and its variants or the Helmholtz-explicit models that are often used when extremely accurate predictions of fluid properties are required.

The nonsmooth reformulation of the vapor-liquid equilibrium problem could also be modified to solve more challenging multiphase equilibrium problems. Initial efforts have already extended the model to the liquid-liquid equilibrium (LLE) case,[94] but the other common case of vapor-liquid-liquid equilibrium (VLLE) problems has not yet been explored. An approach to VLLE problems might build upon existing work by nesting a nonsmooth LLE formulation into the liquid argument of the nonsmooth VLE formulation developed in this work, if appropriate ordering of the two liquid phases and the vapor phase can be established. Additionally, issues with convergence to both the trivial solution and nontrivial but suboptimal (unstable) solutions is well known to be a major challenge in LLE and VLLE calculations, so it is plausible that a variant on the density extrapolation procedure could be developed for this problem. However, reliably differentiating acceptable properties for the two liquid phases could prove to be a challenge in implementing this approach.

Appendix A

Notation

This appendix provides a reference for the most commonly-used abbreviations and variable names in this thesis. Unless otherwise stated, variables and abbreviations should be assumed to have the meanings listed here. Note that some roman letters, particularly $A - G$, are also sometimes used to represent parameters in correlation or surrogate models, e.g. in Chapters 5, 6 and Appendix B, and should be read in context.

A.1 Abbreviations

AD	=	automatic differentiation
BWRS	=	Benedict-Webb-Rubin-Starling
CPU	=	central processing unit
DIPPR	=	Design Institute for Physical Properties
DMR	=	dual mixed refrigerant
EO	=	equation-oriented
EOS	=	equation of state
HPR	=	high pressure refrigerant
IPOPT	=	interior-point optimizer
LNG	=	liquefied natural gas
LD	=	lexicographic directional

LP	=	linear program
LPR	=	low pressure refrigerant
KKT	=	Karush-Kuhn-Tucker
MHEX	=	multistream heat exchanger
MINLP	=	mixed-integer nonlinear program
MPCC	=	mathematical program with complementarity constraints
MR	=	mixed refrigerant
NG	=	natural gas
NGL	=	natural gas liquids
NRTL	=	nonrandom two-liquid
PQ	=	pressure-heat duty (flash calculation)
PR	=	Peng-Robinson
PRICO	=	Poly-Refrigerated Integrated Cycle Operations (process)
PS	=	pressure-entropy (flash calculation)
PSE	=	process systems engineering
PT	=	pressure-temperature (flash calculation)
SM	=	sequential-modular
SMR	=	single mixed refrigerant
SQP	=	sequential-quadratic programming
SRK	=	Soave-Redlich-Kwong
VLE	=	vapor-liquid equilibrium

A.2 Variables

a, b	=	cubic EOS constants
A	=	heat transfer area (m^2)
AP	=	classical pinch location function (W)
C	=	index set of cold streams

C_p = heat capacity ($\mathrm{J{\cdot}mol^{-1}K^{-1}}$)

C^1 = class of continuously-differentiable functions

$\mathbf{D}$ = differential operator

$\mathbf{D}_{\mathrm{L}}$ = lexicographic differential operator

$\mathbf{e}$ = cartesian basis vector

EBP = extended pinch location function (W)

F = total feed molar flowrate ($\mathrm{mol{\cdot}s^{-1}}$)

F_{C_p} = heat capacity flow rate of hot stream ($\mathrm{W{\cdot}K^{-1}}$)

f_{C_p} = heat capacity flow rate of cold stream ($\mathrm{W{\cdot}K^{-1}}$)

f = component feed molar flowrate ($\mathrm{mol{\cdot}s^{-1}}$)

G = total Gibbs free energy (J)

$\bar{G}$ = partial molar Gibbs free energy ($\mathrm{J{\cdot}mol^{-1}}$)

$\mathbf{G}$ = element of generalized derivative

H = index set of hot streams

h = specific enthalpy ($\mathrm{J{\cdot}mol^{-1}}$)

Δh = specific enthalpy departure ($\mathrm{J{\cdot}mol^{-1}}$)

$I_{\mathbf{f}}^{\mathrm{ess}}(\mathbf{x})$ = set of essentially active indices of PC^1 function $\mathbf{f}$ at $\mathbf{x}$

$\mathbf{I}$ = identity matrix

k = equilibrium coefficient

k_b = reference equilibrium coefficient

K = index set of nondifferentiable points in composite curves

L = liquid phase molar flowrate ($\mathrm{mol{\cdot}s^{-1}}$)

l = component liquid phase molar flowrate ($\mathrm{mol{\cdot}s^{-1}}$)

m = number of residuals kept in memory during Anderson acceleration

$\mathbf{M}$ = directions matrix for lexicographic differentiation

n_c = number of components

P = absolute pressure (Pa)

P^{sat} = vapor pressure (Pa)

$\mathcal{P}$ = index set of pinch candidates

$\mathbf{p}$ = parameters in a model

PC^1 = class of piecewise-differentiable functions

Q = heat flowrate (W)

R = inner loop iteration variable

R_G = gas constant ($\mathrm{J} \cdot \mathrm{mol}^{-1} \cdot \mathrm{K}^{-1}$)

s = specific entropy ($\mathrm{J{\cdot}mol^{-1}K^{-1}}$)

S = total entropy ($\mathrm{J{\cdot}K^{-1}}$)

T = temperature (for a hot stream if used along with next entry) (K)

t = temperature of cold stream (K)

ΔT_{LM} = log-mean temperature difference (K)

ΔT_{min} = minimum approach temperature between hot and cold streams (K)

U = overall heat transfer coefficient ($\mathrm{W{\cdot}m^{-2} \cdot K^{-1}}$)

UA = heat exchanger conductance ($\mathrm{W{\cdot}K^{-1}}$)

$\mathbf{u}$ = volatility parameters vector

v = component vapor phase molar flowrate ($\mathrm{mol{\cdot}s^{-1}}$)

V = total vapor phase molar flowrate ($\mathrm{mol{\cdot}s^{-1}}$)

$\mathbf{x}$ = unknown variables in a model

$\mathbf{x}_L$ = liquid phase mole fraction vector

$\mathbf{y}_V$ = vapor phase mole fraction vector

$\mathbf{z}_F$ = feed stream mole fraction vector

Y = indicator variable in disjunctive formulation

Z = compressibility factor

Greek symbols

α = vapor fraction

β = variable used to relax equilibrium constraints

γ = variable in LP objective of the LP-Newton method

∂_{B} = Bouligand subdifferential

∂_{C} = Clarke Jacobian

∂_{L} = Lexicographic subdifferential

Δ = difference

ε = small parameter, usually termination tolerance

ν = molar volume ($m^3 \cdot mol^{-1}$)

$\Psi, \boldsymbol{\Psi}$ = inner loop error function in flash algorithms

ρ = molar density ($mol \cdot m^{-3}$)

$\boldsymbol{\upsilon}$ = outer loop iteration variable vector in flash algorithms

ϕ = fugacity coefficient

ω = Pitzer acentric factor

Ω = outer loop error function in flash algorithms

Subscripts

2p = two-phase regime

BP = bubble point

c = critical property

DP = dew point

F = feed property

i, j = stream or component indices

(i) = ith column of a matrix

$(1:i)$ = first i columns of a matrix

L = liquid phase property

mc = mechanical critical property

mix = mixture property

sub = subcooled liquid regime

sup = superheated vapor regime

V = vapor phase property

Superscripts

EOS = EOS model

extrap = extrapolation model

hi (or U,UB) = upper bound

id = ideal property

IN/OUT = inlet/outlet of physical process stream

in/out = inlet/outlet of heat integration stream

(k) = iteration counter

lo (or L,LB)	=	lower bound
p	=	pinch candidate
ref	=	reference state
0	=	initial (constant) value
$'$	=	distinct from unprimed value
$\hat{}$	=	calculated value
*	=	solution value

Appendix B

Thermophysical property models

This Appendix gives further details regarding the forms of the major equations of state used in Chapters 4-8 of this thesis. Numerical values for all parameters used in this work are available from the databanks of Aspen Plus v8.4.[5]

B.1 Ideal model

The ideal property model used in the examples in this work, which is heavily based on the IDEAL property method from Aspen Plus v8.4,[5] is detailed here. The ideal gas EOS is simply given by:

$$P = \rho R_G T. \tag{B.1}$$

The ideal gas heat capacity of a pure component i is given by the Aly-Lee model, also called DIPPR Equation 107:[1]

$$C_{p,i}^{\mathrm{id}} = C_{1,i} + C_{2,i}\left(\frac{C_{3,i}/T}{\sinh(C_{3,i}/T)}\right)^2 + C_{4,i}\left(\frac{C_{5,i}/T}{\cosh(C_{5,i}/T)}\right)^2, \tag{B.2}$$

where $C_{1,i}, \ldots, C_{5,i}$ are species dependent constants. The ideal gas enthalpy of pure component i is then found by calculating:

$$h_{V,i}^{\mathrm{id}} = \int_{T^{\mathrm{ref}}}^{T} C_{p,i} dT, \tag{B.3}$$

which may be expressed in closed form as:

$$h_{V,i}^{\text{id}} = C_{1,i}(T - T^{\text{ref}}) + C_{2,i}C_{3,i}\left(\coth(C_{3,i}/T) - \coth(C_{3,i}/T^{\text{ref}})\right) - C_{4,i}C_{5,i}\left(\tanh(C_{5,i}/T) - \tanh(C_{5,i}/T^{\text{ref}})\right), \tag{B.4}$$

where T^{ref} is a reference temperature that is assumed to be 298.15 K throughout this thesis. The ideal liquid enthalpy of component i is given by:

$$h_{L,i}^{\text{id}} = h_{V,i}^{\text{id}} - \Delta h_{\text{vap},i}, \tag{B.5}$$

with the heat of vaporization calculated with the nonsmooth reformulation of DIPPR Equation 106 given in Chapter 4 as Equation (4.25). Ideal mixing rules are used to calculate the enthalpy of multicomponent streams:

$$h_L^{\text{id}} = \sum_{i=1}^{n_c} x_{L,i} h_{L,i}^{\text{id}}, \tag{B.6}$$

$$h_V^{\text{id}} = \sum_{i=1}^{n_c} y_{V,i} h_{V,i}^{\text{id}}. \tag{B.7}$$

The ideal entropy of a (liquid/vapor) mixture is given by:

$$s_G^{\text{id}} = \sum_{i=1}^{n_c} z_i \int_{T^{\text{ref}}}^{T} \frac{C_{p,i}}{T} dT - R_G \ln \frac{P}{P^{\text{ref}}} - R_G \sum_{i=1}^{n_c} z_i \ln(z_i), \tag{B.8}$$

where components of $\mathbf{z}$ are placeholders for components of either $\mathbf{x}_L$ or $\mathbf{y}_V$ depending on the desired phase, R_G is the universal gas constant, P^{ref} is a reference pressure assumed to be 0.101325 MPa in this thesis and the integral term may be expressed

in closed form as follows:

$$\int_{T^{\text{ref}}}^{T} \frac{C_{p,i}}{T} dT = C_{1,i} \ln(T/T^{\text{ref}}) + C_{2,i}\left(\frac{C_{3,i}}{T}\coth(C_{3,i}/T) - \ln\left(\sinh(C_{3,i}/T)\right)\right.$$
$$\left. - \frac{C_{3,i}}{T^{\text{ref}}}\coth(C_{3,i}/T^{\text{ref}}) + \ln\left(\sinh(C_{3,i}/T^{\text{ref}})\right)\right)$$
$$- C_{4,i}\left(\frac{C_{5,i}}{T}\tanh(C_{5,i}/T) - \ln\left(\cosh(C_{5,i}/T)\right)\right.$$
$$\left. - \frac{C_{5,i}}{T^{\text{ref}}}\tanh(C_{5,i}/T^{\text{ref}}) + \ln\left(\cosh(C_{5,i}/T^{\text{ref}})\right)\right). \tag{B.9}$$

Equilibrium coefficients for each component i are calculated assuming Raoult's Law holds, so that:

$$k_i^{\text{id}}(T,P) = \frac{P_i^{\text{sat}}(T)}{P}, \tag{B.10}$$

where P_i^{sat} is the vapor pressure of component i and is given by the extended Antoine Equation:

$$P_i^{\text{sat}} = \exp\left(D_{1,i} + D_{2,i}/T + D_{3,i}\ln(T) + D_{4,i}T^{D_{5,i}}\right), \tag{B.11}$$

where $D_{1,i}, \ldots, D_{5,i}$ are species dependent constants. Bubble points and dew points are found be solving the following equations for T_{BP} and T_{DP}, respectively :

$$P_{\text{BP}} = \sum_{i=1}^{n_c} x_{L,i} P_i^{\text{sat}}(T_{\text{BP}}), \tag{B.12}$$

$$P_{\text{DP}} = \frac{1}{\sum_{i=1}^{n_c} \frac{y_{V,i}}{P_i^{\text{sat}}(T_{\text{DP}})}}. \tag{B.13}$$

Isentropic expansion and compression of a single-component ideal gas is governed by the equation:

$$\frac{T^{\text{out}}}{T^{\text{in}}} = \left(\frac{P^{\text{out}}}{P^{\text{in}}}\right)^{\frac{\gamma-1}{\gamma}}, \tag{B.14}$$

where $\gamma \equiv C_p^{\text{id}}/(C_p^{\text{id}} - R_G)$ is the heat capacity ratio. Pumps for single-component ideal liquids are also assumed to be isentropic and operate on an incompressible fluid so that:

$$h_L^{\text{id}}(T^{\text{out}}) = h_L^{\text{id}}(T^{\text{in}}) + \frac{P^{\text{out}} - P^{\text{in}}}{\rho_L}, \tag{B.15}$$

where ρ_L is the liquid's density. Isentropic pressure-changing operations involving ideal gases mixtures may be modeled by equating the inlet and outlet entropy values obtained from Equation (B.8) and solving for the outlet temperature.

B.2 Peng-Robinson EOS

The Peng-Robinson cubic EOS written in pressure-explicit form and in terms of density is as follows:

$$P = \frac{\rho R_G T}{1 - b\rho} - \frac{a\rho^2}{1 + 2b\rho - b^2\rho^2}, \tag{B.16}$$

where a and b are mixture parameters that are calculated from the pure component parameters through mixing rules. Note that, in the following, $\mathbf{z}$ should be replaced by $\mathbf{x}_L$ when calculating liquid phase properties and by $\mathbf{y}_V$ when calculating vapor phase properties. In standard practice (and this thesis), a quadratic mixing rule is used for a and a linear mixing rule is used for b, that is:

$$a \equiv \sum_{i=1}^{n_c} \sum_{j=1}^{n_c} z_i z_j \sqrt{a_i a_j}(1 - k_{i,j}), \tag{B.17}$$

$$b \equiv \sum_{i=1}^{n_c} z_i b_i, \tag{B.18}$$

where $k_{i,j}$ is the binary interaction parameter between species i and species j, and, for each component $i = 1, \ldots, n_c$:

$$a_i = 0.45724 \frac{\alpha_i R_G^2 T_{\mathrm{c},i}^2}{P_{\mathrm{c},i}}, \tag{B.19}$$

$$\alpha_i = \left[1.0 + \left(0.37464 + 1.54226\omega_i - 0.26992\omega_i^2\right)\left(1 - \sqrt{\tfrac{T}{T_{\mathrm{c},i}}}\right)\right]^2, \tag{B.20}$$

$$b_i = 0.07780 \frac{R_G T_{\mathrm{c},i}}{P_{\mathrm{c},i}}, \tag{B.21}$$

where ω_i is the Pitzer acentric factor of component i. Note that these definitions show that a is a function of temperature and composition and that b is a function of composition. The partial derivative of pressure with respect to density for this EOS

is given by:

$$P_\rho = \frac{R_G T}{(b\rho - 1)^2} - \frac{2a\rho(b\rho + 1)}{(b^2\rho^2 - 2b\rho - 1)^2}. \tag{B.22}$$

Assuming now that the EOS has been solved for the density value of a chosen phase (see Chapter 6), the enthalpy departure function for that phase in terms of the compressibility factor $Z \equiv P/(\rho R_G T)$ and parameters $A \equiv (aP)/(R_G^2 T^2)$ and $B \equiv (bP)/(R_G T)$ is as follows:

$$\Delta h = R_G T(Z-1) - \frac{a - T\frac{\partial a}{\partial T}}{2\sqrt{2}b} \ln\left(\frac{Z + (1+\sqrt{2})B}{Z + (1-\sqrt{2})B}\right), \tag{B.23}$$

and the entropy departure function is:

$$\Delta s = R_G \ln(Z - B) + \frac{\frac{\partial a}{\partial T}}{2\sqrt{2}b} \ln\left(\frac{Z + (1+\sqrt{2})B}{Z + (1-\sqrt{2})B}\right). \tag{B.24}$$

The fugacity coefficient of mixture component i is given by:

$$\begin{aligned} \ln\phi_i = &\frac{A}{2\sqrt{2}B}\left(\frac{2\sum_{j=1}^{n_c} z_j\sqrt{a_i a_j}(1-k_{i,j})}{a} - \frac{b_i}{b}\right)\ln\left(\frac{Z + (1+\sqrt{2})B}{Z + (1-\sqrt{2})B}\right) \\ &+ \frac{b_i}{b}(Z-1) - \ln(Z-B), \end{aligned} \tag{B.25}$$

and corresponding equilibrium coefficients are given by:

$$k_i(T, P, \mathbf{x}_L, \mathbf{y}_V) = \frac{\phi_i^L(T, P, \mathbf{x})}{\phi_i^V(T, P, \mathbf{y})}. \tag{B.26}$$

B.3 Other equations of state

Several other equations of state are used briefly in examples throughout Chapters 5 and 6.

Redlich-Kwong and Soave-Redlich-Kwong

The Redlich-Kwong cubic EOS expressed in terms of density is given by:

$$P = \frac{\rho R_G T}{1 - b\rho} - \frac{a\rho^2}{\sqrt{T}(1 + b\rho)}, \tag{B.27}$$

where mixture parameters a and b are calculated from the mixing rules in Equations (B.17) and (B.18) with

$$a_i = 0.42748 \frac{R_G^2 T_{\mathrm{c},i}^{5/2}}{P_{\mathrm{c},i}},$$
$$b_i = 0.08664 \frac{R_G T_{\mathrm{c},i}}{P_{\mathrm{c},i}}.$$

Note that there is no temperature dependence in the species a_i values for this EOS. Soave's modification of this EOS is given by:

$$P = \frac{\rho R_G T}{1 - b\rho} - \frac{a\rho^2}{(1 + b\rho)}, \tag{B.28}$$

where mixture parameters a and b are calculated from the mixing rules in Equations (B.17) and (B.18) with

$$a_i = 0.42748 \frac{\alpha_i R_G^2 T_{\mathrm{c},i}^2}{P_{\mathrm{c},i}},$$
$$\alpha_i = \left[1.0 + \left(0.48508 + 1.55171\omega_i - 0.15613\omega_i^2\right)\left(1 - \sqrt{\frac{T}{T_{\mathrm{c},i}}}\right)\right]^2,$$
$$b_i = 0.08664 \frac{R_G T_{\mathrm{c},i}}{P_{\mathrm{c},i}},$$

where a temperature dependence is again embedded in the EOS parameters, as in the Peng-Robinson EOS. Departure functions and fugacity coefficients may be obtained from these equations of state once they have been solved for density (or compressibility) to yield equations similar to that for the Peng-Robinson EOS.

Benedict-Webb-Rubin-Starling

The BWRS virial EOS written in pressure-explicit form and in terms of density is as follows:

$$\begin{aligned} P = \rho R_G T + \left(BR_G T - A - \tfrac{C}{T^2} + \tfrac{D}{T^3} - \tfrac{E}{T^4}\right)\rho^2 + \left(bR_G T - a - \tfrac{d}{T}\right)\rho^3 \\ + \alpha\left(a + \tfrac{d}{T}\right)\rho^6 + \tfrac{c\rho^3}{T^2}(1+\gamma\rho^2)\exp(-\gamma\rho^2), \end{aligned} \tag{B.29}$$

where the 11 mixture-dependent parameters are obtained from the pure-component parameters by the following identities:

$$B = \sum_{i=1}^{n_c} z_i B_i, \qquad b = \left(\sum_{i=1}^{n_c} z_i b_i^{1/3}\right)^3,$$

$$A = \sum_{i=1}^{n_c}\sum_{j=1}^{n_c} z_i z_j \sqrt{A_i A_j}(1-k_{i,j}), \qquad a = \left(\sum_{i=1}^{n_c} z_i a_i^{1/3}\right)^3,$$

$$C = \sum_{i=1}^{n_c}\sum_{j=1}^{n_c} z_i z_j \sqrt{C_i C_j}(1-k_{i,j})^3, \qquad c = \left(\sum_{i=1}^{n_c} z_i c_i^{1/3}\right)^3,$$

$$D = \sum_{i=1}^{n_c}\sum_{j=1}^{n_c} z_i z_j \sqrt{D_i D_j}(1-k_{i,j})^4, \qquad d = \left(\sum_{i=1}^{n_c} z_i d_i^{1/3}\right)^3,$$

$$E = \sum_{i=1}^{n_c}\sum_{j=1}^{n_c} z_i z_j \sqrt{E_i E_j}(1-k_{i,j})^5, \qquad \alpha = \left(\sum_{i=1}^{n_c} z_i \alpha_i^{1/3}\right)^3,$$

$$\gamma = \left(\sum_{i=1}^{n_c} z_i \gamma_i^{1/2}\right)^2.$$

The pure component parameter values and binary interaction parameter values ($k_{i,j}$) may all be obtained from a database such as Aspen Plus.[5] The partial derivative of pressure with respect to density for this EOS is given by:

$$\begin{aligned} P_\rho = R_G T + 2\left(BR_G T - A - \tfrac{C}{T^2} + \tfrac{D}{T^3} - \tfrac{E}{T^4}\right)\rho + 3\left(bR_G T - a - \tfrac{d}{T}\right)\rho^2 \\ + 6\alpha\left(a + \tfrac{d}{T}\right)\rho^5 + \tfrac{c\rho^2}{T^2}(3(1+\gamma\rho^2) - 2\gamma^2\rho^4)\exp(-\gamma\rho^2). \end{aligned} \tag{B.30}$$

Activity coefficient models

In Chapter 5, some nonideal liquid phases are modeled with either the NRTL or the Wilson activity coefficient model. The NRTL model defines activity coefficients as follows:

$$\ln \gamma_i = \frac{\sum_{j=1}^{n_c} x_{L,j}\tau_{ji}G_{ji}}{\sum_{k=1}^{n_c} x_{L,k}G_{ki}} + \sum_{j=1}^{n_c} \frac{x_{L,j}G_{ij}}{\sum_{k=1}^{n_c} x_{L,k}G_{kj}} \left(\tau_{ij} - \frac{\sum_{m=1}^{n_c} x_{L,m}\tau_{mji}G_{mj}}{\sum_{k=1}^{n_c} x_{L,k}G_{kj}} \right), \quad \text{(B.31)}$$

where:

$$\tau_{ij} = A_{ij} + \frac{B_{ij}}{T} + C_{ij}\ln(T) + D_{ij}T,$$
$$G_{ij} = \exp(-\tau_{ij}(E_{ij} + F_{ij}(T - T^{\text{ref}}))),$$

and $A_{ij}, \ldots, F_{ij}$ are asymmetrical binary interaction parameters.

The Wilson model defines activity coefficients as follows:

$$\ln \gamma_i = 1 - \ln\left(\sum_{j=1}^{n_c} G_{ij}x_{L,j}\right) - \sum_{j=1}^{n_c} \frac{G_{ji}x_{L,j}}{\sum_{k=1}^{n_c} G_{jk}x_{L,k}}, \quad \text{(B.32)}$$

where:

$$\ln G_{ij} = A_{ij} + \frac{B_{ij}}{T} + C_{ij}\ln(T) + D_{ij}T + \frac{E_{ij}}{T^2},$$

where $A_{ij}, \ldots, E_{ij}$ are asymmetrical binary interaction parameters (distinct from the parameters in the NRTL model). In both cases, the excess enthalpy of mixing can be calculated using the identity:

$$h^E \equiv -R_G T^2 \sum_{i=1}^{n_c} \frac{x_{L,i}}{\gamma_i}\frac{\partial \gamma_i}{\partial T}, \quad \text{(B.33)}$$

and equilibrium coefficients for component i are given by:

$$k_i(T, P\mathbf{x}_L, \mathbf{y}_V) = \frac{\gamma_i(T, \mathbf{x}_L)P_i^{\text{sat}}(T)}{\phi_i^V(T, P, \mathbf{y})P}. \quad \text{(B.34)}$$

Appendix C

Prospects for global optimization

This Appendix assumes that the reader is familiar with the fundamentals and nomenclature of the branch-and-bound algorithm for global optimization, as well as methods for bounding the range of factorable functions, including interval analysis, McCormick relaxations and the more recently-developed differentiable McCormick relaxations.[66]

In this first section, it is shown through example how the nonsmooth MHEX model can be used effectively in a branch-and-bound algorithm in which convex underestimators are calculated using the differentiable multivariate relaxations developed by Khan *et al.*[66] As noted previously in this thesis, global optimization problems involving MHEXs are often formulated as an MINLP due to the pinch constraints, which results in the addition of a large number of constraints and binary variables to the problem. Direct use of a nonsmooth formulation avoids this undesirable increase in complexity.

C.1 Global optimization of the multistream heat exchanger model

Consider the optimization of the offshore process concept for LNG production featuring compression and expansion of process streams from Example 3.5. The flowrates, temperature levels and pressure levels of the natural gas streams and carbon dioxide

streams in the process are once again considered fixed based on the preliminary design work described in Wechsung *et al.*[141] As in Example 3.5, in place of using physical property calculations and phase detection mechanisms in the simulation, several of the physical process streams are instead split into substreams of constant heat capacity to approximate the real temperature-enthalpy relationships of their cooling curves. Table C.1 details the values of the fixed process parameters, as well as the unknown stream variables, which are the decision variables in the optimization problem.

Stream	F, f [kg/s]	C_p [kJ/kg]	$T^{\text{in}}, t^{\text{in}}$ [K]	$T^{\text{out}}, T^{\text{out}}$ [K]	P [MPa]
H1 (NG-2–NG-4)	1.00	3.46	319.80	265.15	10.0
H2 (NG-2–NG-4)	1.00	5.14	265.15	197.35	10.0
H3 (NG-2–NG-4)	1.00	3.51	197.35	104.75	10.0
H4 (N_2-8–N_2-9)	F_{N2}	1.15	$T^{\text{in}}_{\text{H4}}$	$T^{\text{out}}_{\text{H4}}$	P_{H4}
C1 (CO_2-2–CO_2-3)	2.46	2.11	221.12	252.55	6.0
C2 (CO_2-2–CO_2-3)	2.46	2.48	252.55	293.15	6.0
C3 (N_2-2–N_2-4)	F_{N2}	2.48	103.45	171.05	10.0
C4 (N_2-2–N_2-4)	F_{N2}	1.80	171.05	218.75	10.0
C5 (N_2-2–N_2-4)	F_{N2}	1.18	218.75	$T^{\text{out}}_{\text{C5}}$	10.0
C6 (N_2-5–N_2-7)	F_{N2}	1.07	$t^{\text{in}}_{\text{C6}}$	$t^{\text{out}}_{\text{C6}}$	P_{C6}
C7 (N_2-10–N_2-12)	F_{N2}	1.04	$t^{\text{in}}_{\text{C7}}$	$t^{\text{out}}_{\text{C7}}$	0.1

Table C.1: Data and unknowns for the global optimization of the offshore LNG production process.

In Wechsung *et al.*,[141] this simultaneous flowsheet simulation and heat integration problem was modeled as an MINLP using the formulation from Yee and Grossmann.[46] This previous work also did not distinguish between the two physical heat exchangers in the flowsheet, and instead considered all streams as being part of a single heat integration problem. For consistency of results, this approach is taken here as well. However, the problem is instead modeled using the framework recently developed in Chapter 3, which can be extended to allow for the presence of utilities as follows:

$$Q_H + \sum_{i \in H} F_{C_p,i}(T_i^{\text{in}} - T_i^{\text{out}}) = Q_C + \sum_{j \in C} f_{C_p,j}(t_j^{\text{out}} - t_j^{\text{in}}), \tag{C.1}$$

$$\min_{p \in \mathcal{P}} \{EBP_C^p - EBP_H^p\} = -Q_c, \tag{C.2}$$

where Q_H is the heating utility required by the process and Q_C is the cooling utility required by the process.

As in the article by Wechsung *et al.*,[141] the process is optimized subject to progressively more stringent sets of constraints that limit the amounts of external utilities and power which the process is allowed to consume. One hot and one cold utility are assumed to be available at 383.15 and 93.15 K, respectively. The objective function in all cases is to minimize the required nitrogen flowrate. The same four cases are studied here:

- Case I: minimize F_{N2},
- Case II: minimize F_{N2} such that $\dot{W}_{\mathrm{net}} \leq 0$,
- Case III: minimize F_{N2} such that $Q_C = 0$ and $\dot{W}_{\mathrm{net}} \leq 0$,
- Case IV: minimize F_{N2} such that $Q_C = Q_H = 0$ and $\dot{W}_{\mathrm{net}} \leq 0$,

where $\dot{W}_{\mathrm{net}}$ is the net power required by the process. In all cases, bounds on temperature variables are given by the utility temperatures. In addition, some constraints on the nitrogen stream were also determined by the preliminary design, as follows: the flowrate of nitrogen is allowed to vary between 0.0 and 2.0 kg/s, the pressure of stream C6 is bounded between 0.3 and 1.0 MPa, and the pressure of stream H4 is constrained between 1.0 and 3.5 MPa.

All cases were first resolved in GAMS v24.5 using BARON v15.9[108] with CPLEX and SNOPT as the LP and NLP solvers, respectively, on an Intel Xeon E5-1650 v2 workstation using six cores at 3.50 GHz and 12 GB RAM under Linux v14.04. Since BARON cannot directly model multivariate max and min functions, the MINLP formulation was used. The model consists of 1244 constraints, 363 binary variables and 173 continuous variables for Case IV, as an example (in this formulation, the number of binary variables is equal to $3n_s^2$, where n_s is the number of streams, which here is 11). The relative termination tolerance in GAMS was set to 10^{-4}. The absolute termination tolerance in GAMS, as well as all feasibility tolerances and SNOPT or CPLEX tolerances, were left at their default values. An optimal solution

with objective value within the optimality tolerance of that reported in Wechsung *et al.*[141] in all cases. The first three rows of Table C.2 summarize the computational results for each of the four cases. It is clear when comparing the present solution times to those reported in Wechsung *et al.*[141] that BARON's performance has improved significantly on this problem in newer versions.

The four cases were then solved using a basic branch-and-bound code implemented in C++ using the differentiable McCormick relaxations developed in Khan *et al*[66] to construct continuously differentiable convex relaxations. Note that differentiability must be defined in the sense of Whitney[142] on closed sets (since the relaxations are necessarily constructed on boxes). The Whitney-C^1 relaxations were minimized using SNOPT v7.2[37] to provide lower bounds on the optimal solution. Using the nonsmooth modeling approach, Case IV requires 9 constraints and 10 continuous variables, a significant reduction compared to the MINLP model. Upper bounds on the solution value were obtained by first finding an approximate solution to the nonsmooth problem with SNOPT in derivative-free mode, and then passing the SNOPT solution to the bundle solver MPBNGC v2.0.[77] The bundle solver was allowed to take a maximum of five iterations to attempt to improve the upper bound before termination. This upper bounding strategy worked well in practice, as the global solution for each of the four cases was found very early in the branch tree. Optimality and feasibility tolerances were set identical to those from GAMS for fair comparison. Branching was performed such that the current box was bisected along the largest current width relative to the original box dimensions, and nodes were selected according to the lowest lower bound heuristic. An optimal solution with objective value within the optimality tolerance of that reported in Wechsung *et al.*[141] was found for all cases. The second three rows of Table C.2 summarize the computational results for each of these numerical tests.

Noting that BARON was solving the problems more efficiently than the in-house software largely because of the use of range reduction techniques, these features of BARON were turned off completely. With this restriction, BARON was not able to solve any problem except Case I in fewer than 100 hours. For a better comparison, the

Solution method	Statistic	Case I	Case II	Case III	Case IV
BARON v15.9 (all features)	Time (s)	0.52	2.04	15.68	31.91
	Iterations	2	6	102	1099
	Max nodes	2	3	22	74
Whitney-C^1 relaxations (no range reduction)	Time (s)	0.0039	3511.05	831.90	363.96
	Iterations	1	339,207	146,665	59,597
	Max nodes	1	33,343	13,126	6,442
BARON v15.9 (DBBT only)	Time (s)	0.38	783.42	6169.10	5989.74
	Iterations	1	45,477	536,407	452,878
	Max nodes	1	2,551	23,287	15,499
Whitney-C^1 relaxations (DBBT only)	Time (s)	0.0040	141.22	35.67	56.61
	Iterations	1	15,851	3,715	7,647
	Max nodes	1	3,561	431	771
Whitney-C^1 relaxations (DBBT & obj. fun. cuts)	Time (s)	0.0040	93.99	27.52	35.91
	Iterations	1	11,229	1,621	2,095
	Max nodes	1	2,702	317	488

Table C.2: Computational results for the global optimization case study.

cases were solved again, this time allowing BARON to only use dual multiplier-based bounds tightening (DBBT, or option `MDo = 1` in BARON) as described by Ryoo and Sahinidis[107] for range reduction (all preprocessing was also left active). DBBT was also implemented and used in the in-house `C++` code using multiplier values calculated by SNOPT in the lower bounding procedure. The results from these experiments are shown in Table C.2. Finally, the branch-and-bound code was also augmented with objective function value cuts in addition to DBBT. The cases were resolved in this framework and the results are also shown in Table C.2. The CPU cost of each iteration averaged between Cases II, III and IV for each method is shown in Table C.3.

Even without employing range reduction, use of the differentiable relaxations provides tight lower bounds for the nonsmooth model and reasonable solution times for each of the four cases, especially if compared with the original results from Wechsung *et al.*[141] or even the current version of BARON running only the standard branch-and-bound algorithm. Case II is somewhat of an exception, as the nonsmooth model

Solution method	Average time per iteration (ms)
BARON v15.9 (all features)	202.42
Whitney-C^1 relaxations (no range reduction)	6.39
BARON v15.9 (DBBT only)	13.99
Whitney-C^1 relaxations (DBBT only)	8.63
Whitney-C^1 relaxations (DBBT & obj. fun. cuts)	14.16

Table C.3: Average time per branch-and-bound iteration for the methods tested in the global optimization case study.

appears to be more significantly affected by the degeneracy of the optimal solution than the MINLP method. With just DBBT enabled as a range reduction technique, the performance of the differentiable relaxations improves significantly and outperforms BARON running with only DBBT in all four cases. When the in-house code uses both DBBT and objective function value cuts, the solution statistics are very comparable to those of BARON with all features enabled on the most constrained case (IV) studied in this example. Additionally, as Table C.3 shows, the average CPU cost per node in full-featured BARON is significantly higher than for any version of the in-house code. For more complicated heat integration problems (involving many streams or including real thermodynamic models), this cost could become prohibitive, owing to both the large model size and the dependence on costly range reductions techniques such as probing.

Overall, these results indicate that the nonsmooth MHEX model can be used successfully for flowsheet optimization in a deterministic branch-and-bound algorithm when equipped with the multivariate differentiable McCormick relaxations. Achieving comparable performance to a state-of-the-art solver using only basic range reduction techniques in an otherwise standard branch-and-bound algorithm indicates that this is a viable strategy for future research efforts. However, realistic flowsheet optimization requires that actual thermodynamic models be included in the process models, which, as will be shown in the following section, currently appears to be a significant challenge.

C.2 Global optimization with flash calculations and thermodynamic models

While the MHEX model itself performed well in global optimization studies in the absence of embedded thermodynamic models, this is not sufficient for realistic liquefaction process optimization. As shown in Chapters 4 through 8, adding ideal and nonideal thermophysical property models to a simulation or optimization problem significantly increases its complexity, with much of the increase in difficulty associated with the need to perform numerous nested flash calculations in the flowsheet. Application of deterministic global optimization methods requires that bounding and relaxation information be available for the flowsheet model, and so interval and McCormick analysis techniques must be applied to calculate this information about the parametric behavior of the solutions of these nested flash models. Unfortunately, this currently appears to be an obstacle for practical global optimization of realistic liquefaction processes, as detailed next.

As discussed in Chapter 5, the (nonsmooth) inside-out methods have a nested loop structure in which the inner loop is converged using a (nonsmooth) Newton type method and the outer loop is converged using a fixed-point iteration such as Anderson Acceleration. Table C.4 shows the analogous procedures that must be applied to these subproblems to calculate either interval bounds or convex relaxations on the results of these algorithms as functions of the problem parameters. Note that in each case, the parametric variant of the given method is indeed needed, as the decision variables in the optimization problem will generally either be or directly affect the parameters in the flash models. The reference ascribed to each of the interval and McCormick based techniques in Table C.4 describes the given method in detail.

Much of the previous work concerned with bounding or relaxing implicit functions has made the assumption that all functions involved are differentiable. Fortunately for the case of the nonsmooth flash models, this need not be true, and moreover, incorporating the PC^1 mid function requires only minor modifications to the established procedures. An interval extension of the mid operator is straightforward. For

Table C.4: Solution, bounding and relaxation methods for the inside-out algorithms.

Evaluation	Inner loop convergence method	Outer loop convergence method
Real values	Semismooth Newton	Anderson Acceleration
Interval bounds	Parametric interval Newton[120]	Parametric interval successive substitution[120]
McCormick relaxations	Mean-value form relaxations[121]	Direct relaxation of fixed-point mapping[121]

intervals $X \equiv [x^L \;\; x^U], Y \equiv [y^L \;\; y^U]$ and $Z \equiv [z^L \;\; z^U]$,

$$\mathrm{mid}(X, Y, Z) \equiv [\mathrm{mid}(x^L, y^L, z^L) \;\; \mathrm{mid}(x^U, y^U, z^U)],$$

by noting that the inequalities

$$x^L \leq x \leq x^U, \;\; y^L \leq y \leq y^U, \;\; z^L \leq z \leq z^U,$$

imply that

$$\mathrm{mid}(x^L, y^L, z^L) \leq \mathrm{mid}(x, y, z) \leq \mathrm{mid}(x^U, y^U, z^U).$$

The identity given in Equation (2.9) also holds without overestimation when all real-valued arguments are replaced with intervals. However, as the mid function is in general a nonconvex function of its three arguments, its convex and concave envelopes are nontrivial to derive. The McCormick extension of the mid function is therefore handled using Equation (2.9) with the multivariate relaxations of Tsoukalas and Mitsos[126] used for the bivariate max and min functions. While this likely results in looser relaxations than the best possible, as will be seen in the following, the failure to generate useful interval bounds is a greater problem at present that precludes the ability to calculate tight relaxations.

The parametric interval Newton method requires an interval extension of the Jacobian of the equation system in the case of differentiable equations. In the nonsmooth

case, the interval extension of the classical Jacobian can be replaced by an interval extension of the Clarke Jacobian.[88;27] For problems in which the only source of nonsmoothness is a single PC^1 function, this object is fortunately simple to calculate. For example, the following procedure can be used for the mid function in the flash equations. First, determine if the values 0 and/or 1 are included in the interval range for the vapor fraction, α, in order to determine which selection functions are active. Then, using AD with an interval subtype, calculate the interval extension of the Jacobian for each essentially active selection function. Finally, take the convex hull of these intervals to yield an interval extension of the Clarke Jacobian that can be used in the method. Similarly, the mean-value theorem for locally Lipschitz functions involving the Clarke Jacobian[27] is used in place of the classical mean-value theorem in order to apply the method of Stuber *et al.*[121] to calculate relaxations of nonsmooth implicit functions. Application of these methods to some simple cases highlights the present limitations of these techniques.

Example C.1. Consider a PT-flash calculation involving a five-component vapor-phase mixture of 10 mol% nitrogen, 10 mol% methane, 20 mol% ethane, 30 mol% propane and 30 mol% n-butane initially at 298.15 K, 0.4 MPa and flowing at 1.0 kmol/s. The flash temperature is 291.5 K and the parameter range for the pressure is given by the interval [0.4 0.8] MPa. The ideal model described in Appendix B is used for the thermophysical property method. In order to be able to test the performance of intervals and McCormick objects in the inside-out procedure, the algorithm is given an intentionally somewhat poor initial guess box for $\mathbf{u}$ (the vector of volatility parameters, which in the PT-flash case is the vector of the logarithms of the species' equilibrium coefficients) in place of performing the initialization in Lines 1-6 of Algorithm 5.4. This is done here by computing the equilibrium coefficients for each species using Raoult's Law in interval arithmetic and then widening each of these intervals by a factor of two around their respective midpoints. The initial interval for R in the inner loop (which is equivalent to the vapor fraction, α in the PT-case) is taken as [0.0 1.0] in Line 7 of the algorithm.

Starting from this initialization, the interval bounds converge (in the Hausdorff

metric) in three iterations of the parametric interval successive substitution method, and yield a conservative final interval [0.31483 1.00000] for the vapor fraction. The techniques of Stuber *et al.*[121] are then used to generate relaxations initialized with the results of the interval calculations. Figure C-1 shows the value as well as the convex and concave relaxations of the implicit function $\alpha \equiv x(\mathbf{p})$, defined by the solution of the PT-flash model, over the whole pressure interval from 0.4 to 0.8 MPa (relaxations denoted $x^{cv}_{P_0}(\mathbf{p})$ and $x^{cc}_{P_0}(\mathbf{p})$). The procedure was then repeated for the nested pressure intervals 0.4 to 0.6 MPa (relaxations denoted $x^{cv}_{P_1}(\mathbf{p})$ and $x^{cc}_{P_1}(\mathbf{p})$) and 0.6 to 0.8 MPa (relaxations denoted $x^{cv}_{P_2}(\mathbf{p})$ and $x^{cc}_{P_2}(\mathbf{p})$), and the resulting tighter relaxations calculated on these smaller intervals are also shown in Figure C-1.

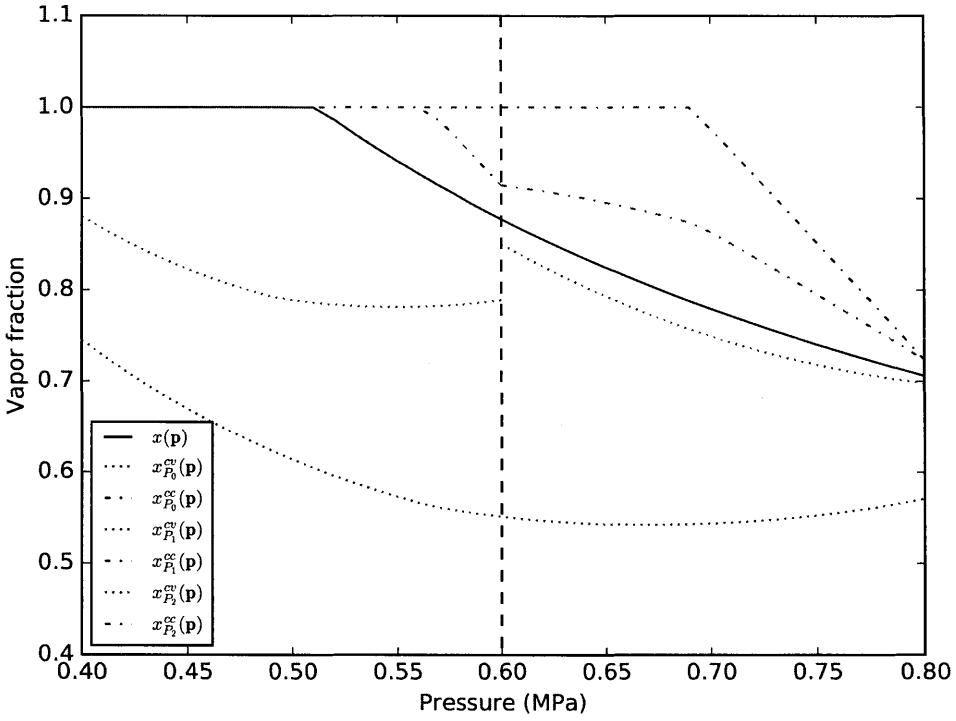


Figure C-1: Convex and concave relaxations of the vapor fraction implicitly defined by the solution of a PT-flash calculated on nested parameter intervals.

For this simple problem, the methods work quite well, producing valid and relatively tight convex and concave relaxations of the nonsmooth implicit function. However, if the problem is modified to include a parameter range for the flash temperature,

then the width of this interval has a significant impact on the success of the bounding and relaxation techniques. It has been observed that for this problem, increasing the width of this temperature interval beyond around 5 K (centered on 291.5 K) produces both very weak interval bounds on α and relaxations that are essentially constant at the lower and upper bound. Increasing the width of the temperature interval beyond around 10 K yields useless bounding and relaxation information when the methods terminate (i.e. $\alpha \in [0.0 \;\; 1.0]$). As the temperature intervals in the optimization problems of Chapter 8 have widths of several hundred degrees, this performance is clearly not acceptable, as each of these intervals would have to be partitioned many times to yield useful relaxations for the PT-flash problems embedded in the flowsheet. If there is also uncertainty in the feed composition, e.g. in a liquefaction process optimization with the refrigerant component flowrates included as decision variables, then the problem becomes even more challenging for the interval methods.

The problem is now attempted with the Peng-Robinson EOS as the thermodynamic model (with no density extrapolation). The interval methods fail to improve upon the initial bounds for nontrivial parameter intervals. Even just a narrow parameter interval for the flash pressure results in nontrivial intervals for the vapor and liquid phase compositions, which in turn leads to the intervals for the composition-dependent EOS parameters having substantial width. The presence of a nontrivial parameter interval for the flash temperature further exacerbates this issue. Calculation of bounds for the density of each phase via the parametric interval Newton method then generally fails to provide useful bounding information. The resulting overly-conservative width of the density interval leads to issues in the fugacity coefficient evaluation in Equation (B.25), as the interval arguments in the logarithmic terms will then almost always contain negative values. Replacing the argument of these logarithmic terms with an argument of the form $\max(\varepsilon, \cdot)$ for some small ε allows the calculation to proceed, but is also problematic as the choice of ε will then almost always have a direct impact on the outcome of the calculation and lead to significant overestimation in the fugacity coefficient calculation. This results in extremely conservative bounds on the equilibrium coefficients and so the bounds on the

vector $\mathbf{u}$ subsequently fail to be improved by interval iteration.

Example C.2. Now consider PQ-flash calculations on the same mixture from the previous example. With the flash temperature as an unknown, an interval is calculated or provided for this quantity as an initial guess and then updated in the inner loop of the algorithm. However, as noted before, even the ideal physical property calculations are a source of difficulty with all but extremely narrow intervals for the flash temperature, and here it is highly unlikely that this will ever be the case in practice. The issues discussed with regards to nonideal thermodynamics also mean such calculations are even more sensitive to the width of the flash temperature interval than in the ideal case. As an example of a calculation with the ideal property method, assume that the pressure parameter interval is given by [0.48 0.52] MPa, and then consider performing an adiabatic PQ-flash on the mixture (with all other flash parameters given by real numbers). The initialization procedure in Algorithm 5.1 is used (with surrogate model coefficients C though F fixed to 0 due to the use of the ideal model) except that initial intervals are provided for T and α instead of performing the calculations in Lines 1 and 2. Providing initial intervals $\alpha \in [0.8 \ 1.0]$ and $T \in [285.0 \ 290.0]$ leads to the outer loop interval iteration converging in 4 iterations and returning $\alpha \in [0.8510 \ 0.9949]$ with no improvement in the temperature interval, both of which significantly overestimate the true range ($\alpha \in [0.8855 \ 0.8937]$, $T \in [287.127 \ 287.265]$) even for this extremely narrow pressure interval and tight initial guess. Increasing the width of the parameter interval(s) or decreasing the quality of the guess tends to result in bounds that do not improve from the initial intervals.

This problem is now attempted in BARON v15.9 to test the performance of a global optimization algorithm on such a calculation. As BARON cannot directly handle nonsmoothness, the classical flash model is used and only problems with two-phase solutions are considered. Equations (4.8)-(4.12) are given as the problem constraints, and the ideal model from Appendix B is used to describe equilibrium and physical properties. Note that the max statement in the heat of vaporization model (Equation (4.25)) must be replaced, e.g. with the smoothing approximation in Equation (3.10) with $\beta := 10^{-4}$. When a constant objective function is provided, the problem is solved

immediately in preprocessing. However, if one of the flash parameters is included as a decision variable, this no longer the case. Consider the problem of minimizing the flash heat duty at constant pressure after adding the constraint that $\alpha \geq 0.5$ to the model, i.e. to determine how much the mixture can be cooled while still producing at least as much vapor as liquid). Full-featured BARON v15.9 fails to solve this problem to the desired tolerance in 10^8 branch-and-reduce iterations, terminating with a relative gap between the lower and upper bounds of 0.25%. If, instead, the problem is to maximize the pressure of the adiabatic flash subject to these constraints, then BARON is able to converge to a global solution after performing 5,545 iterations taking 14.1 seconds.

Since simulating the PRICO process flowsheet with a reasonably fine discretization of the process stream cooling curves requires the solution of over 100 such flash calculations, each described by a cubic EOS, it is clear that the level of performance observed in these examples will not suffice. Some potential opportunities for improvement exist before considering more major changes to the methods. For instance, careful examination and possible rewriting of the mathematical expressions used in the flash calculations algorithms could help to reduce overestimation due to the interval dependency effect. There may also be more optimal preconditioning matrices that can be calculated for use in these methods than the midpoint-inverse preconditioner that was used in these examples. Additionally, a possibly useful modification to the basic interval calculation technique is to use the algorithm of Hua *et al.*[53] to provide interval extensions of mole fraction weighted quantities instead of simply calculating the natural interval extension of the sum of the products. Those authors note that this approach is helpful for bounding the range of the composition-dependent EOS coefficients. However, their work does not consider the parameter-dependent case, in which intervals for the flash pressure and temperature/heat duty can contribute significantly to overestimation of these coefficients in a manner that cannot be mitigated by this technique. However, it seems most likely that more substantial improvements involving the numerical methods themselves are needed for challenging practical problems.

Bibliography

[1] F. A. Aly and L. L. Lee. Self-consistent equations for calculating the ideal gas heat capacity, enthalpy and entropy. *Fluid Phase Equilibria*, 6:169–179, 1981.

[2] D. G. Anderson. Iterative procedures for nonlinear integral equations. *Journal of the Association for Computing Machinery*, 12(4):547–560, 1965.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[4] A. Aspelund, T. Gundersen, J. Myklebust, M. Nowak, and A. Tomasgard. An optimization-simulation model for a simple LNG process. *Computers & Chemical Engineering*, 34(10):1606–1617, 2010.

[5] Aspen Technology Inc. *Aspen Plus v8.4*. Aspen Technology Inc., Cambridge, MA, 2014.

[6] L. Asselineau, G. Bogdanic, and J. Vidal. A versatile algorithm for calculating vapour-liquid equilibria. *Fluid Phase Equilibria*, 3(4):273–290, 1979.

[7] B. Austbø and T. Gundersen. Optimal distribution of temperature driving forces in low-temperature heat transfer. *AIChE Journal*, 61(8):2447–2455, 2015.

[8] B. Austbø and T. Gundersen. Optimization of a single expander LNG process. *Energy Procedia*, 64(Supplement C):63–72, 2015.

[9] B. Austbø and T. Gundersen. Impact of problem formulation on LNG process optimization. *AIChE Journal*, 62(10):3598–3610, 2016.

[10] B. Austbø, S. W. Løvseth, and T. Gundersen. Annotated bibliography – Use of optimization in LNG process design and operation. *Computers and Chemical Engineering*, 71:391–414, 2014.

[11] B. Austbø, P. E. Wahl, and T. Gundersen. Constraint handling in stochastic optimization algorithms for natural gas liquefaction processes. *Computer Aided Chemical Engineering*, 32:445–450, 2013.

[12] S. Balakrishna and L. T. Biegler. Targeting strategies for the synthesis and energy integration of nonisothermal reactor networks. *Industrial & Engineering Chemistry*, 38(3):2152–2164, 1992.

[13] P. I. Barton, K. A. Khan, P. Stechlinski, and H. A. J. Watson. Computationally relevant generalized derivatives: theory, evaluation and applications. *Optimization Methods & Software*. DOI:10.1080/10556788.2017.1374385.

[14] B. T. Baumrucker, J. G. Renfro, and L. T. Biegler. MPEC problem formulations and solution strategies with chemical engineering applications. *Computers & Chemical Engineering*, 32(12):2903–2913, 2008.

[15] B. L. Beegle, M. Modell, and R. C. Reid. Thermodynamic stability criterion for pure substances and mixtures. *AIChE Journal*, 20(6):1200–1206, 1974.

[16] J. Boston and H. Britt. A radically different formulation and solution of the single-stage flash problem. *Computers & Chemical Engineering*, 2(1):109–122, 1978.

[17] J. Boston and S. Sullivan. An improved algorithm for solving the mass balance equations in multistage separation processes. *The Canadian Journal of Chemical Engineering*, 50(2):663–669, 1972.

[18] British Petroleum. *BP Statistical Review of World Energy 2017*. 2017.

[19] R. H. Byrd, J. Nocedal, and R. A. Waltz. *Large-Scale Nonlinear Optimization*, pages 35–59. Springer, Boston, MA, 2006.

[20] M. Castier and E. M. Queiroz. Energy targeting in heat exchanger network synthesis using rigorous physical property calculations. *Industrial & Engineering Chemistry Research*, 41(6):1511–1515, 2002.

[21] L. Castillo and C. Dorao. Influence of the plot area in an economical analysis for selecting small scale LNG technologies for remote gas production. *Journal of Natural Gas Science and Engineering*, 2(6):302–309, 2010.

[22] R. H. Cavett. Application of numerical methods to the convergence of simulated processes involving recycle loops. *Proceedings of the American Petroleum Institute*, 43(57), 1963.

[23] S. Chakraborty and P. Ghosh. Heat exchanger network synthesis: the possibility of randomization. *Chemical Engineering Journal*, 72:209–216, 1999.

[24] W. K. Chan and R. G. H. Prince. Application of the chain rule of differentiation to sequential modular flowsheet optimization. *Computers and Chemical Engineering*, 10(3):223–240, 1986.

[25] C. Chen and O. L. Mangasarian. A class of smoothing functions for nonlinear and mixed complementarity problems. *Computational Optimization and Applications*, 5(2):97–138, 1996.

[26] H.-S. Chen and M. A. Stadtherr. A simultaneous modular approach to process flowsheeting and optimization. Part I: theory and implementation. *AIChE Journal*, 31(11):1843–1856, 1985.

[27] F. H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, Philadelphia, PA, 1990.

[28] ConocoPhillips LNG Technology & Licensing. Optimized Cascade® Process. `http://lnglicensing.conocophillips.com/what-we-do/lng-technology-licensing/Pages/optimized-cascade-process.aspx`, 2016.

[29] F. Del Nogal, J.-K. Kim, S. Perry, and R. Smith. Optimal design of mixed refrigerant cycles. *Industrial & Engineering Chemistry Research*, 47(22):8724–8740, 2008.

[30] A. W. Dowling, C. Balwani, Q. Gao, and L. T. Biegler. Optimization of sub-ambient separation systems with embedded cubic equation of state thermodynamic models and complementarity constraints. *Computers & Chemical Engineering*, 81:323–343, 2015.

[31] A. S. Drud. CONOPT – A Large-Scale GRG Code. *ORSA Journal on Computing*, 6(2):207–216, 1994.

[32] M. A. Duran and I. E. Grossmann. Simultaneous optimization and heat integration of chemical processes. *AIChE Journal*, 32(1):123–138, 1986.

[33] F. Facchinei, A. Fischer, and M. Herrich. An LP-Newton method: nonsmooth equations, KKT systems, and nonisolated solutions. *Mathematical Programming*, 146:1–36, 2014.

[34] F. Facchinei and J.-S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems*, volume 2. Springer, New York, NY, 2003.

[35] A. Fischer, M. Herrich, A. F. Izmailov, and M. V. Solodov. A Globally Convergent LP-Newton Method. *SIAM Journal on Optimization*, 26(4):2012–2033, 2016.

[36] J. R. Friday and B. D. Smith. An analysis of the equilibrium stage separations problem–formulation and convergence. *AIChE Journal*, 10(5):698–707, 1964.

[37] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Review*, 47(1):99–131, 2002.

[38] M. Glass and A. Mitsos. Thermodynamic analysis of formulations to discriminate multiple roots of cubic equations of state in process models. *Computers and Chemical Engineering*. In Press.

[39] G. H. Golub and C. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[40] V. Gopal and L. T. Biegler. Nonsmooth dynamic simulation with linear programming based methods. *Computers & Chemical Engineering*, 21(7):675–689, 1997.

[41] V. Gopal and L. T. Biegler. Smoothing methods for complementarity problems in process engineering. *AIChE Journal*, 45(7):1535–1547, 1999.

[42] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques for Algorithmic Differentiation.* SIAM, Philadelphia, PA, 2nd edition, 2008.

[43] H. Grootjans, R. Nagelvoort, and K. Vink. Liquefying a stream enriched in methane, 2002. US Patent 6,370,910.

[44] I. E. Grossmann, J. A. Caballero, and H. Yeomans. Advances in mathematical programming for the synthesis of process systems. *Latin American Applied Research*, 30(98):263–284, 2000.

[45] I. E. Grossmann and F. Trespalacios. Systematic modeling of discrete-continuous optimization models through generalized disjunctive programming. *AIChE Journal*, 59(9):3276–3295, 2013.

[46] I. E. Grossmann, H. Yeomans, and Z. Kravanja. A rigorous disjunctive optimization model for simultaneous flowsheet optimization and heat integration. *Computers & Chemical Engineering*, 22(98):157–164, 1998.

[47] T. Gundersen. Numerical aspects of the implementation of cubic equations of state in flash calculation routines. *Computers and Chemical Engineering*, 6(3):245–255, 1982.

[48] M. M. F. Hasan and I. A. Karimi. Modeling and simulation of main cryogenic heat exchanger in a base-load liquefied natural gas plant. *Proceedings of the 17th European Symposium on Computer Aided Process Engineering*, pages 1–6, 2007.

[49] M. M. F. Hasan, I. A. Karimi, H. E. Alfadala, and H. Grootjans. Operational modeling of multistream heat exchangers with phase changes. *AIChE Journal*, 55(1):150–171, 2009.

[50] P. Hatcher, R. Khalilpour, and A. Abbas. Optimisation of LNG mixed-refrigerant processes considering operation and design objectives. *Computers & Chemical Engineering*, 41:123–133, 2012.

[51] R. A. Heidemann and A. M. Khalil. The calculation of critical points. *AIChE Journal*, 26(5):769–779, 1980.

[52] G. F. Hewitt and S. J. Pugh. Approximate design and costing methods for heat exchangers. *Heat Transfer Engineering*, 28(2):76–86, 2007.

[53] J. Z. Hua, J. F. Brennecke, and M. A. Stadtherr. Enhanced interval analysis for phase stability: cubic equation of state models. *Industrial & Engineering Chemistry Research*, 37(4):1519–1527, 1998.

[54] J.-H. Hwang, M.-I. Roh, and K.-Y. Lee. Determination of the optimal operating conditions of the dual mixed refrigerant cycle for the LNG FPSO topside liquefaction process. *Computers & Chemical Engineering*, 49:25–36, 2013.

[55] IBM. IBM ILOG CPLEX v12.7. `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html`, 2016.

[56] J. B. Jensen and S. Skogestad. Optimal operation of a simple LNG process. *Proceedings of the International Symposium on Advanced Control of Chemical Processes*, pages 241–246, 2006.

[57] J. B. Jensen and S. Skogestad. Problems with Specifying $\Delta T_{\min}$ in the Design of Processes with Heat Exchangers. *Industrial & Engineering Chemistry Research*, 47:3071–3075, 2008.

[58] J. T. Jensen. *The Development of a Global LNG market.* The Oxford Institute for Energy Studies, Oxford, UK, 2004.

[59] R. S. Kamath, L. T. Biegler, and I. Grossmann. Modeling multistream heat exchangers with and without phase changes for simultaneous optimization and heat integration. *AIChE Journal*, 58(1):190–204, 2012.

[60] R. S. Kamath, L. T. Biegler, and I. E. Grossmann. An equation-oriented approach for handling thermodynamics based on cubic equation of state in process optimization. *Computers & Chemical Engineering*, 34(12):2085–2096, 2010.

[61] F. Kappel and A. Kuntsevich. An implementation of Shor's r-algorithm. *Computational Optimization and Applications*, 15:193–205, 2000.

[62] N. Karmitsa, A. Bagirov, and M. Mäkelä. Comparing different nonsmooth minimization methods and software. *Optimization Methods and Software*, 27(1):131–153, 2012.

[63] K. A. Khan and P. I. Barton. Evaluating an element of the Clarke generalized Jacobian of a composite piecewise differentiable function. *ACM Transactions on Mathematical Software*, 39(4):1–28, 2013.

[64] K. A. Khan and P. I. Barton. A vector forward mode of automatic differentiation for generalized derivative evaluation. *Optimization Methods & Software*, 30(6):1185–1212, 2015.

[65] K. A. Khan and P. I. Barton. Generalized derivatives for hybrid systems. *IEEE Transactions On Automatic Control.*, 62(7):3193–3208, 2017.

[66] K. A. Khan, H. A. J. Watson, and P. I. Barton. Differentiable McCormick relaxations. *Journal of Global Optimization*, 67(4):687–729, 2017.

[67] M. S. Khan, Y. A. Husnil, M. Getu, and M. Lee. Modeling and simulation of multistream heat exchangers using artificial neural networks. *Proceedings of the 11th International Symposium on Process Systems Engineering*, pages 1196–1200, 2012.

[68] M. S. Khan, I. A. Karimi, and M. Lee. Evolution and optimization of the dual mixed refrigerant process of natural gas liquefaction. *Applied Thermal Engineering*, 96:320–329, 2016.

[69] M. S. Khan and M. Lee. Design optimization of single mixed refrigerant natural gas liquefaction process using the particle swarm paradigm with nonlinear constraints. *Energy*, 49:146–155, 2013.

[70] C. J. King. *Separation Processes.* McGraw-Hill, Inc., New York, NY, 2nd edition, 1980.

[71] M. Kojima and S. Shindo. Extensions of Newton and quasi-Newton methods to systems of PC^1 equations. *Journal of the Operations Research Society of Japan*, 29(4):352–374, 1986.

[72] G. C. Lee, R. Smith, and X. X. Zhu. Optimal synthesis of mixed-refrigerant systems for low-temperature processes. *Industrial & Engineering Chemistry Research*, 41(20):5016–5028, 2002.

[73] S. Lee, N. V. D. Long, and M. Lee. Design and optimization of natural gas liquefaction and recovery processes for offshore floating liquefied natural gas plants. *Industrial and Engineering Chemistry Research*, 51(30):10021–10030, 2012.

[74] W. Lim, K. Choi, and I. Moon. Current Status and Perspectives of Liquefied Natural Gas (LNG) Plant Design. *Industrial & Engineering Chemistry Research*, 52(9):3065–3088, 2013.

[75] L. Lukšan and J. Vlček. Algorithm 811: NDA: Algorithms for nondifferentiable optimization. *ACM Transactions on Mathematical Software*, 27:193–213, 2001.

[76] J. Maher and J. Sudduth. Method and apparatus for liquefying gases, 1975. US Patent 3,914,949.

[77] M. M. Mäkelä. Multiobjective proximal bundle method for nonconvex nonsmooth optimization: Fortran subroutine MPBNGC 2.0. *Reports of the Department of Mathematical Information Technology, Series B, Scientific Computing*, (B 13/2003), University of Jyväskylä, Jyväskylä, 2003.

[78] P. M. Mathias and M. S. Benson. Computational aspects of equations of state: Fact and fiction. *AIChE Journal*, 32(12):2087–2090, 1986.

[79] P. M. Mathias, J. F. Boston, and S. Watanasiri. Effective utilization of equations of state for thermodynamic properties in process simulation. *AIChE Journal*, 30(2):182–186, 1984.

[80] F. A. Michelsen, I. J. Halvorsen, B. F. Lund, and P. E. Wahl. Modeling and simulation for control of the TEALARC liquified natural gas process. *Industrial & Engineering Chemistry Research*, 49(16):7389–7397, 2010.

[81] M. L. Michelsen. The isothermal flash problem. Part I. Stability. *Fluid phase Equilibra*, 9:1–19, 1982.

[82] M. L. Michelsen. The isothermal flash problem. Part II. Phase-split calculation. *Fluid Phase Equilibria*, 9:21–40, 1982.

[83] M. B. Mills, M. J. Wills, and V. L. Bhirud. The calculation of density by the BWRS equation of state in process simulation contexts. *AIChE Journal*, 26(6):902–910, 1980.

[84] A. Mitsos and P. I. Barton. A dual extremum principle in thermodynamics. *AIChE Journal*, 53(8):2131–2147, 2007.

[85] S. Mokhatab, J. Y. Mak, J. V. Valappil, and D. A. Wood. *Handbook of Liquefied Natural Gas*. Gulf Professional Publishing, Houston, TX, 2013.

[86] E. J. Moniz, H. D. Jacoby, and A. J. M. Meggs. *The Future of Natural Gas: An Interdisciplinary MIT Study*. 2011.

[87] Y. Nesterov. Lexicographic differentiation of nonsmooth functions. *Mathematical Programming, Series B*, 104:669–700, 2005.

[88] A. Neumaier. *Interval Methods for Systems of Equations*, volume 37 of *Encyclopedia of Mathematics and its Applications*. Cambridge Univ. Press, Cambridge, UK, 1990.

[89] C. Newton. Dual mixed refrigerant natural gas liquefaction with staged compression, 1985. US Patent 4,525,185.

[90] W. Nuttall, R. Clarke, and B. Glowacki. *The Future of Helium as a Natural Resource*. Routledge Explorations in Environmental Economics. Taylor & Francis, Abingdon, UK, 2012.

[91] J. Ortega and W. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. SIAM, Philadelphia, PA, 2000.

[92] V. S. Parekh and P. M. Mathias. Efficient flash calculations for chemical process design – extension of the Boston–Britt inside–out flash algorithm to extreme conditions and new flash types. *Computers & Chemical Engineering*, 22(10):1371–1380, 1998.

[93] G. V. Pasad and G. Venkatarathnam. A method for avoiding trivial roots in isothermal flash calculations using cubic equations of state. *Industrial & Engineering Chemistry Research*, 38(9):3530–3534, 1999.

[94] M. Patrascu and P. I. Barton. Optimal campaigns in end-to-end continuous pharmaceuticals manufacturing. Part 1: nonsmooth dynamic modeling. Submitted.

[95] R. C. Pattison and M. Baldea. Multistream heat exchangers: equation-oriented modeling and flowsheet optimization. *AIChE Journal*, 61(6):1856–1866, 2015.

[96] M. Pillarella, Y.-N. Liu, J. Petrowski, and R. Bower. The C3MR liquefaction cycle: versatility for a fast growing, ever changing LNG industry. 15th International Conference on LNG (LNG-15), Barcelona, 24–27 April, 2007.

[97] B. E. Poling, E. A. Grens, and J. M. Prausnitz. Thermodynamic properties from a cubic equation of state: avoiding trivial roots and spurious derivatives. *Industrial & Engineering Chemistry Process Design and Development*, 20(1978):127–130, 1981.

[98] J. M. Ponce-Ortega, A. Jiménez-Gutiérrez, and I. E. Grossmann. Optimal synthesis of heat exchanger networks involving isothermal process streams. *Computers and Chemical Engineering*, 32(8):1918–1942, 2008.

[99] L. Qi and J. Sun. A nonsmooth version of Newton's method. *Mathematical Programming*, 58:353–367, 1993.

[100] H. H. Rachford Jr. and J. D. Rice. Procedure for use of electronic digital computers in calculating flash vaporization hydrocarbon equilibrium. *Journal of Petroleum Technology*, 4(10):327–328, 1952.

[101] D. Ralph and S. Scholtes. Sensitivity analysis of composite piecewise smooth equations. *Mathematical Programming*, 76:593–612, 1997.

[102] H. N. Rao and I. A. Karimi. A superstructure-based model for multistream heat exchanger design within flowsheet optimization. *AIChE Journal.* In Press.

[103] M. Roberts and R. Agrawal. Dual mixed refrigerant cycle for gas liquefaction, 2000. US Patent 6,119,479.

[104] M. Roberts and R. Agrawal. Hybrid cycle for the production of liquefied natural gas, 2001. US Patent 6,308,531.

[105] J. Rowlinson and F. Swinton. *Liquids and Liquid Mixtures.* Butterworths Monographs in Chemistry and Chemical Engineering. Butterworth Scientific, London, UK, 3rd edition, 1982.

[106] R. A. Russell. A flexible and reliable method solves single-tower and crude distillation column problems. *Chemical Engineering*, 90(21):52–59, 1983.

[107] H. S. Ryoo and N. V. Sahinidis. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Computers & Chemical Engineering*, 19(5):551–566, 1995.

[108] N. V. Sahinidis. *BARON 15.9: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual, 2015. Available at `https://www.gams.com/help/topic/gams.doc/solvers/baron/index.html`.

[109] K. Schittkowski. NLPQLP: A FORTRAN implementation of a sequential quadratic programming algorithm with distributed and non-monotone line Search - User's guide, Version 3.0. *Report, Department of Computer Science, University of Bayreuth, 2009.*

[110] C. Schnepper and M. Stadtherr. Robust process simulation using interval methods. *Computers & Chemical Engineering*, 20(2):187–199, 1996.

[111] S. Scholtes. *Introduction to Piecewise Differentiable Equations.* SpringerBriefs in Optimization, New York, NY, 2012.

[112] J. K. Scott, M. D. Stuber, and P. I. Barton. Generalized McCormick relaxations. *Journal of Global Optimization*, 51(4):569–606, 2011.

[113] G. Skaugen, M. Hammer, P. E. Wahl, and T. Wilhelmsen. Constrained nonlinear optimisation of a process for liquefaction of natural gas including a geometrical and thermo-hydraulic model of a compact heat exchanger. *Computers and Chemical Engineering*, 73:102–115, 2015.

[114] R. Smith. *Chemical Process Design.* McGraw-Hill, Inc., New York, NY, 1995.

[115] K. Starling. *Fluid Thermodynamic Properties for Light Petroleum Systems.* Gulf Publishing Co., Houston, TX, 1973.

[116] R. P. Stateva, S. G. Tsvetkov, and C. B. Boyadjiev. An approach to organizing the EOS model in process simulators. *AIChE Journal*, 36(1):132–136, 1990.

[117] P. Stechlinski, K. A. Khan, and P. I. Barton. Generalized sensitivity analysis of nonlinear programs. *SIAM Journal on Optimization.* In Press.

[118] R. Stockmann, W. Forg, M. Bolt, M. Steinbauer, C. Pfeiffer, P. Paurola, A. Fredheim, and O. Sorensen. Method for liquefying a stream rich in hydrocarbons, 2001. US Patent 6,253,574.

[119] B. A. Stradi, G. Xu, J. F. Brennecke, and M. A. Stadtherr. Modeling and design of an environmentally benign reaction process. *Proceedings of Foundations of Computer Aided Process Design (FOCAPD)*, 1999.

[120] M. D. Stuber and P. I. Barton. Robust simulation and design using parametric interval methods. *4th International Workshop on Reliable Engineering Computing (REC 2010)*, pages 536–553, 2010.

[121] M. D. Stuber, J. K. Scott, and P. I. Barton. Convex and concave relaxations of implicit functions. *Optimization Methods and Software*, 30(3):424–460, 2015.

[122] J. W. Tester and M. Modell. *Thermodynamics and Its Applications.* Prentice-Hall, 3rd edition, 1997.

[123] J. E. Tolsma and P. I. Barton. On computational differentiation. *Computers & Chemical Engineering*, 22(4/5):475–490, 1998.

[124] J. E. Tolsma and P. I. Barton. DAEPACK: an open modeling environment for legacy models. *Industrial & Engineering Chemistry Research*, 39(6):1826–1839, 2000.

[125] J. E. Tolsma, J. A. Clabaugh, and P. I. Barton. Symbolic incorporation of external procedures into process modeling environments. *Industrial & Engineering Chemistry Research*, 41(16):3867–3876, 2002.

[126] A. Tsoukalas and A. Mitsos. Multivariate McCormick relaxations. *Journal of Global Optimization*, 59(2-3):633–662, 2014.

[127] Union Gas. Chemical composition of natural gas, September 2017. Available at: `https://www.uniongas.com/about-us/about-natural-gas/chemical-composition-of-natural-gas`.

[128] University of Manchester Centre for Process Integration. Process integration software suite. `http://www.ceas.manchester.ac.uk/our-research/themes-challenges/themes/pdi/cpi/software/downloads/`, 2017.

[129] U.S. Energy Information Administration. Carbon dioxide emissions coefficients, February 2016. Available at: `https://www.eia.gov/environment/emissions/co2_vol_mass.php`.

[130] U.S. Energy Information Administration. *Annual Energy Outlook 2017 with projections to 2050.* U.S. Energy Information Administration, Washington, DC, 2017.

[131] G. Venkatarathnam. *Cryogenic Mixed Refrigerant Processes.* International Cryogenics Monograph Series. Springer, New York, NY, 2008.

[132] M. Vikse, H. A. J. Watson, T. Gundersen, and P. I. Barton. Robust simulation methods for complex single mixed refrigerant natural gas liquefaction processes. Submitted.

[133] A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[134] H. F. Walker and P. Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011.

[135] H. A. J. Watson and P. I. Barton. Reliable flash calculations: Part 3. A nonsmooth approach to density extrapolation and pseudoproperty evaluation. Submitted.

[136] H. A. J. Watson and P. I. Barton. Modeling phase changes in multistream heat exchangers. *International Journal of Heat and Mass Transfer*, 105:207–219, 2017.

[137] H. A. J. Watson, K. A. Khan, and P. I. Barton. Multistream heat exchanger modeling and design. *AIChE Journal*, 61(10):3390–3403, 2015.

[138] H. A. J. Watson, M. Vikse, T. Gundersen, and P. I. Barton. Optimization of single mixed refrigerant natural gas liquefaction processes described by nondifferentiable models. In preparation.

[139] H. A. J. Watson, M. Vikse, T. Gundersen, and P. I. Barton. Reliable flash calculations: Part 2. Process flowsheeting with nonsmooth models and generalized derivatives. Submitted.

[140] H. A. J. Watson, M. Vikse, T. Gundersen, and P. I. Barton. Reliable flash calculations: Part 1. Nonsmooth inside-out algorithms. *Industrial & Engineering Chemistry Research*, 56(4):960–973, 2017.

[141] A. Wechsung, A. Aspelund, T. Gundersen, and P. I. Barton. Synthesis of heat exchanger networks at subambient conditions with compression and expansion of process streams. *AIChE Journal*, 57(8):2090–2108, 2011.

[142] H. Whitney. Analytic extensions of differentiable functions defined in closed sets. *Transactions of the American Mathematical Society*, 36(1):63–89, 1934.

[143] C. H. Whitson, Ø. Fevang, and T. Yang. Gas condensate PVT – what's really important and why? *Proceedings of IBC Conference on Optimisation of Gas Condensate Fields*, 1999.

[144] C. H. Whitson and M. L. Michelsen. The negative flash. *Fluid Phase Equilibria*, 53:51–71, 1989.

[145] J. H. Wilkinson. *Rounding Errors in Algebraic Processes.* Prentice Hall, Englewood Cliffs, NJ, 1963.

[146] D. Wolbert, X. Joulia, B. Koehret, and L. Biegler. Flowsheet optimization and optimal sensitivity analysis using analytical derivatives. *Computers & Chemical Engineering*, 18(11-12):1083–1095, 1994.

[147] W. Xiao, K. Zhu, W. Yuan, and H. H. Chien. An algorithm for simultaneous chemical and phase equilibrium calculation. *AIChE Journal*, 35(11):1813–1820, 1989.

[148] G. Xu, D. Bluck, D. J. Van Peursem, and I. H. Boys. Thermodynamic process control based on pseudo-density root for equation of state, 2012. US Patent 8,165,860 B2.

[149] T. F. Yee and I. E. Grossmann. Simultaneous optimization models for heat integration – II. Heat exchanger network synthesis. *Computers & Chemical Engineering*, 14(10):1165–1184, 1990.

[150] A. Zavala-Río, R. Femat, and R. Santiesteban-Cos. An analytical study of the logarithmic mean temperature. *Revista Mexicana de Ingeniería Químca*, 4:201–212, 2005.

[151] E. Zhao and S. Saha. Applications of complex domain in vapor-liquid equilibrium calculations using a cubic equation of state. *Industrial & Engineering Chemistry Research*, 37:1625–1633, 1998.