Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering

Doctoral Thesis

Phase Equilibrium Computation and Compositional Simulation

Prague 2021

Tomáš Smejkal

This thesis is submitted to the Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague, in partial fulfilment of the requirements for the degree of Doctor of Philosophy (Ph.D.) in Mathematical Engineering.

Copyright © 2021 Tomáš Smejkal. All Rights Reserved.

Bibliografický záznam

Autor	Ing. Tomáš Smejkal, České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská, Katedra matematiky
Název práce	Výpočet fázové rovnováhy a kompoziční simulace
Studijní program	Aplikace přírodních věd
Studijní obor	Matematické inženýrství
Školitel	doc. Ing. Jiří Mikyška, Ph.D., České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská, Katedra matematiky
Akademický rok	2021
Počet stran	217
Klíčová slova	termodynamika, fázová stabilita, fázová rovnováha, optimalizace, porézní prostředí, kompoziční simulace, smíšená metoda konečných objemů

Bibliographic Entry

Author	Ing. Tomáš Smejkal, Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Department of Mathematics
Title of dissertation	Phase Equilibrium Computation and Compositional Simulation
Degree programme	Application of Natural Sciences
Field of study	Mathematical Engineering
Supervisor	doc. Ing. Jiří Mikyška, Ph.D., Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Department of Mathematics
Academic year	2021
Number of pages	217
Keywords	thermodynamics, phase stability, phase equilibrium, optimization, porous medium, compositional simulation, mixed hybrid finite element method

Abstrakt

Tématem této práce je testování fázové stability a výpočet fázové rovnováhy vícesložkových směsí. Hlavním cílem je představit jednotnou formulaci těchto problémů, numerické algoritmy (založené na Newtonově–Raphsonově metodě) pro řešení těchto problémů a ukázat výkon algoritmů na příkladech různé složitosti. Kromě hlavního cíle je v práci uveden rozsáhlý přehled rovnovážné termodynamiky. Dále jsou představeny alternativní výpočetní algoritmy. V úloze testování fázové stability jsou diskutovány přístupy založené na technikách globální optimalizace (deterministické nebo heuristické). Nakonec je uvedeno efektivní řešení systému lineárních rovnic vznikajících z linearizace úlohy fázové stability. Pro výpočet fázové rovnováhy je odvozena numerická metoda založená na eliminaci omezujících podmínek a Newtonově–Raphsonově metodě. Nakonec tato práce představuje aplikaci výpočtu fázové rovnováhy v proudění tekutin v porézním prostředí. Je uveden matematický model a numerické řešení kompozičního vícefázového stlačitelného izotermického Darcyho proudění pro vícesložkové směsi. Tato práce obsahuje výsledky mnoha testů všech výše uvedených numerických algoritmů.

Abstract

The main topics of this thesis are the phase stability testing and the phase equilibrium computation of multi-component mixtures. The main goal is to present a unified formulation of these problems, numerical algorithms (based on the Newton–Raphson method) for solving these problems, and the performance of the algorithms on examples of various complexity. Besides the main goal, an extensive overview of the equilibrium thermodynamics is given. Moreover, alternative algorithms to the main algorithms are presented. In the phase stability testing problem, approaches based on the global optimization techniques (deterministic or heuristical) are discussed. Lastly, an efficient solution of the system of linear equations arising from the linearization of the phase stability problem is given. In the phase equilibrium computation, a numerical method based on the elimination of the constraints and the Newton–Raphson method is presented. Finally, this thesis presents an application of the phase equilibrium computation in the fluid flow in the porous medium. The mathematical model and the numerical solution for the compositional multi-phase compressible isothermal Darcy's flow for multi-component mixtures are presented. This thesis contains the results of various tests of all the above numerical algorithms.

Acknowledgements

I would like to thank everyone who gave me any kind of support in any of my activities during the time when I was a Ph.D. student. First, I would like to express my thanks to my supportive supervisor, doc. Jiří Mikyška, Ph.D., who has supported me throughout this research project. Second, I would like to express my gratitude to my friends and family. Without their tremendous understanding and encouragement in the past few years, it would be impossible for me to complete my study.

The work related to the topic of this thesis was partially supported by the following projects:

- project "Thermal energy storage materials: thermophysical characteristics for the design of thermal batteries" number 17-08218S of the Czech Science Foundation,
- project "Investigation of shallow subsurface flow with phase transitions" number 17-06759S of the Czech Science Foundation,
- project "Multiphase flow, transport, and structural changes related to water freezing and thawing in the subsurface" number 21-09093S of the Czech Science Foundation,
- project "Advanced supercomputing methods for implementation of mathematical models" number SGS11/161/OHK4/3T/14 of the Student Grant Agency of the Czech Technical University in Prague,
- project "Advanced supercomputing methods for mathematical modeling of natural processes" number SGS14/206/OHK4/3T/14 of the Student Grant Agency of the Czech Technical University in Prague,
- project "Application of advanced supercomputing methods for mathematical modeling of natural processes" number SGS17/194/OHK4/3T/14 of the Student Grant Agency of the Czech Technical University in Prague,
- project "Development and application of advanced methods for mathematical modeling of natural and industrial processes using high-performance computing" number SGS20/184/OHK4/3T/14 of the Student Grant Agency of the Czech Technical University in Prague,
- project "Numerical Methods for Multiphase Flow and Transport in Subsurface Environmental Applications" number Kontakt ME10009 of Czech Ministry of Education, Youth and Sports,
- project "Development and Validation of Porous Media Fluid Dynamics and Phase Transitions Models for Subsurface Environmental Application" number Kontakt II LH14003 of Czech Ministry of Education, Youth, and Sports,
- project "Computational Models and Experimental Investigation of Fluid Dynamics, Mass Transfer and Transport, and Phase Transitions in Porous medium for Environmental Applications" number LTAUSA19021 of the Czech Ministry of Education, Youth, and Sports.

Author's declaration

I confirm having prepared the thesis by my own and having listed all used sources of information in the bibliography.

Prague, 27 July 2021 Tomáš Smejkal

Contents

Nomenclature

Acronyms

BB	**B**ranch and **B**ound
CPA	**C**ubic **P**lus **A**ssociation
DG	**D**iscontinuous **G**alerkin
EOC	**E**xperimental **O**rder of **C**onvergence
FD	**F**inite **D**ifference
FE	**F**inite **E**lement
FV	**F**inite **V**olume
ig	**i**deal **g**as
IMPEC	**Im**plicit in **P**ressure **E**xplicit in **C**oncentration
MHFE	**M**ixed **H**ybrid **F**inite **E**lement
pm	**p**orous **m**edium
PR	**P**eng–**R**obinson
RTN	**R**aviart–**T**homas–**N**édélec
SSI	**S**uccessive **S**ubstitution **I**teration
TPD	**T**angent **P**lane **D**istance

Greek

α_{ik}	estimate coefficient for c_p of the i-th component considered as the ideal gas	
Δt	discrete time step	s
Δx	discrete spatial step	m
$\delta_{i,j}$	Kronecker delta function	
δ_{i-j}	binary interaction parameter between component i and j	-
η	dynamic viscosity	$\mathrm{kg} \cdot \mathrm{m}^{-1} \cdot \mathrm{s}^{-1}$

Γ_p	Dirichlet part of the boundary	
Γ_q	Neumann part of the boundary	
μ_i	chemical potential of i-th component	$\mathrm{J} \cdot \mathrm{mol}^{-1}$
ν_k	molar phase fraction of phase k	-
Ω	computational domain, $\Omega \subset \mathbb{R}^d$	
ω	Grand Potential	J
Φ_i	volume function coefficient of the i-th component	-
Π	number of phases in the equilibrium	
τ_Ω	spatial discretization of Ω	
Θ_i	coefficient in pressure equation, $\Theta_i = \frac{\partial p^{(\mathrm{eq})}}{\partial c_i}$	$\mathrm{Pa} \cdot \mathrm{m}^3 \cdot \mathrm{mol}^{-1}$
Υ^{Γ_p}	set of the Dirichlet boundary sides	
Υ^{Γ_q}	set of the Neumann boundary sides	
Υ_Ω	set of all sides of τ_Ω	
Υ_K	set of all sides of an element $K \in \tau_\Omega$	
ε	numerical tolerance	
φ_i	fugacity coefficient of the i-th component	-

Latin

A	Helmholtz free energy	J
a	Helmholtz free energy density	$\mathrm{J} \cdot \mathrm{m}^{-3}$
b_i	i-th co-volume coefficient of Peng–Robinson equation of state	$\mathrm{m}^3 \cdot \mathrm{mol}^{-1}$
c	total molar concentration (density), $c = \sum\limits_{i=1}^{n} c_i$	$\mathrm{mol} \cdot \mathrm{m}^{-3}$
$\mathbf{c}$	vector of molar concentration, $\mathbf{c} = (c_1, \ldots, c_n)^{\mathrm{T}}$	$\mathrm{mol} \cdot \mathrm{m}^{-3}$
c_i	molar concentration of the i-th component	$\mathrm{mol} \cdot \mathrm{m}^{-3}$
$c_{k,i}$	molar concentration of the i-th component of phase k	$\mathrm{mol} \cdot \mathrm{m}^{-3}$
c_p	molar heat capacity at constant pressure	$\mathrm{J} \cdot \mathrm{mol}^{-1} \cdot \mathrm{K}^{-1}$
c_v	molar heat capacity at constant volume	$\mathrm{J} \cdot \mathrm{mol}^{-1} \cdot \mathrm{K}^{-1}$
$\mathcal{D}$	feasible domain	
F	objective function in the phase equilibrium computation	
f_i	fugacity of the i-th component	Pa

F_i	volume function of the i-th component	Pa
$\mathrm{F}_{\mathrm{red}}$	reduced objective function in the phase equilibrium computation	
G	Gibbs free energy	J
$\widetilde{g}$	Gibbs free energy per mol	$\mathrm{J} \cdot \mathrm{mol}^{-1}$
H	enthalpy	J
$\widetilde{h}$	enthalpy per mol	$\mathrm{J} \cdot \mathrm{mol}^{-1}$
$\mathbf{K}$	intristic permeability	m^2
$k_{r,\alpha}$	relative permeability of phase α	-
K_i	the i-th equilibrium constant	-
N	total number of moles, $N = \sum\limits_{i=1}^{n} N_i$	mol
n	number of components in the mixture	
$N^{(k)}$	total number of moles in phase k, $N^{(k)} = \sum\limits_{i=1}^{n} N_{k,i}$	mol
N_i	mole number of the i-th component	mol
$N_{k,i}$	mole number of the i-th component in phase k	mol
N_A	Avogadro's number	6.02219×10^{23}
NP	size of the population in evolution algorithms	
$\mathcal{P}$	population in evolution algorithms	
$\breve{p}$	pressure trace	Pa
p, P	pressure	Pa
$P_{\mathrm{c},i}$	critical pressure of the i-th component	Pa
$P_i^{(\mathrm{sat})}$	saturation pressure of the i-th component	Pa
k_B	Boltzmann constant	$1.38064852 \times 10^{-23}\ \mathrm{m}^2 \cdot \mathrm{kg} \cdot \mathrm{K}^{-1} \cdot \mathrm{s}^{-2}$
R	universal gas constant	$8.3144598\ \mathrm{J} \cdot \mathrm{K}^{-1} \cdot \mathrm{mol}^{-1}$
S_k	saturation (volume fraction) of phase k	-
T	temperature	K
$T_{\mathrm{c},i}$	critical temperature of the i-th component	K
$T_{\mathrm{r},i}$	reduced temperature of the i-th component, $T_{\mathrm{r},i} = \frac{T}{T_{c,i}}$	-
$\mathbf{u}$	interstitial velocity	$\mathrm{m} \cdot \mathrm{s}^{-1}$
U	internal energy	J

$\mathrm{U_c}(a,b)$	uniform continuous distribution from a to b	
$\mathrm{U_d}(a,b)$	uniform discrete distribution from a to b	
$u_{K,E}$	velocity across side E in the outward direction with respect to element K	$\mathrm{m \cdot s^{-1}}$
u	internal energy density	$\mathrm{J \cdot m^{-3}}$
V	volume	$\mathrm{m^3}$
$\widetilde{v}$	molar volume	$\mathrm{m^3 \cdot mol^{-1}}$
$w_{K,E}$	basis function of $\mathbf{RTN}_0$ space	
$x_{k,i}$	molar fraction of the i-th component of phase k	-
$\mathbf{y}$	vector of unknowns in the general phase stability testing problem	
$\mathbf{y}^{(\mathrm{opt})}$	global minimum of function $\mathrm{TPD}(\mathbf{y};\mathbf{x}^*)$	
$\mathbf{Z}$	vector of unknowns in the reduced phase equilibrium computation	
$\mathbf{z}$	vector of unknowns in the phase equilibrium computation	
Z	compressibility factor	-
Superscripts		
$*$	initial phase indicator	
opt	optimum	
T	transposition	
eq	equilibrium	
ig	ideal gas	
PR	Peng–Robinson	
sat	saturation	
Subscripts		
c	critical property	
i	i-th component	
E	average value on side E	
K	average value on element K	
Other		
$\boldsymbol{\nabla}\cdot$	divergence operator of a vector function	
$\boldsymbol{\nabla}$	gradient operator of a scalar function	
$[\mathbf{x}_1,\dots,\mathbf{x}_m]_\kappa$	convex hull of vectors $\mathbf{x}_1,\dots,\mathbf{x}_m$	
$\widehat{n}$	$\{1,\dots,n\}$	

Introduction 1

The main topics of our research are the phase stability testing, phase equilibrium computation, and the multi-phase compositional simulations. We are mainly interested in numerical methods for solving the phase equilibrium computation and its consequent application in the compositional simulations of a multi-component, compressible Darcy's flow in a porous medium.

Consider a multi-component mixture at given physical conditions, e.g., the pressure, temperature, and mole numbers of the mixture are given. Our focus is to predict whether this mixture remains in one phase (liquid or gas), or the splitting occurs, and the mixture exists in two (or more) phases. This problem is known as the *phase stability testing problem.* Using the laws of thermodynamics, a mathematical formulation of the phase stability testing problem and, consequently, the criterion of stability can be derived. However, the phase stability testing only decides whether the given mixture remains in one phase. If the mixture is in two (or more) phases, the result of the phase stability testing does not say anything about the composition and amounts of the individual phases. If we need to know the composition and amounts of the phases (i.e., the equilibrium state of the mixture), the problem known as *phase equilibrium computation* has to be solved. The phase equilibrium computation is a more complicated task than the phase stability testing; however, using the laws of thermodynamics, a mathematical formulation and the phase equilibrium conditions can still be derived. Using the phase stability testing and phase equilibrium computation, multi-phase compositional fluid flows can be modeled. In these simulations, the composition and the number of phases of the fluid are studied. With the help of the phase equilibrium computation, the individual phases can be modeled as fully miscible, and according to the physical conditions the phases can appear and disappear during the simulations.

These models have broad usage in various areas. Our main focus is in the chemical engineering, e.g., for the CO_2 sequestration (Firoozabadi and Myint (2010); Zhang et al. (2020)) or the enhanced oil recovery (Olajire (2014)). However, we can mention other areas of the research where compositional simulations take place: medicine (Chernyavsky et al. (2010)), agriculture (Dayyani et al. (2010)), or hydrology (Trévisan and Periáñez (2016)). Moreover, this topic includes several branches of mathematics (e.g., mathematical modelling of fluid flow in porous media, optimization or numerical analysis) and combines it with branches of physics (mainly equilibrium thermodynamics).

1.1 Content of this thesis

This thesis is structured in the following way:

1. Chapter 1 introduces the field of the phase equilibrium computation and the compositional simulations. Furthermore, it contains a discussion of our research goals. Finally, it provides

the notation which is used in this thesis.

2. Chapter 2 presents the equilibrium thermodynamics. The thermodynamical postulates are introduced, and the essential equations and potentials are derived. Equations of state used in this thesis are presented. Moreover, individual forms of the most needed thermodynamical functions are presented.
3. Chapter 3 defines the phase stability testing problem. First, the unified formulation is presented. Second, various numerical algorithms for solving the phase stability testing problem are given. Examples showing the performance of the numerical algorithms are given.
4. Chapter 4 defines the phase equilibrium computation problem. The unified formulation is presented, and a numerical strategy for solving the phase equilibrium computation is proposed. A comparison of the individual phase equilibrium specifications is carried out. Examples showing the performance of the numerical algorithms are given.
5. Chapter 5 presents compositional simulation. A multi-phase iso-thermal model with phase equilibrium computation is presented. Conservation laws describing the physical model are given, and the numerical solution is derived. Examples showing the performance of the numerical algorithms are given.
6. Chapter 6 discusses the results of this thesis. Some conclusions are drawn.

This thesis was processed by the author using LaTeX. Technical drawings in this thesis were created using *Inkscape* (inkscape.org) and *Lucidchart* (lucidchart.com). For the visualisation of our numerical results, we use *Matlab* (mathworks.com) and *Paraview* (paraview.org) software.

1.2 Phase equilibrium computation — state of the art

The phase stability testing and the phase equilibrium computation are two of the basic problems of the chemical engineering and are closely related together.

The traditional formulation of these problems uses pressure P^*, temperature T^*, and mole numbers $N_1^*, \ldots, N_n^*$ (or mole fractions $x_1^*, \ldots, x_n^*$) as specification variables – this is the case of the so-called PTN-stability and PTN-flash equilibrium computation. Problems of PTN-flash and PTN-stability were investigated by many authors building mainly on the the classical works of Michelsen. Michelsen (1982a) defined the tangent plane function (TPD) and presented the criterion of phase stability. This criterion leads to an optimization task with the TPD function. Lastly, Michelsen (1982a) proposed a numerical optimization method based on direct substitution. Michelsen (1993, 1982b) described possible algorithms for the computation of the multi-phase equilibrium. One is direct substitution which leads to the well-known Rachford–Rice equation (see Rachford and Rice (1952)). The other proposed algorithm is based on the Newton–Raphson method with a modification of the Hessian matrix to ensure the descent of the objective function. Michelsen (1986a) presented a simplified phase equilibrium computation for an arbitrary cubic equation of state. Baker et al. (1982) presented a geometrical interpretation of the phase stability testing problem. Later, research focused on the development of faster and more robust algorithms. One direction of the research were reduction methods. The idea behind these methods is to exploit the structure of the problem to reduce the number of independent variables. This allows reducing the problem to a lower-dimensional space. As a result of the reduction, the size of the resulting systems of linear algebraic equations, which need to be solved, is substantially reduced. The first reduction method was proposed by Michelsen (1986b), where the binary interaction coefficient in the equation of state had to be zero. In this case, the number of independent variables can be reduced to three. Jensen and Fredenslund (1987) extended the phase equilibrium computation to one non-zero interaction coefficient. Later, different PTN-phase equilibrium computation reduction methods were presented by Nichita et al. (2006a); Pan and Firoozabadi (2003); Petitfrere and Nichita (2015a,b). A critical comparison has been presented by Haugen and Beckner (2013) or Gorucu and Johns (2014). Michelsen et al. (2013) gave a comparative study of reduction methods. The solution of the PTN-phase stability using the reduction algorithms were presented by Firoozabadi and Pan (2002); Hoteit and Firoozabadi (2006b); Nichita and Petitfrere (2013). Global optimization methods have been also investigated. Nichita et al. (2002) used a tunnelling method to find a global minimum. Souza et al. (2006) investigated a global optimization method using interval analysis. Another stability analysis based on the interval methods can be found in Stadtherr et al. (2007), where the PTN specification was used. Solving PTN-specification phase stability using a topographical global optimization method was investigated by Henderson et al. (2015). Zhang et al. (2011) presented a comparison of various approaches for the global optimization methods. Pan and Firoozabadi (1998) presented a solution of phase equilibrium computation based on the simulated annealing. More than two phase splits were investigated by Haugen et al. (2010). They proposed to use a two-dimensional bisection method to provide good initial guesses for the Newton–Raphson algorithm used to solve the three-phase Rachford–Rice equations. With the rise of new formulations, volume-based formulation of the PTN-specification has been investigated in Nichita (2018b,c).

Compared to PTN-stability and PTN-flash, other variables specifications are less common. One of the notable specifications uses volume V^*, temperature T^* and mole numbers $N_1^*, \ldots, N_n^*$ as specification variables (the so called VTN-stability and VTN-flash). First, Mikyška and Firoozabadi (2012) investigated the VTN-stability and derived a criterion for VTN-stability. Jindrová and Mikyška (2013) presented a numerical algorithm of the two-phase VTN-flash computation. Finally, Jindrová and Mikyška (2015) presented a general algorithm of an arbitrary Π-phase split computation and a general strategy for solving the phase equilibrium computations with an a priori unknown number of phases. A global approach using a Lagrangian function and solving the dual problem was presented by Pereira et al. (2010). Castier (2014) presented and compared PTN and VTN phase stability testing. Their approach uses molar volume and mole fractions of $n-1$ components as primary variables or alternatively natural logarithms of the mole fractions. This approach can be advantageous since a lot of thermodynamical models work with these variables. Various work in the VTN-specification was presented by Nichita, e.g., see Nichita (2017, 2018a); Nichita et al. (2009).

Another notable formulation uses the internal energy U^*, volume V^*, and mole numbers $N_1^*, \ldots, N_n^*$ as its specification variables (the so-called UVN-stability and UVN-flash). The UVN-flash specification is useful in non-isothermal problems, e.g., in the dynamic simulation of separation vessels (Castier (2010); Qiu et al. (2014)) or dynamic filling of a process vessel (Saha and Carroll (1997)). There are only few papers concerning the UVN-flash. First, Michelsen (1999a) has proposed a general framework for other variables specifications, including the UVN-specification. His approach used the PTN-flash in the inner loop while pressure and temperature are iterated in an outer loop with the goal of fulfilling the pertinent variable(s) specification(s). The second paper devoted to the UVN-flash equilibrium calculation is the paper Saha and Carroll (1997). Here, a sophisticated heuristics was developed to estimate the pressure and temperature corresponding to the given internal energy, volume, and moles. Initial K-values are estimated using the Wilson correlation and the estimated values of pressure and temperature. The solution method proposed by Saha and Carroll (1997) is not directly applicable to a system consisting from a single component. Therefore, the authors propose a special procedure for a single-component systems. Then, Castier (2009) presented a general strategy for calculating phase equilibrium at given UVN with an a priori unknown number of phases. In this strategy, to get a good initial phase split, one has to use estimates of pressure P and temperature T. In case of numerical difficulties, the algorithm is based on nested loops using the PTN-flash in the inner loop, while pressure and temperature are iterated in the outer loop with the goal of matching the remaining variables specification. Recently, Bi et al. (2020a,b) presented numerical solutions of the UVN-stability and flash based on the successive substitution method.

1.3 Application of the phase equilibrium computation in the multi-phase fluid flow in porous medium — state of the art

One of the possible applications of the phase equilibrium computation are the numerical simulations of a multi-phase flow in porous medium. However, there exist various formulations of the problem. First, the phases are assumed to be immiscible, and the composition of the phases is not studied. This problem in the literature is known as immiscible *two-phase flow*. In this situation, the phases are fully described by their pressures and saturations (volume fractions). For a homogeneous porous medium with neglected gravitational effects, McWhorter and Sunada (1990) derived exact (semi-analytical) solutions in 1D and 2D. Recently, Fučík et al. (2016) showed that the semi-analytical solution can also be obtained for the three-dimensional case. Another analytical solution exists for the two-phase flow without the capillarity effects (known as Buckley-Leverett equation), see Buckley and Leverett (1942); van Duijn et al. (2007). A large number of numerical methods based on finite differences (e.g., Jim Douglas (1983)), finite volumes (e.g., Durlofsky et al. (2007), or finite elements (e.g., Efendiev et al. (2006)) were proposed to solve the immiscible two-phase flow problems. However, the presented methods suffered from a low level of accuracy. Therefore, higher order methods were developed. Nayagum et al. (2004) used an approach based on the mixed-hybrid finite element method. A combination of the mixed-hybrid finite element method with discontinuous Galerkin method was proposed by Fučík et al. (2019); Fučík and Mikyška (2011) or Hoteit and Firoozabadi (2005, 2008).

Second, in contrast to the immiscible *two-phase flow*, we are interested in the composition of the phases. This problem is known as *compositional simulation*. Here, the mixture is generally a multi-component fluid, and the conservation of mass is described by the transport equation for each component of the mixture. In case the mixture is assumed to be in a single phase, the problem was studied extensively. Traditionally, fully implicit methods or sequential methods are used, see, e.g., Chen (2006) or Ewing (1983). Alternatively, the IMPEC approach can be used, see, e.g., Huyakorn (1983) or Acs et al. (1985). In this approach, the equations are solved in two steps. First, the pressure equation is solved implicitly to get the pressure field. Then, the concentrations of the first $n-1$ components are updated explicitly using the pressure from the previous step. The concentration of the last component is updated using the previous ones, the total concentration, and the equation of state. The conservation of mass holds for the $n-1$ components. However, for the last n-th component, the conservation of mass does not hold. Polívka and Mikyška (2011) presented an approach based on a combination of the mixed-hybrid finite element method and the finite volume method which leads to a system of linear algebraic equations. Recently, Chen et al. (2019) presented a novel iterative IMPEC scheme where the conservation of mass of all components is guaranteed. In contrast to the single-phase compositional simulations, multi-phase fluids can be studied as well. However, a thermodynamical model has to be included to determine the composition and amount of each phase. The numerical solution of this multi-phase problem followed a similar path as the single-phase simulations. Coats (1980) presented a scheme based on the finite difference method. However, the large numerical diffusion requires using fine meshes. Hoteit and Firoozabadi (2006a) presented a solution based on the mixed-hybrid finite element method and the discontinuous Galerkin method. The compositions of individual phases are computed using the PTN-specification. Moortgat et al. (2011) extended the numerical solution up to three phases using the unified approach of volume balance of Acs et al. (1985). The PTN-specification approach is the most used in literature. Sun et al. (2012) proposed single-domain and two-domain methods based on the mixed finite element method. Zidane and Firoozabadi (2019, 2020) presented a higher-order numerical model for two-phase compositional flow in fractured media. Specifications other than PTN are less common. Polívka and Mikyška (2014) used the VTN-specification and a fully implicit scheme, which gives the pressure field directly. However, this approach is computationally expensive. Later, Polívka and Mikyška (2015) improved their method with a semi-implicit time discretisation. In contrast to the fully implicit schemes, the size of the linear system does not depend on the number of mixture components.

1.4 Research goals

Based on the lists of scientific results in Sections 1.2 and 1.3, we formulated the following goals of our research. To the best of our knowledge, none of these goals has been achieved before.

1. Present the UVN-based phase stability testing problem.
2. Develop a unified framework for phase stability testing and phase equilibrium computation problems.
3. Develop a global optimization method based on the Branch and Bound strategy for solving the VTN-phase stability testing problem.
4. Develop a multi-phase compositional model for porous media with the VTN-flash computation and the iterative IMPEC scheme.

1.5 Achieved results

The thesis presents the following results that — to the best of our knowledge — have not been achieved before.

1. **A fast and robust algorithm for the general multi-phase equilibrium computation at constant internal energy, volume, and moles (specification UVN).** Unlike the previously published formulations, our method uses results of the UVN-phase stability testing for initialization of the UVN-flash computation. Compared to the previous works, the computational algorithm is simplified, treats both single-component and multi-component mixtures in the same way, and can be performed in real arithmetic only. The numerical difficulties mentioned by Castier (2009), which required some parts of the algorithm to be performed in the complex arithmetic, are thus avoided. We have published our findings in Smejkal and Mikyška (2017).

2. **A unified formulation of the phase-equilibrium and phase-stability problems for multi-component mixtures.** Furthermore, we develop a numerical method for solving the unified formulation of the phase-equilibrium problems. In contrast to other unified solver presented by Paterson et al. (2018), linearisation of the constraints and possibly some approximations as mentioned by Paterson et al. (2018) are not performed. In our formulation, in which we transform the equilibrium computation to a general optimization

problem with quadratic constraints, the solver remains the same for all possible variables specifications. We have published our findings in Smejkal and Mikyška (2018).

3. **A global optimization algorithm for the phase stability testing at constant temperature, volume, and moles (specification VTN).** The algorithm is based on the convex-concave splitting of the Helmholtz free energy density function and the Branch and Bound strategy. In one example, we managed to find a solution that previously used locally convergent algorithms have not detected. However, the computation times are higher than the times with a stand-alone gradient method, so the usability in large scale algorithms is only possible for mixtures with a small number of components. We have published our findings in Smejkal and Mikyška (2020).

4. **A simple procedure for solving systems of linear equations arising from the linearisation of the VTN-phase stability problem.** Using the structure of the Hessian matrix, the solution is obtained by sequential usage of the Sherman-Morrison formula. The great advantage of the method is that the system of equations can be solved without even assembling the system. The numerical experiments showed that the new algorithm was able to reduce the computation times if the number of components was at least 10. For smaller mixtures, the classical algorithm is preferred. We have published our findings in Smejkal and Mikyška (2021).

5. **The performance on solving the VTN-phase stability testing problem of five chosen heuristical algorithms: Differential Evolution, Cuckoo Search, Harmony Search, CMA-ES, and Elephant Herding Optimization.** For the comparison of the evolution strategies, we use the Wilcoxon signed-rank test. Moreover, we presented the expanded mirroring technique, which mirrors the computed solution into a given simplex. Based on the numerical experiments, the Differential Evolution and CMA-ES algorithm had the best performance. However, none of the evolution strategies found the solution in all cases. Therefore, the usage of these algorithms in the area of chemical engineering is limited. Even more, in comparison with the classical Newton–Raphson method with line-search, the computation times of the evolution strategies were significantly higher. We have published our findings in Smejkal et al. (2021).

6. **A numerical solution of a multi-phase compressible Darcy's flow of a multi-component mixture in a porous medium.** The mathematical model consists of mass conservation equation of each component, extended Darcy's law for each phase, and an appropriate set of the initial and boundary conditions. The phase split is computed using the phase equilibrium computation in the VTN-specification (known as VTN-flash). The transport equations are solved numerically using the mixed-hybrid finite element method and a novel iterative IMPEC scheme developed by Chen et al. (2019). We have published our findings in Smejkal and Mikyška (2021).

1.6 Challenges for the future

In the course of our research, we encountered other interesting problems that can be investigated in the future. Here, we mention few of them.

1. Give rigorous mathematically correct proofs of the equivalents principles to the maximum entropy principle. To the best of our knowledge, no rigorous proof has been published in the literature.

2. Implement and test other specifications of the phase equilibrium computation such as HPN or SVN. The HPN-specification can be used in the simulations of the Steam-assisted gravity drainage (SAGD), which is an important application of steam injection for recovery of extra-heavy oil and bitumen. See Zhu and Okuno (2016) and Butler (1991) for details.

3. Implement and test more complex equations of state such as PC-SAFT (Chapman et al. (1989)) or e-CPA (Kontogeorgis (2010)). The PC-SAFT equation of state can be applied to pure components and mixtures of non-associating and weakly polar components. According to Gross and Sadowski (2001), the PC-SAFT equation of state has a good agreement with experimental liquid-liquid data of a polyethylene-pentane mixture. The e-CPA equation of state is used to model the saline water. One of the applications is the modelling of the solubilities of the carbon dioxide in the aqueous solutions of inorganic salts, which is important for carbon storage, gas hydrate formation, and seawater desalination. See Sun et al. (2019) or Li et al. (2020) for details.

4. Extend the mathematical model and numerical solution of the multi-phase compositional flow to the non-isothermal problems. The compositional simulations without constant temperature can be carried out using the conservation of energy and the HPN- or UVN-specification of the phase equilibrium computation.

1.7 System of notation

Vectors and matrices are printed in the bold font, and their components are in the non-bold font, i.e., $\mathbf{v} = (v_1, \ldots, v_d)^{\mathrm{T}} \in \mathbb{R}^d$, where T denotes the transpose. Occasionally, the components of a vector $\mathbf{v}$ and a matrix $\mathbf{M}$ will be also denoted by $[\mathbf{v}]_i$ and $[\mathbf{M}]_{i,j}$, respectively. The Euclidean scalar product is denoted by $\cdot$, i.e.,

$$\mathbf{v}_1 \cdot \mathbf{v}_2 = \sum_{i=1}^{d} [\mathbf{v}_1]_i \, [\mathbf{v}_2]_i \, ,$$

where $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^d$. The open intervals are printed using round brackets. Closed intervals using the square bracket, i.e.,

$$(a, b) = \{x \in \mathbb{R}; a < x < b\}$$
$$[a, b] = \{x \in \mathbb{R}; a \leqslant x \leqslant b\} \, .$$

Set of positive integers not exceeding n is denoted by $\widehat{n}$, i.e.,

$$\widehat{n} = \{1, 2, \ldots, n\} \, .$$

Random variable X that follows some distribution D is expressed by $X \sim \mathrm{D}$. Unless said otherwise, all physical quantities that are used in this thesis are considered to be in the units stated in the Section Nomenclature.

Equilibrium thermodynamics 2

Thermodynamics is described by Callen (1985) as the study of the restrictions on the possible properties of matter that follow from the symmetry properties of the fundamental laws of physics. A more pleasant description is that thermodynamics is a branch of physics that deals with heat and processes related to it. The basic question of thermodynamics is *why is something observed in nature*. Thermodynamics is trying to give mathematical answers (theory) to these questions. The two main branches of thermodynamics are

- statistical,
- classical.

The statistical thermodynamics takes into account the individual particles and the interactions between them. The main object of this branch is to give the molecular theory of equilibrium properties of macroscopic systems (see Hill (1986)). On the other hand, the classical thermodynamics is trying to give answers without the knowledge of the molecular interactions. There exist other divisions of thermodynamics, e.g., equilibrium vs non-equilibrium (see Demirel (2014)) or the Gibbsian vs the Clausius–Kelvin. According to Tisza (1966), the Clausius–Kelvin thermodynamics considers a physical system as a black-box, and all information about the system is computed using the work done on the system and the change in the internal energy. On the other hand, in the Gibbsian thermodynamics the details of the black-box are known (i.e., the entropy function of the system, which will be defined later). In this work, we are interested in the classical Gibbsian equilibrium thermodynamics.

In this chapter, we review the basic principles of the equilibrium thermodynamics. We start with the principles of thermodynamics, which are basic for all derivations. Then, we present necessary equilibrium conditions for different physical systems. The thermodynamical potentials such as Helmholtz free energy or Gibbs free energy will be derived using the Legendre transformation. One of the most important statements is the entropy maximization principle, which describes the behaviour of the thermodynamical systems. In this chapter, we give equivalent principles such as the minimum energy principle or minimum Helmholtz free energy principle. Moreover, using properties of the thermodynamical potentials, constitutive equations such as the Gibbs–Duhem equation will be derived. These constitutive equations will be advantageous in later chapters, where the phase stability testing of the mixtures will be studied. This chapter will end with a presentation of the most needed constitutive relations for computational algorithms.

First, we will define terms used in this thesis. These terms are mainly adopted from Firoozabadi (2016).
Mole number Number of molecules divided by Avogadro's number, $N_A = 6.02219 \times 10^{23}$. The mole number of component i is denoted by N_i.

Mole fraction Mole number of species divided by the total number of moles, $\frac{N_i}{N}$, where N is total number of moles. The mole fraction of component i is denoted by x_i or z_i.

Critical temperature The maximum temperature at which a gas can be liquefied by pressure. Above this temperature, the substance cannot be liquefied, no matter how much pressure is applied. The critical temperature of the substance i is denoted by $T_{\mathrm{c},i}$.

Critical pressure The critical pressure of a substance is the pressure required to liquefy a gas at its critical temperature. The critical pressure of the substance i is denoted by $P_{\mathrm{c},i}$.

Saturation Pressure At a given temperature, saturation pressure is the pressure at which a given liquid and its vapour can co-exist in equilibrium. The saturation pressure of the substance i at temperature T is denoted by $P_i^{(\mathrm{sat})}(T)$.

Wall A physical system idealized as a surface forming the common boundary of two different systems. The walls that completely enclose a system are called enclosures. Walls separating the subsystems of a composite system are called partitions.

Closed system A system with a wall that is restrictive to the flow of matter.

Extensive variables Variables that depend on the total quantity of matter in the system or subsystem. Examples are volume and mole numbers.

Intensive variables Variables that have point values and are independent of the size of the system or subsystem. Examples are temperature and pressure.

Phase A region of space in which all physical properties are uniform. Examples are liquid phase and gas phase.

Internal energy Sum of the kinetic energy of the atoms and molecules in the system and the potential energy of the mutual interactions between the atoms and molecules. The internal energy is denoted by U.

Equilibrium state The state in which the properties are determined by intrinsic properties that are time-independent. A more precise definition will be presented later.

Next, we present the postulates of thermodynamics. These postulates are the basis of thermodynamics and can be stated in different but equivalent ways. Here, we present the formulations from Callen (1985). They are not definitions and cannot be proven. They exist based on the absence of an experience that disproves them.

Postulate 0 *The heat flux to a system in any process (at constant mole numbers) is simply the difference in internal energy between the final and initial states, diminished by the work done in that process.*

Writing the zeroth postulate mathematically, in an infinitesimal quasi-static process at constant mole numbers, the quasi-static heat δQ is defined by

$$\delta Q = \mathrm{d}U - \delta W, \tag{2.1}$$

where U is the internal energy. The previous equation is known as the first law of thermodynamics. The quasi-static work δW is associated with a change in volume and is given by

$$\delta W = -P\mathrm{d}V, \tag{2.2}$$

where P is the pressure and V is the volume.

Postulate I *There exist particular states (called equilibrium states) of systems that are characterized completely by the internal energy U, the volume V, and the mole numbers $N_1, \ldots, N_n$ of the chemical components.*

Postulate II *There exists an additive function S (called the entropy) of the extensive parameters with the following property: the values assumed by the extensive parameters in the absence of an internal constraint are those that maximize the entropy over the manifold of constrained equilibrium states.*

Mathematically, the additive property of S can be written as

$$S^{\Pi} = \sum_{\alpha=1}^{\Pi} S^{(\alpha)} = \sum_{\alpha=1}^{\Pi} S^{(\alpha)}\left(U^{(\alpha)}, V^{(\alpha)}, N_1^{(\alpha)}, \ldots, N_n^{(\alpha)}\right), \tag{2.3}$$

where $S^{(\alpha)}$ is the entropy of subsystem α. Combining Postulates I and II, we can derive that at equilibrium, the entropy $S^{(\Pi)}$ of the system is at maximum with respect to energies $U^{(\alpha)}$, volumes $V^{(\alpha)}$, and mole numbers $N_1^{(\alpha)}, \ldots, N_n^{(\alpha)}$, for $\alpha = 1, \ldots, \Pi$, such that

$$U^* = \sum_{\alpha=1}^{\Pi} U^{(\alpha)}, \tag{2.4}$$

$$V^* = \sum_{\alpha=1}^{\Pi} V^{(\alpha)}, \tag{2.5}$$

$$N_i^* = \sum_{\alpha=1}^{\Pi} N_i^{(\alpha)}, \qquad i \in \hat{n}, \tag{2.6}$$

where the symbol $\hat{n}$ represents a set of positive integers not exceeding n. We will refer to this property as the *maximum entropy principle*. In Section 2.1, we derive the necessary condition of equilibrium for a simple system with $\Pi = 2$. In Section 2.3, we will show equivalent principles based on other thermodynamical functions.

Postulate III *The entropy is continuous, differentiable, and monotonically increasing function of the internal energy such that*

$$\frac{\partial S}{\partial U}\left(U, V, N_1, \ldots, N_n\right) > 0. \tag{2.7}$$

Using the previous equation, a positive quantity called temperature

$$T\left(U, V, N_1, \ldots, N_n\right) = \frac{1}{\frac{\partial S}{\partial U}\left(U, V, N_1, \ldots, N_n\right)} \tag{2.8}$$

can be defined. Lastly, the continuity, differentiability, and monotonic property imply that the entropy function can be inverted with respect to the internal energy. Therefore, the internal energy function

$$U = U\left(S, V, N_1, \ldots, N_n\right) \tag{2.9}$$

can be uniquely defined. Now, we rewrite the first law of thermodynamics (2.1) to include the entropy function. The second law of thermodynamics states that for a quasi-static irreversible process, the change of heat δQ can be expressed as

$$\delta Q = T\mathrm{d}S. \tag{2.10}$$

This law can be derived from postulates I–III (see Callen (1985)). Therefore, combining equations (2.1) and (2.10) results in

$$\mathrm{d}U = T\mathrm{d}S - P\mathrm{d}V. \tag{2.11}$$

Recall that the previous equation holds only for a system with constant mole numbers of all chemical components. To include cases where the chemical composition is not constant, the chemical potential of the i-th component μ_i is defined as

$$\mu_i(S, V, N_1, \ldots, N_n) = \frac{\partial U}{\partial N_i}\left(S, V, N_1, \ldots, N_n\right). \tag{2.12}$$

In other words, the value of the chemical potential μ_i represents a change of the internal energy if we include a small amount of the i-th substance to the system without changing the entropy and volume of the system. The concept of chemical potentials was introduced by Gibbs (1876), see the collected work Gibbs (1948). Combining equations (2.11) and (2.12) gives

$$\mathrm{d}U = T\mathrm{d}S - P\mathrm{d}V + \sum_{i=1}^{n} \mu_i \mathrm{d}N_i \tag{2.13}$$

The previous equation fully describes the conservation of energy and will be the basis of all of our derivations. Moreover, we have

$$T(S, V, N_1, \ldots, N_n) = \frac{\partial U}{\partial S}\left(S, V, N_1, \ldots, N_n\right), \tag{2.14}$$

$$-P(S, V, N_1, \ldots, N_n) = \frac{\partial U}{\partial V}\left(S, V, N_1, \ldots, N_n\right), \tag{2.15}$$

$$\mu_i(S, V, N_1, \ldots, N_n) = \frac{\partial U}{\partial N_i}\left(S, V, N_1, \ldots, N_n\right), \quad i \in \widehat{n}. \tag{2.16}$$

Since the function $S\left(U, V, N_1, \ldots, N_n\right)$ and consequently $U(S, V, N_1, \ldots, N_n)$ have additive property, they have to be homogeneous of the first-order, i.e.,

$$U\left(\lambda S, \lambda V, \lambda N_1, \ldots, \lambda N_n\right) = \lambda U\left(S, V, N_1, \ldots, N_n\right), \quad \forall \lambda > 0. \tag{2.17}$$

Differentiating the previous equation with respect to λ and setting $\lambda = 1$ results in

$$U\left(S, V, N_1, \ldots, N_n\right) = TS - PV + \sum_{i=1}^{n} \mu_i N_i. \tag{2.18}$$

Postulate IV *The entropy of any system vanishes in the state for which*

$$\frac{\partial U}{\partial S}\left(S, V, N_1, \ldots, N_n\right) = 0 \tag{2.19}$$

that is at zero temperature.

The most important consequence is that entropy S has a uniquely defined zero value. This is not the case of internal energy U.

2.1 Necessary equilibrium conditions

In this section, we will present necessary equilibrium conditions for different systems. These conditions are direct consequences of Postulates II and III. In this section, for the sake of clarity, we will assume only a simple system consisting of two subsystems. However, using the additive property of the entropy function, generalization to an arbitrary system is straightforward. In each case, the system with internal energy U^*, mole numbers $N_1^*, \dots, N_n^*$ occupying volume V^* will be studied. The two subsystems will be separated by a wall with different properties. The subsystems can be viewed as individual phases of the system, e.g., the gas and liquid phase.

2.1.1 Thermal equilibrium

First, we consider a closed system separated by a rigid diathermic wall. The system is depicted in Figure 2.1. In this situation, the volumes and mole numbers of both subsystems are constant, and the internal energies obey

$$U^* = U^{(1)} + U^{(2)}. \tag{2.20}$$

Therefore, the first law of thermodynamics (2.13) reduces to

$$\mathrm{d}S^{(i)} = \frac{\mathrm{d}U^{(i)}}{T^{(i)}} \tag{2.21}$$

for both subsystems $i = 1, 2$. Using the additive property of the entropy function, the entropy of the system reads as

$$S^{\Pi} = S^{(1)} + S^{(2)} = S\left(U^{(1)}\right) + S\left(U^{(2)}\right) = S\left(U^{(1)}\right) + S\left(U^* - U^{(1)}\right), \tag{2.22}$$

where we used equation (2.20). Therefore, $S^{\Pi} = S^{\Pi}(U^{(1)})$, and the necessary condition for equilibrium arising from the maximum entropy principle is

$$0 = \frac{\partial S^{\Pi}}{\partial U^{(1)}}. \tag{2.23}$$

Using equations (2.21) and (2.22), the previous condition reads as

$$0 = \frac{1}{T^{(1)}} - \frac{1}{T^{(2)}}, \tag{2.24}$$

which is equivalent to the condition

$$T^{(1)} = T^{(2)}. \tag{2.25}$$

Therefore, at thermal equilibrium, the temperatures of the two subsystems are equal.

2.1.2 Mechanical equilibrium

Now, we consider a closed system consisting of two subsystems separated by a movable diathermic wall. Therefore, the only difference from Section 2.1.1 is that the volumes of the subsystems are not constant. The system is depicted in Figure 2.2. The first law of thermodynamics (2.13) in this case is

$$\mathrm{d}S^{(i)} = \frac{\mathrm{d}U^{(i)}}{T^{(i)}} + \frac{P^{(i)}\mathrm{d}V^{(i)}}{T^{(i)}}, \tag{2.26}$$

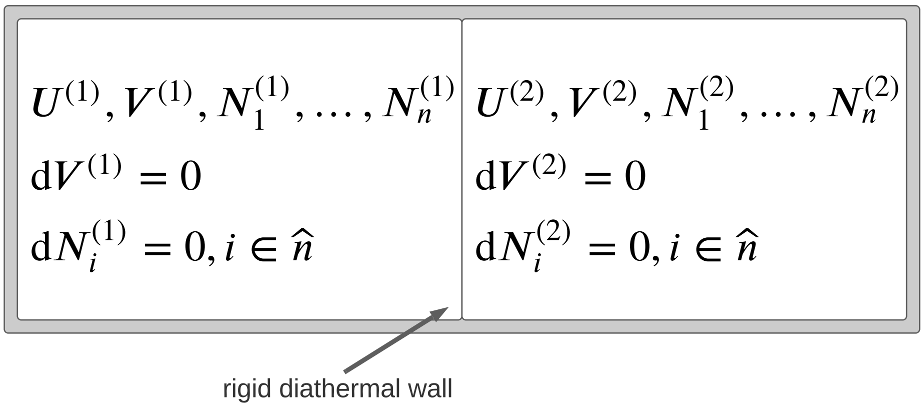

$U^{(1)}, V^{(1)}, N_1^{(1)}, \dots, N_n^{(1)}$
$dV^{(1)} = 0$
$dN_i^{(1)} = 0, i \in \hat{n}$
$U^{(2)}, V^{(2)}, N_1^{(2)}, \dots, N_n^{(2)}$
$dV^{(2)} = 0$
$dN_i^{(2)} = 0, i \in \hat{n}$
rigid diathermal wall

Figure 2.1: Thermal equilibrium scheme. The volumes and mole numbers of both subsystems are constant. The internal energies $U^{(i)}$ satisfy equation (2.20).

for both subsystems $i = 1, 2$. The internal energy and volume of the system are given by

$$U^* = U^{(1)} + U^{(2)}, \tag{2.27}$$
$$V^* = V^{(1)} + V^{(2)}. \tag{2.28}$$

Using a similar approach as in the previous section, the entropy of the system can be expressed as

$$S^{\Pi} = S^{(1)} + S^{(2)} = S\left(U^{(1)}, V^{(1)}\right) + S\left(U^* - U^{(1)}, V^* - V^{(1)}\right). \tag{2.29}$$

Therefore, $S^{\Pi} = S^{\Pi}\left(U^{(1)}, V^{(1)}\right)$, and the necessary conditions for equilibrium are

$$0 = \frac{\partial S^{\Pi}}{\partial U^{(1)}}, \tag{2.30}$$
$$0 = \frac{\partial S^{\Pi}}{\partial V^{(1)}}. \tag{2.31}$$

Using equations (2.26) and (2.29), the conditions read as

$$0 = \frac{1}{T^{(1)}} - \frac{1}{T^{(2)}}, \tag{2.32}$$
$$0 = \frac{P^{(1)}}{T^{(1)}} - \frac{P^{(2)}}{T^{(2)}}, \tag{2.33}$$

or equivalently

$$T^{(1)} = T^{(2)}, \tag{2.34}$$
$$P^{(1)} = P^{(2)}. \tag{2.35}$$

To conclude, at mechanical equilibrium, the temperatures and pressures of both systems are equal.

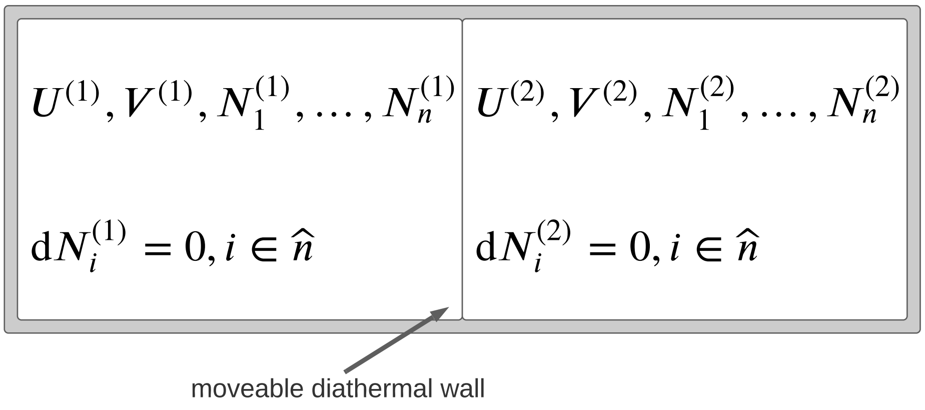

$U^{(1)}, V^{(1)}, N_1^{(1)}, \dots, N_n^{(1)}$
$U^{(2)}, V^{(2)}, N_1^{(2)}, \dots, N_n^{(2)}$
$\mathrm{d}N_i^{(1)} = 0, i \in \hat{n}$
$\mathrm{d}N_i^{(2)} = 0, i \in \hat{n}$
moveable diathermal wall

Figure 2.2: Mechanical equilibrium scheme. The mole numbers of both subsystems are constant. The volumes and internal energies satisfy equations (2.27) and (2.28).

2.1.3 Chemical equilibrium

Lastly, we consider a general case where the wall between two subsystems is movable and permeable to every component i of the mixture. The system is depicted in Figure 2.3. Now, the first law of thermodynamics (2.13) is in full form

$$\mathrm{d}S^{(i)} = \frac{\mathrm{d}U^{(i)}}{T^{(i)}} + \frac{P^{(i)}\mathrm{d}V^{(i)}}{T^{(i)}} - \frac{1}{T^{(i)}} \sum_{j=1}^{n} \mu_j^{(i)} \mathrm{d}N_j^{(i)}, \tag{2.36}$$

for both subsystems $i = 1, 2$. The internal energy, volume, and mole numbers of all components of the system are given by

$$U^* = U^{(1)} + U^{(2)}, \tag{2.37}$$
$$V^* = V^{(1)} + V^{(2)}, \tag{2.38}$$
$$N_i^* = N_i^{(1)} + N_i^{(2)}, \quad i \in \hat{n}. \tag{2.39}$$

Using equations (2.37)–(2.39), the entropy of the system is

$$\begin{aligned} S^{\Pi} &= S^{(1)} + S^{(2)} \\ &= S\left(U^{(1)}, V^{(1)}, N_1^{(1)}, \dots, N_n^{(1)}\right) + S\left(U^* - U^{(1)}, V^* - V^{(1)}, N_1^* - N_1^{(1)}, \dots, N_n^* - N_n^{(1)}\right), \end{aligned} \tag{2.40}$$

Therefore, $S^{\Pi} = S^{\Pi}\left(U^{(1)}, V^{(1)}, N_1^{(1)}, \dots, N_n^{(1)}\right)$, and the necessary conditions for equilibrium are

$$0 = \frac{\partial S}{\partial U^{(1)}}, \tag{2.41}$$
$$0 = \frac{\partial S}{\partial V^{(1)}}, \tag{2.42}$$
$$0 = \frac{\partial S}{\partial N_i^{(1)}}, \quad i \in \hat{n}. \tag{2.43}$$

Using equations (2.36) and (2.40), the conditions read as

$$0 = \frac{1}{T^{(1)}} - \frac{1}{T^{(2)}}, \tag{2.44}$$

$$0 = \frac{P^{(1)}}{T^{(1)}} - \frac{P^{(2)}}{T^{(2)}}, \tag{2.45}$$

$$0 = \frac{\mu_i^{(1)}}{T^{(1)}} - \frac{\mu_i^{(2)}}{T^{(2)}}, \quad i \in \widehat{n}, \tag{2.46}$$

or equivalently

$$T^{(1)} = T^{(2)}, \tag{2.47}$$

$$P^{(1)} = P^{(2)}, \tag{2.48}$$

$$\mu_i^{(1)} = \mu_i^{(2)}, \quad i \in \widehat{n}. \tag{2.49}$$

To conclude, at chemical equilibrium, the temperatures, pressures, and chemical potentials of all components of both subsystems are equal.

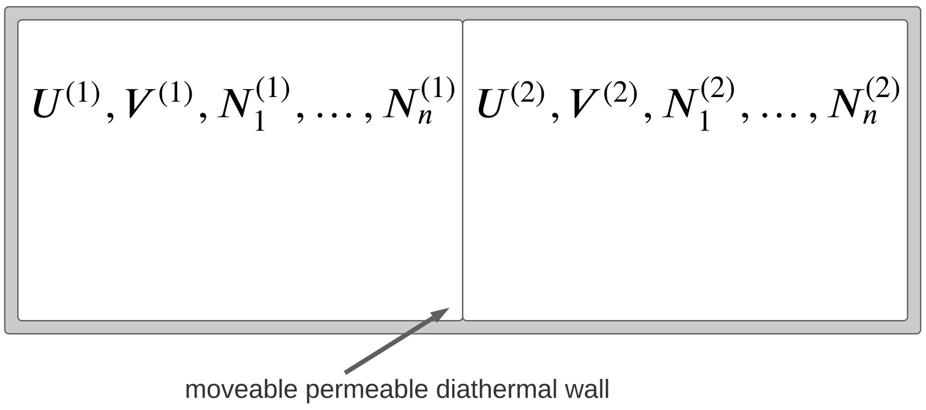

$U^{(1)}, V^{(1)}, N_1^{(1)}, \dots, N_n^{(1)}$
$U^{(2)}, V^{(2)}, N_1^{(2)}, \dots, N_n^{(2)}$
moveable permeable diathermal wall

Figure 2.3: Chemical equilibrium scheme. There is no constant property of the subsystem. The internal energies, volumes, and mole numbers satisfy equations (2.37)–(2.39).

2.2 Thermodynamical potentials

Using the first law of thermodynamics and the Legendre transformation (see, e.g., Sewell (1987)), other thermodynamical potentials can be derived. In this section, we briefly derive the most used ones. Moreover, we will define their densities if they are volume-based or their values per mol in other cases. The densities will be denoted by lower-case letters, the values per mol will be denoted by lower-case letters with a tilde symbol $\sim$.

2.2.1 Helmholtz free energy

The Helmholtz free energy $A = A(T, V, N_1, \dots, N_n)$ can be defined as

$$A = U - TS. \tag{2.50}$$

Therefore, using equation (2.13), the differential of A reads as

$$\mathrm{d}A = -S\mathrm{d}T - P\mathrm{d}V + \sum_{i=1}^{n} \mu_i \mathrm{d}N_i. \tag{2.51}$$

In the previous equation, the entropy S, pressure P, and chemical potentials μ_i for $i \in \widehat{n}$ are functions of variables $T, V, N_1, \ldots, N_n$, i.e.,

$$S = S(T, V, N_1, \ldots, N_n) = \frac{\partial A}{\partial T}(T, V, N_1 \ldots, N_n), \tag{2.52}$$

$$P = P(T, V, N_1, \ldots, N_n) = -\frac{\partial A}{\partial V}(T, V, N_1 \ldots, N_n), \tag{2.53}$$

$$\mu_i = \mu_i(T, V, N_1, \ldots, N_n) = \frac{\partial A}{\partial N_i}(T, V, N_1 \ldots, N_n), \quad i \in \widehat{n}. \tag{2.54}$$

Combining equations (2.18) and (2.50) gives

$$A(T, V, N_1, \ldots, N_n) = -P(T, V, N_1, \ldots, N_n)V + \sum_{i=1}^{n} N_i \mu_i(T, V, N_1, \ldots, N_n). \tag{2.55}$$

Moreover, we define the Helmholtz free energy density $a = a(T, c_1, \ldots, c_n)$ with

$$\mathrm{d}a = \frac{\mathrm{d}A}{V} = -\frac{S}{V}\mathrm{d}T + \sum_{i=1}^{n} \mu_i \mathrm{d}\frac{N_i}{V} = -s\mathrm{d}T + \sum_{i=1}^{n} \mu_i \mathrm{d}c_i, \tag{2.56}$$

where we defined entropy density $s = \frac{S}{V}$ and concentrations $c_i = \frac{N_i}{V}$ for $i \in \widehat{n}$.

Remark 2.1. In the application, where the Helmholtz free energy density is used, the temperature is usually held constant. Therefore, the dependence of a on the temperature is omitted, and we will write

$$a = a(c_1, \ldots, c_n). \tag{2.57}$$

Remark 2.2. The Helmholtz free energy is called 'free energy', because during a process with constant temperature and moles, the potential A is

$$\mathrm{d}A = -P\mathrm{d}V. \tag{2.58}$$

Therefore, up to the sign, the value is the maximal work that the closed system can do during an isothermal process.

2.2.2 Gibbs free energy

The Gibbs free energy $G = G(T, P, N_1, \ldots, N_n)$ can be defined as

$$G = U - TS + PV. \tag{2.59}$$

Therefore, using equation (2.13), the differential of G reads as

$$\mathrm{d}G = -S\mathrm{d}T + V\mathrm{d}P + \sum_{i=1}^{n} \mu_i \mathrm{d}N_i. \tag{2.60}$$

In the previous equation, the entropy S, volume V, and chemical potentials μ_i for $i \in \widehat{n}$ are functions of variables $T, P, N_1, \dots, N_n$, i.e.,

$$S = S(T,P,N_1,\dots,N_n) = -\frac{\partial G}{\partial T}(T,P,N_1,\dots,N_n), \tag{2.61}$$

$$V = V(T,P,N_1,\dots,N_n) = \frac{\partial G}{\partial P}(T,P,N_1,\dots,N_n), \tag{2.62}$$

$$\mu_i = \mu_i(T,P,N_1,\dots,N_n) = \frac{\partial G}{\partial N_i}(T,P,N_1,\dots,N_n), \quad i \in \widehat{n}. \tag{2.63}$$

Combining equations (2.18) and (2.59) gives

$$G(T,P,N_1,\dots,N_n) = \sum_{i=1}^{n} N_i \mu_i(T,P,N_1,\dots,N_n). \tag{2.64}$$

Moreover, we define the Gibbs free energy per one mol $\widetilde{g} = \widetilde{g}(T,P,x_1,\dots,x_n)$ with

$$\mathrm{d}\widetilde{g} = \frac{\mathrm{d}G}{N} = -\widetilde{s}\mathrm{d}T + \widetilde{v}\mathrm{d}P + \sum_{i=1}^{n} \mu_i \mathrm{d}x_i, \tag{2.65}$$

where we defined the entropy per mol $\widetilde{s} = \frac{S}{N}$, molar volume $\widetilde{v} = \frac{V}{N}$, and mole fractions $x_i = \frac{N_i}{N}$.

Remark 2.3. As in the Helmholtz free energy density, the dependence on the intensive variables is usually omitted, i.e., we will write

$$\widetilde{g} = \widetilde{g}(x_1,\dots,x_n). \tag{2.66}$$

Moreover, since $\sum_{i=1}^{n} x_i = 1$, the mole fractions $x_1,\dots,x_n$ are not independent variables, and the Gibbs per mol can be defined as a function of only $n-1$ mole fractions

$$\breve{g}(x_1,\dots,x_{n-1}) = \widetilde{g}(x_1,\dots,x_{n-1}, 1 - \sum_{i=1}^{n-1} x_i). \tag{2.67}$$

Remark 2.4. The Gibbs free energy, first defined by Gibbs (1873), was originally called available energy. Gibbs own description was: *the greatest amount of mechanical work which can be obtained from a given quantity of a certain substance in a given initial state, without increasing its total volume or allowing heat to pass to or from external bodies, except such as at the close of the processes are left in their initial condition.*

2.2.3 Enthalpy

The enthalpy $H = H(S,P,N_1,\dots,N_n)$ can be defined as

$$H = U + PV. \tag{2.68}$$

Therefore, using equation (2.13), the differential of H reads as

$$\mathrm{d}H = T\mathrm{d}S + V\mathrm{d}P + \sum_{i=1}^{n} \mu_i \mathrm{d}N_i. \tag{2.69}$$

In the previous equation, the temperature T, volume V, and chemical potentials μ_i for $i \in \hat{n}$ are functions of variables $S, P, N_1, \ldots, N_n$, i.e.,

$$T = T(S, P, N_1, \ldots, N_n) = \frac{\partial H}{\partial S}(S, P, N_1, \ldots, N_n), \tag{2.70}$$

$$V = V(S, P, N_1, \ldots, N_n) = \frac{\partial H}{\partial P}(S, P, N_1, \ldots, N_n), \tag{2.71}$$

$$\mu_i = \mu_i(S, P, N_1, \ldots, N_n) = \frac{\partial H}{\partial N_i}(S, P, N_1, \ldots, N_n), \quad i \in \hat{n}. \tag{2.72}$$

Combining equations (2.18) and (2.68) gives

$$H(S, P, N_1, \ldots, N_n) = T(S, P, N_1, \ldots, N_n)S + \sum_{i=1}^{n} N_i \mu_i(S, P, N_1, \ldots, N_n). \tag{2.73}$$

Moreover, we define the enthalpy per one mol $\widetilde{h} = \widetilde{h}(\widetilde{s}, P, x_1, \ldots, x_n)$ with

$$\mathrm{d}\widetilde{h} = \frac{\mathrm{d}H}{N} = T\mathrm{d}\widetilde{s} + \widetilde{v}\mathrm{d}P + \sum_{i=1}^{n} \mu_i \mathrm{d}x_i. \tag{2.74}$$

2.2.4 Grand potential

The Grand potential $\omega = \omega(T, V, \mu_1, \ldots, \mu_n)$ can be defined as

$$\omega = U - TS - \sum_{i=1}^{n} \mu_i N_i. \tag{2.75}$$

Therefore, using equation (2.13), the differential of ω reads as

$$\mathrm{d}\omega = -S\mathrm{d}T - P\mathrm{d}V - \sum_{i=1}^{n} N_i \mathrm{d}\mu_i. \tag{2.76}$$

In the previous equation, the entropy S, pressure P, and mole numbers N_i for $i \in \hat{n}$ are functions of variables $T, V, \mu_1, \ldots, \mu_n$, i.e.,

$$S = S(T, V, \mu_1, \ldots, \mu_n) = \frac{\partial \omega}{\partial T}(T, V, \mu_1, \ldots, \mu_n), \tag{2.77}$$

$$P = P(T, V, \mu_1, \ldots, \mu_n) = \frac{\partial \omega}{\partial V}(T, V, \mu_1, \ldots, \mu_n), \tag{2.78}$$

$$N_i = N_i(T, V, \mu_1, \ldots, \mu_n) = \frac{\partial \omega}{\partial \mu_i}(T, V, \mu_1, \ldots, \mu_n), \quad i \in \hat{n}. \tag{2.79}$$

Combining equations (2.18) and (2.75) gives

$$\omega(T, V, \mu_1, \ldots, \mu_n) = -P(T, V, \mu_1, \ldots, \mu_n)V \tag{2.80}$$

2.3 Equivalent principles to the Maximum entropy principle

The maximum entropy principle, given by principles of thermodynamics, can be mathematically written in the following theorem.

Theorem 2.5 (Maximum entropy principle by Callen (1985)). *The equilibrium value of any unconstrained internal parameter is such as to maximize the entropy for the given value of the total internal energy, volume, and moles.*

The following alternative statement is sufficient for our purposes in this thesis.

Theorem 2.6 (Maximum entropy principle)**.** *Consider a mixture of n components with mole numbers $N_1^*, \ldots, N_n^*$, internal energy U^* occupying volume V^*. Then, the system is at equilibrium, if the entropy of the system*

$$S^{\Pi} = \sum_{\alpha=1}^{\Pi} S\left(U^{(\alpha)}, V^{(\alpha)}, N_1^{(\alpha)}, \ldots, N_n^{(\alpha)}\right), \tag{2.81}$$

is maximal with respect to $U^{(\alpha)}, V^{(\alpha)}, N_1^{(\alpha)}, \ldots, N_n^{(\alpha)}$, $\alpha \in \widehat{\Pi}$, such that

$$U^* = \sum_{\alpha=1}^{\Pi} U^{(\alpha)}, \tag{2.82}$$

$$V^* = \sum_{\alpha=1}^{\Pi} V^{(\alpha)}, \tag{2.83}$$

$$N_i^* = \sum_{\alpha=1}^{\Pi} N_i^{(\alpha)}, \quad i \in \widehat{n}. \tag{2.84}$$

In this section, we present equivalent principles in terms of internal energy U, Helmholtz free energy A, or Gibbs free energy G. These equivalent formulations will be advantageous in the phase equilibrium computation. First, we present the minimum internal energy principle, which can be stated in the following way.

Theorem 2.7 (Minimum internal energy principle by Callen (1985))**.** *The equilibrium value of any unconstrained internal parameter is such as to minimize the internal energy for the given value of the total entropy, volume, and moles.*

Alternatively, one can express the principle as:

Theorem 2.8 (Minimum internal energy principle)**.** *Consider a mixture of n components with mole numbers $N_1^*, \ldots, N_n^*$, entropy S^* occupying volume V^*. Then, the system is at equilibrium, if the internal energy of the system*

$$U^{\Pi} = \sum_{\alpha=1}^{\Pi} U\left(S^{(\alpha)}, V^{(\alpha)}, N_1^{(\alpha)}, \ldots, N_n^{(\alpha)}\right), \tag{2.85}$$

is minimal with respect to $S^{(\alpha)}, V^{(\alpha)}, N_1^{(\alpha)}, \ldots, N_n^{(\alpha)}$, $\alpha \in \widehat{\Pi}$, such that

$$S^* = \sum_{\alpha=1}^{\Pi} S^{(\alpha)}, \tag{2.86}$$

$$V^* = \sum_{\alpha=1}^{\Pi} V^{(\alpha)}, \tag{2.87}$$

$$N_i^* = \sum_{\alpha=1}^{\Pi} N_i^{(\alpha)}, \quad i \in \widehat{n}. \tag{2.88}$$

Now, we would like to prove equivalence of Theorems 2.5 and 2.7. A physical argument (proof) is given in Callen (1985):

Physical argument: Let the system be in equilibrium and by the maximum entropy principle has the highest value of the entropy. Moreover, let the internal energy does not have its smallest possible value, i.e., we will give proof by contradiction. Therefore, we can lower the value of the internal energy by withdrawing it in the form of work. This action is possible to take place during constant entropy. In the second step, we return the withdrawn energy in the form of heat. Therefore, the entropy of the system will increase ($\delta Q = T\mathrm{d}S$). To conclude, we were able to increase the value of entropy in equilibrium. This is in contradiction with the maximum entropy principle, and we prove that the maximum entropy principle implies minimum internal energy principle. Now, we show the opposite implication. Let the system be in equilibrium and by the minimum internal energy principle has the lowest possible value of the internal energy. Moreover, let the entropy of the system is not maximal. Then, one can increase the entropy of the system by exchanging heat δQ from a heat source with the same temperature T as the system has. This iso-thermal process increases the internal energy of the system by δQ and the entropy of the system by $\frac{\delta Q}{T}$. Now, we can extract this internal energy from the system in the form of work. This process will not change the value of the entropy. Therefore, at this point, the system has the same internal energy as in the initial (supposedly equilibrium) state and increased value of the entropy. Now, we give back the heat δQ to the heat source which results in decreasing the internal energy by δQ and decreasing the value of entropy by $\frac{\delta Q}{T}$. Therefore, we were able to decrease the value of the internal energy which is in contradiction with the minimum internal energy principle.

To conclude, we were able to prove the equivalence of Theorems 2.5 and 2.7 using physical arguments. We would like to have a rigorous mathematical proof. However, we discovered that this task is beyond the scope of this thesis and it will not be presented here.

Similarly to the minimum internal energy principle, other principles based on the thermodynamical potentials derived in Section 2.2 can be stated. Here, we only state the principles and do not prove their equivalence with the maximum entropy principle.

Theorem 2.9 (Minimum Helmholtz free energy principle). *Consider a mixture of n components with mole numbers $N_1^*, \dots, N_n^*$ with temperature T^* occupying volume V^*. Then, the system is at equilibrium, if the Helmholtz energy of the system*

$$A^{\Pi} = \sum_{\alpha=1}^{\Pi} A\left(T^*, V^{(\alpha)}, N_1^{(\alpha)}, \dots, N_n^{(\alpha)}\right) \tag{2.89}$$

is minimal with respect to $V^{(\alpha)}, N_1^{(\alpha)}, \dots, N_n^{(\alpha)}$, $\alpha \in \widehat{\Pi}$, such that

$$V^* = \sum_{\alpha=1}^{\Pi} V^{(\alpha)}, \tag{2.90}$$

$$N_i^* = \sum_{\alpha=1}^{\Pi} N_i^{(\alpha)}, \quad i \in \widehat{n}. \tag{2.91}$$

Theorem 2.10 (Minimum Gibbs energy principle). *Consider a mixture of n components with mole numbers $N_1^*, \dots, N_n^*$ with temperature T^* and pressure P^*. Then, the system is at equilibrium, if the Gibbs energy of the system*

$$G^{\Pi} = \sum_{\alpha=1}^{\Pi} G\left(T^*, P^*, N_1^{(\alpha)}, \dots, N_n^{(\alpha)}\right), \tag{2.92}$$

is minimal with respect to $N_1^{(\alpha)}, \dots, N_n^{(\alpha)}$, $\alpha \in \widehat{\Pi}$, *such that*

$$N_i^* = \sum_{\alpha=1}^{\Pi} N_i^{(\alpha)}, \quad i \in \hat{n}. \tag{2.93}$$

Theorem 2.11 (Minimum Enthalpy energy principle)**.** *Consider a mixture of n components with mole numbers* $N_1^*, \dots, N_n^*$ *with entropy* S^* *and pressure* P^*. *Then, the system is at equilibrium, if the enthalpy of the system*

$$H^{\Pi} = \sum_{\alpha=1}^{\Pi} H\left(S^*, P^*, N_1^{(\alpha)}, \dots, N_n^{(\alpha)}\right), \tag{2.94}$$

is minimal with respect to $S^{(\alpha)}, N_1^{(\alpha)}, \dots, N_n^{(\alpha)}$, $\alpha \in \widehat{\Pi}$, *such that*

$$S^* = \sum_{\alpha=1}^{\Pi} S^{(\alpha)}, \tag{2.95}$$

$$N_i^* = \sum_{\alpha=1}^{\Pi} N_i^{(\alpha)}, \quad i \in \hat{n}. \tag{2.96}$$

2.4 Equations of state

One of the most critical equations in thermodynamics is the *equation of state* (EOS). This equation relates the state variables pressure P, temperature T, volume V, and mole numbers $N_1, \dots, N_n$. However, there does not exist a single equation that correctly describes every possible system. Therefore, for each system, a specific equation of state has to be used. Here, we present the equations of state used in this thesis. Most of the equations are pressure explicit, i.e.,

$$P = P^{(\mathrm{EOS})}(T, V, N_1, \dots, N_n), \tag{2.97}$$

or using the concentrations $c_i = \frac{N_i}{V}$

$$P = p^{(\mathrm{EOS})}(T, c_1, \dots, c_n) = P^{(\mathrm{EOS})}(T, 1, c_1, \dots, c_n). \tag{2.98}$$

2.4.1 Ideal gas

The ideal gas (ig) equation of state can be written as

$$P^{(\mathrm{ig})}(T, V, N_1, \dots, N_n) = \frac{RT}{V} \sum_{i=1}^{n} N_i, \tag{2.99}$$

where R is the universal gas constant. The ideal gas equation of state or ideal gas law is the equation of state of a hypothetical ideal gas. This equation can be used for many gases. However, its application is limited, e.g., it cannot be used to describe multi-phase states. According to Siccuan (2001), it was first stated by Clapeyron in 1834. Krönig (1856) and Clausius (1857) derived the ideal gas equation of state from the microscopic kinetic theory.

2.4.2 Peng–Robinson

The Peng–Robinson (PR) equation of state was presented by Peng and Robinson (1976). The equation was developed to satisfy:

- The parameters should be expressible in terms of the critical properties and the acentric factor.
- The model should provide reasonable accuracy near the critical point, particularly for calculations of the compressibility factor and liquid density.
- The mixing rules should not employ more than a single binary interaction parameter, which should be independent of temperature, pressure, and composition.
- The equation should be applicable to all calculations of all fluid properties in natural gas processes.

In this thesis, we will use the Peng–Robinson equation of state in the following form

$$p^{(PR)}\left(T, c_1, \ldots, c_n\right) = \frac{cRT}{1 - \sum\limits_{i=1}^{n} b_i c_i} - \frac{\sum\limits_{i,j=1}^{n} a_{ij} c_i c_j}{1 + 2\sum\limits_{i=1}^{n} b_i c_i - \left(\sum\limits_{i=1}^{n} b_i c_i\right)^2}. \tag{2.100}$$

The parameters a_{ij}, b_i of the Peng–Robinson equation of state are defined as

$$a_{ij}(T) = (1 - \delta_{i-j})\sqrt{a_i(T)a_j(T)}, \tag{2.101}$$

$$a_i(T) = 0.45724\frac{R^2 T_{\mathrm{c},i}^2}{P_{\mathrm{c},i}}\left[1 + m_i\left(1 - \sqrt{T_{\mathrm{r},i}}\right)\right]^2, \tag{2.102}$$

$$b_i = 0.0778\frac{RT_{\mathrm{c},i}}{P_{\mathrm{c},i}}, \tag{2.103}$$

$$m_i = \begin{cases} 0.37464 + 1.54226\omega_i - 0.26992\omega_i^2, & \omega_i < 0.5, \\ 0.3796 + 1.485\omega_i - 0.1644\omega_i^2 + 0.01667\omega_i^3, & \omega_i \geqslant 0.5. \end{cases} \tag{2.104}$$

In the above equations, $c = \sum\limits_{i=1}^{n} c_i$ is the molar density, R is the universal gas constant, δ_{i-j} is the binary interaction parameter between components i and j, $T_{\mathrm{c},i}, P_{\mathrm{c},i}$ are the critical temperature and critical pressure of component i, respectively, $T_{\mathrm{r},i}$ is the reduced temperature defined as $T_{\mathrm{r},i} = \frac{T}{T_{\mathrm{c},i}}$, and ω_i is the acentric factor of component i.

2.4.3 Cubic plus association

Associating systems are those which contain compounds capable of hydrogen bonding, e.g., alcohols, water, amines, acids, etc. In this thesis, we are mainly interested in mixtures containing water. Therefore, in this section, water will have an index $i = 1$ in all the expressions. The CPA (Cubic Plus Association) approach combines classical cubic equation of state (e.g., the Peng–Robinson) with an advanced association term

$$p^{(\mathrm{EOS})}\left(T, c_1, \ldots, c_n\right) = p^{(\mathrm{cubic})}\left(T, c_1, \ldots, c_n\right) + p^{(\mathrm{association})}\left(T, c_1, \ldots, c_n\right). \tag{2.105}$$

Kontogeorgis et al. (1996, 1999) developed the association term in a form

$$p^{(\text{association})}\left(T, c_1, \ldots, c_n\right) = 2RT\left(\frac{\eta}{g}\frac{\partial g}{\partial \eta} + 1\right)\sum_{i=1}^{n_{\text{cross}}}\left[c_i(\chi_i - 1)\right], \tag{2.106}$$

where n_{cross} is the number of pseudo-associating components for which the cross association coefficient $s_i \neq 0$. The coefficients a_i and b_i for water read as

$$a_1 = c_0\left[1 + c_1\left(1 - \sqrt{T_{r,1}}\right) + c_2\left(1 - \sqrt{T_{r,1}}\right)^2 + c_3\left(1 - \sqrt{T_{r,1}}\right)^3\right] \tag{2.107}$$

$$b_1 = 1.45843 \times 10^{-5}, \tag{2.108}$$

where $T_{r,1}$ is the reduced temperature of water and $c_0 = 0.09627$, $c_1 = 1.75573$, $c_2 = 0.00352$, $c_3 = -0.27464$. The χ_i coefficients are defined implicitly using

$$\chi_1 = \frac{1}{1 + 2c_1\chi_1\Delta^{\alpha\beta} + \sum\limits_{i=2}^{n} 2c_i\chi_i\Delta^{\alpha\beta_i'}}, \tag{2.109}$$

$$\chi_i = \frac{1}{1 + 2c_1\chi_1\Delta^{\alpha\beta_i'}}, \quad i = 2, \ldots, n, \tag{2.110}$$

where $\Delta^{\alpha\beta}$ is the association strength between molecules, and is given by

$$\Delta^{\alpha\beta} = g\kappa^{\alpha\beta}\left[\exp\left\{\varepsilon^{\alpha\beta}/k_B T - 1\right\}\right], \tag{2.111}$$

where k_B is the Boltzmann constant, $\kappa^{\alpha\beta}$ and $\varepsilon^{\alpha\beta}$ are the bonding volume and energy parameters of water, respectively, and g is the contact value of the radial distribution function of hard-sphere mixture that can be approximated as

$$g = g(\eta) = \frac{1}{1 - 1.9\eta}, \tag{2.112}$$

where $\eta = \sum\limits_{i=1}^{n} b_i c_i$. The association strength between water and pseudo-associating component i is related to the strength between water molecules as

$$\Delta^{\alpha\beta_i'} = s_i\Delta^{\alpha\beta}, \tag{2.113}$$

where s_i is the temperature-dependent cross association coefficient which can be determined together with the binary interaction coefficient δ_{i-j} by fitting the experimental data.

The computation of the χ_i coefficients from equations (2.109)–(2.110) is not a straightforward task. Michelsen (2006) gives a general numerical algorithm for solving system (2.109)–(2.110). Here, we are using successive iterations to solve the system.

Remark 2.12. Each presented equation of state has the following property

$$\lim_{V \to +\infty}\left[P^{(\text{EOS})}(T, V, N_1, \ldots, N_n) - P^{(\text{ig})}(T, V, N_1, \ldots, N_n)\right] = 0. \tag{2.114}$$

In other words, one can say that at the infinite volume, each mixture behaves as the ideal gas.

2.5 Important thermodynamical relations

In this section, we derive some well-known relations from equilibrium thermodynamics. These relations are primarily derived using basic theorems from multi-variable calculus.

2.5.1 Gibbs–Duhem equation

In Section 2.2, different thermodynamical potentials were derived with a transformation of variables of the internal energy, i.e., $S, V, N_1, \dots, N_n$. However, in any case, not all variables can be transformed. Transforming all variables results in

$$\Theta = U - TS + PV - \sum_{i=1}^{n} \mu_i N_i = 0, \tag{2.115}$$

where we used equation (2.18). Moreover,

$$0 = \mathrm{d}\Theta = \mathrm{d}G - \sum_{i=1}^{n} \mu_i \mathrm{d}N_i - \sum_{i=1}^{n} N_i \mathrm{d}\mu_i = -S\mathrm{d}T + V\mathrm{d}P - \sum_{i=1}^{n} N_i \mathrm{d}\mu_i. \tag{2.116}$$

Rearranging the previous equation results in the well-known Gibbs–Duhem equation. In the most common form, the equation reads as

$$S\mathrm{d}T - V\mathrm{d}P + \sum_{i=1}^{n} N_i \mathrm{d}\mu_i = 0. \tag{2.117}$$

This equation states that the intensive variables (temperature, pressure, and chemical potentials) are not independent and gives the relation between them.

2.5.2 Maxwell relations

Since all thermodynamical potentials defined in Section 2.2 are smooth functions, the equality of mixed partial derivatives can be used to derive relations between derivatives. See, e.g., Rudin (1976). The equations resulting from this approach are called *Maxwell relations*. For example, if we use internal energy $U = U(S, V, N_1, \dots, N_1)$ we can derive

$$\frac{\partial^2 U}{\partial S \partial V}(S, V, N_1, \dots, N_n) = \frac{\partial^2 U}{\partial V \partial S}(S, V, N_1, \dots, N_n). \tag{2.118}$$

Using equation (2.13), the first derivatives can be evaluated and the equality

$$-\frac{\partial P}{\partial S}(S, V, N_1, \dots, N_n) = \frac{\partial T}{\partial V}(S, V, N_1, \dots, N_n) \tag{2.119}$$

holds. Similarly, other relations from internal energy can be obtained

$$\frac{\partial T}{\partial N_i}(S, V, N_1, \dots, N_n) = \frac{\partial \mu_i}{\partial S}(S, V, N_1, \dots, N_n), \tag{2.120}$$

$$-\frac{\partial P}{\partial N_i}(S, V, N_1, \dots, N_n) = \frac{\partial \mu_i}{\partial V}(S, V, N_1, \dots, N_n), \tag{2.121}$$

$$\frac{\partial \mu_i}{\partial N_j}(S, V, N_1, \dots, N_n) = \frac{\partial \mu_j}{\partial N_i}(S, V, N_1, \dots, N_n). \tag{2.122}$$

Using a similar approach, sets of relations can be obtained for each thermodynamical potential from Section 2.2. The note-worthy are

$$\frac{\partial S}{\partial V}(T, V, N_1, \dots, N_n) = \frac{\partial P}{\partial T}(T, V, N_1, \dots, N_n), \tag{2.123}$$

$$\frac{\partial S}{\partial P}(T, P, N_1, \dots, N_n) = -\frac{\partial V}{\partial T}(T, P, N_1, \dots, N_n), \tag{2.124}$$

since on the right-hand side of these equations are physically measurable values.

2.5.3 Mayer equation

Mayer equation tells us the relation between molar heat capacities at constant pressure c_p and volume c_v. These values are defined as

$$c_p = \frac{\partial \widetilde{h}^\Diamond}{\partial T}(T, P), \tag{2.125}$$

$$c_v = \frac{\partial \widetilde{u}^\Diamond}{\partial T}(T, \widetilde{v}), \tag{2.126}$$

where $\widetilde{u}^\Diamond$, $\widetilde{h}^\Diamond$ are internal energy and enthalpy per mol, respectively. However, these function are functions of different variables than the ones defined in Section 2.2. Therefore, the superscript $\Diamond$ is used. The enthalpy $\widetilde{h}^\Diamond$ can be expressed using $\widetilde{h}$ by

$$\widetilde{h}^\Diamond(T, P) = \widetilde{h}\left(\widetilde{s}^\Diamond(T, P), P\right). \tag{2.127}$$

Therefore,

$$c_p = \frac{\partial \widetilde{h}^\Diamond}{\partial T}(T, P) = \frac{\partial \widetilde{h}}{\partial \widetilde{s}}\left(\widetilde{s}^\Diamond(T, P), P\right)\frac{\partial \widetilde{s}^\Diamond}{\partial T} = T\frac{\partial \widetilde{s}^\Diamond}{\partial T}(T, P), \tag{2.128}$$

where $T = \frac{\partial \widetilde{h}}{\partial \widetilde{s}}\left(\widetilde{s}^\Diamond(T, P), P\right)$. Moreover,

$$\mathrm{d}\widetilde{s}^\Diamond = \frac{\partial \widetilde{s}^\Diamond}{\partial T}(T, P)\,\mathrm{d}T + \frac{\partial \widetilde{s}^\Diamond}{\partial P}(T, P)\,\mathrm{d}P = \frac{c_p}{T}\mathrm{d}T - \frac{\partial \widetilde{v}}{\partial T}(T, P)\,\mathrm{d}P, \tag{2.129}$$

where we used the Maxwell relation

$$\frac{\partial \widetilde{s}^\Diamond}{\partial P}(T, P) = -\frac{\partial \widetilde{v}}{\partial T}(T, P), \tag{2.130}$$

which is obtained from the Gibbs per mol function. Using a similar procedure, the entropy in variables $T, \widetilde{v}$ can be expressed as

$$\mathrm{d}\widetilde{s}^\Diamond = \frac{c_v}{T}\mathrm{d}T + \frac{\partial P}{\partial T}(T, \widetilde{v})\,\mathrm{d}\widetilde{v}. \tag{2.131}$$

Combining equations (2.129) and (2.131) results in

$$\frac{c_p - c_v}{T}\mathrm{d}T = \frac{\partial \widetilde{v}}{\partial T}(T, P)\,\mathrm{d}P + \frac{\partial P}{\partial T}(T, \widetilde{v})\,\mathrm{d}\widetilde{v}. \tag{2.132}$$

At the same time

$$\mathrm{d}T = \frac{\partial T}{\partial P}(P, \widetilde{v})\,\mathrm{d}P + \frac{\partial T}{\partial \widetilde{v}}(P, \widetilde{v})\,\mathrm{d}\widetilde{v}. \tag{2.133}$$

Comparing the coefficients by $\mathrm{d}\widetilde{v}$ in equations (2.132) and (2.133) results in

$$\frac{c_p - c_v}{T}\frac{\partial T}{\partial \widetilde{v}}(P, \widetilde{v}) = \frac{\partial P}{\partial T}(T, \widetilde{v}), \tag{2.134}$$

or alternatively

$$c_p - c_v = T\frac{\partial P}{\partial T}(T, \widetilde{v})\frac{\partial \widetilde{v}}{\partial T}(P, \widetilde{v}). \tag{2.135}$$

This equation is known as the Mayer equation.

Remark 2.13. Using different equations of state, special forms of the Mayer equation can be obtained. With the ideal gas equation of state from Section 2.4.1, the Mayer equation (2.135) becomes

$$c_p - c_v = R, \tag{2.136}$$

where R is the universal gas constant.

2.5.4 Wilson correlation

Consider a mixture of n components at temperature T^* and pressure P^*. Let the mixture be in a two-phase state. Then, one can define the so-called K-factors or K_i-factors by

$$K_i = \frac{x_i^{(1)}}{x_i^{(2)}}, \quad i \in \widehat{n}, \tag{2.137}$$

where $x_i^{(1)}$ and $x_i^{(2)}$ are the mole fractions of the i-th component in phase one and two, respectively. K_i factors are also called *equilibrium constants* or *Henry's constants.* Determination of these constants is the object of the phase equilibrium computation. However, there exist approximations based on critical properties of the individual components of the mixture. Wilson (1969) approximates the equilibrium K_i-factors as

$$K_i \approx \frac{P_{c,i}}{P^*} \exp\left\{5.42\left(1 - \frac{T_{c,i}}{T^*}\right)\right\}, \tag{2.138}$$

where $T_{\mathrm{c},i}, P_{\mathrm{c},i}$ are the critical temperature and critical pressure of component i, respectively. The previous equation is known as *Wilson correlation.* The correlation gives reasonable results only for mixtures at low pressures and away from critical points. However, this correlation is an excellent instrument to initialize the phase equilibrium or phase stability testing problem. For more information, see Chapter 3.

2.6 Fugacities and volume functions

In Section 2.1.3, we derived that one set of the necessary conditions for chemical equilibrium is

$$\mu_i^{(1)} = \mu_i^{(2)}, \quad i \in \widehat{n}. \tag{2.139}$$

Therefore, it is crucial to be able to evaluate the difference

$$\mu_i^{(1)} - \mu_i^{(2)} \tag{2.140}$$

in the equilibrium computation. In this section, we present two approaches for calculating this difference. First, the two states will be defined by their temperature and pressure (PTN approach). Second, the volume and temperature will be used (VTN approach). In this section, it is going to be necessary to differentiate these two approaches and the denotation between the chemical potential derived from the Helmholtz free energy (defined by equation (2.54)) and chemical potential derived from the Gibbs free energy (defined by equation (2.63)). Therefore, in this section, we denote

$$\mu_i^{(A)}(T, V, N_1, \ldots, N_n) = \frac{\partial A}{\partial N_i}(T, V, N_1, \ldots, N_n), \tag{2.141}$$

$$\mu_i^{(G)}(T, P, N_1, \ldots, N_n) = \frac{\partial G}{\partial N_i}(T, P, N_1, \ldots, N_n), \tag{2.142}$$

where the superscript denotes from which potential the chemical potential is defined. In other sections, it is clear from the context which chemical potential is used, and the superscript would make it difficult to read.

Remark 2.14. The chemical potential is defined up to a constant. However, the difference of chemical potentials is defined precisely.

2.6.1 Fugacities

The thermodynamic quantity fugacity of i-th component $f_i = f_i(T, P, N_1, \dots, N_n)$ is defined as

$$\mu_i^{(G)}(T, P_2, N_1, \dots, N_n) - \mu_i^{(G)}(T, P_1, N_1, \dots, N_n) = RT \ln \frac{f_i(T, P_2, N_1, \dots, N_n)}{f_i(T, P_1, N_1, \dots, N_n)}, \tag{2.143}$$

and

$$\lim_{P \to 0+} \frac{f_i(T, P, N_1, \dots, N_n)}{x_i P} = 1, \tag{2.144}$$

where $x_i = \frac{N_i}{\sum\limits_{i=1}^{n} N_i}$ is the mole fraction of component i. Moreover, we denote

$$\varphi_i(T, P, N_1, \dots, N_n) = \frac{f_i(T, P, N_1, \dots, N_n)}{x_i P}. \tag{2.145}$$

Then, equation (2.144) reads as

$$\lim_{P \to 0} \varphi_i = 1. \tag{2.146}$$

The quantity φ_i is known as the fugacity coefficient. However, the computation of the fugacity from the definition is not pleasant if we do not know how to compute the chemical potentials. Therefore, we derive a formula for the computation of the fugacity from the state variables $T, P, N_1, \dots, N_n$. However, since most of the equations of state are pressure explicit, in the second part of this section, another formula based on $T, V, N_1, \dots, N_n$ will be given. Combining the equation

$$\mu_i^{(G)}(T, P_2, N_1, \dots, N_n) - \mu_i^{(G)}(T, P_1, N_1, \dots, N_n) = \int\limits_{P_1}^{P_2} \frac{\partial \mu_i^{(G)}}{\partial P}(T, P, N_1, \dots, N_n) \, \mathrm{d}P, \tag{2.147}$$

with equation (2.143) results in

$$RT \ln \frac{f_i(T, P_2, N_1, \dots, N_n)}{f_i(T, P_1, N_1, \dots, N_n)} = \int\limits_{P_1}^{P_2} \frac{\partial \mu_i^{(G)}}{\partial P}(T, P, N_1, \dots, N_n) \, \mathrm{d}P. \tag{2.148}$$

Using the Maxwell relation

$$\frac{\partial \mu_i^{(G)}}{\partial P}(T, P, N_1, \dots, N_n) = \frac{\partial V}{\partial N_i}(T, P, N_1, \dots, N_n), \tag{2.149}$$

equation (2.148) reads as

$$RT \ln \frac{f_i(T, P_2, N_1, \dots, N_n)}{f_i(T, P_1, N_1, \dots, N_n)} = \int\limits_{P_1}^{P_2} \frac{\partial V}{\partial N_i}(T, P, N_1, \dots, N_n) \, \mathrm{d}P. \tag{2.150}$$

Since

$$\frac{f_i(T, P_2, N_1, \dots, N_n)}{f_i(T, P_1, N_1, \dots, N_n)} = \frac{\varphi_i(T, P_2, N_1, \dots, N_n)}{\varphi_i(T, P_1, N_1, \dots, N_n)} \frac{P_2}{P_1}, \tag{2.151}$$

equation (2.150) can be written as

$$\ln \varphi_i(T, P_1, N_1, \ldots, N_n) = \int\limits_{P_1}^{P_2} \left[\frac{1}{P} - \frac{1}{RT} \frac{\partial V}{\partial N_i} (T, P, N_1, \ldots, N_n) \right] \mathrm{d}P + \ln \varphi_i(T, P_2, N_1, \ldots, N_n). \tag{2.152}$$

Taking limit $P_2 \to 0$ and using equation (2.146) results in

$$\ln \varphi_i\left(T, P_1, N_1, \ldots, N_n\right) = \int\limits_{P_1}^{0} \left[\frac{1}{P} - \frac{1}{RT} \frac{\partial V}{\partial N_i} (T, P, N_1, \ldots, N_n) \right] \mathrm{d}P. \tag{2.153}$$

Lastly, switching integration limits and setting $P_1 = P$ gives us the final formula for the computation of the fugacity coefficient

$$\ln \varphi_i\left(T, P, N_1, \ldots, N_n\right) = \int\limits_{0}^{P} \left[\frac{1}{RT} \frac{\partial V}{\partial N_i} \left(T, \breve{P}, N_1, \ldots, N_n\right) - \frac{1}{\breve{P}} \right] \mathrm{d}\breve{P}. \tag{2.154}$$

Therefore, the fugacity coefficient can be computed if we know the relation

$$V = V(T, P, N_1, \ldots, N_n). \tag{2.155}$$

However, this relation is not a standard relation, since the most equations of state are explicit in pressure, i.e., $P = P^{(\mathrm{EOS})}(T, V, N_1, \ldots, N_n)$. Therefore, to have a usable formula, a different derivation is necessary. Let the equation of state $P = P^{(\mathrm{EOS})}\left(T, V, N_1, \ldots, N_n\right)$ be given. Then, the difference of two chemical potentials at different pressures can be written as

$$\begin{aligned} &\mu_i^{(G)}\left(T, P_2, N_1, \ldots, N_n\right) - \mu_i^{(G)}\left(T, P_1, N_1, \ldots, N_n\right) \\ &\qquad = \mu_i^{(G)}\left(T, P_2, N_1, \ldots, N_n\right) - \mu_i^{(A)}\left(T, V_2, N_1, \ldots, N_n\right) \\ &\qquad\quad + \mu_i^{(A)}\left(T, V_2, N_1, \ldots, N_n\right) - \mu_i^{(A)}\left(T, V_1, N_1, \ldots, N_n\right) \\ &\qquad\quad + \mu_i^{(A)}\left(T, V_1, N_1, \ldots, N_n\right) - \mu_i^{(G)}\left(T, P_1, N_1, \ldots, N_n\right), \end{aligned} \tag{2.156}$$

where the volumes V_1, V_2 are defined as

$$P_1 = P^{(\mathrm{EOS})}\left(T, V_1, N_1, \ldots, N_n\right), \tag{2.157}$$

$$P_2 = P^{(\mathrm{EOS})}\left(T, V_2, N_1, \ldots, N_n\right). \tag{2.158}$$

Since the chemical potentials at the same physical conditions have to have identical value, we can write

$$\mu_i^{(G)}\left(T, P_2, N_1, \ldots, N_n\right) = \mu_i^{(A)}\left(T, V_2, N_1, \ldots, N_n\right), \tag{2.159}$$

$$\mu_i^{(G)}\left(T, P_1, N_1, \ldots, N_n\right) = \mu_i^{(A)}\left(T, V_1, N_1, \ldots, N_n\right). \tag{2.160}$$

Combining equations (2.156) and (2.159) gives

$$\begin{aligned} &\mu_i^{(G)}\left(T, P_2, N_1, \ldots, N_n\right) - \mu_i^{(G)}\left(T, P_1, N_1, \ldots, N_n\right) \\ &\qquad = \mu_i^{(A)}\left(T, V_2, N_1, \ldots, N_n\right) - \mu_i^{(A)}\left(T, V_1, N_1, \ldots, N_n\right) \\ &\qquad = \int_{V_1}^{V_2} \frac{\partial \mu_i^{(A)}}{\partial V} \left(T, V, N_1, \ldots, N_n\right) \mathrm{d}V \\ &\qquad = -\int_{V_1}^{V_2} \frac{\partial P}{\partial N_i} \left(T, V, N_1, \ldots, N_n\right) \mathrm{d}V, \end{aligned} \tag{2.161}$$

where we used Maxwell relation

$$\frac{\partial \mu_i^{(A)}}{\partial V}(T, V, N_1, \ldots, N_n) = -\frac{\partial P}{\partial N_i}(T, V, N_1, \ldots, N_n). \tag{2.162}$$

Moreover, from equation (2.151) we obtain

$$RT \ln \frac{f_i(T, P_2, N_1, \ldots, N_n)}{f_i(T, P_1, N_1, \ldots, N_n)} = RT \ln \frac{\varphi_i(T, P_2, N_1, \ldots, N_n)}{\varphi_i(T, P_1, N_1, \ldots, N_n)} + RT \ln \frac{P_2}{P_1}. \tag{2.163}$$

The fraction $\frac{P_2}{P_1}$ can be written as

$$\begin{aligned}\frac{P_2}{P_1} &= \frac{P^{(\mathrm{EOS})}(T, V_2, N_1, \ldots, N_n)}{P^{(\mathrm{EOS})}(T, V_1, N_1, \ldots, N_n)} \frac{P^{(\mathrm{ig})}(T, V_2, N_1, \ldots, N_n)}{P^{(\mathrm{ig})}(T, V_2, N_1, \ldots, N_n)} \frac{P^{(\mathrm{ig})}(T, V_1, N_1, \ldots, N_n)}{P^{(\mathrm{ig})}(T, V_1, N_1, \ldots, N_n)} \\ &= \frac{Z(T, V_2, N_1, \ldots, N_n)}{Z(T, V_1, N_1, \ldots, N_n)} \frac{V_1}{V_2}\end{aligned} \tag{2.164}$$

where we used the ideal gas equation of state (2.99), and defined the compressibility factor Z with

$$Z(T, V, N_1, \ldots, N_n) = \frac{P^{(\mathrm{EOS})}(T, V, N_1, \ldots, N_n)}{P^{(\mathrm{ig})}(T, V, N_1, \ldots, N_n)}. \tag{2.165}$$

Combining equations (2.150), (2.161), (2.163), and (2.164) gives

$$\begin{aligned}RT \ln \frac{\varphi_i(T, P_2, N_1, \ldots, N_n)}{\varphi_i(T, P_1, N_1, \ldots, N_n)} &+ RT \ln \left[\frac{Z(T, V_2, N_1, \ldots, N_n)}{Z(T, V_1, N_1, \ldots, N_n)} \frac{V_1}{V_2}\right] \\ &= -\int_{V_1}^{V_2} \frac{\partial P}{\partial N_i}(T, V, N_1, \ldots, N_n)\, \mathrm{d}V,\end{aligned} \tag{2.166}$$

or equivalently

$$\begin{aligned}RT \ln \frac{\varphi_i(T, P_2, N_1, \ldots, N_n)}{\varphi_i(T, P_1, N_1, \ldots, N_n)} &+ RT \ln \left[\frac{Z(T, V_2, N_1, \ldots, N_n)}{Z(T, V_1, N_1, \ldots, N_n)}\right] \\ &= \int_{V_1}^{V_2} \left[\frac{RT}{V} - \frac{\partial P}{\partial N_i}(T, V, N_1, \ldots, N_n)\right] \mathrm{d}V,\end{aligned} \tag{2.167}$$

Moreover, in limit $V_1 \to +\infty$, we have

$$\lim_{V_1 \to +\infty} Z(T, V_1, N_1, \ldots, N_n) = 1. \tag{2.168}$$

Therefore, combining equations (2.146), (2.167), and (2.168) gives the final expression in terms of $T, V, N_1, \ldots, N_n$

$$\begin{aligned}\ln \varphi_i(T, P_2, N_1, \ldots, N_n) &= \int_{+\infty}^{V_2} \left[\frac{1}{V} - \frac{1}{RT} \frac{\partial P}{\partial N_i}(T, V, N_1, \ldots, N_n)\right] \mathrm{d}V \\ &\qquad - \ln Z(T, V_2, N_1, \ldots, N_n) \\ &= \int_{V_2}^{+\infty} \left[\frac{1}{RT} \frac{\partial P}{\partial N_i}(T, V, N_1, \ldots, N_n) - \frac{1}{V}\right] \mathrm{d}V \\ &\qquad - \ln Z(T, V_2, N_1, \ldots, N_n),\end{aligned} \tag{2.169}$$

where the volume V_2 is given by the chosen equation of state

$$P_2 = P^{(\mathrm{EOS})}\left(T, V_2, N_1, \ldots, N_n\right). \tag{2.170}$$

Equation (2.169) in this form is written in many thermodynamics books, e.g., Firoozabadi (2016); Michelsen and Mollerup (2007); Prausnitz et al. (1999). However, this form is not entirely correct since on the left-hand side is a function of $T, P, N_1, \ldots, N_n$, and on the right-hand side is a function of $T, V, N_1, \ldots, N_n$. The precise form of equation (2.169) is

$$\begin{aligned}\ln \varphi_i\left(T, V_2, N_1, \ldots, N_n\right) = \int_{V_2}^{+\infty} \left[\frac{1}{RT}\frac{\partial P}{\partial N_i}\left(T, V, N_1, \ldots, N_n\right) - \frac{1}{V}\right] \mathrm{d}V \\ - \ln Z\left(T, V_2, N_1, \ldots, N_n\right),\end{aligned} \tag{2.171}$$

or

$$\begin{aligned}&\ln \varphi_i\Big(T, P^{(\mathrm{EOS})}\left(T, V_2, N_1, \ldots, N_n\right), N_1, \ldots, N_n\Big) = \\ &\int_{V_2}^{+\infty} \left[\frac{1}{RT}\frac{\partial P}{\partial N_i}\left(T, V, N_1, \ldots, N_n\right) - \frac{1}{V}\right] \mathrm{d}V - \ln Z\left(T, V_2, N_1, \ldots, N_n\right).\end{aligned} \tag{2.172}$$

To conclude, if we want to compute the fugacity coefficients for the given $T, P, N_1, \ldots, N_n$, we have to proceed in two steps. First, we have to invert the equation of state to find the corresponding volume V that satisfies

$$P = P^{(\mathrm{EOS})}\left(T, V, N_1, \ldots, N_n\right). \tag{2.173}$$

If the equation of state is a cubic equation, the inversion is straightforward (see, e.g., Firoozabadi (1999)). On the other hand, if the equation of state is not a cubic equation, a more complex method has to be used to separate the roots. Second, having the volume V, one can use equation (2.171) to compute the fugacity coefficient of the state.

2.6.2 Volume functions

Volume functions presented by Mikyška and Firoozabadi (2011) are equivalent functions to fugacities in variables $T, V, N_1, \ldots, N_n$. The volume function $F_i = F_i\left(T, V, N_1, \ldots, N_n\right)$ of the i-th component is defined as

$$\mu_i^{(A)}\left(T, V_2, N_1, \ldots, N_n\right) - \mu_i^{(A)}\left(T, V_1, N_1, \ldots, N_n\right) = -RT \ln \frac{F_i\left(T, V_2, N_1, \ldots, N_n\right)}{F_i\left(T, V_1, N_1, \ldots, N_n\right)} \tag{2.174}$$

and

$$\lim_{V \to +\infty} \frac{F_i\left(T, V, N_1, \ldots, N_n\right)}{V} = 1. \tag{2.175}$$

Similarly, the ratio $\Phi_i = \frac{F_i(T,V,N_1,\ldots,N_n)}{V}$ is called volume function coefficient. Then, one can derive

$$RT \ln \frac{F_i\left(T, V_2, N_1, \ldots, N_n\right)}{F_i\left(T, V_1, N_1, \ldots, N_n\right)} = -\int_{V_1}^{V_2} \frac{\partial \mu_i^{(A)}}{\partial V}\left(T, V, N_1, \ldots, N_n\right) \mathrm{d}V. \tag{2.176}$$

Using the Maxwell relation

$$\frac{\partial \mu_i^{(A)}}{\partial V}\left(T, V, N_1, \ldots, N_n\right) = -\frac{\partial P}{\partial N_i}\left(T, V, N_1, \ldots, N_n\right), \tag{2.177}$$

equation (2.176) reads as

$$RT \ln \frac{F_i\left(T, V_2, N_1, \ldots, N_n\right)}{F_i\left(T, V, N_1, \ldots, N_n\right)} = \int_{V_1}^{V_2} \frac{\partial P}{\partial N_i}\left(T, V, N_1, \ldots, N_n\right) \mathrm{d}V. \tag{2.178}$$

Using similar procedures as in Section 2.6.1, we can derive

$$\begin{aligned}\ln \frac{F_i\left(T, V_1, N_1, \ldots, N_n\right)}{V} = & \int_{V_1}^{V_2} \left[\frac{1}{V} - \frac{1}{RT} \frac{\partial P}{\partial N_i}\left(T, V, N_1, \ldots, N_n\right)\right] \mathrm{d}V \\ & + \ln \Phi_i\left(T, V_2, N_1, \ldots, N_n\right).\end{aligned} \tag{2.179}$$

Taking limit $V_2 \rightarrow +\infty$ results in

$$\ln \Phi_i\left(T, V_1, N_1, \ldots, N_n\right) = \int_{V_1}^{+\infty} \left[\frac{1}{V} - \frac{1}{RT} \frac{\partial P}{\partial N_i}\left(T, V, N_1, \ldots, N_n\right)\right] \mathrm{d}V. \tag{2.180}$$

Therefore, the coefficient Φ_i can be computed if the relation

$$P = P\left(T, V, N_1, \ldots, N_n\right) \tag{2.181}$$

is known. Since most of the equations of state have this form, the computation of the volume function coefficient is a straightforward task.

2.7 Constitutive equations

In Section 2.2, we presented various thermodynamical potentials. For our applications, the most critical is the Helmholtz free energy. Any given system is fully described with this potential. However, to this point, we have never presented a full description of the potential, i.e., the equation

$$A = A(T, V, N_1, \ldots, N_n), \tag{2.182}$$

or

$$a = a(T, c_1, \ldots, c_n). \tag{2.183}$$

In equation (2.55), we presented a formula to compute the Helmholtz free energy; however, in this equation, unknown functions, e.g., $\mu_i(T, V, N_1, \ldots, N_n)$ are used. Therefore, in practical computations, this formula is not satisfactory. In this section, we derive the complete formula for the Helmholtz free energy density function. Moreover, using this formula, chemical potentials will be expressed as functions of the concentrations. At the end of this section, thermal equation $U = U(T, V, N_1, \ldots, N_n)$ and entropy equation $S = S\left(T, V, N_1, \ldots, N_n\right)$ will be given. In each derivation, a general equation of state $P = P^{(\mathrm{EOS})}(T, V, N_1, \ldots, N_n)$ will be assumed. Then, the Peng–Robinson equation of state (see Section 2.4.2) will be used.

2.7.1 Helmholtz free energy density $a = a(c_1, \dots, c_n)$

From equation (2.55), the Helmholtz free energy density function $a = a(c_1, \dots, c_n)$ has to obey

$$a(c_1, \dots, c_n) = \sum_{i=1}^{n} c_k \frac{\partial a}{\partial c_i}(c_1, \dots, c_n) - P(c_1, \dots, c_n) \tag{2.184}$$

This equation can be solved in general using the method of characteristics (see, e.g., Strauss (2008)). The solution for any equation of state is

$$a(c_1, \dots, c_n) = c \int_{c_0}^{c} \frac{1}{(\breve{c})^2} P(\breve{c}x_1, \dots, \breve{c}x_n) \,\mathrm{d}\breve{c}, \tag{2.185}$$

where

$$c = \sum_{i=1}^{n} c_i, \tag{2.186}$$

$$x_i = \frac{c_i}{c}, \quad i \in \hat{n}, \tag{2.187}$$

and $c_0 > 0$ is the value of total concentration for which

$$a(c_0 x_1, \dots, c_0 x_n) = 0. \tag{2.188}$$

The choice of $c_0 > 0$ is arbitrary. In this thesis (if not stated otherwise), we are using $c_0 = 1$. Using the Peng–Robinson equation of state (2.100), the integral in (2.185) can be evaluated, and the solution is

$$\begin{aligned} a(c_1, \dots, c_n) = {} & RT \sum_{i=1}^{n} c_i \ln \frac{c_i}{c_0} - RTc \ln\left(1 - \sum_{i=1}^{n} b_i c_i\right) \\ & - \frac{\sum_{i,j=1}^{n} c_i c_j a_{ij}}{2\sqrt{2} \sum_{i=1}^{n} b_i c_i} \ln\left(\frac{1 + \left(1 + \sqrt{2}\right) \sum_{i=1}^{n} b_i c_i}{1 + \left(1 - \sqrt{2}\right) \sum_{i=1}^{n} b_i c_i}\right). \end{aligned} \tag{2.189}$$

For a more compact notation, we denote

$$\mathbf{c} = (c_1, \dots, c_n)^{\mathrm{T}}, \tag{2.190}$$

$$\mathbf{b} = (b_1, \dots, b_n)^{\mathrm{T}}, \tag{2.191}$$

$$\psi_1(\mathbf{c}) = \sum_{i,j=1}^{n} a_{ij} c_i c_j, \tag{2.192}$$

$$\psi_2(x) = \frac{1}{2\sqrt{2}x} \ln\left(\frac{1 + \left(1 + \sqrt{2}\right) x}{1 + \left(1 - \sqrt{2}\right) x}\right). \tag{2.193}$$

Then, equation (2.189) can be rewritten as

$$a(\mathbf{c}) = RT \left(\sum_{i=1}^{n} c_i \ln \frac{c_i}{c_0} - c \ln(1 - \mathbf{b} \cdot \mathbf{c})\right) - \psi_1(\mathbf{c}) \psi_2(\mathbf{b} \cdot \mathbf{c}), \tag{2.194}$$

where the symbol $\cdot$ represents Euclidean scalar product.

Remark 2.15. The solution of the system (2.184) is not unique. For example, an alternative Helmholtz free energy density can be defined as

$$a(\mathbf{c}) = RTc \ln \frac{c}{c_0} - RTc \ln(1 - \mathbf{b} \cdot \mathbf{c}) - \psi_1(\mathbf{c}) \psi_2(\mathbf{b} \cdot \mathbf{c}), \tag{2.195}$$

However, this Helmholtz free energy describes a different system and is not equivalent to (2.194).

2.7.2 Chemical potential $\mu_i = \mu_i(c_1, \ldots, c_n)$

Using relation

$$\mu_i(\mathbf{c}) = \frac{\partial a}{\partial c_i}(\mathbf{c}), \tag{2.196}$$

the chemical potential of the i-th component μ_i for $i \in \widehat{n}$ reads as

$$\mu_i(\mathbf{c}) = RT\left(\ln\frac{c_i}{c_0} + 1 - \ln(1 - \mathbf{b}\cdot\mathbf{c}) + \frac{cb_i}{1 - \mathbf{b}\cdot\mathbf{c}}\right) - \frac{\partial \psi_1}{\partial c_i}(\mathbf{c})\,\psi_2(\mathbf{b}\cdot\mathbf{c}) - \psi_2'(\mathbf{b}\cdot\mathbf{c})b_i, \tag{2.197}$$

where

$$\frac{\partial \psi_1}{\partial c_i}(\mathbf{c}) = 2\sum_{j=1}^{n} a_{ij}c_j = 2\left[\mathbf{A}\mathbf{c}\right]_i, \tag{2.198}$$

$$\psi_2'(x) = -\frac{1}{2\sqrt{2}x^2}\ln\left(\frac{1 + (1+\sqrt{2})x}{1 + (1-\sqrt{2})x}\right) + \frac{1}{x(1 + 2x - x^2)}, \tag{2.199}$$

where the ij-th element of the matrix $\mathbf{A} \in \mathbb{R}^{n,n}$ is defined as

$$a_{ij} = \left[\mathbf{A}\right]_{ij} = (1 - \delta_{i-j})\sqrt{a_i}\sqrt{a_j}, \tag{2.200}$$

where δ_{i-j} is the binary interaction coefficient between components i and j, and $a_i \in \mathbb{R}^+$ is a parameter of the equation of state for the i-th component. The definition is given in Section 2.4.2.

2.7.3 Thermal equation $U = U(T, V, N_1, \ldots, N_n)$

Since U does not have a defined zero value, we have to choose it. In order to be consistent with the literature Reid et al. (1987), the zero value will be defined from the enthalpy potential as

$$\widetilde{h}_0 = \widetilde{h}^{(\mathrm{ig})}(T_0, P_0) = 0, \tag{2.201}$$

where $T_0 = 298.15$ K, $P_0 = 1$ bar. Then,

$$0 = \widetilde{h}^{(\mathrm{ig})}(T_0, P_0) = u_0 + P_0^{(\mathrm{ig})}\widetilde{v}_0 = \widetilde{u}_0 + RT_0, \tag{2.202}$$

where we used equation (2.74) and the ideal gas equation of state (2.99). Therefore,

$$\widetilde{u}_0 = -RT_0 = -2478.95687512\,\mathrm{J\ mol}^{-1}. \tag{2.203}$$

Now, we can start the derivation. First, we derive the equation for the molar energy

$$\mathrm{d}\widetilde{u} = c_v\mathrm{d}T + \left(T\frac{\partial P}{\partial T}(T, \widetilde{v}) - P\right)\mathrm{d}\widetilde{v}. \tag{2.204}$$

Let $\widetilde{u} = \widetilde{u}(T, \widetilde{v})$ be a function of temperature T and molar volume $\widetilde{v}$

$$\mathrm{d}\widetilde{u} = \frac{\partial \widetilde{u}}{\partial T}\mathrm{d}T + \frac{\partial \widetilde{u}}{\partial \widetilde{v}}\mathrm{d}\widetilde{v} = c_v\mathrm{d}T + \frac{\partial \widetilde{u}}{\partial \widetilde{v}}\mathrm{d}\widetilde{v}, \tag{2.205}$$

where we used the definition of the molar heat capacity c_v (see equation (2.126)). Then, one may use

$$\widetilde{u}(T, \widetilde{v}) = u(s(T, \widetilde{v}), \widetilde{v}). \tag{2.206}$$

Therefore,

$$\frac{\partial \widetilde{u}}{\partial \widetilde{v}}(T,\widetilde{v}) = \frac{\partial u}{\partial s}(s(T,\widetilde{v}),\widetilde{v})\frac{\partial s}{\partial \widetilde{v}}(T,\widetilde{v}) + \frac{\partial u}{\partial \widetilde{v}}(s(T,\widetilde{v}),\widetilde{v}) = T\frac{\partial s}{\partial \widetilde{v}}(T,\widetilde{v}) - P. \tag{2.207}$$

Using Maxwell relation (2.123) in the previous equation gives

$$\frac{\partial \widetilde{u}}{\partial \widetilde{v}}(T,\widetilde{v}) = T\frac{\partial P}{\partial T}(T,\widetilde{v}) - P. \tag{2.208}$$

Combining equations (2.205) and (2.208) gives equation (2.204) which we wanted to prove. Multiplying equation (2.204) with the total number of mole N results in

$$\mathrm{d}U = Nc_v\mathrm{d}T + \left(T\frac{\partial P}{\partial T}(T,V) - P\right)\mathrm{d}V. \tag{2.209}$$

Let $\mathbf{N} = (N_1,\ldots,N_n)^{\mathrm{T}}$ is the vector of mole numbers. Then, we can write

$$\begin{aligned} U(T,V,\mathbf{N}) = {} & U(T,V,\mathbf{N}) - U(T,V',\mathbf{N}) \\ & + U(T,V',\mathbf{N}) - U^{(\mathrm{ig})}(T,V',\mathbf{N}) \\ & + U^{(\mathrm{ig})}(T,V',\mathbf{N}) - U^{(\mathrm{ig})}(T,V_0,\mathbf{N}) \\ & + U^{(\mathrm{ig})}(T,V_0,\mathbf{N}) - U^{(\mathrm{ig})}(T_0,V_0,\mathbf{N}) \\ & + U^{(\mathrm{ig})}(T_0,V_0,\mathbf{N}). \end{aligned} \tag{2.210}$$

The differences in the previous equation can be evaluated using equation (2.209). Therefore,

$$U(T,V,\mathbf{N}) - U(T,V',\mathbf{N}) = \int\limits_{V'}^{V}\left(T\,\frac{\partial P}{\partial T}\left(T,\check{V},\mathbf{N}\right) - P\left(T,\check{V},\mathbf{N}\right)\right)\mathrm{d}\check{V}, \tag{2.211}$$

$$U^{(\mathrm{ig})}(T,V',\mathbf{N}) - U^{(\mathrm{ig})}(T,V_0,\mathbf{N}) = \int\limits_{V_0}^{V'}\left(T\frac{\partial P^{(\mathrm{ig})}}{\partial T}(T,V,\mathbf{N}) - P^{(\mathrm{ig})}(T,V,\mathbf{N})\right)\mathrm{d}V, \tag{2.212}$$

$$U^{(\mathrm{ig})}(T,V_0,\mathbf{N}) - U^{(\mathrm{ig})}(T_0,V_0,\mathbf{N}) = \int\limits_{T_0}^{T} Nc_v^{(\mathrm{ig})}\mathrm{d}T. \tag{2.213}$$

Moreover,

$$U^{(\mathrm{ig})}(T_0,V_0,\mathbf{N}) = N\widetilde{u}^{(\mathrm{ig})}(T_0,V_0) = N\widetilde{u}_0 = -NRT_0. \tag{2.214}$$

Using the ideal gas equation of state (2.99), one can obtain

$$T\frac{\partial P^{(\mathrm{ig})}}{\partial T}(T,V,\mathbf{N}) = \frac{NRT}{V} = P^{(\mathrm{ig})}(T,V,\mathbf{N}). \tag{2.215}$$

Therefore, the equation (2.212) reads as

$$U^{(\mathrm{ig})}(T,V',\mathbf{N}) - U^{(\mathrm{ig})}(T,V_0,\mathbf{N}) = \int\limits_{V_0}^{V'}\left(P^{(\mathrm{ig})}(T,V,\mathbf{N}) - P^{(\mathrm{ig})}(T,V,\mathbf{N})\right)\mathrm{d}V = 0. \tag{2.216}$$

Moreover, the molar heat capacity of an ideal gas is

$$c_v^{(\mathrm{ig})} = \sum_{i=1}^{n} x_i c_{v,i}^{(\mathrm{ig})}, \tag{2.217}$$

i.e., the heat capacity of the mixture can be calculated as a weighted sum of individual heat capacities of each species. Therefore, equation (2.213) reads as

$$U^{(\mathrm{ig})}(T, V_0, \mathbf{N}) - U^{(\mathrm{ig})}(T_0, V_0, \mathbf{N}) = -NR\,(T - T_0) + \sum_{i=1}^{n} N_i \int_{T_0}^{T} c_{p,i}^{(\mathrm{ig})} \mathrm{d}T, \tag{2.218}$$

where we used the Mayer relation (2.135) for the ideal gas

$$c_p^{(\mathrm{ig})} - c_v^{(\mathrm{ig})} = R, \tag{2.219}$$

The molar heat capacity at constant pressure considered as the ideal gas $c_{p,i}^{(\mathrm{ig})}$ can be evaluated using the correlation

$$c_{p,i}^{(\mathrm{ig})}(T) = \sum_{k=0}^{3} \alpha_{ik} T^k, \tag{2.220}$$

where α_{ik} are the correlation coefficients. Reid et al. (1987) gives large database of the correlation coefficients for numerous components. Combining equations (2.210), (2.211), (2.216), and (2.218) results in

$$\begin{aligned} U(T, V, \mathbf{N}) = \int_{V'}^{V} \left(T \frac{\partial P}{\partial T} \left(T, \check{V}, \mathbf{N}\right) - P\left(T, \check{V}, \mathbf{N}\right)\right) \mathrm{d}\check{V} + U(T, V', \mathbf{N}) - U^{(\mathrm{ig})}(T, V', \mathbf{N}) \\ -NR\,(T - T_0) + \sum_{i=1}^{n} N_i \sum_{k=0}^{3} \alpha_{ik} \frac{T^{k+1} - T_0^{k+1}}{k+1} + N\widetilde{u}_0, \end{aligned} \tag{2.221}$$

for an arbitrary volume V'. Since

$$\lim_{V' \to +\infty} \left(U(T, V', \mathbf{N}) - U^{(\mathrm{ig})}(T, V', \mathbf{N}) \right) = 0, \tag{2.222}$$

equation (2.221) in limit $V' \to +\infty$ reads as

$$\begin{aligned} U(T, V, \mathbf{N}) = &\int_{+\infty}^{V} \left(T \frac{\partial P}{\partial T} \left(T, \check{V}, \mathbf{N}\right) - P\left(T, \check{V}, \mathbf{N}\right)\right) \mathrm{d}\check{V} \\ &- NR\,(T - T_0) + \sum_{i=1}^{n} N_i \sum_{k=0}^{3} \alpha_{ik} \frac{T^{k+1} - T_0^{k+1}}{k+1} + N\widetilde{u}_0. \end{aligned} \tag{2.223}$$

Using the Peng–Robinson equation of state (2.100), the integral in the previous equation can be evaluated, and the internal energy as a function of $T, V, \mathbf{N}$ reads as

$$\begin{aligned} U^{(\mathrm{PR})}(T, V, \mathbf{N}) = N \frac{T \frac{\partial a}{\partial T} - a}{2\sqrt{2} \sum\limits_{i=1}^{n} b_i x_i} \ln \left(\frac{V + \left(1 + \sqrt{2}\right) \sum\limits_{i=1}^{n} b_i N_i}{V + \left(1 - \sqrt{2}\right) \sum\limits_{i=1}^{n} b_i N_i} \right) - NR\,(T - T_0) \\ + \sum_{i=1}^{n} N_i \sum_{k=0}^{3} \alpha_{ik} \frac{T^{k+1} - T_0^{k+1}}{k+1} + N\widetilde{u}_0, \end{aligned} \tag{2.224}$$

where the coefficient a is defined as

$$a(N_1, \ldots, N_n, T) = \sum_{i,j=1}^{n} \frac{N_i N_j}{N^2} a_{ij}(T), \tag{2.225}$$

where $N = \sum_{i=1}^{n} N_i$.

2.7.4 Entropy equation $S = S(T, V, N_1, \ldots, N_n)$

Let $\mathbf{N} = (N_1, \ldots, N_n)^{\mathrm{T}}$. Then, the entropy at given $T, V, \mathbf{N}$ can be written as

$$\begin{aligned} S(T, V, \mathbf{N}) &= S(T, V, \mathbf{N}) - S(T, V', \mathbf{N}) \\ &+ S(T, V', \mathbf{N}) - S^{(\mathrm{ig})}(T, V', \mathbf{N}) \\ &+ S^{(\mathrm{ig})}(T, V', \mathbf{N}) - S^{(\mathrm{ig})}(T, V_0, \mathbf{N}) \\ &+ S^{(\mathrm{ig})}(T, V_0, \mathbf{N}) - S^{(\mathrm{ig})}(T_0, V_0, \mathbf{N}) \\ &+ S^{(\mathrm{ig})}(T_0, V_0, \mathbf{N}). \end{aligned} \tag{2.226}$$

Then, the first and third difference can be computed using

$$S(T, V, \mathbf{N}) - S(T, V', \mathbf{N}) = \int_{V'}^{V} \frac{\partial S}{\partial V}\left(T, \check{V}, \mathbf{N}\right) \mathrm{d}\check{V} = \int_{V'}^{V} \frac{\partial P}{\partial T}\left(T, \check{V}, \mathbf{N}\right) \mathrm{d}\check{V}, \tag{2.227}$$

$$\begin{aligned} S^{(\mathrm{ig})}(T, V', \mathbf{N}) - S^{(\mathrm{ig})}(T, V_0, \mathbf{N}) &= \int_{V_0}^{V'} \frac{\partial S^{(\mathrm{ig})}}{\partial V}(T, V, \mathbf{N}) \, \mathrm{d}V = \int_{V_0}^{V'} \frac{\partial P^{(\mathrm{ig})}}{\partial T}(T, V, \mathbf{N}) \, \mathrm{d}V \\ &= R \sum_{i=1}^{n} N_i \int_{V_0}^{V'} \frac{\mathrm{d}V}{V}. \end{aligned} \tag{2.228}$$

where we used the ideal gas equation of state and the Maxwell relation (2.123). The fourth difference in equation (2.226) reads as

$$\begin{aligned} S^{(\mathrm{ig})}(T, V_0, \mathbf{N}) - S^{(\mathrm{ig})}(T_0, V_0, \mathbf{N}) &= \sum_{i=1}^{n} \left(S_i^{(\mathrm{ig})}(T, V_0, N_i) - S_i^{(\mathrm{ig})}(T_0, V_0, N_i) \right) \\ &= \sum_{i=1}^{n} N_i \left(s_i^{(\mathrm{ig})}(T, V_0) - s_i^{(\mathrm{ig})}(T_0, V_0) \right) \\ &= \sum_{i=1}^{n} N_i \int_{T_0}^{T} \frac{\partial s_i^{(\mathrm{ig})}}{\partial T}\left(\check{T}, V_0\right) \mathrm{d}\check{T}. \end{aligned} \tag{2.229}$$

Using the Mayer relation for ideal gas (2.135) gives

$$\frac{\partial s_i^{(\mathrm{ig})}}{\partial T}(T, V_0) = \frac{c_{v,i}^{(\mathrm{ig})}}{T} = \frac{-R + c_{p,i}^{(\mathrm{ig})}}{T}. \tag{2.230}$$

Combining equations (2.229) and (2.230) results in

$$S^{(\mathrm{ig})}(T, V_0, \mathbf{N}) - S^{(\mathrm{ig})}(T_0, V_0, \mathbf{N}) = -NR \ln \frac{T}{T_0} + \sum_{i=1}^{n} N_i \int_{T_0}^{T} \frac{c_{p,i}^{(\mathrm{ig})}}{\check{T}} \mathrm{d}\check{T}. \tag{2.231}$$

Substituting equations (2.227),(2.228) and (2.231) to (2.226) gives

$$\begin{aligned} S(T,V,\mathbf{N}) = &\int\limits_{V'}^{V} \frac{\partial P}{\partial T}\left(T,\check{V},\mathbf{N}\right)\mathrm{d}\check{V} + S(T,V',\mathbf{N}) - S^{(\mathrm{ig})}(T,V',\mathbf{N}) \\ &+ R\sum_{i=1}^{n} N_i \int\limits_{V_0}^{V'} \frac{\mathrm{d}V}{V} - NR\ln\frac{T}{T_0} + \sum_{i=1}^{n} N_i \int\limits_{T_0}^{T} \frac{c_{p,i}^{(\mathrm{ig})}}{\check{T}}\mathrm{d}\check{T} + S^{(\mathrm{ig})}(T_0,V_0,\mathbf{N}). \end{aligned} \tag{2.232}$$

Using the integral additive property, one can derive

$$\int\limits_{V_0}^{V'} \frac{\mathrm{d}V}{V} = \int\limits_{V_0}^{V} \frac{\mathrm{d}\check{V}}{\check{V}} + \int\limits_{V}^{V'} \frac{\mathrm{d}\check{V}}{\check{V}}. \tag{2.233}$$

Therefore, equation (2.232) is

$$\begin{aligned} S(T,V,\mathbf{N}) = &\int\limits_{V}^{V'} \left[\frac{NR}{\check{V}} - \frac{\partial P}{\partial T}\left(T,\check{V},\mathbf{N}\right)\right]\mathrm{d}\check{V} + R\sum_{i=1}^{n} N_i \ln\frac{V}{V_0} - NR\ln\frac{T}{T_0} \\ &+ \sum_{i=1}^{n} N_i \int\limits_{T_0}^{T} \frac{c_{p,i}^{(\mathrm{ig})}}{\check{T}}\mathrm{d}\check{T} + S(T,V',\mathbf{N}) - S^{(\mathrm{ig})}(T,V',\mathbf{N}) + S^{(\mathrm{ig})}(T_0,V_0,\mathbf{N}). \end{aligned} \tag{2.234}$$

Since

$$\lim_{V\to+\infty}\left[S(T,V,\mathbf{N}) - S^{(\mathrm{ig})}(T,V,\mathbf{N})\right] = 0, \tag{2.235}$$

taking limit $V' \to +\infty$ in equation (2.234) results in

$$S(T,V,\mathbf{N}) = \int_{V}^{+\infty} \left[\frac{NR}{\check{V}} - \frac{\partial P}{\partial T}\left(T,\check{V},\mathbf{N}\right)\right]\mathrm{d}\check{V} + R\sum_{i=1}^{n} N_i \ln\frac{VP_0}{N_iRT_0} + \sum_{i=1}^{n} N_i \int_{T_0}^{T} \frac{c_{p,i}^{(\mathrm{ig})}}{\check{T}}\mathrm{d}\check{T}, \tag{2.236}$$

where we used $V_0 = \frac{N_iRT}{P_0}$, and the last term in (2.234) is equal to zero. The previous equation gives a general expression for entropy using an arbitrary equation of state. Using the Peng–Robinson equation of state (2.100), the integral can be analytically computed, and the previous entropy equation results in

$$\begin{aligned} S^{(\mathrm{PR})}(T,V,\mathbf{N}) = NR\ln\left(\frac{V - \sum\limits_{i=1}^{n} N_ib_i}{V}\right) + N\frac{\frac{\partial a}{\partial T}}{2\sqrt{2}\sum\limits_{i=1}^{n} N_ib_i}\ln\left(\frac{V + \left(1+\sqrt{2}\right)\sum\limits_{i=1}^{n} N_ib_i}{V + \left(1-\sqrt{2}\right)\sum\limits_{i=1}^{n} N_ib_i}\right) \\ + R\sum_{i=1}^{n} N_i\ln\frac{VP_0}{N_iRT} + \sum_{i=1}^{n} N_i \int\limits_{T_0}^{T} \frac{c_{p,i}^{(\mathrm{ig})}}{\check{T}}\mathrm{d}\check{T}, \end{aligned} \tag{2.237}$$

where the coefficient a is defined as

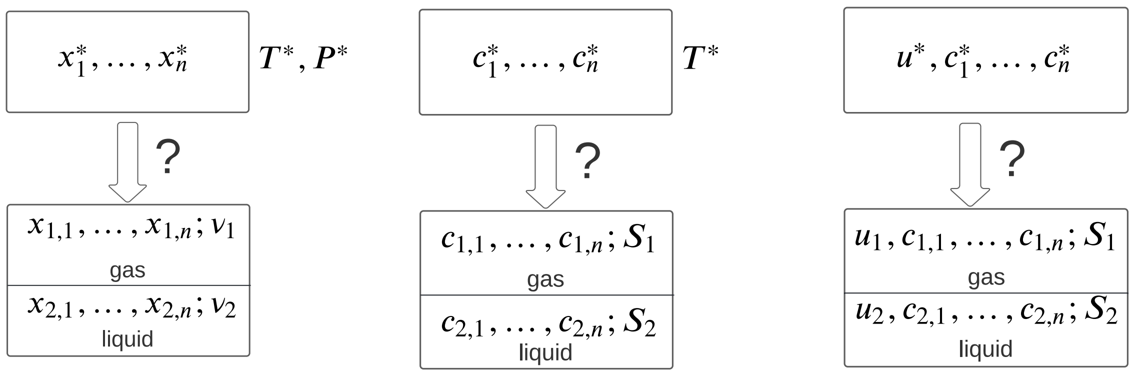
$$a(N_1,\ldots,N_n,T) = \sum_{i,j=1}^{n} \frac{N_iN_j}{N^2}a_{ij}(T), \tag{2.238}$$

where $N = \sum\limits_{i=1}^{n} N_i$.

Phase stability testing 3

The phase stability testing is a basic problem of thermodynamics. Consider a mixture of n components with temperature T^*, pressure P^*, and the overall mole fractions $x_1^*, \dots, x_n^*$. In the phase stability testing problem, the goal is to predict whether the given mixture (state) is stable and remains in one phase or if this state is unstable and splitting will occur. In Figure 3.1a, the problem is depicted. This formulation of the problem is known as PTN-specification or PTN-phase stability testing as the mixture is described by pressure P, temperature T, and mole fraction (numbers) $x_1, \dots, x_n$ ($N_1, \dots, N_n$). However, in recent years other formulations rose to popularity. The most used and studied are known as VTN-, and UVN-specifications. In the VTN-phase stability testing (see Figure 3.1b), the mixture is described by its volume V^*, temperature T^*, and mole numbers $N_1^*, \dots, N_n^*$. Alternatively, the concentrations $c_i^* = \frac{N_i^*}{V}$ and the temperature T^* can be used to describe the mixture. Thus, in the literature, the term cT-phase stability testing problem is used. Lastly, in the UVN-specification (see Figure 3.1c), the mixture is described by its internal energy U^*, volume V^*, and mole numbers $N_1^*, \dots, N_n^*$. As in the VTN-formulation, the concentrations c_i^* and internal energy density $u^* = \frac{U^*}{V}$ can be used and the term uc-phase stability testing is used.

Remark 3.1. Mikyška and Firoozabadi (2012) have shown that the VTN-specification has advantages over the PTN-specification. The main reason is that the equation of state is pressure explicit. Therefore, only one value of the pressure is corresponding for the given volume. On the other hand, if the equation of state is cubic, up to three volumes are possible for a given pressure.


$x_1^*, \dots, x_n^*$
T^*, P^*
?
$x_{1,1}, \dots, x_{1,n}; v_1$
gas
$x_{2,1}, \dots, x_{2,n}; v_2$
liquid
$c_1^*, \dots, c_n^*$
T^*
?
$c_{1,1}, \dots, c_{1,n}; S_1$
gas
$c_{2,1}, \dots, c_{2,n}; S_2$
liquid
$u^*, c_1^*, \dots, c_n^*$
?
$u_1, c_{1,1}, \dots, c_{1,n}; S_1$
gas
$u_2, c_{2,1}, \dots, c_{2,n}; S_2$
liquid

(a) PTN-specification (b) VTN-specification (c) UVN-specification

Figure 3.1: Phase stability testing diagram in different specifications. The question is whether the given state is stable and remains in one-phase or splitting occurs.

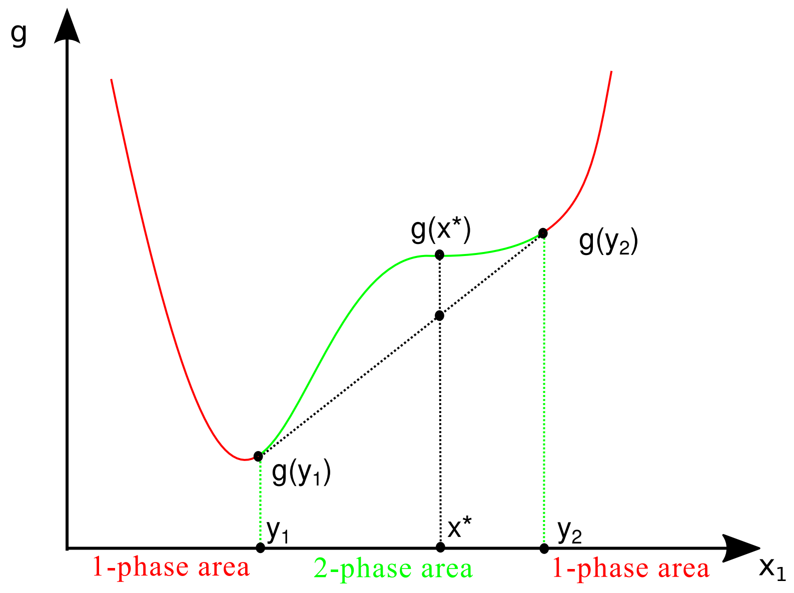


Figure 3.2: A possible graph of the Gibbs free energy per mol. In the red area, the mixture is stable. In the green area, the value of the Gibbs free energy per mol can be lowered, and the mixture is unstable.

Remark 3.2. In the literature, other specifications are studied. For example, the HPN-specification (constant entalpy, pressure, moles) was investigated by Zhu and Okuno (2016) or Gupta et al. (1990). The SPN-specification (constant entropy, pressure, moles) was investigated by Michelsen (1987). These formulations will fit into our general framework. However, they are not studied in this thesis.

In this chapter, the mathematical formulation of the phase stability problem will be presented. Then, various numerical algorithms for solving the phase stability testing problem will be given. Lastly, numerical examples showing the performance of the individual algorithms will be presented.

3.1 Mathematical formulation

Using the laws of thermodynamics, the mixture will split if there exists a two-phase state with a lower value of an appropriate thermodynamical potential compared to the one corresponding to the one-phase state. In Figure 3.2, we depicted the Gibbs free energy per mol of a single component mixture as a function of mole fraction x_1. Here, the unstable area and the stable area are depicted in green and red, respectively. In the unstable area, the Gibbs energy of the state $g(x^*)$ can be lowered by combining two states denoted by y_1 and y_2.

In the case of the PTN-stability testing, we can use Theorem 2.10, and the appropriate thermodynamical potential is the previously mentioned Gibbs free energy. If we find two states

$(x_{1,1},\ldots,x_{1,n})$ and $(x_{2,1},\ldots,x_{2,n})$ satisfying

$$x_i^* = \nu_1 x_{1,i} + \nu_2 x_{2,i}, \quad i \in \widehat{n}, \tag{3.1}$$

$$1 = \nu_1 + \nu_2, \tag{3.2}$$

$$1 = \sum_{i=1}^{n} x_{k,i}, \quad k \in \widehat{2}, \tag{3.3}$$

$$g\left(T^*, P^*, x_1^*, \ldots, x_n^*\right) > \nu_1 g(T^*, P^*, x_{1,1}, \ldots, x_{1,n}) + \nu_2 g(T^*, P^*, x_{2,1}, \ldots, x_{2,n}), \tag{3.4}$$

then the initial state is unstable. In the previous equations, $x_{k,i}$ is the mole fraction of the i-th component in phase k, ν_1, ν_2 are the phase mole fractions defined as

$$\nu_k = \frac{N^{(k)}}{N^*}, \quad k \in \widehat{n}, \tag{3.5}$$

where $N^{(k)}$ is the total number of moles in phase k. Using the relation $\nu_2 = 1 - \nu_1$, the PTN-phase stability testing conditions (3.1)–(3.4) can be rewritten as

$$x_i^* = \nu_1 x_{1,i} + (1-\nu_1) x_{2,i}, \quad i \in \widehat{n} \tag{3.6}$$

$$1 = \sum_{i=1}^{n} x_{k,i}, \quad k \in \widehat{2}, \tag{3.7}$$

$$g\left(T^*, P^*, x_1^*, \ldots, x_n^*\right) > \nu_1 g(T^*, P^*, x_{1,1}, \ldots, x_{1,n}) + (1-\nu_1) g(T^*, P^*, x_{2,1}, \ldots, x_{2,n}). \tag{3.8}$$

However, in order to test condition (3.8) directly, all feasible states have to be used. This is not a suitable option. Therefore, the classical approach is to transform the problem into a more acceptable form. For this reason, a function known as TPD (Tangent Plane Distance) is defined. Using this function, condition (3.8) can be solved easier. The formal definition will be given in the next section.

Other phase stability testing formulations can be mathematically described in a similar fashion. In the VTN-specification, the appropriate thermodynamical potential is the Helmholtz free energy density, in the UVN-specification, the entropy density function is chosen. In the next section, we present a unified formulation, which will cover all specifications.

3.1.1 Unified formulation of the phase stability testing

In Smejkal and Mikyška (2018), we presented this unified formulation:

Let $f : \mathbb{R}^m \to \mathbb{R}$ be a function defined on a convex set $\mathcal{D} \subset \mathbb{R}^m$ and $\mathbf{x}^* \in \mathcal{D}$. The task is to find out whether there exist vectors $\mathbf{y}, \mathbf{z} \in \mathcal{D}$ and a coefficient $\beta \in (0,1)$ satisfying

$$\beta \mathbf{y} + (1-\beta)\mathbf{z} = \mathbf{x}^*, \tag{3.9}$$

$$\beta f(\mathbf{y}) + (1-\beta) f(\mathbf{z}) < f(\mathbf{x}^*). \tag{3.10}$$

Eliminating the unknown vector $\mathbf{z}$ using (3.9) and (3.10), we obtain

$$\beta f(\mathbf{y}) + (1-\beta) f\left(\frac{1}{1-\beta}(\mathbf{x}^* - \beta\mathbf{y})\right) < f(\mathbf{x}^*). \tag{3.11}$$

Using the Taylor expansion of f around $\mathbf{x}^*$ (assuming that f is smooth enough), we obtain

$$f\left(\frac{1}{1-\beta}(\mathbf{x}^* - \beta\mathbf{y})\right) = f(\mathbf{x}^*) + \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}(\mathbf{x}^*)\left(\frac{1}{1-\beta}(\mathbf{x}^* - \beta\mathbf{y}) - \mathbf{x}^*\right) + o(\beta), \quad \beta \to 0+, \tag{3.12}$$

where $o(\beta)$ is the reminder in the Taylor expansion. Now, equation (3.11) can be rewritten as follows

$$\beta f(\mathbf{y}) - \beta f(\mathbf{x}^*) + \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}(\mathbf{x}^*)\,\beta\,(\mathbf{x}^* - \mathbf{y}) + o(\beta) < 0. \tag{3.13}$$

Finally, we introduce a function TPD of variables $\mathbf{y} = (y_1, \ldots, y_m)^{\mathrm{T}}$ by

$$\mathrm{TPD}(\mathbf{y};\mathbf{x}^*) = \lim_{\beta \to 0+} \frac{1}{\beta}\left(\beta f(\mathbf{y}) - \beta f(\mathbf{x}^*) + \beta \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}(\mathbf{x}^*)\,(\mathbf{x}^* - \mathbf{y}) + o(\beta)\right). \tag{3.14}$$

This function can be also rewritten as

$$\mathrm{TPD}(\mathbf{y};\mathbf{x}^*) = f(\mathbf{y}) - f(\mathbf{x}^*) + \sum_{i=1}^{m} \frac{\partial f}{\partial x_i}(\mathbf{x}^*)\,(x_i^* - y_i)\,. \tag{3.15}$$

The state $\mathbf{x}^*$ is stable, if $\mathrm{TPD}(\mathbf{y};\mathbf{x}^*) \geqslant 0$ for all feasible $\mathbf{y} = (y_1, \ldots, y_m)^{\mathrm{T}}$. One can seek the global minimum $\mathbf{y}^{(\mathrm{opt})}$ of the TPD to test the above condition. If $\mathrm{TPD}(\mathbf{y}^{(\mathrm{opt})};\mathbf{x}^*) \geqslant 0$, the state is stable. On the other hand, if $\mathrm{TPD}(\mathbf{y}^{(\mathrm{opt})};\mathbf{x}^*) < 0$, then the system is unstable, and splitting will occur. The name TPD stands for **T**angent **P**lane **D**istance. In other words, we measure the distance between the function and the tangent hyperplane at point $\mathbf{x}^*$. In Figure 3.3, two possibilities are depicted. The state $\mathbf{x}^{**}$ is stable since all points of the tangent hyperplane at point $\mathbf{x}^{**}$ lie below the graph of f. On the other hand, state $\mathbf{x}^*$ is unstable since there are points of the tangent hyperplane at point $\mathbf{x}^*$, which lie above the graph of f. Therefore, the distance and consequently the value of TPD are negative.

Remark 3.3. Since $\mathrm{TPD}(\mathbf{x}^*;\mathbf{x}^*) = 0$, the value $\mathrm{TPD}(\mathbf{y}^{(\mathrm{opt})};\mathbf{x}^*)$ can not be positive. The solution $\mathbf{y} = \mathbf{x}^*$ is known as *trivial solution.*

In the unstable case, it follows from the definition of function TPD that a small positive β exists for which

$$\beta f(\mathbf{y}) + (1-\beta) f\left(\frac{1}{1-\beta}(\mathbf{x}^* - \beta\mathbf{y})\right) < f(\mathbf{x}^*)\,. \tag{3.16}$$

The coefficient β can be found by the bisectioning. Once β has been found, denoting

$$\begin{aligned} \mathbf{x}^{(1)} &= \mathbf{y}, \\ \mathbf{x}^{(2)} &= \frac{1}{1-\beta}(\mathbf{x}^* - \beta\mathbf{y})\,, \\ \alpha_1 &= \beta, \\ \alpha_2 &= 1-\beta, \end{aligned}$$

we have constructed a two-phase split with a lower energy than the energy of the initial phase. The phase stability testing not only decides on the phase stability but, in the unstable case, it also provides a tool for constructing an initial two-phase split. This will be advantageous for the initialization of the phase equilibrium computation.

Now, we will verify that the TPD function (3.15) is a generalization of the TPD functions of the most commonly used stability testing approaches. To derive the TPD function for the VTN-phase stability testing, we choose

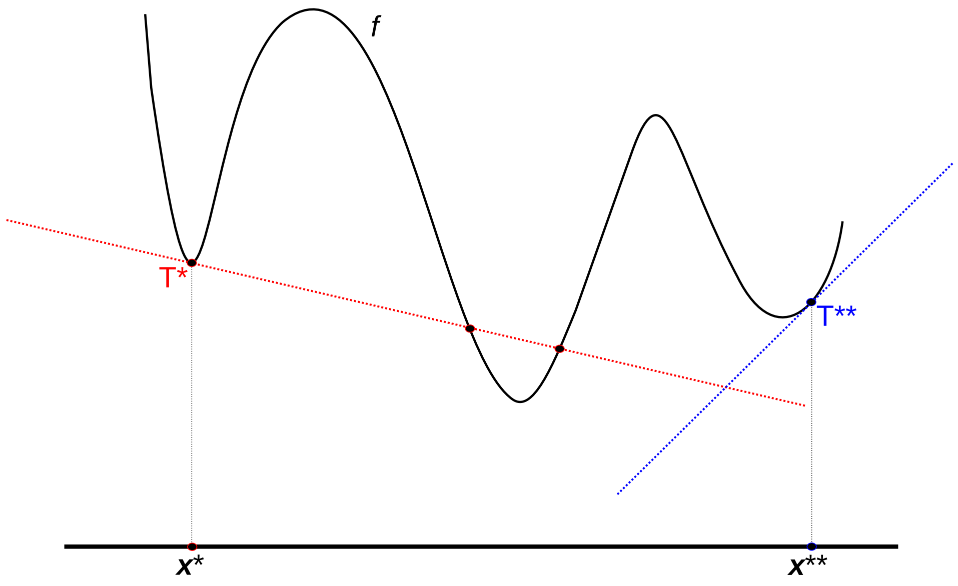


Figure 3.3: Geometrical interpretation of the TPD function.

$$\mathcal{D} = \left\{ \left(c_1, \ldots, c_n\right)^{\mathrm{T}}; \left(\forall i \in \widehat{n}\right) \left(c_i \geqslant 0\right) \wedge \sum_{i=1}^{n} c_i b_i < 1 \right\}, \tag{3.17a}$$

$$\mathbf{x}^* = \left(c_1^*, \ldots, c_n^*\right)^{\mathrm{T}}, \tag{3.17b}$$

$$f\left(\mathbf{y}\right) = a\left(\mathbf{y}\right). \tag{3.17c}$$

In equation (3.17a), b_i is the co-volume parameter of component i from the Peng–Robinson equation of state defined in Section 2.4.2. The partial derivatives of the Helmholtz free energy density read as

$$\frac{\partial a}{\partial c_i}\left(c_1, \ldots, c_n\right) = \mu_i\left(c_1, \ldots, c_n\right), \quad i \in \widehat{n}. \tag{3.18}$$

Substituting these formulas into the general TPD function (3.15), we derive

$$\begin{aligned} \mathrm{TPD}_a\left(c_1, \ldots, c_n; \mathbf{x}^*\right) &= \sum_{i=1}^{n} \mu_i\left(\mathbf{x}^*\right)\left(c_i^* - c_i\right) + \sum_{i=1}^{n} c_i \mu_i\left(c_1, \ldots, c_n\right) \\ &\qquad - P\left(c_1, \ldots, c_n\right) - \sum_{i=1}^{n} c_i^* \mu_i\left(\mathbf{x}^*\right) + P\left(x^*\right) \\ &= \sum_{i=1}^{n} c_i\left[\mu_i\left(c_1, \ldots, c_n\right) - \mu_i\left(\mathbf{x}^*\right)\right] - \left[P\left(c_1, \ldots, c_n\right) - P\left(\mathbf{x}^*\right)\right], \end{aligned} \tag{3.19}$$

which is in agreement with the TPD function in Mikyška and Firoozabadi (2012). To derive the

UVN-stability TPD function, we choose

$$D = \left\{ (u, c_1, \ldots, c_n)^{\mathrm{T}} ; (\forall i \in \widehat{n}) \, (c_i \geqslant 0) \ \wedge \ \sum_{i=1}^{n} c_i b_i < 1 \ \wedge \right. \\ \left. (\exists T > 0) \left(u = U^{(EOS)} (T, 1, c_1, \ldots, c_n) \right) \right\}, \tag{3.20a}$$

$$\mathbf{x}^* = (u^*, c_1^*, \ldots, c_n^*)^{\mathrm{T}}, \tag{3.20b}$$

$$f(\mathbf{y}) = -s(\mathbf{y}), \tag{3.20c}$$

In equation (3.20a), $U^{(EOS)}$ denotes the internal energy equation of state. Its form is given in Section 2.7.3.

Remark 3.4. Since the unified formulation is presented as a minimization problem, the minus sign in equation (3.20c) before the entropy function has to be included. The entropy function is maximal at the equilibrium.

The partial derivatives of the entropy density read as

$$\frac{\partial s}{\partial c_i}(u, c_1, \ldots, c_n) = -\frac{\mu_i}{T}(u, c_1, \ldots, c_n), \quad i \in \widehat{n}, \tag{3.21}$$

$$\frac{\partial s}{\partial u}(u, c_1, \ldots, c_n) = \frac{1}{T}(u, c_1, \ldots, c_n). \tag{3.22}$$

Denote $\mathbf{y} = (u, c_1, \ldots, c_n)^{\mathrm{T}}$ and substituting these formulas into the general TPD function (3.15), we derive

$$\begin{aligned} \mathrm{TPD}_s(\mathbf{y};\mathbf{x}^*) &= -\frac{1}{T}(\mathbf{x}^*)(u^* - u) + \sum_{i=1}^{n} \frac{\mu_i}{T}(\mathbf{x}^*)(c_i^* - c_i) - \frac{u}{T}(\mathbf{y}) - \frac{P}{T}(\mathbf{y}) \\ &\quad + \sum_{i=1}^{n} c_i \frac{\mu_i}{T}(\mathbf{y}) + \frac{u^*}{T}(\mathbf{x}^*) + \frac{P}{T}(\mathbf{x}^*) - \sum_{i=1}^{n} c_i^* \frac{\mu_i}{T}(\mathbf{x}^*) \\ &= \sum_{i=1}^{n} c_i \left[\frac{\mu_i}{T}(\mathbf{y}) - \frac{\mu_i}{T}(\mathbf{x}^*) \right] - \left[\frac{P}{T}(\mathbf{y}) - \frac{P}{T}(\mathbf{x}^*) \right] - \left[\frac{1}{T}(\mathbf{y}) - \frac{1}{T}(\mathbf{x}^*) \right] u. \end{aligned} \tag{3.23}$$

This result is in agreement with the TPD function derived in Smejkal and Mikyška (2017). For the PTN-stability testing we choose f, D and x^* as

$$\mathcal{D} = \left\{ (x_1, \ldots, x_{n-1})^{\mathrm{T}} ; \sum_{i=1}^{n-1} x_i \leqslant 1 \wedge \left(\forall i \in \widehat{n-1} \right) (x_i \geqslant 0) \right\}, \tag{3.24a}$$

$$\mathbf{x}^* = \left(x_1^*, \ldots, x_{n-1}^* \right)^{\mathrm{T}}, \tag{3.24b}$$

$$f(\mathbf{y}) = \breve{g}(\mathbf{y}) = \widetilde{g}\left(x_1, \ldots, x_{n-1}, 1 - \sum_{i=1}^{n-1} x_i \right). \tag{3.24c}$$

The partial derivatives of the Gibbs energy per one mol $\breve{g}$ using the Gibbs-Duhem equation (2.117) read as

$$\frac{\partial \breve{g}}{\partial x_i}(x_1, \ldots, x_{n-1}) = \mu_i(x_1, \ldots, x_{n-1}) - \mu_n(x_1, \ldots, x_{n-1}), \quad i \in \widehat{n-1}. \tag{3.25}$$

Substituting these formulas into general TPD function (3.15), we derive

$$\begin{aligned}\mathrm{TPD}_g\left(x_1,\ldots,x_{n-1};\mathbf{x}^*\right) = &\sum_{i=1}^{n-1} x_i\left[\mu_i\left(x_1,\ldots,x_{n-1}\right)-\mu_i\left(\mathbf{x}^*\right)\right] \\ &+\mu_n\left(\mathbf{x}^*\right)\left(\sum_{i=1}^{n-1} x_i - 1\right) - \mu_n\left(x_1,\ldots,x_{n-1}\right)\left(\sum_{i=1}^{n-1} x_i - 1\right).\end{aligned} \tag{3.26}$$

Denoting $x_n = 1 - \sum_{i=1}^{n-1} x_i$, we derive same TPD function as in Michelsen (1982a).

3.2 Numerical algorithms for the phase stability testing

To find out whether for a given $\mathbf{x}^* \in \mathcal{D}$ there exists a state $\mathbf{y}$ for which $\mathrm{TPD}(\mathbf{y};\mathbf{x}^*) < 0$, we will seek for global or at least local minima of TPD. In this section, various optimization methods for solving the phase stability testing problem will be presented. In the literature, the PTN-phase stability testing is mostly solved using a method known as SSI (Successive Substitution Iteration). Therefore, first, we present this method in the PTN-specification. Then, we extend this method to VTN- and UVN-specifications. The VTN-specification was discussed by Mikyška and Firoozabadi (2012), the UVN-specification by Bi et al. (2020b). The main reason why we present these methods is the comparison with the Newton–Raphson method, which will be presented in Section 3.2.3. Using the unified formulation, the Newton–Raphson method is identical in all specifications. However, the SSI and as well as the Newton–Raphson method are only local methods. Therefore, more than one initial approximation has to be used, and even then, we have no guarantee that we find the global minimum. To have a guarantee, we have to use a global optimization algorithm that is designed to find the global minimum. Therefore, later, we present two global optimization approaches: one deterministic and various heuristical methods. These global algorithms depend on the structure of the objective functions; therefore, only the VTN-specification will be discussed.

3.2.1 SSI method

First, we present one of the most popular methods of solving the phase stability testing problem that is the SSI (Successive Substitution Iteration) method, see, e.g., Firoozabadi (1999); Firoozabadi and Pan (2002); Hoteit and Firoozabadi (2006b); Michelsen (1982a). This method is based on writing the chemical potentials using the fugacity or volume functions and satisfying the chemical equilibrium condition (2.49). The popularity is caused by the simple derivation, straightforward implementation, and robust behaviour. However, the popularity of the SSI method is mainly in the PTN-specification. In other specifications, the use of the SSI method is not common. Recently, Bi et al. (2020b) presented the implementation of the SSI method in the UVN-phase stability testing problem. Therefore, in this thesis, we will present a critical comparison of the SSI method with the Newton–Raphson method in each individual specification. See Section 3.3.5 for the results. Here, we first present the PTN-phase stability testing. Then, the VTN-, and UVN-formulation will be discussed.

PTN-specification

The SSI method (or its combination with the Newton–Raphson iterations) in the PTN specification is one of the most frequently used algorithms, see, e.g., Li and Firoozabadi (2012), Sherafati and Jessen (2017), or Nichita et al. (2007). Here, we derive the SSI method based on Firoozabadi (2016). In the PTN-phase stability testing, the stationary conditions are

$$0 = \frac{\partial \mathrm{TPD}_g}{\partial x_j}\left(\mathbf{x};\mathbf{x}^*\right), \quad j \in \widehat{n-1}, \tag{3.27}$$

where $\mathbf{x} = (x_1, \ldots, x_{n-1})^{\mathrm{T}}$. Using equation (3.26) and denoting $x_n = 1 - \sum\limits_{i=1}^{n-1} x_i$, the partial derivative of TPD_g reads as

$$\frac{\partial \mathrm{TPD}_g}{\partial x_j}\left(\mathbf{x};\mathbf{x}^*\right) = \mu_j\left(\mathbf{x}\right) - \mu_j(\mathbf{x}^*) + \sum_{i=1}^{n} x_i \frac{\partial \mu_i}{\partial x_j}\left(\mathbf{x}\right) + \mu_n\left(\mathbf{x}^*\right) - \mu_n\left(\mathbf{x}\right), \quad j \in \widehat{n-1}. \tag{3.28}$$

Using the Gibbs-Duhem equation (2.117) at constant T and P gives

$$\sum_{i=1}^{n} x_i \frac{\partial \mu_i}{\partial x_j}\left(\mathbf{x}\right) = 0, \quad j \in \widehat{n-1}. \tag{3.29}$$

Combining equations (3.28) and (3.29) results in

$$\frac{\partial \mathrm{TPD}_g}{\partial x_j}\left(\mathbf{x};\mathbf{x}^*\right) = \mu_j\left(\mathbf{x}\right) - \mu_j\left(\mathbf{x}^*\right) + \mu_n\left(\mathbf{x}^*\right) - \mu_n\left(\mathbf{x}\right), \quad j \in \widehat{n-1}. \tag{3.30}$$

Therefore, the stability conditions (3.27) read as

$$\mu_j\left(\mathbf{x}\right) - \mu_j(\mathbf{x}^*) = \mu_n\left(\mathbf{x}\right) - \mu_n\left(\mathbf{x}^*\right) = K\left(\mathbf{x};\mathbf{x}^*\right), \quad j \in \widehat{n-1}, \tag{3.31}$$

where we define $K\left(\mathbf{x};\mathbf{x}^*\right) = \mu_j\left(\mathbf{x}\right) - \mu_j(\mathbf{x}^*)$. One can see that K is independent on j. The stationary conditions (3.31) can be written in a single equation as

$$\mu_i\left(\mathbf{x}\right) - \mu_i(\mathbf{x}^*) = K\left(\mathbf{x};\mathbf{x}^*\right), \quad i \in \widehat{n}. \tag{3.32}$$

Moreover, at the stationary points (i.e., points that satisfy conditions (3.27)) the function TPD_g can be evaluated using

$$\mathrm{TPD}_g\left(\mathbf{x};\mathbf{x}^*\right) = \sum_{i=1}^{n-1} x_i K\left(\mathbf{x};\mathbf{x}^*\right) - \left(\sum_{i=1}^{n-1} x_i - 1\right) K\left(\mathbf{x};\mathbf{x}^*\right) = K\left(\mathbf{x};\mathbf{x}^*\right), \tag{3.33}$$

where we used equation (3.31). Therefore, K represents the value of the function TPD_g at stationary points. Using the fugacity and fugacity coefficients introduced in Section 2.6.1, the conditions (3.32) can be rewritten to

$$\ln \frac{x_i}{x_i^*} + \ln \frac{\varphi_i(\mathbf{x})}{\varphi_i(\mathbf{x}^*)} = k, \quad i \in \widehat{n}, \tag{3.34}$$

where $k = \frac{K(\mathbf{x};\mathbf{x}^*)}{RT}$, and we omit the dependence on $\mathbf{x}$. Let us define new variables $\mathbf{X} = (X_1, \ldots, X_n)^{\mathrm{T}}$ with

$$\ln X_i = \ln x_i - k, \quad i \in \widehat{n}, \tag{3.35}$$

or equivalently

$$x_i = X_i e^k, \quad i \in \widehat{n}. \tag{3.36}$$

Moreover, using $\sum\limits_{i=1}^{n} x_i = 1$, the term e^k can be expressed as

$$e^k = \frac{1}{\sum\limits_{i=1}^{n} X_i}. \tag{3.37}$$

Combining equations (3.36) and (3.37) gives

$$x_i = \frac{X_i}{\sum\limits_{j=1}^{n} X_j}. \tag{3.38}$$

Therefore, the new independent variables X_i can formally be interpreted as mole numbers. Then, equation (3.34) reads as

$$\frac{\ln X_i}{\ln \mathbf{x}_i^*} + \ln \frac{\varphi_i(\mathbf{X})}{\varphi_i(\mathbf{x}^*)} = 0, \quad i \in \widehat{n}. \tag{3.39}$$

Rearranging the previous equation to express X_i results in

$$X_i = \exp\left\{\ln x_i^* + \ln \frac{\varphi_i(\mathbf{x}^*)}{\varphi_i(\mathbf{X})}\right\}, \quad i \in \widehat{n}. \tag{3.40}$$

The previous equation can be used to iterate X_i as

$$X_i^{(k+1)} = \exp\left\{\ln x_i^* + \ln \frac{\varphi_i(\mathbf{x}^*)}{\varphi_i\left(\mathbf{X}^{(k)}\right)}\right\}, \quad i \in \widehat{n}, \tag{3.41}$$

where k is the iteration index, and $\mathbf{X}^{(0)} = \left(X_1^{(0)}, \ldots, X_n^{(0)}\right)^{\mathrm{T}}$ is an initial approximation. However, more than one initial approximation has to be used. The strategy for the initialization is presented in Section 3.2.2. Lastly, a stopping criterion has to given. In this thesis, we stop the SSI iterations given by equation (3.41) if

$$\left\|\mathbf{X}^{(k+1)} - \mathbf{X}^{(k)}\right\|_2 < \varepsilon, \tag{3.42}$$

where ε is a given tolerance, e.g., $\varepsilon = 10^{-8}$, and the norm $\|\cdot\|_2$ is the Euclidean norm. The complete algorithm of the SSI method in the PTN-phase stability testing problem is summarized in Algorithm 1.

VTN-specification

The SSI algorithm for the VTN-specification was briefly described by Mikyška and Firoozabadi (2012). However, they reported poor behaviour of the SSI iterations. In the VTN-phase stability testing, the stationary conditions are

$$0 = \frac{\partial \mathrm{TPD}_a}{\partial c_i}(\mathbf{c}; \mathbf{c}^*) = \mu_i(\mathbf{c}) - \mu_i(\mathbf{c}^*), \quad i \in \widehat{n}. \tag{3.43}$$

Using the volume functions introduced in Section 2.6.2, the previous conditions can be rewritten as

$$\ln \frac{c_i}{c_i^*} + \ln \frac{\Phi_i(\mathbf{c}^*)}{\Phi_i(\mathbf{c})} = 0, \quad i \in \widehat{n}. \tag{3.44}$$

Algorithm 1: SSI method for PTN-phase stability testing.

```
   Input  : $P^*, T^* > 0$, $\mathbf{x}^* = (x_1^*, \dots, x_n^*)^{\mathrm{T}}$, $\varepsilon > 0$
   Output : true = state $\mathbf{x}^*$ is stable, false = state $\mathbf{x}^*$ is unstable
 1 compute the fugacities of the initial state $\varphi_i(\mathbf{x}^*)$ using equation (2.169)
 2 compute $s$ initial approximations using the strategy from Section 3.2.2
 3 for $j \in \hat{s}$ do
 4     set $\mathbf{X}^{(0)}$ as the $j$-th initial approximation
 5     set iteration counter $k = 0$
 6     while $k <$ maximum number of iterations do
 7         compute the fugacities of the trial state $\varphi_i\left(\mathbf{X}^{(k)}\right)$ using equation (2.169)
 8         compute $\mathbf{X}^{(k+1)}$ using equation (3.41)
 9         if $\left\|\mathbf{X}^{(k+1)} - \mathbf{X}^{(k)}\right\|_2 < \varepsilon$ then
10             break
11         end
12         set $k = k + 1$
13     end
14     if $\mathrm{TPD}_g\left(\mathbf{x}^{(k+1)};\mathbf{x}^*\right) < 0$ then
15         return false
16     end
17 end
18 return true
```

Rearranging the previous equation to express the trial phase concentration c_i results in

$$c_i = \exp\left\{\ln c_i^* - \ln \frac{\Phi_i(\mathbf{c}^*)}{\Phi_i(\mathbf{c})}\right\}, \quad i \in \hat{n}. \tag{3.45}$$

The previous equation can be used to iterate the concentrations as

$$c_i^{(k+1)} = \exp\left\{\ln c_i^* - \ln \frac{\Phi_i(\mathbf{c}^*)}{\Phi_i\left(\mathbf{c}^{(k)}\right)}\right\}, \quad i \in \hat{n}. \tag{3.46}$$

The strategy for the initialization is presented in Section 3.2.2. Lastly, a stopping criterion has to be given. In this thesis, we stop the SSI iteration given by equation (3.46) if

$$\left\|\mathbf{c}^{(k+1)} - \mathbf{c}^{(k)}\right\|_{pn} < \varepsilon, \tag{3.47}$$

where ε is a given tolerance, e.g., $\varepsilon = 10^{-8}$, and the phase norm $\|\cdot\|_{pn}$ is in the VTN-specification defined as

$$\left\|(c_1, \dots, c_n)^{\mathrm{T}}\right\|_{pn}^2 = \sum_{i=1}^{n} \frac{c_i^2}{(c_i^*)^2}. \tag{3.48}$$

The complete algorithm of the SSI method in the VTN-phase stability testing problem is summarized in Algorithm 2.

Algorithm 2: SSI method for VTN-phase stability testing.

```
   Input  : $T^* > 0$, $\mathbf{c}^* = (c_1^*, \dots, c_n^*)^{\mathrm{T}}$, $\varepsilon > 0$
   Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
 1 compute the volume functions of the initial state $\Phi_i(\mathbf{c}^*)$ using equation (2.180)
 2 compute $s$ initial approximations using the strategy from Section 3.2.2
 3 for $j \in \widehat{s}$ do
 4     set $\mathbf{c}^{(0)}$ as the $j$-th initial approximation
 5     set iteration counter $k = 0$
 6     while $k <$ maximum number of iterations do
 7         compute the volume functions of the trial state $\Phi_i\left(\mathbf{c}^{(k)}\right)$ using equation (2.180)
 8         compute $\mathbf{c}^{(k+1)}$ using equation (3.46)
 9         if $\left\|\mathbf{c}^{(k+1)} - \mathbf{c}^{(k)}\right\|_{pn} < \varepsilon$ then
10             break
11         end
12         set $k = k + 1$
13     end
14     if $\mathrm{TPD}_a\left(\mathbf{c}^{(k+1)};\mathbf{c}^*\right) < 0$ then
15         return false
16     end
17 end
18 return true
```

UVN-specification

In the UVN-phase stability testing, the stationary conditions are

$$0 = \frac{\partial \mathrm{TPD}_s}{\partial c_i}(\mathbf{y};\mathbf{y}^*) = -\frac{\mu_i(\mathbf{y})}{T(\mathbf{y})} + \frac{\mu_i(\mathbf{y}^*)}{T(\mathbf{y}^*)}, \quad i \in \widehat{n}, \tag{3.49}$$

$$0 = \frac{\partial \mathrm{TPD}_s}{\partial u}(\mathbf{y};\mathbf{y}^*) = -\frac{1}{T(\mathbf{y})} + \frac{1}{T(\mathbf{y}^*)}, \tag{3.50}$$

where $\mathbf{y} = (u, c_1, \dots, c_n)^{\mathrm{T}}$ and $T(\mathbf{y})$ is the temperature given to the state $\mathbf{y}$. Equation (3.50) implies

$$T(\mathbf{y}) = T(\mathbf{y}^*), \tag{3.51}$$

i.e., the temperature of the trial phase $T(\mathbf{y})$ has to be equal to the temperature of the initial state $T(\mathbf{y}^*)$. Therefore, the internal energy density u of the trial phase can be calculated using the thermal energy equation of state

$$u = U^{(EOS)}\left(T(\mathbf{y}^*), 1, c_1, \dots, c_n\right), \tag{3.52}$$

and only the trial phase concentrations $c_1, \dots, c_n$ have to be found. These concentrations are found using the SSI method in the VTN-specification. The complete algorithm of the SSI method in the UVN-phase stability testing problem is summarized in Algorithm 3.

3.2.2 Initial approximations

In the SSI and other local methods, multiple initial approximations have to be constructed. The selection of the initial approximations $\mathbf{y}^{(0)}$ depends on the function f as the VTN-, UVN-, and PTN-formulations use different strategies. Here, we describe each of them.

Algorithm 3: SSI method for UVN-phase stability testing.

```
Input   : $u^*$, $\mathbf{c}^* = (c_1^*, \dots, c_n^*)^{\mathrm{T}}$, $\varepsilon > 0$
Output : true = state $(u^*, c_1^*, \dots, c_n^*)$ is stable, false = state $(u^*, c_1^*, \dots, c_n^*)$ is unstable
1  compute initial temperature $T^*$ using thermal energy $u = U^{(EOS)}(T^*, 1, c_1^*, \dots, c_n^*)$
2  compute the volume functions of the initial state $\Phi_i(\mathbf{c}^*)$ with temperature $T^*$ using
     equation (2.180)
3  compute $s$ initial approximations using the strategy from Section 3.2.2
4  for $j \in \widehat{s}$ do
5      set $\mathbf{c}^{(0)}$ as the $j$-th initial approximation
6      set iteration counter $k = 0$
7      while $k <$ maximum number of iterations do
8          compute the volume functions of the trial state $\Phi_i\left(\mathbf{c}^{(k)}\right)$ using temperature $T^*$
             and equation (2.180)
9          compute $\mathbf{c}^{(k+1)}$ using equation (3.46)
10         compute $u^{(k+1)} = U^{(EOS)}\left(T^*, 1, \mathbf{c}^{(k+1)}\right)$
11         if $\left\|\mathbf{c}^{(k+1)} - \mathbf{c}^{(k)}\right\|_{pn} < \varepsilon$ then
12             break
13         end
14         set $k = k + 1$
15     end
16     denote $\mathbf{y} = \left(u^{(k+1)}, c_1^{(k+1)}, \dots, c_n^{(k+1)}\right)^{\mathrm{T}}$
17     if $\mathrm{TPD}_s(\mathbf{y};\mathbf{c}^*) < 0$ then
18         return false
19     end
20 end
21 return true
```

***PTN*-specification**

In the PTN-specification, we employ a strategy from Michelsen (1982a). Two initial approximations are used, and the derivation is based on the Wilson correlation (see Section 2.5.4) that approximates the equilibrium K_i-factors. Then, the first initial mole fractions are chosen as

$$x_i^{(0)} = K_i x_i^*, \quad i \in \widehat{n}, \tag{3.53}$$

where we assume that the initial phase $\mathbf{x}^*$ is liquid. Second, we assume that the initial phase is vapour, and the the second initial approximation is defined as

$$x_i^{(0)} = \frac{x_i^*}{K_i}, \quad i \in \widehat{n}. \tag{3.54}$$

***VTN*-specification**

In the VTN-specification, we employ a strategy from Mikyška and Firoozabadi (2012), which is based on the saturation pressure $P_i^{\mathrm{sat}}(T)$ of each component i. At a given temperature T, the saturation pressure is the pressure at which a given liquid and its vapour can co-exist in equilibrium. In this thesis, the saturation pressure of the i-th component is estimated using

$$P_i^{(\mathrm{sat})}(T) = P_{c,i} \exp\left\{5.37\,(1+\omega_i)\left(1 - \frac{T_{c,i}}{T}\right)\right\}. \tag{3.55}$$

First, the initial pressure is estimated as

$$P_{\mathrm{ini}} = \sum_{i=1}^{n} P_i^{(\mathrm{sat})}(T^*) x_i^*. \tag{3.56}$$

Then, the initial mole fraction of component i is estimated as

$$x_i^{(0)} = \frac{P_i^{(\mathrm{sat})}(T^*)}{P_{\mathrm{ini}}} x_i^*, \quad i \in \widehat{n}. \tag{3.57}$$

Lastly, the initial total concentration $c^{(0)}$ is evaluated from the equation of state. In the case of the cubic equation of state, up to three possible roots are given. To ensure convergence toward the global minimum, we add another up to three initial approximations. In this case, we estimate the initial mole fractions as

$$x_i^{(0)} = \frac{x_i^*}{P_i^{(\mathrm{sat})}(T^*)} \frac{1}{\sum\limits_{j=1}^{n} \frac{x_j^*}{P_j^{(\mathrm{sat})}(T^*)}}, \quad i \in \widehat{n}. \tag{3.58}$$

Then, the initial pressure is estimated by

$$P_{\mathrm{ini}} = \sum_{i=1}^{n} P_i^{(\mathrm{sat})}(T^*) x_i^{(0)}. \tag{3.59}$$

The initial total concentration $c^{(0)}$ is again evaluated from the equation of state. To conclude, in the case of a cubic equation of state, up to six initial approximations are used.

***UVN*-specification**

In the UVN-specification, we employ a strategy from Smejkal and Mikyška (2017). Since the temperature of the trial phase has to be equal to the temperature of the initial phase, we only have to set the initial concentration $c_i^{(0)}$. The initial internal energy density $u^{(0)}$ is then computed using equation (2.223). The initial concentrations $c_i^{(0)}$ are computed using the same strategy as in the VTN-specification.

3.2.3 Modified Newton–Raphson method

Second, we present a numerical method based on the Newton–Raphson iterations. According to Hoteit and Firoozabadi (2006b) and others, the main problem of the SSI method in the PTN-specification is at the vicinity of the critical points. Here, a large number (thousands) of iterations have to be used, and the computational time rises unacceptably. Therefore, we present a second-order convergent method based on the Newton–Raphson iterations (Raphson (1697)). This method is designed for the unified formulation from Section 3.1.1; therefore, the core of the algorithm is identical to all specifications that fit the unified formulation.

In general, the Newton–Raphson method is an iterative root-finding algorithm. Here, we use it to find the points where the necessary extreme conditions

$$\frac{\partial \mathrm{TPD}}{\partial y_i}(\mathbf{y};\mathbf{x}^*) = 0, \quad i \in \widehat{m}, \tag{3.60}$$

are satisfied. Components of the gradient $\boldsymbol{\nabla}$TPD of the function TPD can be derived by differentiation of TPD with respect to y_i, which results in

$$[\boldsymbol{\nabla}\mathrm{TPD}\,(\mathbf{y};\mathbf{x}^*)]_i = \frac{\partial \mathrm{TPD}}{\partial y_i}\,(\mathbf{y};\mathbf{x}^*) = -\frac{\partial f}{\partial y_i}\,(\mathbf{x}^*) + \frac{\partial f}{\partial y_i}\,(\mathbf{y})\,, \quad i \in \widehat{m}. \tag{3.61}$$

Then, we are going to solve a system of m non-linear algebraic equations

$$\frac{\partial \mathrm{TPD}}{\partial y_i}\,(\mathbf{y};\mathbf{x}^*) = 0, \quad i \in \widehat{m}. \tag{3.62}$$

The system (3.62) can be solved iteratively using the modified Newton–Raphson method, which reads as

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \lambda^{(k)}\boldsymbol{\Delta}\mathbf{y}^{(k)}. \tag{3.63}$$

Here $\lambda^{(k)} \in (0,1]$ is a damping parameter, and $\boldsymbol{\Delta}\mathbf{y}^{(k)}$ is an increment, which is determined as a solution of the system

$$\mathbf{H}\left(\mathbf{y}^{(k)}\right)\boldsymbol{\Delta}\mathbf{y}^{(k)} = -\boldsymbol{\nabla}\mathrm{TPD}\left(\mathbf{y}^{(k)};\mathbf{x}^*\right), \tag{3.64}$$

where $\mathbf{H} \in \mathbb{R}^{m,m}$ is the Hessian matrix of function TPD and $\boldsymbol{\nabla}$TPD is the gradient of function TPD. Using equation (3.61) the elements of the Hessian matrix can be computed as

$$[\mathbf{H}]_{i,j}\,(\mathbf{y}) = \frac{\partial^2 f}{\partial y_i \partial y_j}\,(\mathbf{y})\,, \quad i,j \in \widehat{m}. \tag{3.65}$$

Once the increment direction $\boldsymbol{\Delta}\mathbf{y}^{(k)}$ has been established, the damping factor $\lambda^{(k)}$ has to be determined with the so-called line-search technique. This strategy will be presented later.

Modified Cholesky decomposition

When using the modified Newton–Raphson method, iterates may not converge towards a local minimum but also to a local maximum or towards a saddle point. This problem can be avoided by modifying the Hessian matrix. The aim of the modification is to guarantee that the value of function TPD will decrease in each iteration. The decrease of the objective function can be enforced by the following theorem.

Theorem 3.5 (Boyd and Vandenberghe (2004)). *Let $f : \mathbb{R}^m \to \mathbb{R}$ be a C^2 function (has continuous first and second partial derivatives), $\boldsymbol{\nabla} f$ be the gradient function, and $\mathbf{H}$ be the Hessian matrix of the function f. Let $\mathbf{H}(\mathbf{y})$ be positive definite. Then, the solution $\Delta\mathbf{y}$ of the system*

$$\mathbf{H}(\mathbf{y})\Delta\mathbf{y} = -\boldsymbol{\nabla} f(\mathbf{y}), \tag{3.66}$$

gives a direction in which function the function f will decrease. Mathematically,

$$\lim_{t\to 0+} \frac{\mathrm{d}}{\mathrm{d}t} f(\mathbf{y} + t\Delta\mathbf{y}) < 0. \tag{3.67}$$

Proof. Using properties of f, the derivative chain rule, and the continuity of the gradient function, one can directly obtain

$$\begin{aligned}
\lim_{t\to 0+} \frac{\mathrm{d}}{\mathrm{d}t} f(\mathbf{y} + t\Delta\mathbf{y}) &= \lim_{t\to 0+} \sum_{k=1}^{m} \frac{\partial f}{\partial y_k}\,(\mathbf{y} + t\Delta\mathbf{y})\,[\Delta\mathbf{y}]_k \\
&= \lim_{t\to 0+} \left[\boldsymbol{\nabla} f\,(\mathbf{y} + t\Delta\mathbf{y})\right]^{\mathrm{T}} \Delta\mathbf{y} \\
&= \left[\boldsymbol{\nabla} f\,(\mathbf{y})\right]^{\mathrm{T}} \Delta\mathbf{y}.
\end{aligned} \tag{3.68}$$

Using equation (3.66) the gradient $\boldsymbol{\nabla} f(\mathbf{y})$ can be expressed and the limit reads as

$$\lim_{t\to 0+} \frac{\mathrm{d}}{\mathrm{d}t} f(\mathbf{y} + t\Delta\mathbf{y}) = -(\Delta\mathbf{y})^{\mathrm{T}} \mathbf{H}(\mathbf{y})^{\mathrm{T}} \Delta\mathbf{y} = -(\Delta\mathbf{y})^{\mathrm{T}} \mathbf{H}(\mathbf{y}) \Delta\mathbf{y} < 0. \tag{3.69}$$

In the last two steps, the symmetry and the positive-definiteness of the matrix $\mathbf{H}(\mathbf{y})$ was used. □

Therefore, using Theorem 3.5 if the Hessian matrix is positive definite, for sufficiently low values of the damping parameter $\lambda^{(k)}$, we have TPD($\mathbf{y}^{(k+1)}$;$\mathbf{x}^*$) < TPD($\mathbf{y}^{(k)}$;$\mathbf{x}^*$). If the Hessian is not positive definite, then the value of TPD in the next iteration can be either higher or lower than in the previous iteration. In this case, we have to modify the Hessian matrix so that it becomes positive definite. The positive definiteness of the modified Hessian is ensured using the modified Cholesky decomposition of the Hessian matrix $\mathbf{H}$. The standard Cholesky decomposition of a symmetric positive definite matrix $\mathbf{H}$ factorizes the matrix into a product of a lower triangular matrix $\mathbf{L}$ and its transpose, i.e.,

$$\mathbf{H} = \mathbf{L}\mathbf{L}^{\mathrm{T}}. \tag{3.70}$$

The numerical algorithm known as Crout–Cholesky is presented in Algorithm 4. The presented version is known as *in-place*, where the resulting matrix $\mathbf{L}$ replaces the given matrix $\mathbf{H}$.

Algorithm 4: In-place Crout–Cholesky decomposition algorithm.

```
   Input  : $\mathbf{A} \in \mathbb{R}^{m,m}$ symmetric positive definite stored in lower triangle
   Output : $\mathbf{L}$ satisfying $\mathbf{L}\mathbf{L}^{\mathrm{T}} = \mathbf{A}$
 1 for $j = 1, \dots, m$ do
 2     (perform the $j$-th iteration)
 3     $[\mathbf{A}]_{j,j} = \sqrt{[\mathbf{A}]_{j,j}} \quad \left(= [\mathbf{L}]_{j,j}\right)$
 4     for $i = j+1, \dots, m$ do
 5         $[\mathbf{A}]_{i,j} = \frac{[\mathbf{A}]_{i,j}}{[\mathbf{A}]_{j,j}} \quad \left(= [\mathbf{L}]_{i,j}\right)$
 6         for $k = j+1, \dots, i$ do
 7             $[\mathbf{A}]_{i,k} = [\mathbf{A}]_{i,k} - [\mathbf{A}]_{i,j} [\mathbf{A}]_{k,j}$
 8         end
 9     end
10 end
```

If $\mathbf{H}$ is not positive definite, its Cholesky decomposition may not exist or may be unstable (see Gill and Murray (1974)). The modified Cholesky decomposition is performed in the same way as the standard Cholesky decomposition; however, when a negative or a too-small element appears at the diagonal of the factorized matrix, the corresponding diagonal element of $\mathbf{H}$ is increased so as to become sufficiently large. This way, we obtain a Cholesky factorization of a modified matrix $\mathbf{H} + \mathbf{E}$, where $\mathbf{E}$ is a diagonal matrix with non-negative elements. For the modification, we impose the following properties.

1. If $\mathbf{H}$ is sufficiently positive definite, then $\mathbf{E} = 0$.
2. If $\mathbf{H}$ is not positive definite, then $\|\mathbf{E}\|$ is as small as possible.
3. The matrix $\mathbf{H} + \mathbf{E}$ is well-conditioned.
4. Low computation cost.

The first two properties guarantee fast convergence of the Newton–Raphson method. The third one implies a good numerical solution of the linear system arising from the Newton–Raphson method. In the literature, there exists a large number of numerical algorithms for the modified Cholesky computation, e.g., Gill and Murray (1974) or Schnabel and Eskow (1990). Fang and O'Leary (2007) provided a catalog and gave a critical comparison of several algorithms. In this thesis, we use the algorithm presented by Schnabel and Eskow (1999). Now, we present the essential steps. The pseudo-code of the algorithm is presented in Algorithm 5. Let a symmetric $\mathbf{H} \in \mathbb{R}^{m,m}$ be given. The algorithm has two stages. In stage one, the classical Cholesky decomposition is performed. This algorithm starts from the upper left corner of the matrix and continues to calculate the resulting matrix column by column. The algorithm is presented in Algorithm 4. Moreover, a pivoting process on the maximum diagonal of the remaining sub-matrix is performed to ensure the numerical stability. In stage two, the algorithm performs modification. The switch from stage one to stage two is made if, for the current j, one of three conditions is met:

$$\max_{j \leqslant i \leqslant m} \left[\mathbf{H}^{(j)}\right]_{i,i} < \tilde{\tau}\gamma, \tag{3.71}$$

$$\min_{j \leqslant i \leqslant m} \left[\mathbf{H}^{(j)}\right]_{i,i} < -\mu \left(\max_{j \leqslant i \leqslant m} \left[\mathbf{H}^{(j)}\right]_{i,i} \right), \tag{3.72}$$

$$\min_{j+1 \leqslant i \leqslant m} \left(\left[\mathbf{H}^{(j)}\right]_{i,i} - \frac{\left[\mathbf{H}^{(j)}\right]_{i,j}^2}{\left[\mathbf{H}^{(j)}\right]_{j,j}} \right) < -\mu\gamma. \tag{3.73}$$

where $\mu = 0.1$, $\gamma = \max_{1 \leqslant i \leqslant m} \left|[\mathbf{H}]_{i,i}\right|$, $\tilde{\tau} = \sqrt[3]{\varepsilon}$, ε is the machine precision, and $\left[\mathbf{H}^{(j)}\right]_{i,i}$ is the remaining modified sub-matrix after $j-1$ iterations. The first condition defines the limit, where the diagonal element is too small. According to Schnabel and Eskow (1999), the condition (3.72) stops the stage one earlier and leads to a lower value of $\|\mathbf{E}\|$. Lastly, Schnabel and Eskow (1999) state that if the third condition (3.73) is not satisfied, the values of $\mathbf{E}$ resulting from the next iteration (for $j := j+1$) would be too large.

Remark 3.6. If the in-place algorithm of the Cholesky decomposition is used, one can use $\mathbf{H} = \mathbf{H}^{(j)}$ for all j, and the presentation and implementation of the Cholesky decomposition is simple. However, in the presentation, we strictly use the superscript to differentiate between the original matrix and the remaining modified sub-matrix.

In stage two, we have to compute the values $\delta_j = [\mathbf{E}]_{jj}$, i.e., the positive values that are added to the diagonal of $\mathbf{H}$ to be sufficiently positive. Based on the iteration j, a different strategy is used:

- If $j = m$, i.e., only the final 1×1 sub-matrix remains, we set

$$\delta_m = -\left[\mathbf{H}^{(m)}\right]_{m,m} + \max\left\{ -\tau \frac{\left[\mathbf{H}^{(m)}\right]_{m,m}}{1-\tau}, \tilde{\tau}\gamma \right\}. \tag{3.74}$$

- If $j < m-2$, we set

$$\delta_j = \max\left\{ 0, -\left[\mathbf{H}^{(j)}\right]_{j,j} + \max\left\{q_j, \tilde{\tau}\ \right\}, \delta_{j-1} \right\}, \tag{3.75}$$

  where

$$q_j = \sum_{i=j+1}^{n} \left| \left[\mathbf{H}^{(j)}\right]_{i,j} \right|. \tag{3.76}$$

- If $j = m - 2$, i.e., the final 2×2 sub-matrix remains, we find its eigenvalues λ_1, λ_2, $\lambda_1 \leqslant \lambda_2$. Then, we set

$$\delta_j = \max\left\{0, -\lambda_1 + \max\left\{\tau\frac{\lambda_2 - \lambda_1}{1 - \tau}, \tilde{\tau}\gamma\right\}, \delta_{j-1}\right\}. \tag{3.77}$$

These values ensure that the resulting matrix $\mathbf{H} + \mathbf{E}$ is sufficiently positive definite. Moreover, in each iteration j a pivoting process on maximum lower Gerschgorin bound estimate g_i is made. These lower bounds are first computed using

$$g_i = \left[\mathbf{H}^{(j)}\right]_{i,i} - \sum_{l=k+1}^{i-1} \left|\left[\mathbf{H}^{(j)}\right]_{i,l}\right| - \sum_{l=i+1}^{m} \left|\left[\mathbf{H}^{(j)}\right]_{l,i}\right|, \tag{3.78}$$

for $i = k + 1, \ldots, m$, where k is the number of successful iterations performed in stage one. Then, after each iteration j, the lower bounds are updated using

$$g_i = g_i + \left|\left[\mathbf{H}^{(j)}\right]_{i,j}\right| \left(1 - \frac{\sum_{i=j+1}^{m} \left|[\mathbf{H}^{(j)}]_{i,j}\right|}{[\mathbf{H}^{(j)}]_{j,j}}\right). \tag{3.79}$$

Line-search strategy in the Newton–Raphson iterations

The classical Newton–Raphson algorithm iterates the solution using

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \boldsymbol{\Delta}\mathbf{y}^{(k)}. \tag{3.80}$$

However, using this formula, the new solution $\mathbf{y}^{(k+1)}$ can be outside the feasible domain $\mathcal{D}$ (the so-called overshooting). To prevent this situation, a line-search technique with a damping parameter $\lambda^{(k)}$ is employed. First, we set $\lambda^{(k)} = 1$ and test whether

$$\mathbf{y}^{(k)} + \lambda^{(k)}\boldsymbol{\Delta}\mathbf{y}^{(k)} \in \mathcal{D}. \tag{3.81}$$

If the condition does not hold, we iterate $\lambda^{(k)} := \frac{\lambda^{(k)}}{2}$, until the previous condition is fulfilled. Moreover, if we are using the modified Cholesky decomposition, the line-search technique can be extended to ensure the descent of the objective function TPD. In other words, we iterate $\lambda^{(k)} := \frac{\lambda^{(k)}}{2}$ until

$$\mathrm{TPD}(\mathbf{y}^{(k)} + \lambda^{(k)}\boldsymbol{\Delta}\mathbf{y}^{(k)};\mathbf{x}^*) < \mathrm{TPD}(\mathbf{y}^{(k)};\mathbf{x}^*). \tag{3.82}$$

Since we are using the modified Cholesky decomposition, such $\lambda^{(k)} > 0$ has to exist.

Stopping criteria

The Newton–Raphson iterations are stopped if the increment of the solution $\Delta\mathbf{y}^{(k)}$ satisfies

$$\left\|\boldsymbol{\Delta}\mathbf{y}^{(k)}\right\|_{pn} < \varepsilon, \tag{3.83}$$

where $\varepsilon = 10^{-6}$ is a given tolerance. The phase norm $\|\cdot\|_{pn}$ is specification-dependent. Now, we present the definition for the VTN-, UVN-, and PTN-specifications. In the VTN-formulation, $\mathbf{y} = (c_1, \ldots, c_n)^{\mathrm{T}}$, and the norm $\|\cdot\|_{pn}$ is defined by

$$\|\mathbf{y}\|_{pn}^2 = \sum_{i=1}^{n} \frac{c_i^2}{(c_i^*)^2}. \tag{3.84}$$

Algorithm 5: Modified Cholesky decomposition from Schnabel and Eskow (1999). The in-place version of the algorithm is presented where the resulting matrix **L** overwrites the given matrix **H**.

```
   Input  : $\mathbf{H} \in \mathbb{R}^{m,m}$ symmetric
   Output : $\mathbf{L}$ satisfying $\mathbf{L}\mathbf{L}^{\mathrm{T}} = \mathbf{H} + \mathbf{E}$
1  set $j = 1$, stageOne = True
2  while $j \in \widehat{m}$ and stageOne = True do
3      if condition (3.71) is satisfied or condition (3.72) is satisfied then
4          stageOne = False
5      end
6      else
7          pivot on the maximum diagonal of the remaining submatrix
8          if condition (3.73) is satisfied then
9              stageOne = False
10         end
11         else
12             perform $j$-th iteration of the classical Cholesky decomposition using Alg. 4
13         end
14     end
15     j = j+1
16 end
17 if stageOne = False and $j = m$ then
18     calculate $\delta_m$ using equation (3.74) and add it to the diagonal
19         $$[\mathbf{H}]_{m,m} := [\mathbf{H}]_{m,m} + \delta_m$$
       perform $m$-th iteration of the Cholesky decomposition using Alg. 4
20 end
21 if stageOne = False and $j < m$ then
22     set $k = j - 1$
23     calculate lower Gerschgorin bounds of $\mathbf{H}$ using equation (3.78)
24     pivot on maximum lower Gerschgorin bound estimate
25     for $j = k + 1, \dots, n - 2$ do
26         pivot on maximum lower Gerschgorin bound estimate
27         calculate $\delta_j$ using equation (3.75) and add it to diagonal
               $$[\mathbf{H}]_{j,j} := [\mathbf{H}]_{j,j} + \delta_j$$
           update Gerschgorin bound estimates using equation (3.79)
28         perform $j$-th iteration of Cholesky decomposition using Alg. 4
29     end
30     compute eigenvalues $\lambda_1, \lambda_2$
31     compute $\delta$ using (3.77) and for $i = m - 1, m$ add to the diagonal
           $$[\mathbf{H}]_{i,i} := [\mathbf{H}]_{i,i} + \delta$$
32     perform $(m - 1)$-th and $m$-th iteration of the Cholesky decomposition using Alg. 4
33 end
```

In the UVN-formulation, $\mathbf{y} = (u, c_1, \ldots, c_n)^{\mathrm{T}}$, and

$$\|\mathbf{y}\|_{pn}^2 = \sum_{i=1}^{n} \frac{c_i^2}{(c_i^*)^2} + \frac{u^2}{(u^*)^2}. \tag{3.85}$$

In the PTN-formulation, $\mathbf{y} = (x_1, \ldots, x_{n-1})^{\mathrm{T}}$, and

$$\|\mathbf{y}\|_{pn}^2 = \sum_{i=1}^{n-1} x_i^2. \tag{3.86}$$

Summary of the numerical algorithm for the general phase stability testing

In Algorithm 6, we summarize the essential steps of the algorithm for the general single-phase stability testing.

Algorithm 6: Modified Newton–Raphson method for the phase stability testing.

```
   Input  : $f : \mathbb{R}^m \to \mathbb{R}$, $\mathcal{D} \subset \mathbb{R}^m$ and $\mathbf{x}^* \in \mathcal{D}$
   Output : true = state $\mathbf{x}^*$ is stable, false = state $\mathbf{x}^*$ is unstable
1  compute $s$ initial approximations using the strategy from Section 3.2.2
2  for $j \in \widehat{s}$ do
3      set $\mathbf{y}^{(0)}$ as the $j$-th initial approximation
4      set iteration counter $k = 0$
5      while $k <$ maximum number of iterations do
6          assemble the Hessian matrix and gradient of the function TPD using equations
            (3.65) and (3.61)
7          evaluate the increment of the solution $\Delta\mathbf{y}^{(k)} \in \mathbb{R}^m$ by solving the system of linear
            algebraic equations (3.62) using the modified Cholesky decomposition
8          determine $\lambda^{(k)} > 0$ using the line-search strategy
9          update the solution using
                 $\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \lambda^{(k)}\Delta\mathbf{y}^{(k)}$
           if $\left\|\Delta\mathbf{y}^{(k)}\right\|_{pn} < \varepsilon$ then
10             break
11         end
12         set $k := k + 1$
13     end
14     if TPD$\left(\mathbf{y}^{(k+1)};\mathbf{x}^*\right) < 0$ then
15         return false
16     end
17 end
18 return true
```

3.2.4 Efficient solution of linear systems arising from the linearization of the VTN-phase stability

In Section 3.2.3, we presented a local algorithm for solving the phase stability testing problem based on the Newton–Raphson method. The two basic steps in each iteration are

1. Assemble gradient and Hessian matrix.

2. Solve the system of m linear equations.

Using the modified Cholesky decomposition, the second step requires $O(m^3)$ operations. In the case of the VTN-specification, $m = n$, and is equal to the number of components in the mixture. If the system is large enough, the computational cost of the factorization can exceed the sustainable limit. Therefore, in this section, we derive a simple procedure for solving systems of linear equations arising from the linearization of the VTN-phase stability problem. Using the structure of the Hessian matrix, the solution is obtained by sequential usage of the Sherman–Morrison formula. The great advantage of the method is that the system of equations can be solved even without assembling the system. The method has been published in Smejkal and Mikyška (2021).

Sherman–Morrison formula

In this section, we review the Sherman–Morrison formula (Bartlett (1951); Sherman (1978)), for the calculation of the inverse matrix and use the symmetric variant of the formula to develop a novel procedure for the solution of the linear systems arising from the VTN-phase stability testing.

Theorem 3.7. *Let $\mathbf{B} \in \mathbb{R}^{n,n}$ be a non-singular matrix, and $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ be two non-zero vectors. Then the matrix $\mathbf{B} + \mathbf{u}\mathbf{v}^{\mathrm{T}}$ is non-singular if and only if $1 + \mathbf{v}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{u} \neq 0$. If the condition is satisfied, then*

$$(\mathbf{B} + \mathbf{u}\mathbf{v}^{\mathrm{T}})^{-1} = \mathbf{B}^{-1} - \frac{1}{1 + \mathbf{v}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{u}} \left(\mathbf{B}^{-1}\mathbf{u}\mathbf{v}^{\mathrm{T}}\mathbf{B}^{-1}\right). \tag{3.87}$$

The proof is given, e.g., in Sherman (1978). If $\mathbf{B}$ is symmetric, then it is natural to consider symmetric updates, for which $\mathbf{v} = \alpha\mathbf{u}$, where $\alpha \in \mathbb{R}$. A necessary and sufficient condition for the positive definiteness of a symmetric update of a symmetric positive definite matrix $\mathbf{B}$ is given below.

Theorem 3.8. *Let $\mathbf{B} \in \mathbb{R}^{n,n}$ be a symmetric positive definite matrix, $\alpha \in \mathbb{R}$, and $\mathbf{u} \in \mathbb{R}^n$ be a non-zero vector. Then the matrix $\mathbf{B} + \alpha\mathbf{u}\mathbf{u}^{\mathrm{T}}$ is symmetric and positive definite if and only if $1 + \alpha\mathbf{u}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{u} > 0$. If the condition is satisfied, then*

$$(\mathbf{B} + \alpha\mathbf{u}\mathbf{u}^{\mathrm{T}})^{-1} = \mathbf{B}^{-1} - \beta\mathbf{p}\mathbf{p}^{\mathrm{T}}, \tag{3.88}$$

where

$$\mathbf{p} = \mathbf{B}^{-1}\mathbf{u}, \qquad \beta = \frac{\alpha}{1 + \alpha\mathbf{u}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{u}}. \tag{3.89}$$

To prove equation (3.88), one only has to use Theorem 3.7 and set $\mathbf{v} = \alpha\mathbf{u}$. The proof of the positive definiteness is given, e.g., in Wu et al. (2001). Now, we generalise the problem by considering rank-k updates for $k \geqslant 1$. There are two ways to construct the inverse matrix of a matrix after a rank-k update. The first way is to use the following generalisation of the Sherman–Morrison formula, the so-called Woodbury formula (see Woodbury (1950) or Hager (1989)).

Theorem 3.9 (Woodbury formula)**.** *Let $\mathbf{B} \in \mathbb{R}^{n,n}$ be a non-singular matrix, $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{n,k}$, where $k \leqslant n$, and $\mathbf{I}_k$ be the identity matrix of order k. If the matrix $\mathbf{I}_k + \mathbf{V}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{U} \in \mathbb{R}^{k,k}$ is non-singular, then*

$$(\mathbf{B} + \mathbf{U}\mathbf{V}^{\mathrm{T}})^{-1} = \mathbf{B}^{-1} - \mathbf{B}^{-1}\mathbf{U}(\mathbf{I}_k + \mathbf{V}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{U})^{-1}\mathbf{V}^{\mathrm{T}}\mathbf{B}^{-1}. \tag{3.90}$$

To use this formula, one has to solve a system of k linear equations with matrix $\mathbf{I}_k + \mathbf{V}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{U}$. An alternative procedure for obtaining the inverse of matrix $\mathbf{B} + \sum_{i=1}^{k} \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}$ is to perform a series of k rank-one updates of matrix $\mathbf{B}$. This algorithm is summarized as follows. Define $\mathbf{B}_0 = \mathbf{B}$ and

$$\mathbf{B}_i = \mathbf{B}_{i-1} + \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}, \tag{3.91}$$

for $i > 0$. Then

$$\mathbf{B}_i^{-1} = \mathbf{B}_{i-1}^{-1} + \beta_i \mathbf{p}_i \mathbf{p}_i^{\mathrm{T}}, \tag{3.92}$$

where the auxiliary coefficients $\beta_i \in \mathbb{R}$ and vectors $\mathbf{p}_i \in \mathbb{R}^n$ are defined as

$$\mathbf{p}_i = \mathbf{B}_{i-1}^{-1} \mathbf{u}_i, \tag{3.93}$$

$$\beta_i = \frac{\alpha_i}{1 + \alpha_i \mathbf{u}_i^{\mathrm{T}} \mathbf{B}_{i-1}^{-1} \mathbf{u}_i}. \tag{3.94}$$

Using recursion and equations (3.92)–(3.94), the inverse matrices $\mathbf{B}_i^{-1}$ can be calculated as

$$\mathbf{B}_i^{-1} = \mathbf{B}_0^{-1} - \sum_{j=1}^{i} \beta_j \mathbf{p}_j \mathbf{p}_j^{\mathrm{T}}, \tag{3.95}$$

where

$$\mathbf{p}_i = \mathbf{B}_0^{-1} \mathbf{u}_i - \sum_{j=1}^{i-1} \beta_j \mathbf{p}_j \mathbf{p}_j^{\mathrm{T}} \mathbf{u}_i, \tag{3.96}$$

$$\beta_i = \frac{\alpha_i}{1 + \alpha_i \mathbf{u}_i^{\mathrm{T}} \mathbf{p}_i}. \tag{3.97}$$

Therefore, in order to find the matrix

$$\left(\mathbf{B}_0 + \sum_{j=1}^{k} \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}} \right)^{-1}, \tag{3.98}$$

the assembly of the auxiliary matrices $\mathbf{B}_i^{-1}$ (given by equation (3.92)) is not necessary during the computation, and only the auxiliary coefficients β_i and vectors $\mathbf{p}_i$ are calculated and stored. In Algorithm 7, the complete algorithm for solving the system

$$\left(\mathbf{B}_0 + \sum_{i=1}^{k} \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}} \right) \mathbf{x} = \mathbf{f} \tag{3.99}$$

is summarised. Note that in the application that we discuss below, $\mathbf{B}_0$ will be a diagonal matrix with positive elements on the diagonal. Therefore, its inverse is diagonal and the evaluation of $\mathbf{B}_0^{-1}\mathbf{f}$ requires just n multiplications.

Application of the Sherman–Morrison updates in VTN-phase stability testing

In this section, we use Algorithm 7 to solve systems of linear algebraic equations arising from the Newton–Raphson linearization of the VTN-phase stability testing problem. As discussed

Algorithm 7: Solution of $\left(\mathbf{B}_0 + \sum_{i=1}^{k} \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}\right) \mathbf{x} = \mathbf{f}$.

```
1 for $i = 1, \dots, k$ do
2     evaluate vector $\mathbf{p}_i$ using
          $$\mathbf{p}_i = \mathbf{B}_0^{-1}\mathbf{u}_i - \sum_{j=1}^{i-1} \beta_j \mathbf{p}_j (\mathbf{p}_j^{\mathrm{T}} \mathbf{u}_i).$$
3     evaluate coefficient $\beta_i$ using
          $$\beta_i = \frac{\alpha_i}{1 + \alpha_i \mathbf{u}_i^{\mathrm{T}} \mathbf{p}_i}.$$
4 end
  Output:
          $$\mathbf{x} = \mathbf{B}_0^{-1}\mathbf{f} - \sum_{j=1}^{k} \beta_j \mathbf{p}_j (\mathbf{p}_j^{\mathrm{T}} \mathbf{f}).$$
```

in Section 3.2.3, in order to find the new approximation of the solution $\mathbf{c}^{(j+1)}$ in each Newton–Raphson iteration, a linear system with the matrix $\mathbf{H}$ has to be solved. In this section, we will show how to transform the Hessian matrix $\mathbf{H}$ to the desired form

$$\mathbf{H} = \mathbf{B}_0 + \sum_{i=1}^{k} \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}, \tag{3.100}$$

for which Algorithm 7 can be used. Moreover, at the end of this section, the complete algorithm for solving the VTN-phase stability testing problem will be presented. In the VTN-phase stability testing, the Hessian matrix is

$$[\mathbf{H}]_{i,j}(\mathbf{c}) = \frac{\partial^2 a}{\partial c_i \partial c_j}(\mathbf{c}) = \frac{\partial \mu_i}{\partial c_j}(\mathbf{c}). \tag{3.101}$$

Denoting $\sigma = \frac{1}{1-\mathbf{b}\cdot\mathbf{c}}$ and using equation (2.197), the Hessian matrix can be written in a matrix form as

$$\begin{aligned}\mathbf{H}(\mathbf{c}) = & RT\mathbf{D}^{-1}(\mathbf{c}) + RT\sigma(\mathbf{b}\mathbf{e}^{\mathrm{T}} + \mathbf{e}\mathbf{b}^{\mathrm{T}}) + RTc\sigma^2\mathbf{b}\mathbf{b}^{\mathrm{T}} \\ & - 2\psi_2(\mathbf{b}\cdot\mathbf{c})\mathbf{A} - 2\psi_2'(\mathbf{b}\cdot\mathbf{c})\left((\mathbf{A}\mathbf{c})\mathbf{b}^{\mathrm{T}} + \mathbf{b}(\mathbf{A}\mathbf{c})^{\mathrm{T}}\right) \\ & - \psi_1(\mathbf{c})\psi_2''(\mathbf{b}\cdot\mathbf{c})\mathbf{b}\mathbf{b}^{\mathrm{T}},\end{aligned} \tag{3.102}$$

where $\mathbf{e} = (1, \dots, 1)^{\mathrm{T}} \in \mathbb{R}^n$, and $\mathbf{D}^{-1}(\mathbf{c})$ is the inverse matrix to a diagonal matrix with components of the vector $\mathbf{c}$ on the diagonal, i.e.,

$$\mathbf{D}^{-1}(\mathbf{c}) = \begin{pmatrix} \frac{1}{c_1} & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{c_2} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & \frac{1}{c_{n-1}} & 0 \\ 0 & \dots & 0 & 0 & \frac{1}{c_n} \end{pmatrix}. \tag{3.103}$$

Next, let us denote

$$\mathbf{H}^{(1)}(\mathbf{c}) = RT\sigma(\mathbf{b}\mathbf{e}^{\mathrm{T}} + \mathbf{e}\mathbf{b}^{\mathrm{T}}), \tag{3.104}$$

$$\mathbf{H}^{(2)}(\mathbf{c}) = \left(RTc\sigma^2 - \psi_1(\mathbf{c})\psi_2''(\mathbf{b}\cdot\mathbf{c})\right)\mathbf{b}\mathbf{b}^{\mathrm{T}}, \tag{3.105}$$

$$\mathbf{H}^{(3)}(\mathbf{c}) = -2\psi_2'(\mathbf{b}\cdot\mathbf{c})\left((\mathbf{A}\mathbf{c})\mathbf{b}^{\mathrm{T}} + \mathbf{b}(\mathbf{A}\mathbf{c})^{\mathrm{T}}\right). \tag{3.106}$$

Using equations (3.102)–(3.106), the Hessian matrix can be written in a compact form

$$\mathbf{H}(\mathbf{c}) = RT\mathbf{D}^{-1}(\mathbf{c}) + \sum_{i=1}^{3}\mathbf{H}^{(i)}(\mathbf{c}) - 2\psi_2(\mathbf{b}\cdot\mathbf{c})\mathbf{A}. \tag{3.107}$$

Now, we can start defining the matrix $\mathbf{B}_0$, coefficients α_i, and vectors $\mathbf{u}_i$ in equation (3.100). First, the initial matrix $\mathbf{B}_0$ is chosen as

$$\mathbf{B}_0 = RT\mathbf{D}^{-1}(\mathbf{c}). \tag{3.108}$$

In this case, the matrix $\mathbf{B}_0$ is a diagonal matrix. Therefore, its inverse

$$\mathbf{B}_0^{-1} = \frac{1}{RT}\mathbf{D}(\mathbf{c}) \tag{3.109}$$

is readily available. Then, the decomposition of matrices $\mathbf{H}^{(i)}$ will be given. The matrix $\mathbf{H}^{(2)}$ is already in the desired form

$$\mathbf{H}^{(2)} = \alpha\mathbf{u}\mathbf{u}^{\mathrm{T}} \tag{3.110}$$

with

$$\alpha = RTc\sigma^2 - \psi_1(\mathbf{c})\psi_2''(\mathbf{b}\cdot\mathbf{c}), \tag{3.111}$$

$$\mathbf{u} = \mathbf{b}. \tag{3.112}$$

The matrices $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(3)}$ have the following form

$$\beta\left(\mathbf{r}\mathbf{s}^{\mathrm{T}} + \mathbf{s}\mathbf{r}^{\mathrm{T}}\right), \tag{3.113}$$

where $\mathbf{r}, \mathbf{s} \in \mathbb{R}^n$. Therefore, in order to transform the rank-two matrices $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(3)}$ into the sum of two symmetric rank-one updates, their eigenvalues and eigenvectors have to be found. The following theorem gives us the answer.

Theorem 3.10. *Let* $\mathbf{B} = \beta\left(\mathbf{r}\mathbf{s}^T + \mathbf{s}\mathbf{r}^T\right) \in \mathbb{R}^{m,m}$ *for an arbitrary* $\beta \in \mathbb{R}$, $\beta \neq 0$, *and* $\mathbf{r}, \mathbf{s} \in \mathbb{R}^m$, $\mathbf{r} \neq \mathbf{s}$. *Then, the eigenvalues* λ_i *and eigenvectors* $\tilde{\mathbf{v}}_i$ *are*

$$\lambda_1 = \beta\left(\mathbf{s}\cdot\mathbf{r} + \|\mathbf{r}\|\|\mathbf{s}\|\right), \quad \tilde{\mathbf{v}}_1 = \|\mathbf{s}\|\mathbf{r} + \|\mathbf{r}\|\mathbf{s}, \tag{3.114}$$

$$\lambda_2 = \beta\left(\mathbf{s}\cdot\mathbf{r} - \|\mathbf{r}\|\|\mathbf{s}\|\right), \quad \tilde{\mathbf{v}}_2 = \|\mathbf{s}\|\mathbf{r} - \|\mathbf{r}\|\mathbf{s}, \tag{3.115}$$

where $\|\cdot\|$ *is the Euclidean norm.*

Proof. From the definition, we show $\mathbf{B}\tilde{\mathbf{v}}_i = \lambda_i\tilde{\mathbf{v}}_i$ for $i = 1, 2$. For $i = 1$ we have

$$\mathbf{B}\tilde{\mathbf{v}}_1 = \beta\left(\mathbf{r}\mathbf{s}^{\mathrm{T}} + \mathbf{s}\mathbf{r}^{\mathrm{T}}\right)\left(\|\mathbf{s}\|\mathbf{r} + \|\mathbf{r}\|\mathbf{s}\right) = 2\beta\|\mathbf{r}\|\|\mathbf{s}\|\left(\|\mathbf{r}\|\mathbf{s} + \|\mathbf{s}\|\mathbf{r}\right), \tag{3.116}$$

$$\lambda_1\tilde{\mathbf{v}}_1 = \beta\left(\mathbf{s}\cdot\mathbf{r} + \|\mathbf{r}\|\|\mathbf{s}\|\right)\left(\|\mathbf{s}\|\mathbf{r} + \|\mathbf{r}\|\mathbf{s}\right) = 2\beta\|\mathbf{r}\|\|\mathbf{s}\|\left(\|\mathbf{r}\|\mathbf{s} + \|\mathbf{s}\|\mathbf{r}\right). \tag{3.117}$$

matrix	$\mathbf{r}$	$\mathbf{s}$	i	α_i	$\tilde{\mathbf{u}}_i$	$\mathbf{u}_i$
$\mathbf{H}^{(1)}$	$\mathbf{b}$	$\mathbf{e}$	1	$RT\sigma\left(\mathbf{e}\cdot\mathbf{b}+\|\mathbf{e}\|\|\mathbf{b}\|\right)$	$\|\mathbf{e}\|\mathbf{b}+\|\mathbf{b}\|\mathbf{e}$	$\tilde{\mathbf{u}}_1/\|\tilde{\mathbf{u}}_1\|$
			2	$RT\sigma\left(\mathbf{e}\cdot\mathbf{b}-\|\mathbf{e}\|\|\mathbf{b}\|\right)$	$\|\mathbf{e}\|\mathbf{b}-\|\mathbf{b}\|\mathbf{e}$	$\tilde{\mathbf{u}}_2/\|\tilde{\mathbf{u}}_2\|$
$\mathbf{H}^{(2)}$	-	-	3	$RTc\sigma^2-f_1(\mathbf{c})f''(\mathbf{b}\cdot\mathbf{c})$	-	$\mathbf{b}$
$\mathbf{H}^{(3)}$	$\mathbf{Ac}$	$\mathbf{b}$	4	$-2f_2'(\mathbf{b}\cdot\mathbf{c})\left(\mathbf{b}\cdot\mathbf{Ac}+\|\mathbf{Ac}\|\|\mathbf{b}\|\right)$	$\|\mathbf{b}\|\mathbf{Ac}+\|\mathbf{Ac}\|\mathbf{b}$	$\tilde{\mathbf{u}}_4/\|\tilde{\mathbf{u}}_4\|$
			5	$-2f_2'(\mathbf{b}\cdot\mathbf{c})\left(\mathbf{b}\cdot\mathbf{Ac}-\|\mathbf{Ac}\|\|\mathbf{b}\|\right)$	$\|\mathbf{b}\|\mathbf{Ac}-\|\mathbf{Ac}\|\mathbf{b}$	$\tilde{\mathbf{u}}_5/\|\tilde{\mathbf{u}}_5\|$

Table 3.1: Form of the individual rank-one updates for matrices $\mathbf{H}^{(1)}$–$\mathbf{H}^{(3)}$.

For $i = 2$ we have

$$\mathbf{B}\tilde{\mathbf{v}}_2 = \beta\left(\mathbf{r}\mathbf{s}^{\mathrm{T}}+\mathbf{s}\mathbf{r}^{\mathrm{T}}\right)\left(\|\mathbf{s}\|\mathbf{r}-\|\mathbf{r}\|\mathbf{s}\right) = 2\beta\|\mathbf{r}\|\|\mathbf{s}\|\left(\|\mathbf{r}\|\mathbf{s}-\|\mathbf{s}\|\mathbf{r}\right), \tag{3.118}$$

$$\lambda_2\tilde{\mathbf{v}}_2 = \beta\left(\mathbf{s}\cdot\mathbf{r}-\|\mathbf{r}\|\|\mathbf{s}\|\right)\left(\|\mathbf{s}\|\mathbf{r}-\|\mathbf{r}\|\mathbf{s}\right) = 2\beta\|\mathbf{r}\|\|\mathbf{s}\|\left(\|\mathbf{r}\|\mathbf{s}-\|\mathbf{s}\|\mathbf{r}\right). \tag{3.119}$$

□

Therefore, the desired transformation is

$$\beta\left(\mathbf{r}\mathbf{s}^{\mathrm{T}}+\mathbf{s}\mathbf{r}^{\mathrm{T}}\right) = \sum_{i=1}^{2}\lambda_i\mathbf{v}_i\mathbf{v}_i^{\mathrm{T}}, \tag{3.120}$$

where for $i = 1, 2$ the vector $\mathbf{v}_i$ is normalised vector $\tilde{\mathbf{v}}_i$

$$\mathbf{v}_i = \frac{\tilde{\mathbf{v}}_i}{\|\tilde{\mathbf{v}}_i\|}. \tag{3.121}$$

Individual forms of the updates are presented in Table 3.1. In total, five rank-one updates have to be performed to transform $\mathbf{B}_0$ using the matrices $\mathbf{H}^{(1)}$, $\mathbf{H}^{(2)}$, and $\mathbf{H}^{(3)}$. Lastly, the term

$$-2\psi_2(\mathbf{b}\cdot\mathbf{c})\mathbf{A}, \tag{3.122}$$

has to be transformed into the desired form. The elements of the matrix $\mathbf{A}$ have the following form (see equation (2.101))

$$[\mathbf{A}]_{i,j} = (1-\delta_{i-j})\sqrt{a_i}\sqrt{a_j}. \tag{3.123}$$

Let us define matrix $\mathbf{M} \in \mathbb{R}^{n,n}$ with elements $[\mathbf{M}]_{i,j} = 1 - \delta_{i-j}$. Then, the matrix $\mathbf{A}$ can be written as

$$\mathbf{A} = \mathbf{D}(\sqrt{\mathbf{a}})\mathbf{M}\mathbf{D}(\sqrt{\mathbf{a}}), \tag{3.124}$$

where $\mathbf{D}(\sqrt{\mathbf{a}})$ is a diagonal matrix with components of the vector $\sqrt{\mathbf{a}} = \left(\sqrt{a_1},\ldots,\sqrt{a_n}\right)^{\mathrm{T}}$ on the diagonal. We will discuss two cases. First, all binary interaction coefficients δ_{i-j} are zero. Second, there are non-zero coefficients δ_{i-j}.

If all binary interaction coefficients are zero, the matrix $\mathbf{M}$ is a constant matrix with the elements $[\mathbf{M}]_{i,j} = 1$. Then, the matrix $\mathbf{M}$ has only one non-zero eigenvalue $\lambda = n$ with the corresponding orthonormal eigenvector

$$\mathbf{q} = \frac{1}{\sqrt{n}}\left(1,\ldots,1\right)^{\mathrm{T}} \in \mathbb{R}^n. \tag{3.125}$$

Second, we will discuss the situation with non-zero coefficients δ_{i-j}. In this case, the matrix $\mathbf{M}$ is a symmetric matrix with real eigenvalues λ_i. The eigenvalues and the corresponding eigenvectors are found using the TNT numerical library, see Pozo (1997). Let m be the number of non-zero eigenvalues. Then,

$$\mathbf{M} = \mathbf{Q}\Lambda\mathbf{Q}^{\mathrm{T}} = \sum_{i=1}^{m} \lambda_i \mathbf{q}_i \mathbf{q}_i^{\mathrm{T}}, \tag{3.126}$$

where Λ is a diagonal matrix with vector $(\lambda_1 \ldots, \lambda_m, 0, \ldots, 0)^{\mathrm{T}} \in \mathbb{R}^n$ on the diagonal, and $\mathbf{q}_1, \ldots, \mathbf{q}_m$ are the corresponding orthonormal eigenvectors of the matrix $\mathbf{M}$.

To conclude, in both cases, the matrix $-\psi_2(\mathbf{b}\cdot\mathbf{c})\mathbf{A}$ can be written in a form

$$\begin{aligned} -\psi_2(\mathbf{b}\cdot\mathbf{c})\mathbf{A} &= -\psi_2(\mathbf{b}\cdot\mathbf{c})\mathbf{D}\left(\sqrt{\mathbf{a}}\right)\mathbf{Q}\Lambda\mathbf{Q}^{\mathrm{T}}\mathbf{D}\left(\sqrt{\mathbf{a}}\right) \\ &= -\psi_2(\mathbf{b}\cdot\mathbf{c})\sum_{i=1}^{m} \lambda_i \mathbf{u}_{5+i}\mathbf{u}_{5+i}^{\mathrm{T}} \\ &= \sum_{i=1}^{m} \alpha_{5+i}\mathbf{u}_{5+i}\mathbf{u}_{5+i}^{\mathrm{T}}, \end{aligned} \tag{3.127}$$

where for $i = 1, \ldots, m$ the coefficients α_{5+i} and $\mathbf{u}_{5+i}$ are defined as

$$\alpha_{5+i} = -\psi_2(\mathbf{b}\cdot\mathbf{c})\lambda_i, \tag{3.128}$$

$$\mathbf{u}_{5+i} = \left(\sqrt{a_1}\,[\mathbf{q}_i]_1, \ldots, \sqrt{a_n}\,[\mathbf{q}_i]_n\right)^{\mathrm{T}}. \tag{3.129}$$

Therefore, to transform the last term of the Hessian matrix, we need to perform m updates, where m is the number of non-zero eigenvalues of the matrix $\mathbf{M}$. If the binary interaction coefficients are zero, then $m = 1$. If there are non-zero binary interaction coefficients, frequently, there is only a small number of components that have non-zero interaction coefficients with all others. In this situation, m is generally more than 1, but still substantially less than n. In summary, we are able to write the Hessian matrix $\mathbf{H}$ in the form

$$\mathbf{H} = \mathbf{B}_0 + \sum_{i=1}^{5+m} \alpha_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}, \tag{3.130}$$

where α_i and $\mathbf{u}_i$ for $i \in \widehat{5}$ are defined in Table 3.1. For $i > 5$, equations (3.128) and (3.129) are used to calculate α_i and $\mathbf{u}_i$. Therefore, to solve the system arising from the linearization of the VTN-phase stability testing, Algorithm 7 can be used. In this algorithm, the Hessian matrix is not assembled. Therefore, the computational time can be reduced if the size of the Hessian matrix is large enough.

Let us discuss the computational complexity of our algorithm. In the classical solution procedure, $O(n^2)$ operations are needed to form the Hessian matrix, $O(n^3)$ operations are needed for performing the (possibly modified) Cholesky factorization, and another $O(n^2)$ operations are needed for obtaining the solution using the forward and backward substitutions (solving the systems with triangular matrices). The total computational complexity of one Newton–Raphson iteration is thus $O(n^3)$. Now, we discuss the complexity of the presented method. To assemble $\mathbf{p}_i$ and β_i, one has to perform $3in + 1$ multiplications and $in + 2$ additions. Therefore, to assemble every $\mathbf{p}_i$ and β_i one has to perform

$$\sum_{i=1}^{k} \left[3in + 1 + in + 2\right] = O(k^2 n) \tag{3.131}$$

operations, where $k = 5 + m$ is the number of the rank-one updates. After the auxiliary vectors $\mathbf{p}_i$ and coefficients β_i are assembled, the last step of Algorithm 7 is to evaluate the resulting vector $\mathbf{x}$. This task requires

$$n + 1 + k(3n) = O(kn) \tag{3.132}$$

operations. To conclude, the complexity of Algorithm 7 is $O(k^2 n)$. Finally, we will discuss the complexity of the assembly of the individual updates. To create vectors $\mathbf{u}_i$ and coefficients α_i, one has to perform operations such as scalar product or evaluation of the Euclidean norm, which require $O(n)$ operations. In the algorithm, there are only two more complex operations. The first operation is finding the eigenvalues of the matrix $\mathbf{M}$. As this is achieved using iterative methods, it is not possible to determine the CPU cost a priori. Finding eigenvalues is generally a costly task; however, the matrix $\mathbf{M}$ remains constant in all iterations. Therefore, its eigenvalues and eigenvectors can be found once at the beginning, stored and reused in subsequent calculations whenever needed. Second is the product $\mathbf{Ac}$ which has to be evaluated in the chemical potential (equation (2.197)) and in the matrix $\mathbf{H}^{(3)}$ (equation (3.106)). The standard matrix-vector product algorithm requires $O(n^2)$ operations.

Overall, the complexity of assembling all $\mathbf{u}_i$ and α_i for $i = 1, \ldots, k$ is $O(n^2)$. To conclude, the presented method for solving the system of linear equations arising from the Newton–Raphson linearization has complexity $O(n^2)$, where n is the number of components in the mixture.

Complete algorithm

In Algorithm 8, we summarize the numerical method for the VTN-phase stability testing using the Sherman–Morrison formula. The core of the algorithm is identical to the Algorithm 6. The only difference is the computation of the increment $\mathbf{\Delta c}^{(k)}$.

3.2.5 Global deterministic method based on the Branch and Bound approach

In the previous Sections 3.2.1 and 3.2.3, we presented local minimization methods with multiple initial approximations. The idea was that one of the initial approximations should converge toward the global minimum. However, this cannot be guaranteed. Therefore, here, we propose a global optimization method that has to converge toward the global minimum. The use of the global method is not common in the phase stability testing; however, a few papers exist. In the VTN-specification, Nichita et al. (2009, 2002) used a tunneling method. Another global approach using a Lagrangian function and solving the dual problem was presented by Pereira et al. (2010). Souza et al. (2006) investigated a global optimization method using the interval analysis. In the PTN-specification, Stadtherr et al. (2007) presented a method based on the interval methods. Solving the PTN-specification phase stability using a topographical global optimization method was investigated by Henderson et al. (2015). Zhang et al. (2011) presented a comparison of various approaches for the global optimization methods.

In this section, we present a global optimization algorithm for the VTN-phase stability testing based on the Branch and Bound (BB) strategy published by Androulakis et al. (1995). This strategy provides a general framework on how to solve a non-convex optimization problem on a convex domain. Our numerical algorithm has been published in Smejkal and Mikyška (2020). However, recently, we have found an error in our convex-concave splitting strategy. Here, we present a fixed version. The Branch and Bound strategy can be described as

0. Set $\mathcal{E} = \mathcal{D}$ and $\mathbf{z}$ as the barycenter of $\mathcal{D}$.

Algorithm 8: Modified Newton–Raphson method for the VTN-phase stability testing using the Sherman–Morrison iterations.

```
   Input  : $T^*$, $\mathbf{c}^* \in \mathcal{D}$
   Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
 1 compute $s$ initial approximations using the strategy from Section 3.2.2
 2 find the eigenvalues and eigenvectors of the matrix $\mathbf{M}$ defined by equation (3.124)
 3 for $j \in \widehat{s}$ do
 4     set $\mathbf{c}^{(0)}$ as the $j$-th initial approximation
 5     set iteration counter $k = 0$
 6     while $k <$ maximum number of iterations do
 7         calculate the matrix $\mathbf{B}_0^{-1}$ using equation (3.109)
 8         for $i \in \widehat{5+m}$ calculate the coefficients $\alpha_i$ and vectors $\mathbf{u}_i$ using Table 3.1 and
             equations (3.128)–(3.129)
 9         evaluate the increment of the solution $\Delta\mathbf{c}^{(j)} \in \mathbb{R}^n$ by solving the system arising
             from the Newton–Raphson linearization using Algorithm 7
10         determine $\lambda^{(k)} > 0$ using the line-search strategy
11         update the solution using
               $$\mathbf{c}^{(k+1)} = \mathbf{c}^{(k)} + \lambda^{(k)}\mathbf{\Delta c}^{(k)}.$$
             if $\left\|\mathbf{\Delta c}^{(k)}\right\|_{pn} < \varepsilon$ then
12               break
13           end
14         set $k := k + 1$
15     end
16     if TPD $\left(\mathbf{c}^{(k+1)};\mathbf{x}^*\right) < 0$ then
17         return false
18     end
19 end
20 return true
```

1. Find an upper bound `UBD` of the sought minimum by solving the minimization problem locally. The initial approximation is set to $\mathbf{z}$.

2. Branch the computation set $\mathcal{E}$ into two, $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$.

3. Find and store lower bounds $\texttt{LBD}_1$ and $\texttt{LBD}_2$ by solving the underestimate convex optimization problems on $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively.

4. If $\texttt{LBD}_1 > \texttt{UBD}$ or $\texttt{LBD}_2 > \texttt{UBD}$, discard the corresponding set $\mathcal{E}_1$, $\mathcal{E}_2$, respectively, since the global minimum cannot be in that set.

5. Choose a new current computation set $\mathcal{E}$ and a new initial approximation $\mathbf{z}$.

6. Repeat 1–5 until the termination condition is met.

In Figure 3.4, we present the scheme of this algorithm in case $n = 2$, i.e., the feasible domain $\mathcal{D}$ is a triangle in 2D. In the next subsections, we describe each step of the Branch and Bound algorithm.

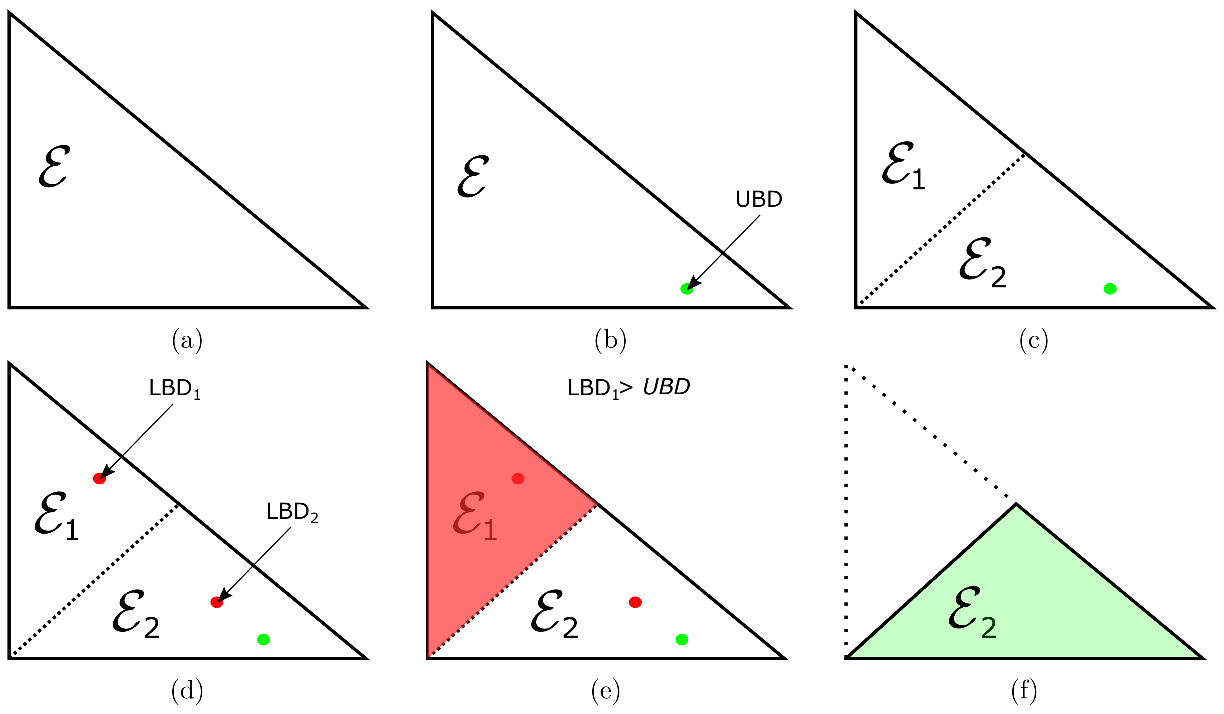


Figure 3.4: Branch and Bound strategy. 3.4a Let $\mathcal{E}$ be the current computation set. 3.4b Find an upper bound of the minimum by solving the minimization problem locally. 3.4c Branch the computation set $\mathcal{E}$ into two, $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$. 3.4d Find a lower bound by solving the underestimate convex optimization problem on $\mathcal{E}_1$ and $\mathcal{E}_2$. 3.4e If any lower bound is greater than the upper bound, discard the corresponding set. 3.4f Choose a new current computation set $\mathcal{E}$.

Local minimization of the TPD function

For the local minimization of the TPD function, we are using the Newton–Raphson method, which was described in Section 3.2.3. In that Section, we used this method stand-alone with multiple initial approximations $\mathbf{c}^{(0)}$ to reach the global minimum. In this variation, we are searching locally; therefore, the strategies for choosing a good initial approximation are not needed. In the first iteration of the Branch and Bound, the initial approximation is the barycenter of the feasible simplex $\mathcal{D}$. In the next iterations, the position of the lower bound is set as the new initial approximation.

Branching of computation set $\mathcal{E}$

As the feasible domain $\mathcal{D}$ is an $(n+1)$-simplex, our branching algorithm is based on dividing the simplex through its longest side. Let $\mathcal{E} = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n+1)}\right]_\kappa$. Find indices $k, l \in \widehat{n+1}, k < l$ satisfying

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(l)}\right\| \geqslant \left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|, \quad \forall i, j \in \widehat{n+1}, i \neq j. \tag{3.133}$$

Then define a new vertex $\mathbf{x}^{(\mathrm{middle})}$ by

$$\mathbf{x}^{(\mathrm{middle})} = \frac{\mathbf{x}^{(k)} + \mathbf{x}^{(l)}}{2}. \tag{3.134}$$

New simplices $\mathcal{E}_1$, $\mathcal{E}_2$ are then defined by

$$\mathcal{E}_1 = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(l-1)}, \mathbf{x}^{(\mathrm{middle})}, \mathbf{x}^{(l+1)}, \ldots, \mathbf{x}^{(n+1)}\right]_\kappa, \tag{3.135}$$

$$\mathcal{E}_2 = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k-1)}, \mathbf{x}^{(\mathrm{middle})}, \mathbf{x}^{(k+1)}, \ldots, \mathbf{x}^{(n+1)}\right]_\kappa. \tag{3.136}$$

Convex optimization of the $\mathrm{TPD}^{(\mathrm{under})}$ function

In the third step of the Branch and Bound strategy, the lower bound of the global minimum is found. This lower bound is found using the minimization of function $\mathrm{TPD}^{(\mathrm{under})}$ on a simplex $\mathcal{E} \subset \mathcal{D}$, which has the following properties

1. $\mathrm{TPD}^{(\mathrm{under})}$ is a convex function on $\mathcal{E}$.
2. $\mathrm{TPD}^{(\mathrm{under})}(\mathbf{c};\mathbf{x}^*) \leqslant \mathrm{TPD}(\mathbf{c};\mathbf{x}^*)$, $\forall \mathbf{c} \in \mathcal{E}$.

Now, we present a construction of the $\mathrm{TPD}^{(\mathrm{under})}$ function based on rewriting the Helmholtz free energy density a as the difference of two convex functions

$$a(\mathbf{c}) = f_1(\mathbf{c}) - f_2(\mathbf{c}), \tag{3.137}$$

where f_1, f_2 are convex functions. The form in the previous equation is known as the *difference of two convex functions* (DC split) or *convex-concave split.* This split is not unique and even does not have to exist for a given function. First, two different splits denoted by DC1 and DC2 will be presented. The first one, denoted DC1, has been developed by Kou and Sun (2018). The second one, denoted DC2, is our own and has been presented in Smejkal and Mikyška (2020). After the presentation of these two splitting strategies, the construction of the $\mathrm{TPD}^{(\mathrm{under})}$ will be given.

DC1: split from Kou and Sun (2018)
Here, we present a convex-concave split from Kou and Sun (2018). Let us denote

$$a^{(\mathrm{ideal})}(\mathbf{c}) = RT\sum_{i=1}^{n} c_i \ln\frac{c_i}{c_0}, \tag{3.138}$$

$$a^{(\mathrm{repulsion})}(\mathbf{c}) = -RTc\ln\left(1 - \sum_{i=1}^{n} b_i c_i\right), \tag{3.139}$$

$$\chi_1(\mathbf{c}) = \frac{\sum\limits_{i,j=1}^{n} c_i c_j a_{ij}}{\sum\limits_{i=1}^{n} b_i c_i}, \tag{3.140}$$

$$\breve{\chi}_2(\mathbf{c}) = \ln\left(\frac{1 + (1-\sqrt{2})\sum\limits_{i=1}^{n} b_i c_i}{1 + (1+\sqrt{2})\sum\limits_{i=1}^{n} b_i c_i}\right). \tag{3.141}$$

Then, the Helmholtz free energy (2.189) reads as

$$a(\mathbf{c}) = a^{(\mathrm{ideal})}(\mathbf{c}) + a^{(\mathrm{repulsion})}(\mathbf{c}) + \frac{1}{2\sqrt{2}}\chi_1(\mathbf{c})\,\breve{\chi}_2(\mathbf{c}). \tag{3.142}$$

The splitting strategy from Kou and Sun (2018) is based on an additional term

$$a^{(\mathrm{SR})}(\mathbf{c}) = -RT\sum_{i=1}^{n} c_i \ln(1 - b_i c_i). \tag{3.143}$$

Adding and consequently subtracting this term in equation (3.142) results in convex-concave split

$$a(\mathbf{c}) = f_1(\mathbf{c}) - f_2(\mathbf{c}), \tag{3.144}$$

with

$$f_1(\mathbf{c}) = a^{(\text{ideal})}(\mathbf{c}) + a^{(\text{repulsion})}(\mathbf{c}) + \alpha^{(KS)}\left(a^{(\text{ideal})}(\mathbf{c}) + a^{(\text{SR})}(\mathbf{c})\right), \tag{3.145}$$

$$f_2(\mathbf{c}) = -\frac{1}{2\sqrt{2}}\chi_1(\mathbf{c})\check{\chi}_2(\mathbf{c}) + \alpha^{(\text{KS})}\left(a^{(\text{ideal})}(\mathbf{c}) + a^{(\text{SR})}(\mathbf{c})\right), \tag{3.146}$$

where $\alpha^{(\text{KS})}$ is a splitting parameter. Kou and Sun (2018) report that the split is valid for large $\alpha^{(KS)}$. The suggested value is $\alpha^{(\text{KS})} = 1$.

DC2: split from Smejkal and Mikyška (2020)
Next, we present a different convex-concave split presented in Smejkal and Mikyška (2020). We start our derivation from equation (3.142). Here, the Helmholtz free energy has three terms. The sum of the first two terms is convex; therefore, these terms will be included in the convex part in the convex-concave splitting. Only the third term in equation (3.142) is not convex. The goal now is to split this term into the convex and the concave part. Let us denote

$$\chi_2(\mathbf{c}) = \check{\chi}_2(\mathbf{c}) + C, \tag{3.147}$$

where C is a positive constant large enough so the function χ_2 is positive for all $\mathbf{c}$. Choosing $C = \ln\left((2+\sqrt{2})/(2-\sqrt{2})\right)$ is sufficient. Then,

$$a(\mathbf{c}) = a^{(\text{ideal})}(\mathbf{c}) + a^{(\text{repulsion})}(\mathbf{c}) - C\frac{1}{2\sqrt{2}}\chi_1(\mathbf{c}) + \frac{1}{2\sqrt{2}}\chi_1(\mathbf{c})\chi_2(\mathbf{c}). \tag{3.148}$$

It is tempting to define the convex-concave split with

$$f_1(\mathbf{c}) = a^{(\text{ideal})}(\mathbf{c}) + a^{(\text{repulsion})}(\mathbf{c}) + \frac{1}{2\sqrt{2}}\chi_1(\mathbf{c})\chi_2(\mathbf{c}), \tag{3.149}$$

$$f_2(\mathbf{c}) = C\frac{1}{2\sqrt{2}}\chi_1(\mathbf{c}). \tag{3.150}$$

However, it can be proven that the product $\chi_1(\mathbf{c})\chi_2(\mathbf{c})$ is not convex. The necessary and sufficient conditions have been presented by Marchi (2010). Consequently, we can not prove that the f_1 function in (3.149) is convex. Therefore, a more complex strategy has to be used. The product $\chi_1(\mathbf{c})\chi_2(\mathbf{c})$ can be written as

$$\chi_1(\mathbf{c})\chi_2(\mathbf{c}) = \frac{1}{2}\left(\chi_1(\mathbf{c}) + \chi_2(\mathbf{c})\right)^2 - \frac{1}{2}\left(\chi_1^2(\mathbf{c}) + \chi_2(\mathbf{c})^2\right). \tag{3.151}$$

Since both χ_1 and χ_2 are positive and convex functions, it can be proven that the split in equation (3.151) is a convex-concave split (see Bačák and Borwein (2011); Veselý and Zajíček (2009)), i.e., the two terms are convex functions (without the minus sign). As χ_1 and χ_2 have different physical units, the split from equation (3.151) is modified by a scaling unit corrector parameter α

$$\chi_1(\mathbf{c})\chi_2(\mathbf{c}) = \frac{\alpha}{2}\left(\frac{\chi_1(\mathbf{c})}{\alpha} + \chi_2(\mathbf{c})\right)^2 - \frac{1}{2\alpha}\left(\chi_1^2(\mathbf{c}) + \alpha^2\chi_2^2(\mathbf{c})\right). \tag{3.152}$$

The convex-concave split is valid for an arbitrary positive α. Having the convex-concave split for the third term in equation (3.142), the Helmholtz free energy density function can be written as

$$a(\mathbf{c}) = f_1(\mathbf{c}) - f_2(\mathbf{c}), \tag{3.153}$$

where function f_1 and f_2 are defined as

$$f_1(\mathbf{c}) = a^{(\text{ideal})}(\mathbf{c}) + a^{(\text{repulsion})}(\mathbf{c}) + \frac{1}{2\sqrt{2}}\frac{\alpha}{2}\left(\frac{\chi_1(\mathbf{c})}{\alpha} + \chi_2(\mathbf{c})\right)^2, \tag{3.154}$$

$$f_2(\mathbf{c}) = \frac{1}{2\sqrt{2}}\left[C\chi_1(\mathbf{c}) + \frac{1}{2\alpha}\left(\chi_1^2(\mathbf{c}) + \alpha^2\chi_2^2(\mathbf{c})\right)\right]. \tag{3.155}$$

From the previous discussion, it is obvious that both functions f_1 and f_2 are convex; therefore, a convex-concave split for the Helmholtz free energy density function is made. This convex-concave split is valid for an arbitrary $\alpha > 0$. However, later, in the construction of the TPD$^{(\text{under})}$ function, we show that a favourable choice of α exists.

Construction of TPD$^{(\text{under})}$
Having the convex-concave of the Helmholtz free energy density $a(\mathbf{c})$ in hand, the TPD$^{(\text{under})}$ can be defined. Let $\mathcal{E} = [\mathbf{x}_1, \ldots, \mathbf{x}_{n+1}]_\kappa$ be a given simplex. For $\mathbf{c} = \sum_{i=1}^{n+1} \alpha_i \mathbf{x}^{(i)}$, where α_i are the coefficients of the convex combination, we get

$$a(\mathbf{c}) = f_1(\mathbf{c}) - f_2(\mathbf{c}) = f_1(\mathbf{c}) - f_2\left(\sum_{i=1}^{n+1}\alpha_i\mathbf{x}^{(i)}\right) \geqslant f_1(\mathbf{c}) - \sum_{i=1}^{n+1}\alpha_i f_2\left(\mathbf{x}^{(i)}\right), \tag{3.156}$$

where we used the Jensen inequality (see Jensen (1906)). Now, we can define function $a^{(\text{under})}$ on simplex $\mathcal{E}$ by

$$a^{(\text{under})}(\mathbf{c}) = f_1(\mathbf{c}) - \sum_{i=1}^{n+1}\alpha_i f_2\left(\mathbf{x}^{(i)}\right), \tag{3.157}$$

where $\mathbf{c} = \sum_{i=1}^{n+1}\alpha_i\mathbf{x}^{(i)}$. Having the function $a^{(\text{under})}(\mathbf{c})$ for which inequality

$$a^{(\text{under})}(\mathbf{c}) \leqslant a(\mathbf{c}) \tag{3.158}$$

holds for all $\mathbf{c} \in \mathcal{E}$, one can define the function TPD$^{(\text{under})}(\mathbf{c};\mathbf{x}^*)$ by

$$\text{TPD}^{(\text{under})}(\mathbf{c};\mathbf{x}^*) = a^{(\text{under})}(\mathbf{c}) - a(\mathbf{x}^*) + \sum_{i=1}^{n}\frac{\partial a}{\partial c_i}(\mathbf{x}^*)(c_i^* - c_i). \tag{3.159}$$

It is easily seen that function TPD$^{(\text{under})}$ is a convex function on a simplex $\mathcal{E}$ which underestimates the function TPD on simplex $\mathcal{E}$. In the Branch and Bound algorithm, we need a good underestimation of the objective function. Therefore, the parameter α should be chosen that the difference between the TPD and TPD$^{(\text{under})}$ is minimal. Since

$$\frac{\partial f_2}{\partial\alpha}(\mathbf{c};\alpha) = \frac{1}{4\sqrt{2}}\left(-\frac{\chi_1^2(\mathbf{c})}{\alpha^2} + \chi_2^2(\mathbf{c})\right), \tag{3.160}$$

$$\frac{\partial^2 f_2}{\partial\alpha^2}(\mathbf{c};\alpha) = \frac{1}{2\sqrt{2}}\frac{\chi_1^2(\mathbf{c})}{\alpha^3}, \tag{3.161}$$

one can see, that $\alpha = \frac{\chi_1}{\chi_2}$ is the local minimum of function f_2. Since χ_1 and χ_2 are both positive functions, α will always be a positive number. Therefore, on a simplex $\mathcal{E}$ (in general the feasible domain $\mathcal{D}$ for concentrations), we set coefficient α by relation

$$\alpha = \frac{\chi_1(c_1', \ldots, c_n')}{\chi_2(c_1', \ldots, c_n')}, \tag{3.162}$$

where $(c_1', \ldots, c_n')^{\mathrm{T}}$ is the barycenter of the simplex $\mathcal{E}$.

Minimization of $\mathrm{TPD}^{(\mathrm{under})}$ **function**

Now, the goal is to find a minimum of $\mathrm{TPD}^{(\mathrm{under})}$ on a given simplex $\mathcal{E}$. Since $\mathrm{TPD}^{(\mathrm{under})}$ is a convex function on $\mathcal{E}$, the minimum has to exist. This optimization problem is solved using the Barrier method. Our implementation is based on Boyd and Vandenberghe (2004). The Barrier method is built on the unconstrained minimization of the function

$$h(\mathbf{c}; t) = t\mathrm{TPD}^{(\mathrm{under})}(\mathbf{c};\mathbf{x}^*) + \Phi(\mathbf{c}), \tag{3.163}$$

where t is a parameter and Φ represents the given inequality constraints. If the constraints have the form

$$\mathbf{Bc} \preceq \mathbf{g} \overset{\mathrm{def}}{\Leftrightarrow} [\mathbf{Bc}]_i \leqslant [\mathbf{g}]_i, \quad i \in \widehat{n+1}, \tag{3.164}$$

where $\mathbf{B} \in \mathbb{R}^{n+1,n}$, $\mathbf{g} \in \mathbb{R}^{n+1}$, then the Φ function reads as

$$\Phi(\mathbf{c}) = -\sum_{i=1}^{n+1} \ln\Big([\mathbf{g}]_i - [\mathbf{B}]_{i,*}^{\mathrm{T}}\, \mathbf{c}\Big), \tag{3.165}$$

where $[\mathbf{B}]_{i,*}$ is the i-th row of matrix $\mathbf{B}$. In our case, the constraints represent the condition $\mathbf{c} \in \mathcal{E}$. Now, we give a strategy to compute the matrix $\mathbf{B}$ and vector $\mathbf{g}$. Let a given simplex $\mathcal{E}$ be defined as $\mathcal{E} = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n+1)}\right]_\kappa$. First, a matrix $\breve{\mathbf{B}} \in \mathbb{R}^{n,n}$ is defined using

$$\breve{\mathbf{B}} = \left(\mathbf{x}^{(2)} - \mathbf{x}^{(1)}, \mathbf{x}^{(3)} - \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n+1)} - \mathbf{x}^{(1)}\right), \tag{3.166}$$

where $\mathbf{x}^{(i)}$ is the i-th vertex of the simplex $\mathcal{E}$. Then, $\mathbf{c} \in \mathcal{E}$ if and only if

$$\mathbf{c} = \mathbf{x}^{(1)} + \breve{\mathbf{B}}\Theta, \tag{3.167}$$

where the parameter $\Theta \in \mathbb{R}^n$ satisfies conditions

$$\sum_{i=1}^{n} [\Theta]_i \leqslant 1, \tag{3.168}$$

$$[\Theta]_i \geqslant 0, \quad i \in \hat{n}. \tag{3.169}$$

Multiplying equation (3.167) with $\breve{\mathbf{B}}^{-1}$ results in

$$\breve{\mathbf{B}}^{-1}\mathbf{c} = \breve{\mathbf{B}}^{-1}\mathbf{x}^{(1)} + \Theta. \tag{3.170}$$

From the Θ constraints (3.168) and (3.169) we get inequalities

$$-\breve{\mathbf{B}}^{-1}\mathbf{c} \leqslant -\breve{\mathbf{B}}^{-1}\mathbf{x}^{(1)}, \tag{3.171}$$

$$\mathbf{1}^{\mathrm{T}}\breve{\mathbf{B}}^{-1}\mathbf{c} \leqslant 1 + \mathbf{1}^{\mathrm{T}}\breve{\mathbf{B}}^{-1}\mathbf{x}^{(1)}, \tag{3.172}$$

where $\mathbf{1} = (1, 1, \ldots, 1)^{\mathrm{T}} \in \mathbb{R}^n$. Therefore, we can define the matrix $\mathbf{B} \in \mathbb{R}^{n+1,n}$ and vector $\mathbf{g} \in \mathbb{R}^{n+1}$ by

$$\mathbf{B} = \begin{pmatrix} -\breve{\mathbf{B}}^{-1} \\ \mathbf{1}^{\mathrm{T}}\breve{\mathbf{B}}^{-1} \end{pmatrix}, \tag{3.173}$$

$$\mathbf{g} = \begin{pmatrix} -\breve{\mathbf{B}}^{-1}\mathbf{x}^{(1)} \\ 1 + \mathbf{1}^{\mathrm{T}}\breve{\mathbf{B}}^{-1}\mathbf{x}^{(1)} \end{pmatrix}. \tag{3.174}$$

The unconstrained minimization of function h is solved using the Newton–Raphson method with line-search. The gradient and Hessian of the function Φ can be written as

$$\boldsymbol{\nabla}\Phi(\mathbf{c}) = \mathbf{B}^{\mathrm{T}}\mathbf{d}, \tag{3.175}$$

$$\mathbf{H}^{\Phi}(\mathbf{c}) = \mathbf{B}^{\mathrm{T}}\left(\mathbf{D}(\mathbf{d})\right)^2\mathbf{B}, \tag{3.176}$$

where $\mathbf{d} = (d_1, \ldots, d_{n+1}) \in \mathbb{R}^{n+1}$ is defined by

$$[\mathbf{d}]_i = \frac{1}{[\mathbf{g}]_i - [\mathbf{B}]_{i,*}^{\mathrm{T}}\mathbf{c}}, \tag{3.177}$$

and the matrix $\mathbf{D}(\mathbf{d}) \in \mathbb{R}^{n+1,n+1}$ is a diagonal matrix with the vector $\mathbf{d}$ on the diagonal. Therefore, using equation (3.163), the gradient and the Hessian matrix of the h function read as

$$\boldsymbol{\nabla}h\left(\mathbf{c};\mathbf{x}^*\right) = t\boldsymbol{\nabla}\mathrm{TPD}^{(\mathrm{under})}\left(\mathbf{c};\mathbf{x}^*\right) + \mathbf{B}^{\mathrm{T}}\mathbf{d}, \tag{3.178}$$

$$\mathbf{H}^{h}\left(\mathbf{c}\right) = t\mathbf{H}^{\mathrm{TPD}^{(\mathrm{under})}}\left(\mathbf{c}\right) + \mathbf{B}^{\mathrm{T}}\mathbf{D}(\mathbf{d})^2\mathbf{B}. \tag{3.179}$$

Next, we provide details about the gradient and Hessian of the $\mathrm{TPD}^{(\mathrm{under})}$ function. Using the definition of function $\mathrm{TPD}^{(\mathrm{under})}(\mathbf{c};\mathbf{x}^*)$ given by equation (3.159) on a simplex $\mathcal{E} = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n+1)}\right]_{\kappa}$, we get

$$\begin{aligned}\left[\boldsymbol{\nabla}\mathrm{TPD}^{(\mathrm{under})}\right]_i(\mathbf{c}) &= -\frac{\partial a}{\partial c_i}\left(\mathbf{x}^*\right) + \frac{\partial a^{(\mathrm{under})}}{\partial c_i}\left(\mathbf{c}\right)\\ &= -\frac{\partial a}{\partial c_i}\left(\mathbf{x}^*\right) + \frac{\partial f_1}{\partial c_i}\left(\mathbf{c}\right) - \sum_{j=1}^{n+1}\frac{\partial \alpha_j}{\partial c_i}\left(\mathbf{c}\right) f_2\left(\mathbf{x}^{(j)}\right),\end{aligned} \tag{3.180}$$

where α_i for $i \in \widehat{n+1}$ are defined as

$$\sum_{i=1}^{n+1}\alpha_i\mathbf{x}^{(i)} = \mathbf{c}. \tag{3.181}$$

Differentiation of the previous equation with respect to c_j for $j \in \widehat{n}$ results in

$$\sum_{i=1}^{n+1}\frac{\partial \alpha_i}{\partial c_j}\left[\mathbf{x}^{(i)}\right]_k = \frac{\partial c_k}{\partial c_j} = \delta_{k,j}, \quad k \in \widehat{n}, \tag{3.182}$$

where $\delta_{k,j}$ is the Kronecker delta function. Since $\sum\limits_{i=1}^{n+1}\alpha_i = 1$, an additional condition for the derivative of α_i is

$$\sum_{i=1}^{n+1}\frac{\partial \alpha_i}{\partial c_j}\left(\mathbf{c}\right) = 0, \quad j \in \widehat{n}. \tag{3.183}$$

In conclusion, to calculate $\frac{\partial \alpha_i}{\partial c_j}$ one has to solve a system of linear equations with unknowns $\frac{\partial \alpha_i}{\partial c_j}$ for $i \in \widehat{n+1}$ and $j \in \widehat{n}$ in form

$$\begin{pmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(n+1)}\\ x_2^{(1)} & x_2^{(2)} & \cdots & x_2^{(n+1)}\\ \vdots & \vdots & \ddots & \vdots\\ x_{n-1}^{(1)} & x_{n-1}^{(2)} & \cdots & x_{n-1}^{(n+1)}\\ x_n^{(1)} & x_n^{(2)} & \cdots & x_n^{(n+1)}\\ 1 & 1 & \cdots & 1\end{pmatrix}\begin{pmatrix}\frac{\partial\alpha_1}{\partial c_1} & \frac{\partial\alpha_1}{\partial c_2} & \cdots & \frac{\partial\alpha_1}{\partial c_n}\\ \frac{\partial\alpha_2}{\partial c_1} & \frac{\partial\alpha_2}{\partial c_2} & \cdots & \frac{\partial\alpha_2}{\partial c_n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial\alpha_{n-1}}{\partial c_1} & \frac{\partial\alpha_{n-1}}{\partial c_2} & \cdots & \frac{\partial\alpha_{n-1}}{\partial c_n}\\ \frac{\partial\alpha_n}{\partial c_1} & \frac{\partial\alpha_n}{\partial c_2} & \cdots & \frac{\partial\alpha_n}{\partial c_n}\\ \frac{\partial\alpha_{n+1}}{\partial c_1} & \frac{\partial\alpha_{n+1}}{\partial c_2} & \cdots & \frac{\partial\alpha_{n+1}}{\partial c_n}\end{pmatrix} = \begin{pmatrix}1 & 0 & 0 & \cdots & 0\\ 0 & 1 & 0 & \cdots & 0\\ \vdots & \vdots & \ddots & \ddots & \vdots\\ 0 & \ldots & 0 & 1 & 0\\ 0 & \ldots & 0 & 0 & 1\\ 0 & \ldots & 0 & 0 & 0\end{pmatrix}. \tag{3.184}$$

For the Hessian matrix of $\mathrm{TPD}^{(\mathrm{under})}(\mathbf{c};\mathbf{x}^*)$ we get

$$\frac{\partial^2 \mathrm{TPD}^{(\mathrm{under})}}{\partial c_j \partial c_k}(\mathbf{c}) = \frac{\partial^2 a^{(\mathrm{under})}}{\partial c_j \partial c_k}(\mathbf{c}) = \frac{\partial^2 f_1}{\partial c_j \partial c_k}(\mathbf{c}) - \sum_{i=1}^{n+1} \frac{\partial^2 \alpha_i}{\partial c_j \partial c_k}(\mathbf{c})\, f_2\left(\mathbf{x}^{(i)}\right), \quad j,k \in \widehat{n}. \tag{3.185}$$

Since $\frac{\partial^2 \alpha_i}{\partial c_j \partial c_k}(\mathbf{c}) = 0$ for all $i \in \widehat{n+1}$, $j,k \in \widehat{n}$, the elements of the Hessian matrix of the $\mathrm{TPD}^{(\mathrm{under})}$ read as

$$\frac{\partial^2 \mathrm{TPD}^{(\mathrm{under})}}{\partial c_j \partial c_k}(\mathbf{c}) = \frac{\partial^2 f_1}{\partial c_j \partial c_k}(\mathbf{c}). \tag{3.186}$$

Now, we summarize the basic steps of the Barrier method.

1. Let $\mathbf{c}^{(0)} \in \mathcal{E}$, $t^{(0)} > 0$, $\mu > 1$, and precision `prec` > 0 be given. Set $k = 0$.
2. Compute $\mathbf{c}(t^{(k)})$ by minimizing the function $h(\mathbf{c};t)$, starting at $\mathbf{c}^{(k)}$. Minimize this function using the Newton–Raphson method.
3. Check if $(n+1)/t^{(k)} <$ `prec`. If yes, stop the computation. If not, update $\mathbf{c}^{(k+1)} = \mathbf{c}(t^{(k)})$, $t^{(k+1)} = \mu t^{(k)}$, and $k := k+1$ and go to step 2.

In this thesis, we are using suggested values from Boyd and Vandenberghe (2004): $\mathbf{c}^{(0)}$ is the barycenter of $\mathcal{E}$, $\mu = 20$, $t^{(0)} = 30$, and `prec` $= 10^{-15}$.

Selection of the new computation set $\mathcal{E}$

In the selection step, the current computation set $\mathcal{E}$ is the one with the minimal lower bound approximation and the initial approximation for the local minimization $\mathbf{z}$ is set as the point where the minimal lower bound is attained.

Stopping criteria

In our algorithm, we are using three stopping criteria. If any of these is met, the computation is stopped. The first criterion is met if

$$\frac{|\mathtt{UBD} - \mathtt{LBD}|}{|\mathtt{UBD}| + 1} < 10^{-8}, \tag{3.187}$$

where `UBD` is the upper bound (the result of the local optimization) and `LBD` is the lowest lower bound (the result of the optimization of the underestimated problem). In the ideal case, this criterion is sufficient. However, since we are limited by the computational time, we are using other two criteria. One stops the computation if 1000 iterations of the Branch and Bound algorithm (i.e., points 1–5 of the algorithm that includes one local optimization, two convex global optimizations, and one splitting of the computation simplex $\mathcal{E}$) are performed. Second, if each computation set was divided more than $2n$ times. Without these two additional criteria, the computation can be very time-consuming.

Concerning the stopping criteria, we would like to emphasize that in the phase stability testing, the mixture is unstable if there exists a state with a negative value of the TPD function. Therefore, in principle, there is no need to continue the global minimum search when the negative state is found. However, using this algorithm we are interested in whether the algorithm can find a minimum that algorithms which we used before (e.g., Jindrová and Mikyška (2015) or algorithm from Section 3.2.3) could not. For this reason, we used the stopping criteria presented above. Nonetheless, in the examples, we also run simulations with an additional condition that stops calculation if `UBD` $< -10^{-4}$ to see the speed-up in the unstable region.

3.2.6 Global heuristical approach

Thus far, we have presented three numerical strategies for finding the global minimum of the function TPD. In Section 3.2.1, the SSI method was presented. This method is robust in the PTN-specification and simple to implement. However, it suffers from slow convergence. In Section 3.2.3, a local method based on the Newton–Raphson iterations was presented. This method exhibits an excellent computation time. However, we do not have a guarantee that the found minimum is the global minimum. In Section 3.2.5, a global deterministic method based on the Branch and Bound framework was given. Using this method, the global minimum is always found. However, the computational time can exceed reasonable limits, and its usage is limited. Therefore, here we present global minimization methods based on the heuristical approach. In contrast to the deterministic approaches, the heuristical approach should solve the problem faster. However, this is only a trade-off of the mathematical certainty of finding the minimum for speed.

In this section, we present various global heuristical algorithms which can be used to solve the phase stability testing problem. We have published our findings in Smejkal et al. (2021). Primary, we are focused on the VTN-specification. However, these algorithms can be used in other specifications without any significant change. All presented heuristics are the so-called evolution algorithms which are inspired by natural selection and evolution. Let $\mathcal{P} = \left\{\mathbf{x}^{(i)}; i \in \widehat{NP}\right\} \subset \mathcal{D}$ be a given population, i.e., the initial approximations of the solution, where $NP > 0$ is the size of the population. Then, the three basic steps can be summarized as follows.

1. Evaluate the fitness of each individual $\mathbf{x}^{(i)}$, i.e., values TPD $\left(\mathbf{x}^{(i)};\mathbf{x}^{*}\right)$.

2. Select the best individuals for reproduction.

3. Breed new offspring using the best individuals and replace the worst ones with them.

In the literature, there exist numerous evolution algorithms. See, e.g., Pétrowski (2017); Sarker (2002); Simon (2013); Yu and Gen (2010). Here, we choose to test five algorithms based on our previous experience with them: Differential Evolution, Cuckoo Search, Harmony Search, CMA-ES, and Elephant Herding Optimization. In our implementation of each algorithm, the initial population $\mathcal{P}$ is generated randomly inside the feasible domain $\mathcal{D}$.

Differential Evolution

The Differential Evolution (DE) is an evolution algorithm developed by Storn and Price (1997). The DE algorithm consists of three steps: mutation, crossover and selection. In the mutation step, a parent $\mathbf{x} \in \mathcal{P}$ is chosen from the current population, and then a new mutant $\mathbf{y}$ is created. In our computations, we are using the version *DE/RAND/2* from Price et al. (2005) where the mutation operator has the form

$$\mathbf{y} = \mathbf{x}^{(r_1)} + F\left(\mathbf{x}^{(r_2)} - \mathbf{x}^{(r_3)}\right) + F\left(\mathbf{x}^{(r_4)} - \mathbf{x}^{(r_5)}\right), \tag{3.188}$$

where $r_i \sim \mathrm{U_d}(1, NP)$ are independent and identically distributed randomly generated integers from 1 to NP and F is a scaling parameter. In the literature, there are many suggestions on how to choose the parameter F. According to Price et al. (2005), the optimum range of F is usually between 0.4 and 1. In Storn and Price (1997), the authors suggested $F = 0.5$. In recent years, authors considered an adaptation of the parameter during calculation. In SaDE variation (see Qin et al. (2009)) of the Differential Evolution, the parameter F is approximated by a normal distribution with the mean value 0.5 and standard deviation 0.3. In our computations, we are changing F in each generation using the formula

$$F = F_{\max} \left(\frac{F_{\min}}{F_{\max}} \right)^{\frac{i_{\text{cur}}}{i_{\max}}}, \tag{3.189}$$

where $F_{\max} = 1.0$, $F_{\min} = 0.1$, i_{cur} is the current iteration index, and $i_{\max}$ is the maximum number of iterations. The main idea behind formula (3.189) is that at the beginning, the coefficient F is close to 1, and we are exploring the whole search space. As generations progress, parameter F is getting smaller, and we expect faster convergence. In the crossover step, the mutant vector is crossbred with its parent $\mathbf{x}$ to create a crossover child $\mathbf{z}$ by formula

$$[\mathbf{z}]_j = \begin{cases} [\mathbf{y}]_j, & \eta \leqslant CR \text{ or } j = j_{\text{rand}}, \\ [\mathbf{x}]_j, & \text{otherwise}, \end{cases} \tag{3.190}$$

where $j_{\text{rand}} \sim \mathrm{U_d}(1, n)$ is one random integer chosen for all $j \in \hat{n}$, $\eta \sim \mathrm{U_c}(0, 1)$ is independent and identically distributed randomly generated real number between 0 and 1, and CR is the crossover rate parameter. Similarly as in the case of F, there exist many suggestions on how to choose this parameter. We are using a strategy proposed by Mohamed et al. (2012), where the parameter CR is calculated using formula

$$CR = CR_{\max} + (CR_{\min} - CR_{\max}) \left(1 - \frac{i_{\text{cur}}}{i_{\max}} \right)^k. \tag{3.191}$$

In our computation, we are using $CR_{\max} = 0.9$, $CR_{\min} = 0.1$, and $k = 4$. In the last step, selection, the crossover child $\mathbf{z}$ replaces its parent in the population if its value of the objective function is lower. These three steps are performed for each parent $\mathbf{x} \in \mathcal{P}$ until a stopping criterion is met. In our computation, the population size is set to $NP = 20$. A pseudo-code of the Differential Evolution algorithm is presented in Algorithm 9.

Cuckoo Search

The Cuckoo Search is an evolution algorithm introduced by Yang (2009) and is based on the brood parasitic behaviour of cuckoos. In contrast to the Differential Evolution, the new member of the population in the Cuckoo Search is created using the Levy flight. The Levy flight is a process driven by an α-stable distribution, and it was shown by Pavlyukevich (2007a,b); Reynolds and Frye (2007) that this process has a similar behaviour as a flight of many animals and insects. The n-dimensional α-stable distribution $\mathrm{L}(\alpha, n)$ has probability density function (PDF)

$$\mathrm{g}(\mathbf{x}, \alpha, n) = \frac{1}{(2\pi)^n} \mathcal{F}_n \left\{ \exp\left(-\|\boldsymbol{\omega}\|^{\alpha} \right) \right\}, \tag{3.192}$$

where $\alpha \in (0, 2)$, $\boldsymbol{\omega} \in \mathbb{R}^m$ is the angular frequency in the Fourier domain, and $\mathcal{F}_n$ is the n-dimensional Fourier transformation. The Cuckoo Search algorithm consists of two steps: the local search and the global search. At the local search, one member of the population (representing cuckoo nest), say $\mathbf{x}^{(i)} \in \mathcal{P}$, is chosen, and from this position, the Levy flight is performed to create a new member $\mathbf{y}$

$$\mathbf{y} = \mathbf{x}^{(i)} + \theta \xi, \tag{3.193}$$

where θ is a scaling parameter, which should be related to the size of the search space, and $\xi \sim \mathrm{L}(\alpha, n)$. In this thesis, we are using $\theta = \frac{\|\Omega\|}{10^4}$, where the norm of the rectangular search space $\|\Omega\|$ is defined as the length of the largest side. If $\mathrm{TPD}(\mathbf{y}; \mathbf{x}^*) < \mathrm{TPD}\left(\mathbf{x}^{(k)}; \mathbf{x}^*\right)$, where $k \sim \mathrm{U_d}(1, NP)$, then $\mathbf{x}^{(k)}$ is replaced in the population $\mathcal{P}$ by $\mathbf{y}$. At the global search, the

Algorithm 9: Differential Evolution.

```
   Input  : $NP > 0$, $F_{\min} \in (0,1]$, $F_{\max} \in (0,1]$, $CR_{\min} \in (0,1]$, $CR_{\max} \in (0,1]$
   Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
 1 randomly initialize population $\mathcal{P} = \left\{\mathbf{x}^{(i)}; i \in \widehat{NP}\right\}$,
 2 while stop criterion do
 3     update values of $F$ and $CR$ using equations (3.189) and (3.191)
 4     for all members $\mathbf{x}^{(i)} \in \mathcal{P}$ do
 5         create a mutant member $\mathbf{y}$ using equation (3.188)
 6         create a crossover member $\mathbf{z}$ from $\mathbf{x}^{(i)}$ and $\mathbf{y}$ using equation (3.190)
 7         if TPD$(\mathbf{z};\mathbf{x}^*) <$ TPD$\left(\mathbf{x}^{(i)};\mathbf{x}^*\right)$ then
 8             replace $\mathbf{x}^{(i)}$ in the population with $\mathbf{z}$
 9         end
10     end
11 end
12 compute minimum $M = \min_{i \in \widehat{NP}} \text{TPD}\left(\mathbf{x}^{(i)};\mathbf{x}^*\right)$
13 if $M < 0$ then
14     return false
15 else
16     return true
17 end
```

population is sorted with respect to its TPD values and the worst p percent of the population are replaced with new ones using Levy flights, i.e.,

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k)} + \theta\xi, \tag{3.194}$$

where $\xi \sim \mathrm{L}(\alpha, n)$ and $k > (1-p)NP$. These two steps are repeated until a stopping criterion is met. Suggested values from the literature are $\alpha = 1$, population size $NP = 15$, and $p = 0.25$. These values are used in our computations. The summary of this algorithm is presented in Algorithm 10.

Harmony Search

The third heuristic for the comparison is the Harmony Search, which was created by Geem (2009). Among the presented algorithms, the Harmony Search is the simplest one. In each iteration, a new member $\mathbf{y}$ of the population is created using improvisation. In this technique, each component y_k of the new member $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$ is created using the current population or is chosen randomly. First, a random number $i \sim \mathrm{U_c}(0,1)$ is generated. If i is greater or equal than a given parameter $HCMR$ (harmony memory consideration rate) then y_k is chosen randomly in the search space. On the other hand, if $i < HCMR$, then the k-th component is copied from a randomly chosen member of the population, i.e., $y_k = \left[\mathbf{x}^{(j)}\right]_k$, where $j \sim \mathrm{U_d}(1, NP)$. Moreover, in the second situation, a random number $q \sim \mathrm{U_c}(0,1)$ is generated, and if q is lower than a given parameter PAR (pitch adjusting rate), then the solution is perturbed using equation

$$y_k = y_k + BW\Delta, \tag{3.195}$$

where BW (bandwidth) is given parameter and $\Delta \sim \mathrm{U_c}(-1,1)$. In our computations, we are using suggested values by Geem (2009): $NP = 6$, $HCMR = 0.85$, $PAR = 0.5$, and $BW = 0.001$. The

Algorithm 10: Cuckoo Search.

```
   Input  : $NP > 0$, $p \in (0,1)$, $\alpha \in (0,2)$, $\theta \in \mathbb{R}^+$
   Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
 1 randomly initialize population $\mathcal{P} = \left\{\mathbf{x}^{(i)}; i \in \widehat{NP}\right\}$,
 2 while stop criterion do
 3     choose randomly $j \in \mathrm{U_d}(1, NP)$
 4     create a new member $\mathbf{y}$ using equation (3.193)
 5     choose randomly $k \in \mathrm{U_d}(1, NP)$
 6     if $\mathrm{TPD}\left(\mathbf{y};\mathbf{x}^*\right) < \mathrm{TPD}\left(\mathbf{x}^{(k)};\mathbf{x}^*\right)$ then
 7         replace $\mathbf{x}^{(k)}$ in the population with $\mathbf{y}$
 8     end
 9     sort the population with respect to the TPD value
10     replace worst $p$ percent of the population using Levy flight
               $\mathbf{x}^{(k)} := \mathbf{x}^{(k)} + \theta\xi, \quad \xi \sim L\left(\alpha, n\right)$
        for $k > (1-p)NP$
11 end
12 compute minimum $M = \min_{i \in \widehat{NP}} \mathrm{TPD}\left(\mathbf{x}^{(i)};\mathbf{x}^*\right)$
13 if $M < 0$ then
14     return false
15 else
16     return true
17 end
```

pseudo-code of the improvisation technique is provided in Algorithm 11. After the improvisation, if the new member $\mathbf{y}$ has a lower value of the objective function than the worst member of the population $\mathcal{P}$, then the worst member of the population is replaced with y in the population. These steps are repeated until a stopping criterion is met. The summary of the Harmony Search algorithm is presented in Algorithm 12. In our computations, the size of the population is set to the suggested value $NP = 6$ (Geem (2009)). Therefore, compared to the size of the population in the Differential Evolution or the Cuckoo Search, the population in the Harmony Search is considerably smaller.

Covariant Matrix Adaptation

In the Covariant Matrix Adaptation (CMA-ES) developed by Hansen and Ostermeier (2001); Ostermeier et al. (1994), the next generation of the population is generated from the normal distribution

$$\mathbf{x}_i^{(g+1)} \sim \mathbf{m}^{(g)} + \sigma^{(g)}\mathcal{N}(0, \mathbf{C}^{(g)}), \tag{3.196}$$

where $\mathbf{m}^{(g)}$ is the mean value, $\mathbf{C}^{(g)}$ is the covariance matrix, and $\sigma^{(g)}$ is the step size of the current generation. The mean value of the next generation $\mathbf{m}^{(g+1)}$ is calculated using

$$\mathbf{m}^{(g+1)} = \mathbf{m}^{(g)} + c_m \sum_{i=1}^{\beta} w_i \left(\mathbf{x}_i^{(g+1)} - \mathbf{m}^{(g)}\right), \tag{3.197}$$

Algorithm 11: Harmony Search improvisation.

```
   Input  : $HMCR \in (0,1)$, $PAR \in (0,1)$, $BW \in (0,1)$
   Output : $\mathbf{y} = (y_1, \dots, y_n)^{\mathrm{T}}$
1  set $k = 1$
2  for $k \leqslant n$ do
3      randomly choose $i \in \mathrm{U_c}(0,1)$
4      if $i < HMCR$ then
5          randomly choose $j \in \mathrm{U_d}(1, NP)$
6          set $k$-th component of the $\mathbf{y}$ using: $[\mathbf{y}]_k = \left[\mathbf{x}^{(j)}\right]_k$
7          randomly choose $q \in \mathrm{U_c}(0,1)$
8          if $q < PAR$ then
9              randomly choose $\Delta \in \mathrm{U_c}(-1,1)$
10             reset $k$-th component using: $[\mathbf{y}]_k := [\mathbf{y}]_k + BW\Delta$
11         end
12     else
13         set $k$-th component of the $\mathbf{y}$ randomly inside feasible area
14     end
15     k := k+1
16 end
17 return y
```

Algorithm 12: Harmony Search.

```
   Input  : $NP > 0$, $HMCR \in (0,1)$, $PAR \in (0,1)$, $BW \in (0,1)$
   Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
1  randomly initialize population $\mathcal{P} = \left\{\mathbf{x}^{(i)}; i \in \widehat{NP}\right\}$,
2  while stop criterion do
3      create a new member $\mathbf{y}$ using improvisation from Algorithm 11
4      if $\mathrm{TPD}\,(\mathbf{y};\mathbf{x}^*) < \mathrm{TPD}\left(\mathbf{x}^{(\mathrm{worst})};\mathbf{x}^*\right)$ then
5          replace $\mathbf{x}^{(\mathrm{worst})}$ in the population with $\mathbf{y}$
6      end
7  end
8  compute minimum $M = \min_{i \in \widehat{NP}} \mathrm{TPD}\left(\mathbf{x}^{(i)};\mathbf{x}^*\right)$
9  if $M < 0$ then
10     return false
11 else
12     return true
13 end
```

where β is the number of parents, c_m is the learning rate, and w_i are the weights, which have to satisfy $\sum\limits_{i=1}^{\beta} w_i = 1$. The covariance matrix $\mathbf{C}$ is updated using

$$\mathbf{C}^{(g+1)} = (1-2c_1)\mathbf{C}^{(g)} + c_\beta \sum_{i=1}^{\beta} w_i \mathbf{y}_i \mathbf{y}_i^{\mathrm{T}} + c_1 \mathbf{p}_c^{(g+1)} \left(\mathbf{p}_c^{(g+1)}\right)^{\mathrm{T}}, \tag{3.198}$$

where $\mathbf{p}_c$ and $\mathbf{y}_i$ are calculated using

$$\mathbf{p}_c^{(g+1)} = (1-c_c)\mathbf{p}_c^{(g)} + \sqrt{c_c(2-c_c)\beta_{eff}} \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}, \tag{3.199}$$

$$\mathbf{y}_i = \frac{1}{\sigma^{(g)}} \left(\mathbf{x}_i^{(g+1)} - \mathbf{m}^{(g)}\right). \tag{3.200}$$

Lastly, the step size $\sigma^{(g)}$ is updated using

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left\{ \frac{c_\sigma}{d_\sigma} \left(\frac{||\mathbf{p}_\sigma^{(g+1)}||}{\mathrm{E}||\mathcal{N}(0,\mathbf{I})||} - 1 \right) \right\}, \tag{3.201}$$

where

$$\mathbf{p}_\sigma^{(g+1)} = (1-c_\sigma)\mathbf{p}_\sigma^{(g)} + \sqrt{c_c(2-c_c)\beta_{eff}}\, \mathbf{C}^{-\frac{1}{2}} \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}, \tag{3.202}$$

and $\mathrm{E}||\mathcal{N}(0,\mathbf{I})||$ is estimated using Hansen (2016)

$$\mathrm{E}||\mathcal{N}(0,\mathbf{I})|| = \sqrt{n}\left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right). \tag{3.203}$$

In the previous equations, c_1, c_β, c_σ, d_σ,c_c, and β_{eff} are numerical coefficients. These coefficients depends on the size of the population and number of parents. In this thesis, we have taken over the strategy from Hansen (2016). The size of population is set to

$$NP = 4 + 3\lfloor \ln n \rfloor, \tag{3.204}$$

where $\lfloor \ln n \rfloor$ is the floor function, i.e., the greatest integer less than or equal to $\ln n$. The number of parents is set to $\beta = \lfloor \frac{NP}{2} \rfloor$. Then, the weights w_i are computed using

$$\breve{w}_i = \ln\left(\frac{NP+1}{2}\right) + \ln(i+1), \quad i \in \widehat{\beta}, \tag{3.205}$$

and

$$w_i = \frac{\breve{w}_i}{\sum\limits_{j=1}^{\beta} \breve{w}_j}, \quad i \in \widehat{\beta}. \tag{3.206}$$

Then, the remaining numerical coefficients are computed using

$$\beta_{eff} = \frac{\left(\sum_{i=1}^{\beta} w_i\right)^2}{\sum_{i=1}^{\beta} w_i^2}, \tag{3.207}$$

$$c_c = \frac{4 + \frac{\beta_{eff}}{n}}{n + 4 + 2\frac{\beta_{eff}}{n}}, \tag{3.208}$$

$$c_\sigma = \frac{\beta_{eff} + 2}{n + \beta_{eff} + 5}, \tag{3.209}$$

$$c_1 = \frac{2}{(n + 1.3)^2 + \beta_{eff}}, \tag{3.210}$$

$$c_\beta = \min\left\{1 - c_1, 2\frac{\beta_{eff} - 2 + \frac{1}{\beta_{eff}}}{(n+2)^2 + \beta_{eff}}\right\}, \tag{3.211}$$

$$d_\sigma = 1 + 2\max\left\{0, \sqrt{\frac{\beta_{eff} - 1}{n + 1}} - 1\right\} + c_\sigma. \tag{3.212}$$

The summary of the CMA-ES algorithm is presented in Algorithm 13.

Algorithm 13: CMA-ES.

```
   Input  : $NP > 0$, $\beta > 0$
   Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
 1 randomly initialize population $\mathcal{P} = \left\{\mathbf{x}^{(i)}; i \in \widehat{NP}\right\}$,
 2 compute the auxiliary coefficients using equations (3.207)–(3.212)
 3 while stop criterion do
 4     for all members $\mathbf{x}^{(i)} \in \mathcal{P}$ do
 5         update a population member $\mathbf{x}^{(i)}$ using equation (3.196)
 6     end
 7     update the mean using equation (3.197)
 8     update the covariance matrix using equation (3.198)
 9     update the step size using equation (3.201)
10 end
11 compute minimum $M = \min_{i \in \widehat{NP}} \mathrm{TPD}\left(\mathbf{x}^{(i)};\mathbf{x}^*\right)$
12 if $M < 0$ then
13     return false
14 else
15     return true
16 end
```

Elephant Herding Optimization

In the Elephant Herding Optimization algorithm, the population of size NP is divided into NC clans. Let us denote the number of elephants in the k-th clan by NE_k, $\mathbf{x}^{(k,i)}$ the i-th elephant in the k-th clan, and let $\mathbf{x}^{(k,1)}$ be the elephant with the lowest value of the objective function in the k-th clan. Then, the next generation $\mathbf{x}^{(k,i),(g+1)}$ of the population is created using

$$\mathbf{x}^{(k,i),(g+1)} = \mathbf{x}^{(k,1),(g)} + r\,\delta\left(\mathbf{x}^{(k,1),(g)} - \mathbf{x}^{(k,i),(g)}\right), \tag{3.213}$$

Algorithm 14: Elephant Herding Optimization.

```
Input  : $NP > 0$, $NC > 0$, $\gamma > 0$, $\delta \in (0,1)$, $\beta \in (0,1]$
Output : true = state $\mathbf{c}^*$ is stable, false = state $\mathbf{c}^*$ is unstable
1  randomly initialize population $\mathcal{P} = \left\{\mathbf{x}^{(k,i)}; k \in \widehat{NC}, i \in \widehat{NE_k}\right\}$,
2  while stop criterion do
3      store the best $\gamma$ member of the population
4      divide the population into clans
5      for all members $\mathbf{x}^{(k,i)} \in \mathcal{P}$ do
6          if $i = 1$ then
7              update member $\mathbf{x}^{(k,1)}$ using equation (3.214)
8          end
9          else
10             update member $\mathbf{x}^{(k,i)}$ using equation (3.213)
11         end
12         replace the worst member in each clan with randomly created
13         replace the $\gamma$ worst members with the stored ones
14     end
15 end
16 compute minimum $M = \min_{i \in \widehat{NP}} \mathrm{TPD}\left(\mathbf{x}^{(i)};\mathbf{x}^*\right)$
17 if $M < 0$ then
18     return false
19 else
20     return true
21 end
```

for $i \in \widehat{NE_k} \backslash \{1\}$ and $k \in \widehat{NC}$. In the previous equation, $\delta \in (0,1)$ is a scale factor and $r \sim \mathrm{U_c}(0,1)$. The best elephants in each clan $\mathbf{x}^{(k,1)}$ are updated using

$$\mathbf{x}^{(k,1),(g+1)} = \frac{\beta}{NE_k}\left(\mathbf{x}^{(k,1),(g)} + \sum_{j=2}^{NE_k} \mathbf{x}^{(k,j),(g+1)}\right), \tag{3.214}$$

where β is a scale factor. The second step of the algorithm is the separation of the worst member of the population. In each clan, the member with the highest value of the objective function is replaced in the population with a randomly created solution. Lastly, the algorithm includes an elitism strategy. At the beginning of each iteration, the best γ members of the population are stored, and at the end of the iteration, the worst γ members of the population are replaced. The pseudo-code of the Elephant Herding Optimization algorithm is provided in Algorithm 14. In this thesis, we are using values $\delta = 0.5$, $\beta = 0.1$, $NP = 30$, $NC = 5$, and $\gamma = 2$.

Mirroring

All presented heuristics are designed to find the global minimum of a function in a rectangular domain Ω. In the case of the VTN-specification of the phase stability testing, the feasible domain $\mathcal{D}$ is inside an n-dimensional rectangle with sides $\mathbf{a} = (0,\ldots,0)^{\mathrm{T}}$ and $\mathbf{b} = \left(\frac{1}{b_1},\ldots,\frac{1}{b_n}\right)^{\mathrm{T}}$, where b_k for $k \in \hat{n}$ is the co-volume parameter of the Peng–Robinson equation of state (see equation (2.103)). In many cases, it may happen that the new member of the population (mutant, cuckoo, improvisation) is created outside of this domain Ω or outside the feasible domain $\mathcal{D} \subset \Omega$. To overcome this situation, a method called mirroring is adopted for our algorithms. If the new member of the population will be placed outside of Ω, it will be bounced back from the boundary to the domain Ω. This method is well-known and used in many optimization algorithms. However, since in the phase stability problems, the feasible domain is a simplex $\mathcal{D} \subset \Omega$, using only the mirroring technique will not be satisfactory. Principally, three possible techniques can be used:

- penalty function in the unfeasible area,
- redefinition of the objective function outside the simplex,
- generalization of the mirroring technique to handle simplex geometry.

The first possibility can not be used alone since the objective function is not defined outside the feasible area. In the second technique, we can define TPD function as a big positive constant outside the feasible simplex $\mathcal{D}$. However, in our experiments, this technique did not provide satisfactory results. Therefore, we expanded the mirroring technique to mirror a vector into a simplex. The scheme of our strategy is presented in Figure 3.5. The solution $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$ of the phase stability testing has to be in a simplex $\mathcal{D}$ defined by equations

$$y_k \geqslant 0, \quad k \in \widehat{n}, \tag{3.215}$$

$$\sum_{i=1}^{n} b_i y_i \leqslant 1 - \varepsilon, \tag{3.216}$$

where $\varepsilon > 0$ is a small positive number as the solution has to satisfy $\sum\limits_{i=1}^{n} b_i c_i < 1$ (see equation (3.17)). In this thesis, we are using $\varepsilon = 10^{-6}$. In the first step of our algorithm, we transform the solution with relation $x_k = \frac{(1-\varepsilon) y_k}{b_k}$ for $k \in \widehat{n}$. Now, the problem is to mirror the transformed solution $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ into the unit simplex defined by

$$x_k \geqslant 0, \quad k \in \widehat{n}, \tag{3.217}$$

$$\sum_{i=1}^{n} x_i \leqslant 1. \tag{3.218}$$

In the second step, the solution is mirrored to satisfy $\mathbf{x} \in [0,1]^n$, where $[0,1]^n$ is a Cartesian product of n identical intervals $[0,1]$. This mirroring can be done using the standard mirroring technique. If (3.218) does not hold, the next task is to find a vector $\mathbf{z} = (z_1, \ldots, z_n)^{\mathrm{T}}$, which is on the boundary of the simplex (3.217) and (3.218) and its distance from the solution $\mathbf{x}$ is minimal. Therefore, the vector $\mathbf{z}$ has to satisfy

$$\sum_{i=1}^{n} z_i = 1, \tag{3.219}$$

$$\mathbf{z} = \operatorname*{arg\,min}_{\mathbf{y} \in \mathbb{R}^n} \frac{1}{2} \|\mathbf{x} - \mathbf{y}\|_2^2, \tag{3.220}$$

$$z_k \geqslant 0, \quad k \in \widehat{n}, \tag{3.221}$$

where the norm in (3.220) is the Euclidean norm. The factor $\frac{1}{2}$ in (3.220) simplifies derivation and does not change the resulting vector $\mathbf{z}$. The vector $\mathbf{z}$ can be found using the Lagrange multipliers method and reads as

$$z_k = x_k - \frac{1}{n} \left(\sum_{i=1}^{n} x_i - 1 \right), \quad k \in \widehat{n}. \tag{3.222}$$

The resulting novel mirroring process begins with $\mathbf{x} \in \Omega$ and terminates for $\mathbf{x}^{(\text{new})}$, which satisfies equations (3.217)–(3.218), using the iteration scheme

$$\mathbf{x}^{(\text{new})} = |\mathbf{x} - 2\mathbf{z}| = \left| \mathbf{x} - \frac{2}{n} \left(\sum_{i=1}^{n} x_i - 1 \right) \mathbf{1} \right|, \tag{3.223}$$

which terminates in a finite number of steps. In equation (3.223), $\mathbf{1} = (1, \dots, 1)^{\mathrm{T}} \in \mathbb{R}^n$, and the absolute value is applied component-wise. In the last step of our algorithm, after a solution $\mathbf{x}^{(\text{new})}$ has been found, the reverse transformation $y_k^{(\text{new})} = \frac{x_k^{(\text{new})} b_k}{1-\varepsilon}$ is applied to create $\mathbf{y}^{(\text{new})} \in \mathcal{D}$.

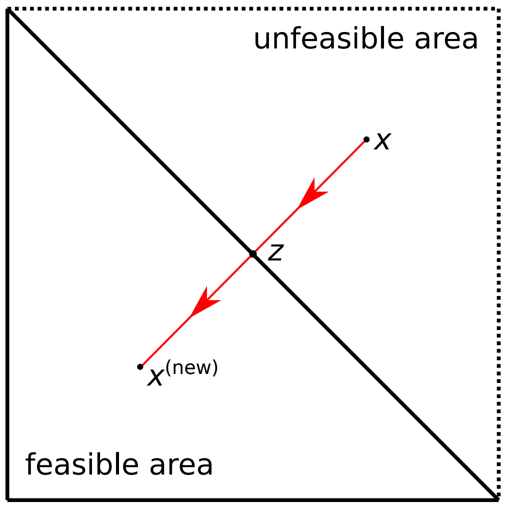


Figure 3.5: Geometric interpretation of the mirroring technique into the feasible simplex.

3.3 Examples

In this section, we provide examples showing the performance of the phase stability testing algorithms presented in the previous sections. First, in Section 3.3.1, we present the performance of the modified Newton–Raphson method presented in Section 3.2.3. We give two examples using the VTN-specification, and one example using the PTN-specification. Then, in Section 3.3.2, examples with the efficient solution of linear systems arising from the linearization of the VTN-phase stability using the Sherman–Morrison iteration will be investigated. Two examples will be given. Then, in Section 3.3.3, the examples showing the performance of the global method based on the Branch and Bound strategy will be given. In Section 3.3.4, the performance of the heuristics algorithms will be presented. Lastly, in Section 3.3.5, numerical examples showing the performance of the SSI method in individual formulations will be given. Comparison with the Newton–Raphson method will be presented. The physical properties of all components used in the examples are presented in Appendix A. Unless said otherwise, the Peng–Robinson equation of state given in Section 2.4.2 will be used.

3.3.1 Performance of the unified Newton–Raphson method

In this section, we present examples showing the performance of the modified Newton–Raphson method presented in Section 3.2.3. Since we are using the unified formulation, an identical code can compute the VTN-, UVN-, or PTN-specification. First, we present two examples with the VTN-specification. Then, example showing the performance of the PTN-specification will be given. A computer with Intel(R) Core(TM) i7-9700K (3.60GHz) processor was used.

Example A1: mixture C_1–C_5

In the first example, we investigate a binary mixture of methane (C_1) and pentane (C_5) with mole fractions $x^*_{C_1} = 0.489575$ and $x^*_{C_5} = 0.510425$. The binary interaction coefficient is $\delta_{C_1-C_5} = 0.041$. The VTN approach is used in this examples. In Figure 3.6a, the minima of the function TPD in the range $T^* \in [250, 450]$ K and a whole range of possible total concentrations c^* are presented. A computation grid with 100×100 points was used, i.e., total 10000 phase stability computations were performed. Moreover, in Figure 3.6b, the phase boundary is depicted. The results are in agreement with Mikyška and Firoozabadi (2012). All stability computations proceeded without any difficulties. On average, the Newton–Raphson method needed approximately 10.2 iterations to converge. Moreover, the mean computation time per one stability was 3.7743×10^{-4} s.

Example A2: mixture H_2O–CO_2

In the second example, we investigate a binary mixture of water (H_2O) and carbon dioxide (CO_2) with mole fractions $x^*_{H_2O} = 0.95$ and $x^*_{CO_2} = 0.05$. In this example, we use the VTN approach and the Peng–Robinson equation of state with association presented in Section 2.4.3. The cross association coefficient s_{CO_2} and the binary interaction coefficient $\delta_{H_2O-CO_2}$ are computed using correlation from Li et al. (2020)

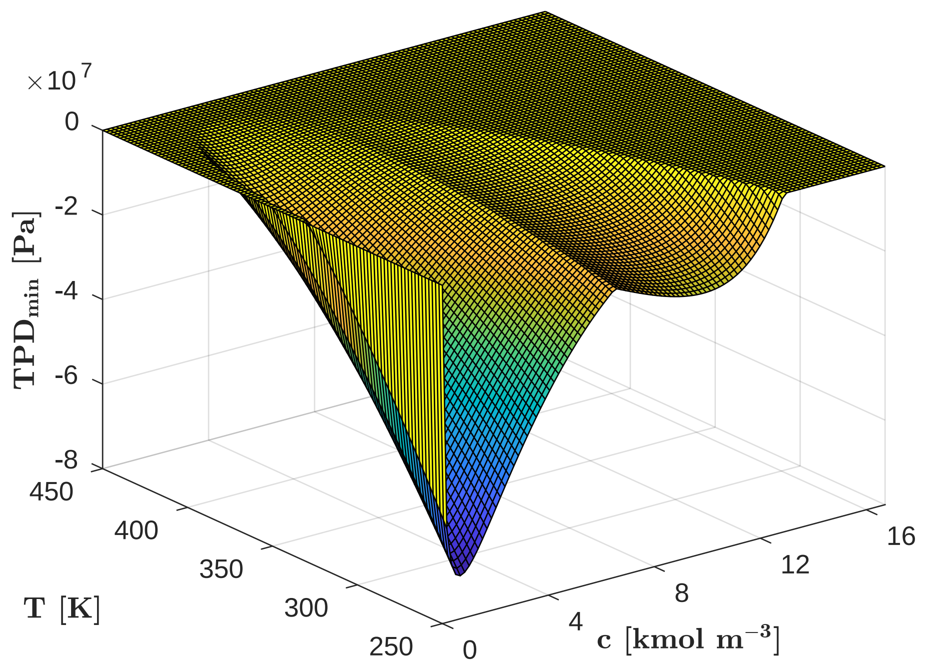
$$s_{CO_2} = -0.425T^3_{\mathrm{red},CO_2} + 1.6922T^2_{\mathrm{red},CO_2} - 1.9815T_{\mathrm{red},CO_2} + 0.7380, \tag{3.224}$$

$$\delta_{H_2O-CO_2} = 0.6546T_{\mathrm{red},CO_2} - 0.6165, \tag{3.225}$$

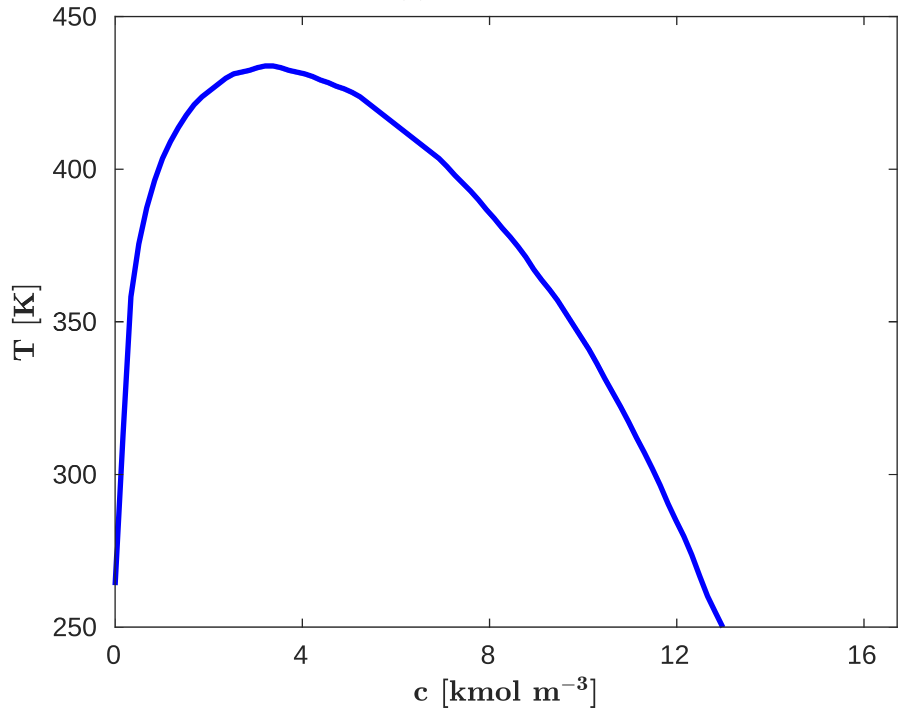
where $T_{\mathrm{red},CO_2} = \frac{T}{T_{c,CO_2}}$ is the reduced temperature of CO_2. In Figure 3.7a, the minima of the function TPD in the range $T^* \in [300, 700]$ K and a whole range of possible total concentrations c^* are presented. A computation grid with 100×100 points was used, i.e., total 10000 phase stability computations were performed. Moreover, in Figure 3.7b, the phase boundary is depicted. All stability computations proceeded without any difficulties. On average, the Newton–Raphson method needed approximately 10.7 iterations to converge. Therefore, almost identical value as in the first example. However, the mean computation time per one stability was 0.0181 s. Therefore, the computation time is higher in comparison with the first example. This behaviour is expected, since the association model is computationally intensive.

Example A3: mixture N_2–C_i

In the third example, we investigate a seven component mixture from Michelsen (1982a) of nitrogen (N_2) and alkans from methane (C_1) to hexane (C_6) with mole fractions $x^*_{N_2} = 0.0140$, $x^*_{C_1} = 0.9430$, $x^*_{C_2} = 0.0270$, $x^*_{C_3} = 0.0074$, $x^*_{C_4} = 0.0049$, $x^*_{C_5} = 0.0027$, and $x^*_{C_6} = 0.0010$. The binary interaction coefficients between all components are presented in Table 3.2. The PTN approach is used in this examples. In Figure 3.8a, the minima of the function TPD in the range $T^* \in [150, 260]$ K and $P^* \in [1, 9]$ MPa are presented. A computation grid with 100×100 points was used. Moreover, in Figure 3.8b, the phase boundary is depicted. All stability computations

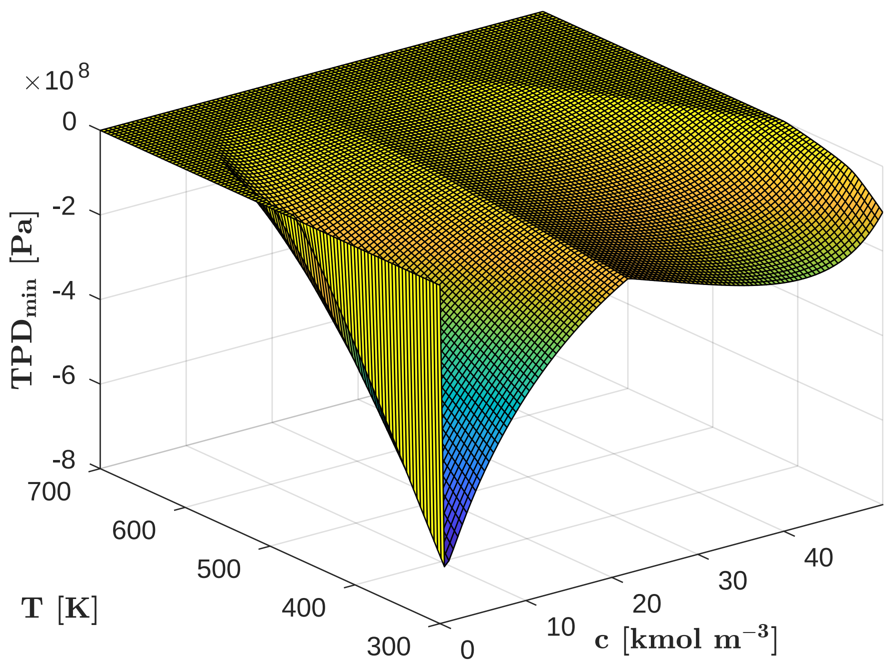


(a) $\mathrm{TPD_{min}}$

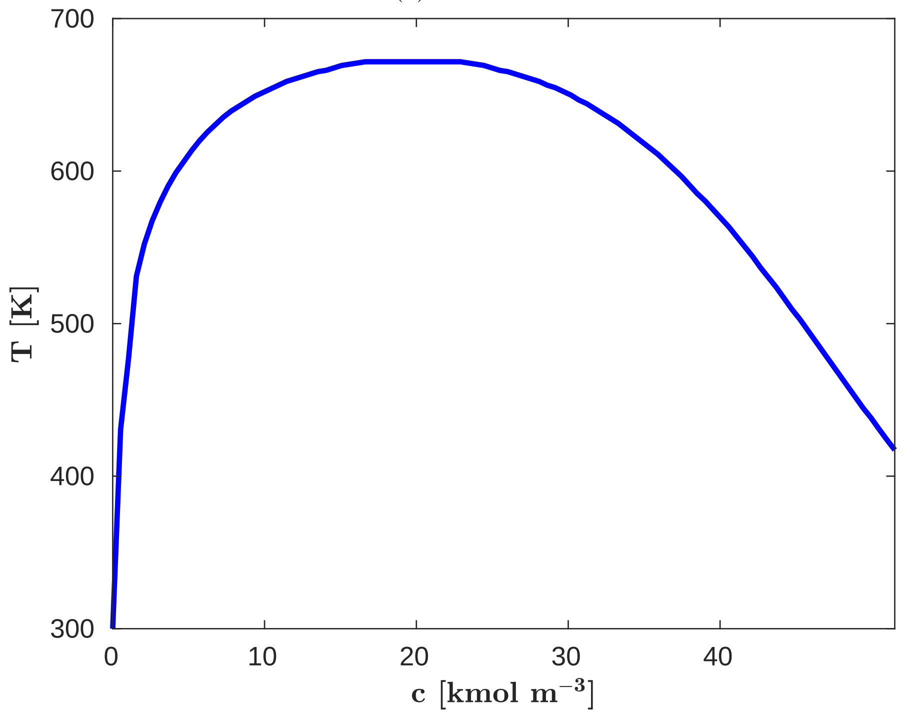


(b) phase boundary

Figure 3.6: Global minima of the TPD function (top). Approximate boundary between the stable and unstable area in the cT space. Below the line the states are unstable (bottom). Example A1: mixture C_1–C_5.



(a) TPD_{min}



(b) phase boundary

Figure 3.7: Global minima of the TPD function (top). Approximate boundary between the stable and unstable area in the *cT* space. Below the line the states are unstable (bottom). Example A2: mixture H_2O-CO_2.

	N_2	C_1	C_2	C_3	C_4	C_5	C_6
N_2	0	0.100	0.100	0.100	0.100	0.100	0.100
C_1	0.100	0	0.034	0.036	0.038	0.041	0.043
C_2	0.100	0.034	0	0	0	0	0
C_3	0.100	0.036	0	0	0	0	0
C_4	0.100	0.038	0	0	0	0	0
C_5	0.100	0.041	0	0	0	0	0
C_6	0.100	0.043	0	0	0	0	0

Table 3.2: The binary interaction coefficients between all components. Data taken over from Michelsen (1982a). Example A3: mixture $N_2{-}C_i$.

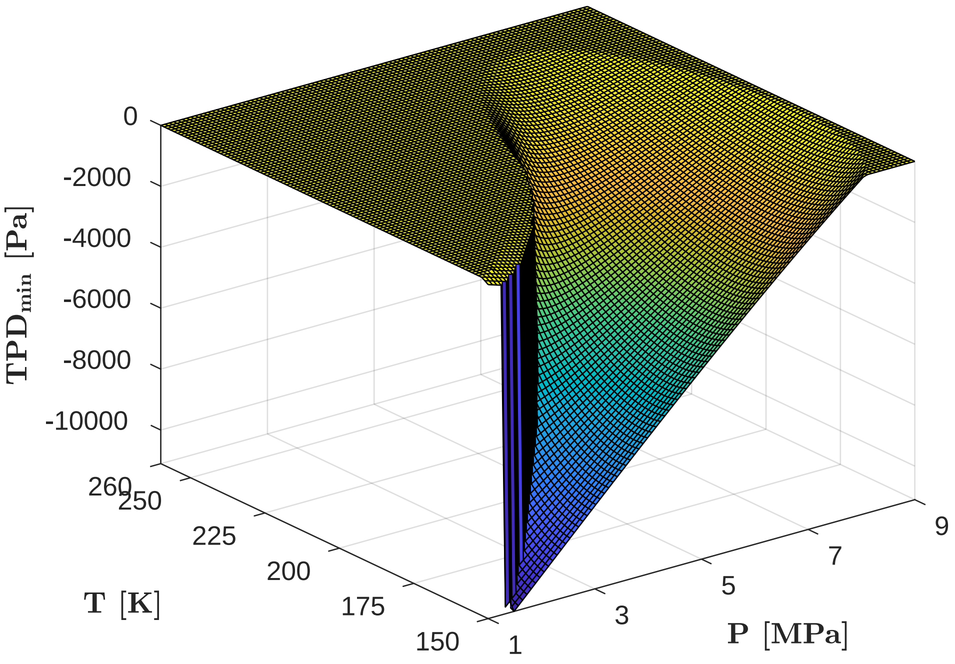
proceeded without any difficulties. The comparison with the original results of Michelsen (1982a) is not straightforward. They used different equation of state (Soave–Redlich–Kwong, see Soave (1972)), and some parameters were not given. On average, the Newton–Raphson method needed approximately 9.5 iterations to converge. Moreover, the mean computation time per one stability was 0.0101 s.

3.3.2 Efficient solution of the linearized system using the Sherman–Morrison iterations

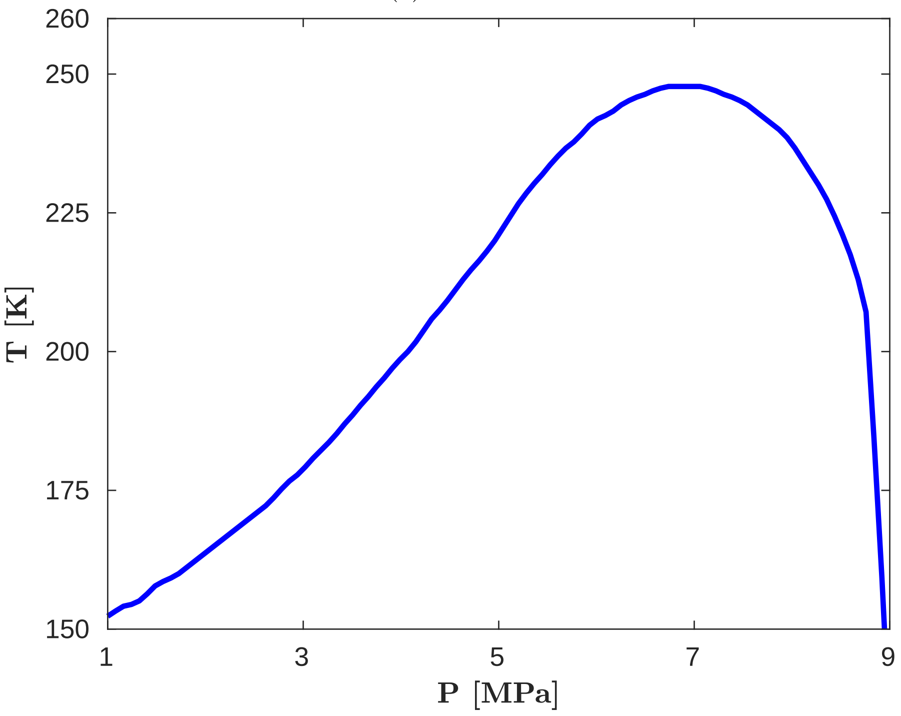
In this section, we present computation results on several examples from the literature showing the performance of the numerical algorithm presented in Section 3.2.4. To measure the speed-up of the computation, we test the algorithm on two examples. The first example is a computational study on the ten component mixture Y10 presented in Hoteit and Firoozabadi (2006b); Petitfrere and Nichita (2015b), and the strategy given in those papers. The second example is a computational study on mixtures with 4, 8, 12, 17, 25, 35, and 45 components. In both examples, the mixtures will be investigated in a cT space in a given range of molar densities and temperatures. A grid with 100×100 points will be used. Therefore, 10000 phase stability computations are performed on each mixture. The computation times and the speed-ups in the stable and unstable regions will be discussed. A computer with Intel(R) Core(TM) i7-8700 (3.20 GHz) processor was used. First, we run the algorithm presented in Section 3.2.3. In the next paragraphs, we will denote this algorithm as 'classic'. Second, we run the algorithm presented in Section 3.2.4. We denote the algorithm as 'update'.

Example B1: mixture Y10

In the first example, the algorithm is tested on mixture Y10 (see Hoteit and Firoozabadi (2006b); Petitfrere and Nichita (2015b)). The mixture Y10, which is a ten component mixture consisting of normal alkanes (denoted as C_i) from methane C_1 to octane C_8, decane C_{10}, and tetradecane C_{14}. The initial mole fractions are $x^*_{C_1} = 0.35$, $x^*_{C_2} = 0.03$, $x^*_{C_3} = 0.04$, $x^*_{C_4} = 0.06$, $x^*_{C_5} = 0.04$, $x^*_{C_6} = 0.03$, $x^*_{C_7} = 0.05$, $x^*_{C_8} = 0.05$, $x^*_{C_{10}} = 0.30$, and $x^*_{C_{14}} = 0.05$. The only non-zero binary interaction coefficients are between C_1 and other components. The used values are given in Table 3.3. To increase the number of components, the following algorithm from Petitfrere and Nichita (2015b) is adopted. First, a component, say i, different from C_1 is randomly chosen. The properties of this component define component $n + 1$. Then, the initial mole fraction x^*_i is split in half, and each half is assigned to component i and $n + 1$. Thus, from a mixture of n component, a mixture of $n + 1$ component is created. This process is repeated until the



(a) $\mathrm{TPD_{min}}$



(b) phase boundary

Figure 3.8: Global minima of the TPD function (top). Approximate boundary between the stable and unstable area in the *PT* space. Below the line the states are unstable (bottom). Example A3: mixture N_2–C_i.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_{10}	C_{14}
C_1	0	0	0	0.02	0.02	0.025	0.025	0.035	0.045	0.045

Table 3.3: The binary interaction coefficients between methane (C_1) and other alkans. Data taken over from Hoteit and Firoozabadi (2006b). Example B1: mixture Y10.

maximum number of components is achieved. In our computation, we stopped when $n = 45$. We investigated these mixtures in the cT space with a temperature range $T^* \in [150, 600]$ K and the whole feasible range of the molar density c^*. In Figure 3.9, the phase boundary is presented. As the mixture for arbitrary n physically represents the identical situation, the phase boundary is the same for all n. In Table 3.4, the numbers of stability computations in the stable and unstable areas are presented for the chosen number of components. One can again observe that the mixture represents the identical physical situation for an arbitrary number of components as the numbers of stable and unstable stability computations are identical. Moreover, both algorithms identically predict the number of stable and unstable computations. Precisely, the algorithms predicted that 3170 states are stable, and 6830 are unstable. In Figure 3.10, the computation times are presented. In all figures, we show the results in the log scale on the y-axis. In Figure 3.10a, the mean time per one stability computation is presented. For a mixture with fewer components, the performance of the algorithms is similar, and the mean computation time per one stability test is around 0.005 seconds for both algorithms. However, as the number of components increases, the performance of the algorithm starts to differ. For a mixture with 30 components, the mean time for the 'classic' algorithm is 2.12×10^{-2} s, and 6.57×10^{-3} for the 'update' algorithm. The greatest difference is for the last mixture with 45 components. In this case, the mean times are 6.23×10^{-2} seconds for the 'classic', and 1.31×10^{-2} seconds for the 'update' algorithm, respectively. However, the computation time can strongly differ in the stable and unstable area because, in the unstable area, the computation is stopped when a negative value of the TPD function is found while in the stable area, all initial guesses have to be tested. Therefore, in Figure 3.10b, the mean computation time per one stability computation in the stable and unstable area is presented. The most time consuming was the 'classic' algorithm in the stable area. For the 45 component mixture, the 'classic' algorithm in the stable area needs, on average, 1.41×10^{-1} seconds for one stability computation. In contrast, the 'update' algorithm needed on average only 2.90×10^{-2} seconds for one stability computation. In the unstable area, the situation is similar. However, the computation times are shorter. For the 45 component mixture, the 'classic' algorithm in the unstable area needs, on average, 2.56×10^{-2} seconds for one stability computation. Therefore, the 'classic' algorithm in the unstable area is as fast as the 'update' algorithm in the unstable area. The 'update' algorithm needs, on average, only 5.74×10^{-3} seconds for one stability computation. In Table 3.5, we present the total computation times with the measured speed-ups. One can observe that for the 45 component mixture, the computation with the Sherman–Morrison formula is about five times faster.

Example B2: mixture N_2–CO_2–H_2S–C_i

In the second example, the algorithm is tested on seven mixtures with an increasing number of components. The mixtures are denoted as `MIXT1`–`MIXT7`. Every mixture consists of nitrogen (N_2), carbon dioxide (CO_2), hydrogen sulfide (H_2S), and a certain number of hydrocarbon components or pseudo-components. The chemical composition with the initial mole fractions of mixtures `MIXT1`–`MIXT7` is presented in Tables 3.7 and 3.8. In Appendix in Table A.1, the parameters needed in the equation of state are presented. The chemical composition was created using the

n	stable		unstable		total	
	classic	update	classic	update	classic	update
10	3170	3170	6830	6830	10000	10000
20	3170	3170	6830	6830	10000	10000
30	3170	3170	6830	6830	10000	10000
40	3170	3170	6830	6830	10000	10000
45	3170	3170	6830	6830	10000	10000

Table 3.4: Number of computations in the stable and unstable area. The numbers are identical for each mixture as they represent an identical physical situation. Example B1: mixture Y10.

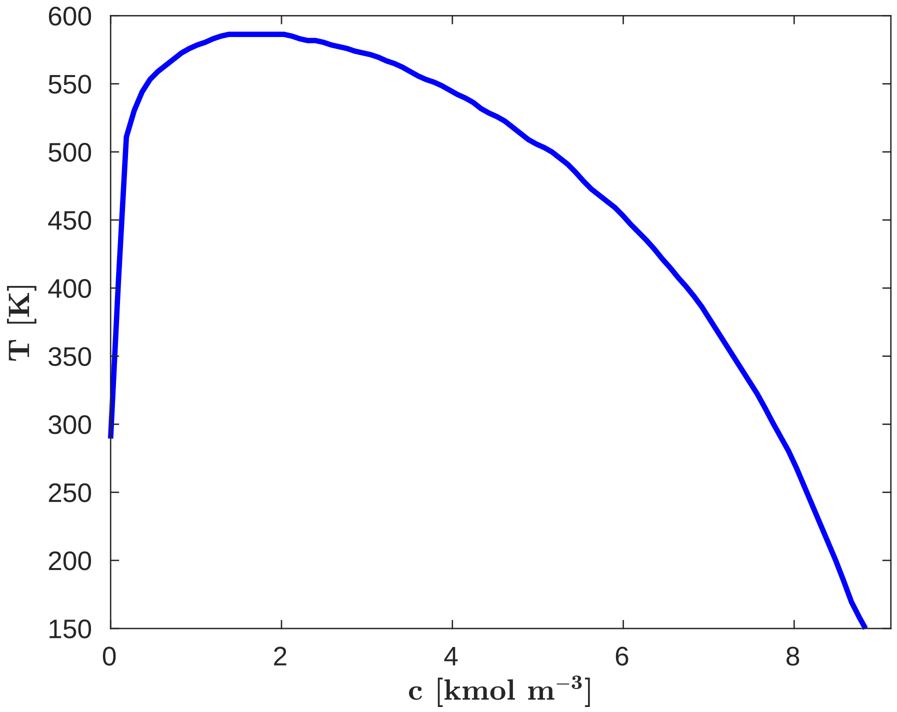
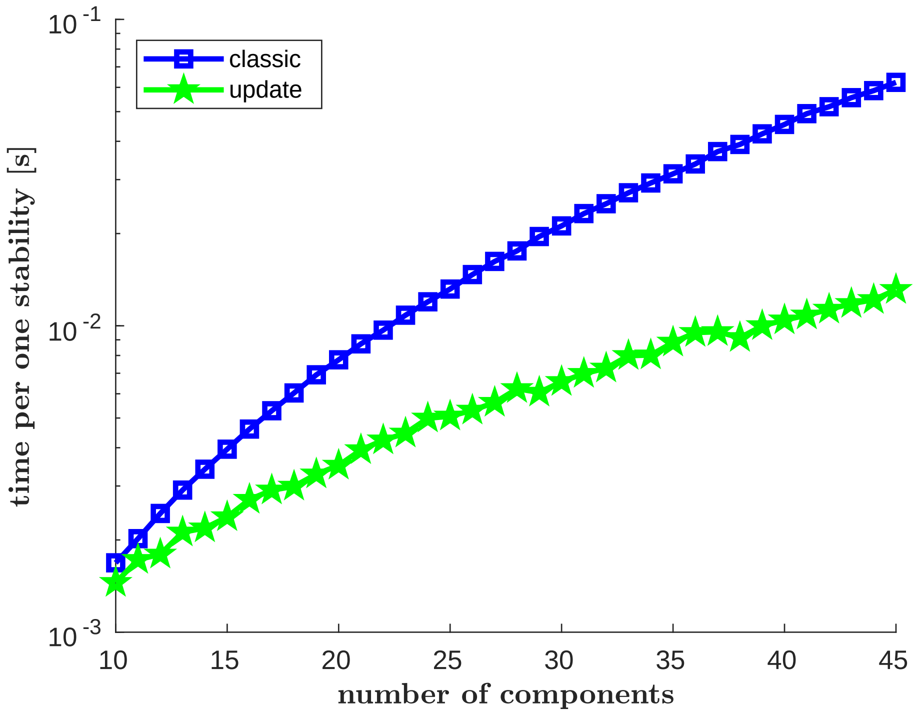


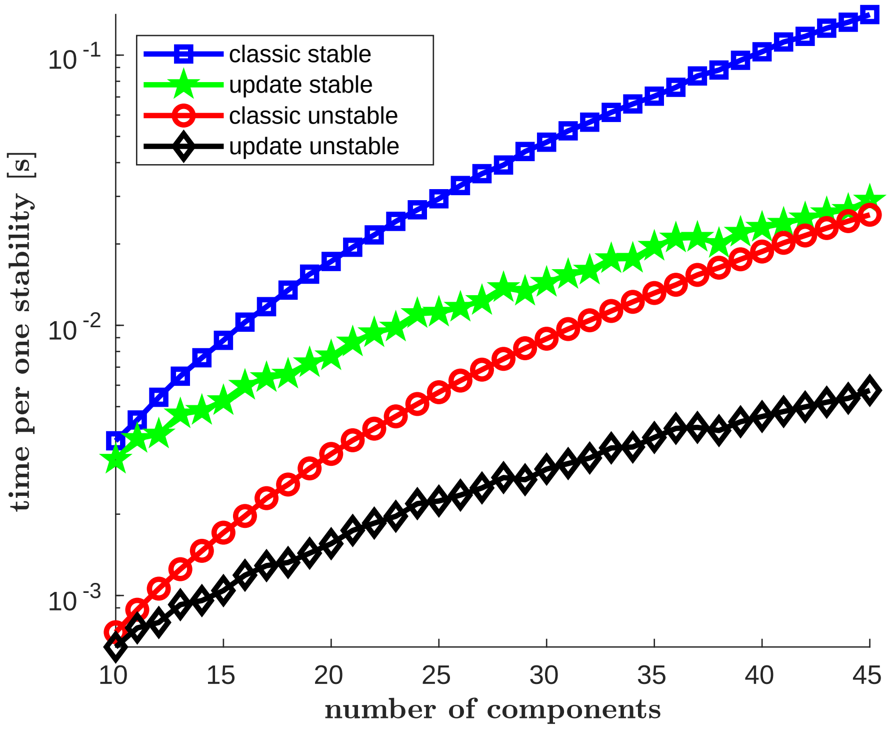
Figure 3.9: Approximate boundary between the stable and unstable area in the cT space. Below the line the states are unstable. Example B1: mixture Y10.

n	overall			stable			unstable		
	t$_c$ [s]	t$_u$ [s]	s-u [-]	t$_c$ [s]	t$_u$ [s]	s-u [-]	t$_c$ [s]	t$_u$ [s]	s-u [-]
10	16.84	14.51	**1.16**	11.85	10.11	**1.17**	4.99	4.40	**1.13**
20	77.44	35.02	**2.21**	54.62	24.40	**2.24**	22.82	10.62	**2.15**
30	211.84	65.69	**3.22**	150.92	45.63	**3.31**	60.92	20.06	**3.04**
40	454.16	104.37	**4.35**	326.10	72.98	**4.47**	128.06	31.40	**4.08**
45	623.05	131.19	**4.75**	447.87	91.96	**4.87**	175.18	39.23	**4.47**

Table 3.5: Total computation times of the 10000 stability testing using the 'classic' algorithm (t_c) and the 'update' algorithm (t_u) with the speed-up (s-u) of the computation on the chosen number of components. Example B1: mixture Y10.



(a) overall



(b) stable and unstable

Figure 3.10: Mean time per one stability computation. Example B1: mixture Y10.

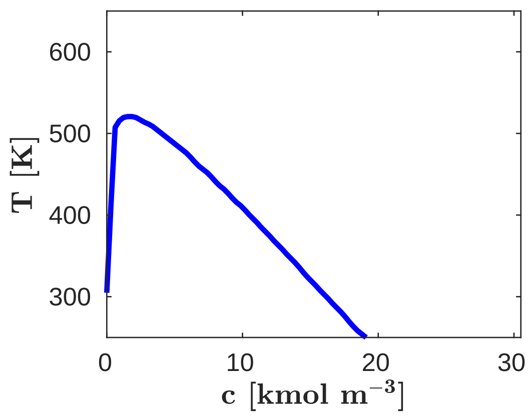
following strategy. First, the chemical composition of the MIXT7 was set. Then, the chemical composition of the MIXT6 is identical to the MIXT7 except for the molar fraction of C_{30+}, which is set to the sum of the mole fractions C_{30}–C_{40+} from the MIXT7. In the same fashion, the chemical compositions of mixtures MIXT5–MIXT1 are created. Finally, the binary interaction coefficients are defined by the following rules:

- binary interaction coefficient between N_2 and CO_2 is zero,
- binary interaction coefficient between N_2 and H_2S is zero,
- binary interaction coefficient between N_2 and any other component is 0.1,
- binary interaction coefficient between CO_2 and H_2S is zero,
- binary interaction coefficient between CO_2 and any other component is 0.15,
- binary interaction coefficient between H_2S and any other component is 0.1,
- binary interaction coefficients between C_1 and other components are presented in Table 3.6,
- any other non-listed binary interaction coefficient is zero.

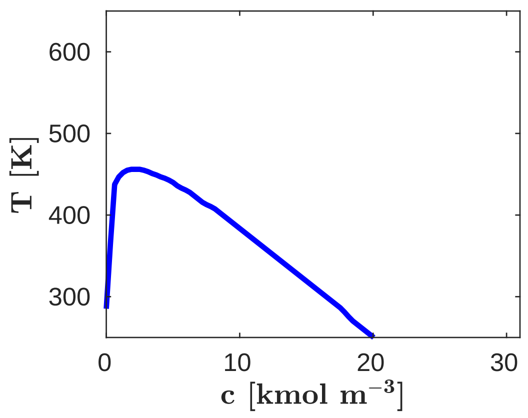
We investigated these mixtures in the cT space with a temperature range $T^* \in [250, 650]$ K and the whole feasible range of the molar density c^*. In Figure 3.11, the phase boundaries of the individual mixtures are presented. The case $n = 4$ (MIXT1) is not presented, because in the given area, all states are stable. Therefore, the phase boundary cannot be depicted. In Figure 3.11, one can observe that the phase boundaries are not identical, and the limits of the usage of the pseudo-components, which should represent higher alkanes, is given. In Table 3.9, the numbers of states in the stable and unstable area are presented. As in the first example, the numbers are identical for both algorithms. However, as previously stated, the numbers of stable or unstable stability computations differ with respect to the number of components. In Figure 3.12, the mean computation times are presented. First, in Figure 3.12a, the overall mean computation time is depicted. The curves are similar to the first example. With a lower number of components, the algorithms have similar performance. The 'classic' algorithm is faster up to $n = 12$. Then, the 'update' algorithm has better performance. Similarly to the first example, in Figure 3.12b, the mean computation times per one stability computation in the stable and unstable areas are presented. The values for $n = 4$ in the stable area are not depicted since all states are stable. The profile of the curves is similar to the first example. The highest computation times has the 'classic' algorithm in the stable area. With $n = 45$, the mean computation time of the 'classic' algorithm in the stable area is 3.08×10^{-1} seconds, in the unstable 4.96×10^{-2} seconds. In contrast, the 'update' algorithm achieves the mean computation times 8.35×10^{-2} seconds and 1.43×10^{-2} seconds in the stable and unstable areas, respectively. The precise speed-up and total computation times of the computation are presented in Table 3.10. One can observe that for the 45 component mixture, the computation with the Sherman–Morrison formula is again about 3.7 times faster compared to the standard method based on the Cholesky decomposition.

3.3.3 Branch and Bound algorithm

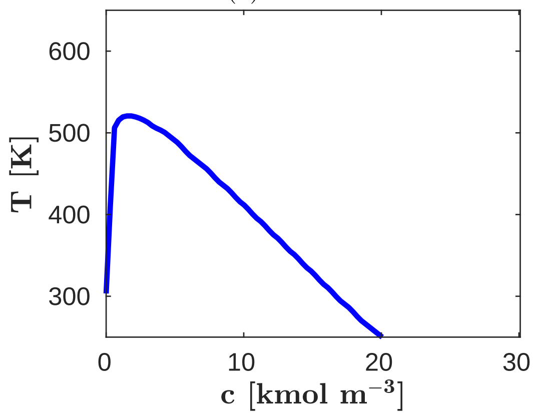
In this section, we present computation results of the Branch and Bound algorithm presented in Section 3.2.5 on examples from the literature. Furthermore, we compare the performance of the Branch and Bound algorithm using two different convex-concave splitting strategies. Both strategies were presented in Section 3.2.5. The strategy denoted as DC1 represents splitting strategy from Kou and Sun (2018), the DC2 splitting strategy represents strategy from Smejkal



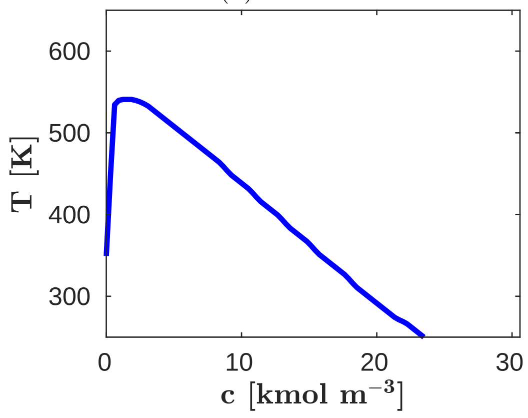
(a) $n = 8$



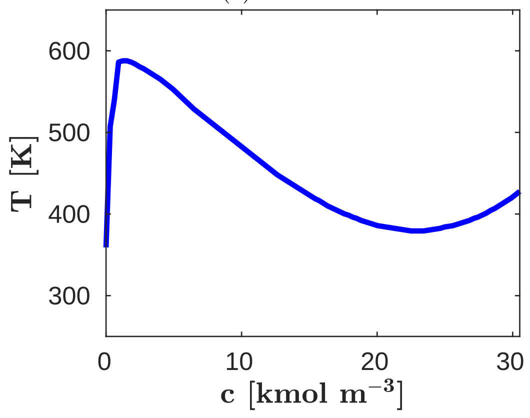
(b) $n = 12$



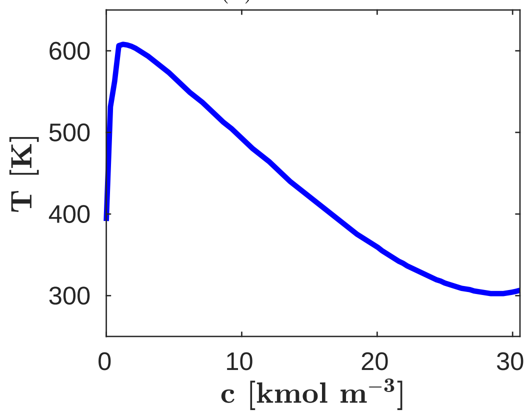
(c) $n = 17$



(d) $n = 25$

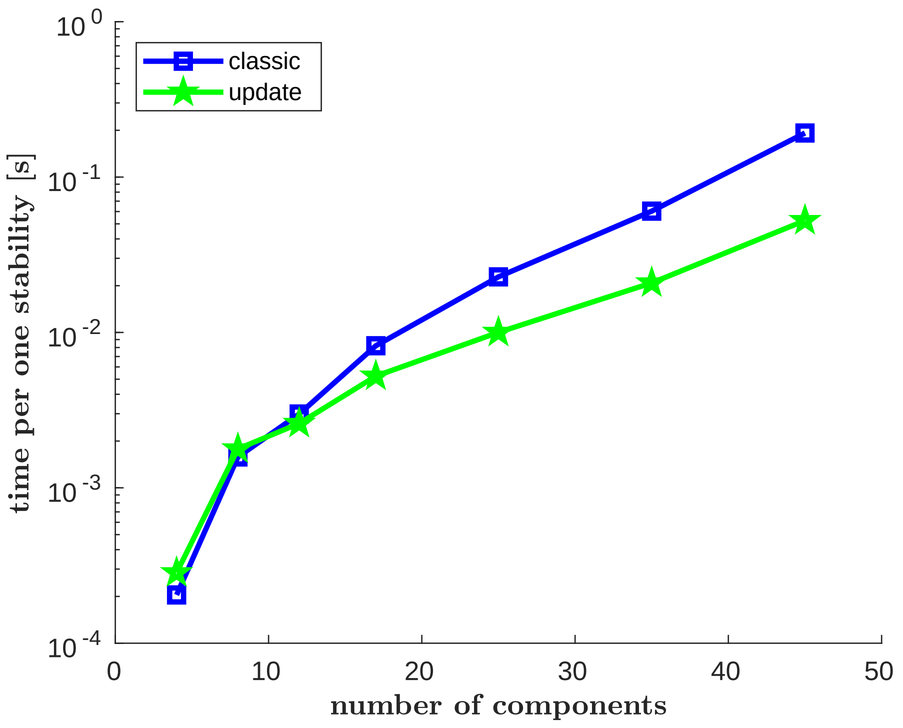


(e) $n = 35$

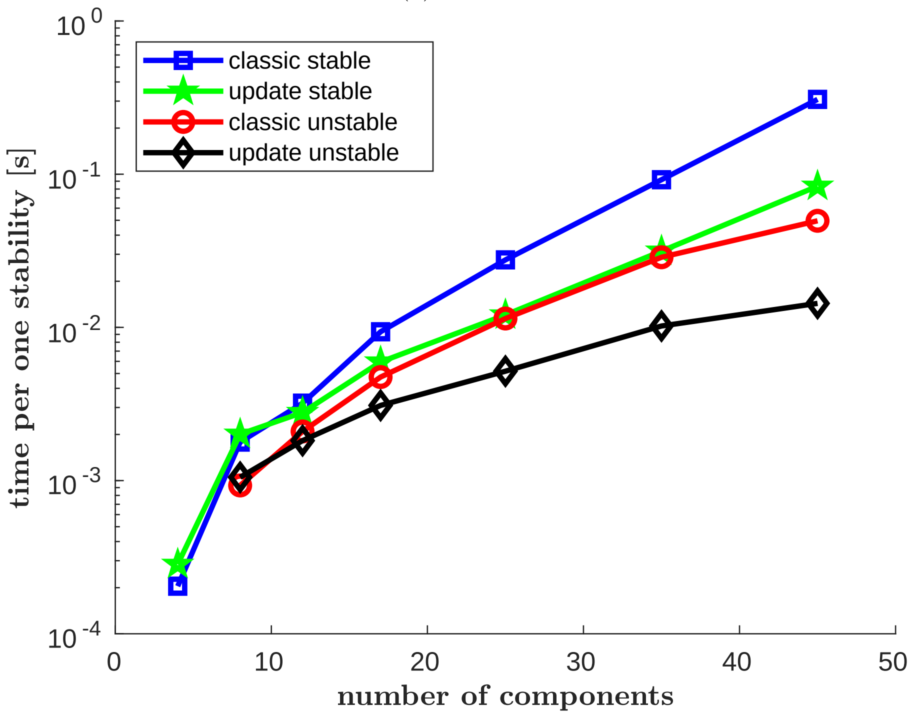


(f) $n = 45$

Figure 3.11: Approximate boundaries between the stable and unstable area in the cT space. Below the line the states are unstable. Example B2: mixture N_2–CO_2–H_2S–C_i.



(a) overall



(b) stable and unstable

Figure 3.12: Mean time per one stability computation. Example B2: mixture N_2–CO_2–H_2S–C_i.

ID	δ_{C_1-*}	ID	δ_{C_1-*}	ID	δ_{C_1-*}
N_2	0.100	C_{10}	0.052	C_{24}	0.084
CO_2	0.150	C_{11}	0.054	C_{25}	0.086
H_2S	0.100	PC_3	0.049	C_{26}	0.089
C_1	0.000	C_{12}	0.057	C_{27}	0.091
C_2	0.034	C_{12+}	0.069	C_{28}	0.093
C_3	0.036	C_{13}	0.059	C_{29}	0.096
PC_1	0.035	C_{14}	0.061	C_{30}	0.098
iC_4	0.038	C_{15}	0.064	C_{30+}	0.192
C_4	0.038	C_{16}	0.066	C_{31}	0.100
iC_5	0.041	C_{17}	0.068	C_{32}	0.102
C_5	0.041	C_{18}	0.070	C_{33}	0.105
C_6	0.043	C_{19}	0.073	C_{34}	0.107
PC_2	0.040	C_{20}	0.075	C_{35}	0.109
C_7	0.045	C_{20+}	0.083	C_{36}	0.112
C_{7+}	0.056	C_{21}	0.077	C_{37}	0.114
C_8	0.048	C_{22}	0.080	C_{38}	0.116
C_9	0.050	C_{23}	0.082	C_{39}	0.118
				C_{40+}	0.121

Table 3.6: Binary interaction coefficients between C_1 and other components. Data taken over from Firoozabadi (2016). Example B2: mixture $N_2{-}CO_2{-}H_2S{-}C_i$.

and Mikyška (2020). A computer with Intel(R) Core(TM) i7-9700K (3.60GHz) processor was used.

Example C1: mixture $C_1{-}C_3$

In the first example, we investigate a binary mixture of methane (C_1) and propane (C_3) with mole fractions $x^*_{C_1} = 0.547413$ and $x^*_{C_3} = 0.452587$. The binary interaction coefficient is $\delta_{C_1-C_3} = 0.0365$. In Figure 3.13, the minima of the function TPD in the range $T^* = [250, 330]$ K and a whole range of possible total concentrations c^* are presented. A computation grid with 51×51 points was used. Both splitting strategies give identical results, which are in agreement with results reported in Mikyška and Firoozabadi (2012). However, the numbers of iterations differ significantly. In Figure 3.14, the comparison of these two splittings is presented. If the Branch and Bound strategy uses the DC1 splitting, the mean number of iterations is 138.19, the maximum is 466, and the minimum is 22. With the DC2 splitting strategy, the number of iterations decreases greatly. The mean number of iterations is 24.9, the maximum value is 234, and the minimum is 4. Therefore, using the DC2 splitting strategy, the Branch and Bound algorithm needs approximately ten times fewer iterations. In all cases, the solution is identical to the solution found using only the local Newton–Raphson method from Section 3.2.3. In Figure 3.15, the number of iterations of the Branch and Bound strategy with an additional stopping condition `UBD` $< -10^{-4}$ is depicted, i.e., the upper bound is sufficiently negative. If this condition is satisfied the mixture is surely unstable, therefore, the numbers of iterations with and without this condition in the stable area have to be identical. For this test, only the DC2 splitting strategy is used. It can be observed that only one iteration is needed in most points of the unstable area. In Figure 3.15, also the difference in the number of iterations with and without this additional condition is presented. At best, the computation is shortened by 239 iterations,

ID	MIXT7	MIXT6	MIXT5	MIXT4	MIXT3	MIXT2	MIXT1
N_2	0.00325	0.00325	0.00325	0.00325	0.00325	0.00325	0.00325
CO_2	0.01556	0.01556	0.01556	0.01556	0.01556	0.01556	0.01556
H_2S	0.03329	0.03329	0.03329	0.03329	0.03329	0.03329	0.03329
C_1	0.82829	0.82829	0.82829	0.82829	0.82829	0.82829	0.94790
C_2	0.05850	0.05850	0.05850	0.05850	0.05850	-	-
C_3	0.02702	0.02702	0.02702	0.02702	0.02702	-	-
PC_1	-	-	-	-	-	0.08552	-
iC_4	0.00543	0.00543	0.00543	0.00543	0.00543	-	-
C_4	0.00562	0.00562	0.00562	0.00562	0.00562	-	-
iC_5	0.00248	0.00248	0.00248	0.00248	0.00248	-	-
C_5	0.00198	0.00198	0.00198	0.00198	0.00198	-	-
C_6	0.00174	0.00174	0.00174	0.00174	0.00174	-	-
PC_2	-	-	-	-	-	0.01725	-
C_7	0.00179	0.00179	0.00179	0.00179	-	-	-
C_{7+}	-	-	-	-	0.01684	-	-
C_8	0.00275	0.00275	0.00275	0.00275	-	-	-
C_9	0.00233	0.00233	0.00233	0.00233	-	-	-
C_{10}	0.00191	0.00191	0.00191	0.00191	-	-	-
C_{11}	0.00137	0.00137	0.00137	0.00137	-	-	-
PC_3	-	-	-	-	-	0.01015	-
C_{12}	0.00112	0.00112	0.00112	-	-	-	-
C_{12+}	-	-	-	0.00669	-	0.00669	-
C_{13}	0.00098	0.00098	0.00098	-	-	-	-
C_{14}	0.00087	0.00087	0.00087	-	-	-	-
C_{15}	0.00076	0.00076	0.00076	-	-	-	-

Table 3.7: Initial mole fractions (chemical composition). Example B2: mixture $N_2-CO_2-H_2S-C_i$. Part I.

on average by 23.9 iterations. In Table 3.11, the computation times are presented. Using the Branch and Bound strategy, the total computation times (i.e., 2601 stability tests) with the DC1 splitting strategy and DC2 splitting strategy are 666.38 and 104.23 seconds, respectively. With the additional stopping criteria and the DC2 splitting, the computation time is shortened to 78.96 seconds. In comparison, the stand-alone Newton–Raphson method with multiple initial approximations needs 0.85 seconds.

Example C2: mixture CO_2-C_{10}

In the second example, we investigate a binary mixture of carbon dioxide (CO_2) and normal decane (C_{10}) with mole fractions $x^*_{CO_2} = 0.547413$ and $x^*_{C_{10}} = 0.452587$. The binary interaction coefficient is $\delta_{CO_2-C_{10}} = 0.15$. In Figure 3.16, the minima of the function TPD in the range $T^* = [250, 650]$ K and a whole range of possible total concentrations c^* are presented. Again, a computation grid with 51×51 points was used. The results are in agreement with Mikyška and Firoozabadi (2012). The difference in the number of iterations between the two splitting strategies is similar to the previous example and is depicted in Figure 3.17. If the DC1 splitting strategy is used, the mean value of iterations is 160.1, the maximum value is 346, and the minimum value is 33. On the other hand, using the DC2 splitting strategy results in the mean

ID	MIXT7	MIXT6	MIXT5	MIXT4	MIXT3	MIXT2	MIXT1
C_{16}	0.00053	0.00053	0.00053	-	-	-	-
C_{17}	0.00042	0.00042	0.00042	-	-	-	-
C_{18}	0.00039	0.00039	0.00039	-	-	-	-
C_{19}	0.00031	0.00031	0.00031	-	-	-	-
C_{20}	0.00022	0.00022	-	-	-	-	-
C_{20+}	-	-	0.00131	-	-	-	-
C_{21}	0.00017	0.00017	-	-	-	-	-
C_{22}	0.00014	0.00014	-	-	-	-	-
C_{23}	0.00011	0.00011	-	-	-	-	-
C_{24}	9×10^{-5}	9×10^{-5}	-	-	-	-	-
C_{25}	7×10^{-5}	7×10^{-5}	-	-	-	-	-
C_{26}	6×10^{-5}	6×10^{-5}	-	-	-	-	-
C_{27}	5×10^{-5}	5×10^{-5}	-	-	-	-	-
C_{28}	4×10^{-5}	4×10^{-5}	-	-	-	-	-
C_{29}	3×10^{-5}	3×10^{-5}	-	-	-	-	-
C_{30}	3×10^{-5}	-	-	-	-	-	-
C_{30+}	-	0.00033	-	-	-	-	-
C_{31}	3×10^{-5}	-	-	-	-	-	-
C_{32}	3×10^{-5}	-	-	-	-	-	-
C_{33}	3×10^{-5}	-	-	-	-	-	-
C_{34}	3×10^{-5}	-	-	-	-	-	-
C_{35}	4×10^{-5}	-	-	-	-	-	-
C_{36}	2×10^{-5}	-	-	-	-	-	-
C_{37}	3×10^{-5}	-	-	-	-	-	-
C_{38}	4×10^{-5}	-	-	-	-	-	-
C_{39}	3×10^{-5}	-	-	-	-	-	-
C_{40+}	2×10^{-5}	-	-	-	-	-	-

Table 3.8: Initial mole fractions (chemical composition). Example B2: mixture $N_2-CO_2-H_2S-C_i$. Part II.

	stable		unstable		total	
n	classic	update	classic	update	classic	update
4	10000	10000	0	0	10000	10000
8	7615	7615	2385	2385	10000	10000
12	8056	8056	1944	1944	10000	10000
17	7510	7510	2490	2490	10000	10000
25	7029	7029	2971	2971	10000	10000
35	4999	4999	5001	5001	10000	10000
45	5518	5518	4482	4482	10000	10000

Table 3.9: Number of computation in the stable and unstable area. Example B2: mixture $N_2-CO_2-H_2S-C_i$.

	overall			stable			unstable		
n	t$_c$ [s]	t$_u$ [s]	s-u [-]	t$_c$ [s]	t$_u$ [s]	s-u [-]	t$_c$ [s]	t$_u$ [s]	s-u [-]
4	2.04	2.84	**0.72**	2.04	2.84	**0.72**	0.00	0.00	-
8	15.80	17.80	**0.89**	13.58	15.30	**0.89**	2.22	2.51	**0.88**
12	29.86	25.97	**1.15**	25.79	22.42	**1.15**	4.07	3.55	**1.15**
17	78.86	50.20	**1.57**	67.26	42.71	**1.57**	11.60	7.50	**1.55**
25	227.89	100.14	**2.28**	193.94	84.76	**2.29**	33.95	15.38	**2.21**
35	603.53	208.39	**2.90**	460.06	157.32	**2.92**	143.47	51.07	**2.81**
45	1921.31	525.02	**3.66**	1698.83	460.71	**3.69**	222.48	64.31	**3.46**

Table 3.10: Total computation times of the 10000 stability testing using the 'classic' algorithm (t_c) and the 'update' algorithm (t_u) with the speed-up (s-u) of the computation. Example B2: mixture $N_2{-}CO_2{-}H_2S{-}C_i$.

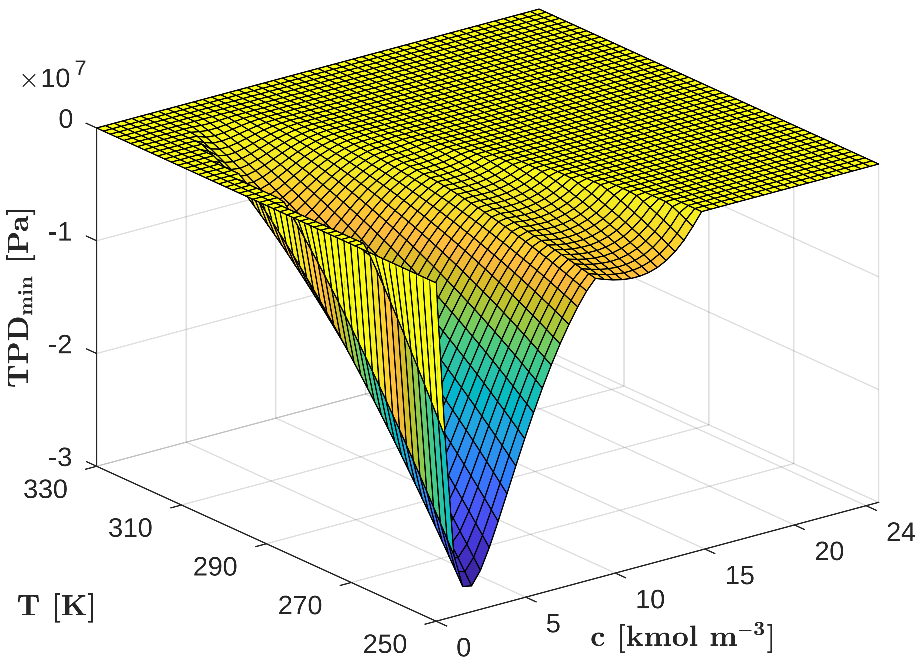
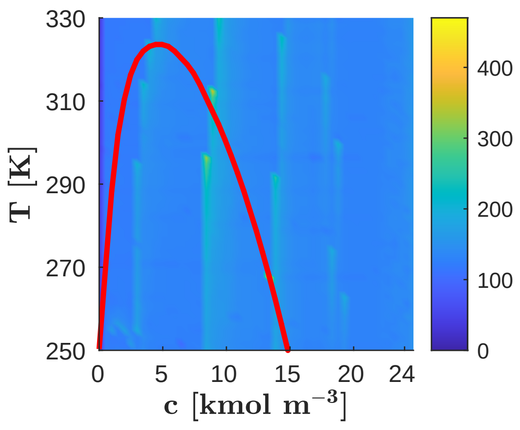


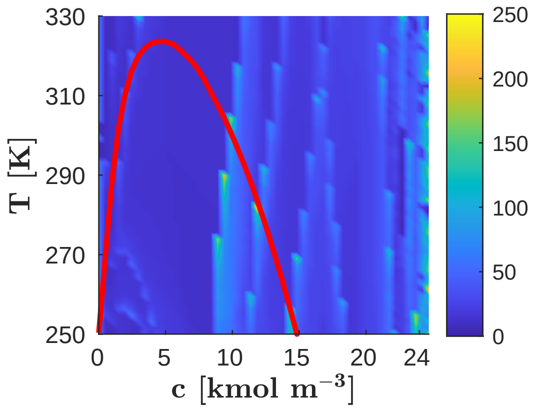
Figure 3.13: The global minima of the TPD function in the cT-space. Example C1: mixture $C_1{-}C_3$.

	Example C1		Example C2		Example C3	
setting	time [s]	mean [s]	time [s]	mean [s]	time [s]	mean [s]
(a)	666.38	0.256	815.85	0.314	33 039.35	12.70
(b)	104.23	0.04	95.41	0.037	21 001.27	8.07
(c)	78.96	0.03	61.12	0.023	12 715.31	4.89
(d)	0.85	3.25×10^{-4}	1.10	4.23×10^{-4}	2.45	9.43×10^{-4}

Table 3.11: Total computation times and mean time per one stability testing in all examples using different settings. Setting (a) - the Branch and Bound strategy with the DC1 splitting, Setting (b) - the Branch and Bound strategy with the DC2 splitting, Setting (c) - the Branch and Bound strategy with the DC2 splitting and with an additional stopping condition `UBD` $< -10^{-4}$, Setting (d) - the stand-alone modified Newton–Raphson from Section 3.2.3. Examples C1–C3.

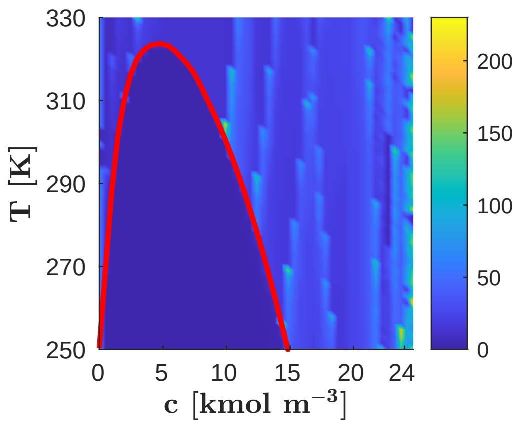


(a) DC1 splitting

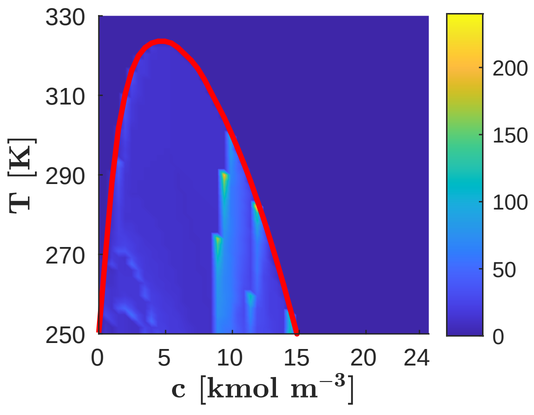


(b) DC2 splitting

Figure 3.14: The number of iterations of the Branch and Bound strategy using different convex-concave splitting strategies of the objective function. The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example C1: mixture C_1–C_3.



(a) number of iterations



(b) difference

Figure 3.15: The number of iterations of the Branch and Bound strategy with an additional stopping condition `UBD` $< -10^{-4}$ using the DC2 splitting (left) and the difference in iterations of the Branch and Bound strategy without this condition (right). The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example C1: mixture C_1–C_3.

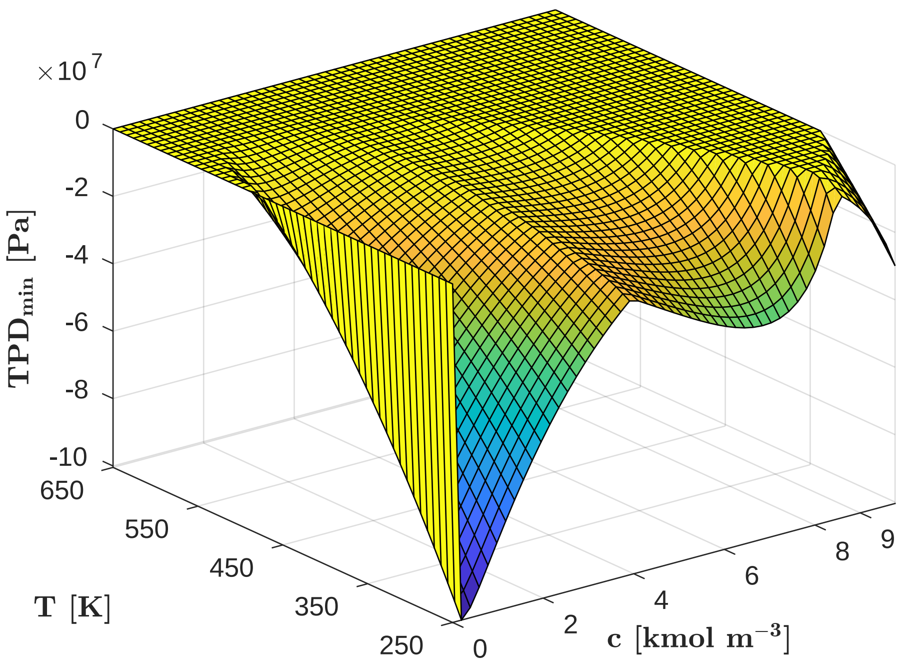


Figure 3.16: The global minima of the TPD function in the cT-space. Example C2: mixture CO_2-C_{10}.

value of 29.1 iterations, the maximum value is 395, and the minimum value is 4. Therefore, the DC2 splitting strategy is about five times faster. In Figure 3.17, the phase envelope representing the phase boundary is also depicted. As in the previous example, we compare results with the modified Newton–Raphson method from Section 3.2.3. In Figure 3.18, the difference between found minima are presented. In one case, the results differ. If $T^* = 314$ K and $c^* = 9573.82$ mol m^{-3}, the Newton–Raphson method found a minimum equal to zero and declared the state as stable. However, the global method was able to find a minimum with TPD $= -2345570$, and therefore the mixture is unstable. A similar problem was reported by Nichita (2018a) that the initial approximations based on Wilson correlation (see Section 2.5.4) used in local Newton methods led to wrongly detected single-phase state. The correct solution (liquid-liquid equilibrium) was obtained by considering the incipient phase as being almost pure CO_2. Our method does not suffer from these issues and will detect phase instability in both vapour-liquid or liquid-liquid equilibrium problems. Furthermore, in Figure 3.19, the number of iterations with an addition stopping condition `UBD` $< -10^{-4}$ is presented. Similarly to the previous example, in most cases, only one iteration of the Branch and Bound strategy is needed in the unstable area. At best, the computation is shortened by 176 iterations, on average by 22.3 iterations. In Table 3.11, the computation times are presented. Using the Branch and Bound strategy, the total computation times (i.e., 2601 stability tests) with the DC1 splitting strategy and DC2 splitting strategy are 815.85 and 95.41 seconds, respectively. In comparison with the first example, the computation time of the Branch and Bound strategy with the DC2 splitting is comparable. However, the computation time of the Branch and Bound strategy with DC1 splitting strategy is approximately 200 seconds higher than the computation time from first example. Moreover, the computation time reduction with the additional stopping criteria is comparable with the first example. The computation time with the DC2 splitting strategy is shortened to 61.12 seconds, i.e., by approximately 30 seconds. The computation time of the stand-alone Newton–Raphson method with multiple initial approximations is 1.10 seconds.

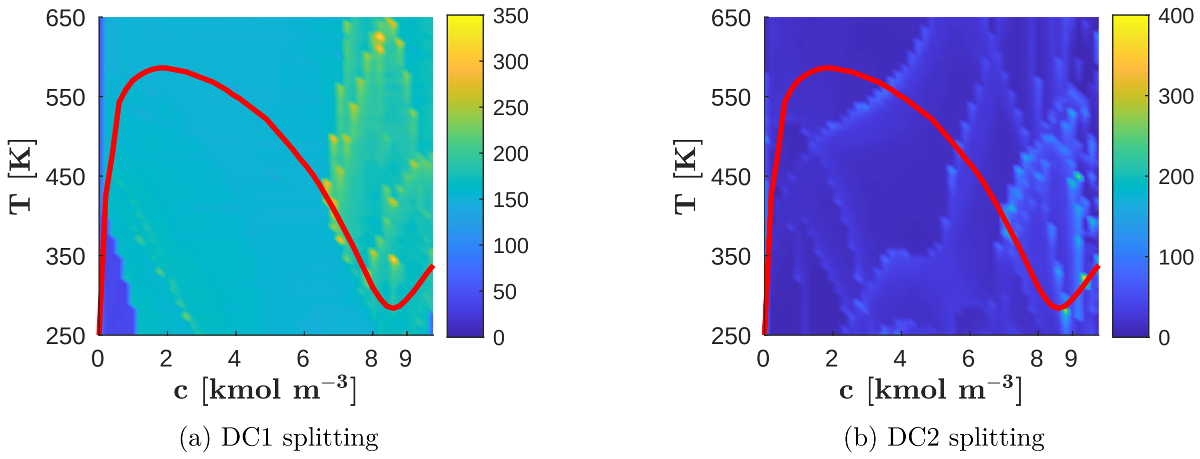


Figure 3.17: The number of iterations of the Branch and Bound strategy using different convex-concave splitting strategies of the objective function. The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example C2: mixture CO_2–C_{10}.

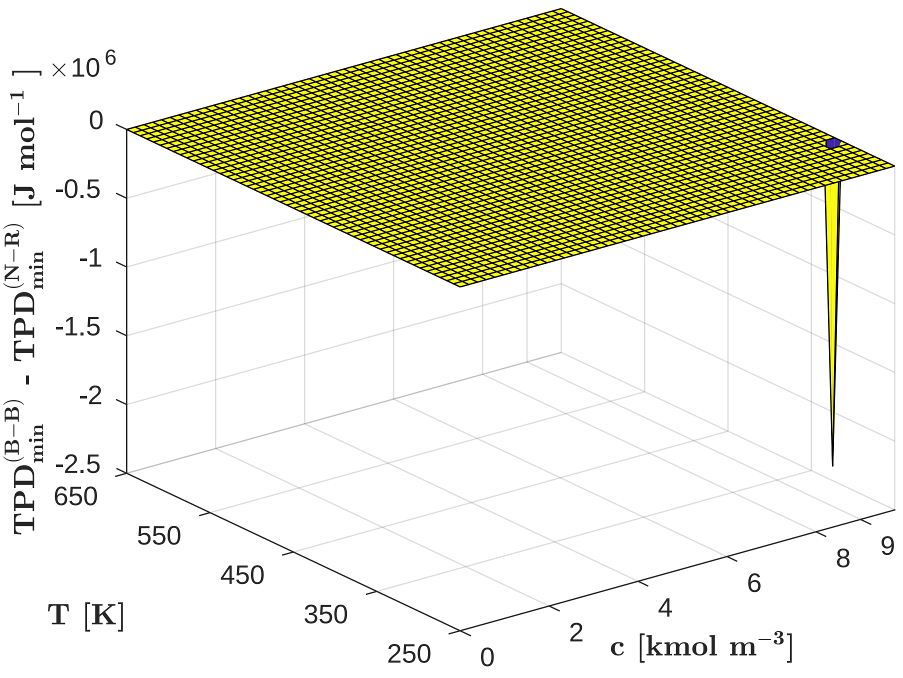
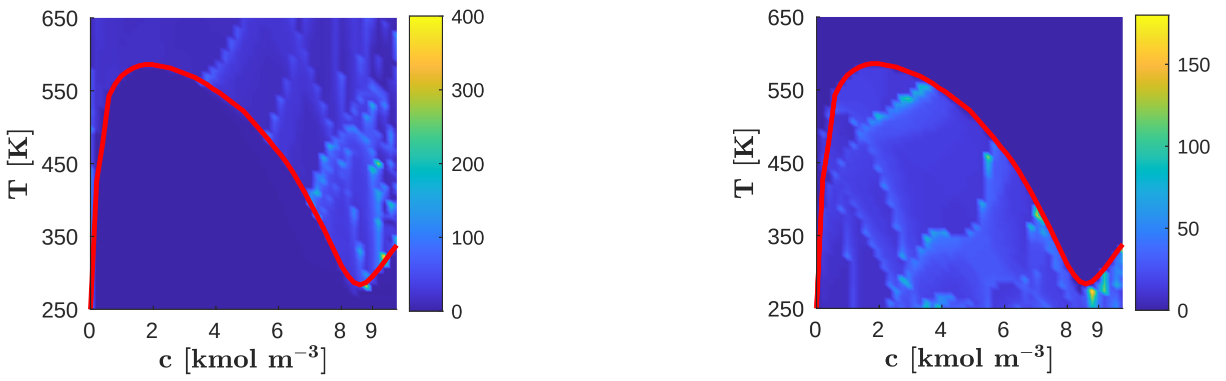


Figure 3.18: The differences between the found minima using the Branch and Bound strategy and the Newton–Raphson method from Section 3.2.3. Example C2: mixture CO_2–C_{10}.



(a) number of iterations (b) difference

Figure 3.19: The number of iterations of the Branch and Bound strategy with an additional stopping condition `UBD` $< -10^{-4}$ using the DC2 splitting (left) and the difference in iterations of the Branch and Bound strategy without this condition (right). The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example C2: mixture CO_2–C_{10}.

Example C3: mixture N_2–C_1–C_3–C_{10}

In the third example, we investigate a four-component mixture of nitrogen (N_2), methane (C_1), propane (C_3), and normal decane (C_{10}) with mole fractions $x^*_{N_2} = 0.2463$, $z^*_{C_1} = 0.2208$, $z^*_{C_3} = 0.2208$, $z^*_{C_{10}} = 0.3121$. The binary interaction coefficients are presented in Table 3.12. In Figure 3.20, the minima of the function TPD in the range $T^* = [250, 650]$ K and a whole range of possible total concentrations c^* are presented. Again, a computation grid with 51×51 points was used. The results are in agreement with Mikyška and Firoozabadi (2012). Both splitting strategies converged towards the same solutions. In Figure 3.21, the needed iterations are provided. It can be observed that the number of iterations increases more than 10 times in comparison to previous binary mixtures examples.

Using the DC1 splitting strategy, the Branch and Bound strategy needs on average 992 iterations. In almost every case (97.7 %), the strategy needs the maximum number of iterations (1000). The number of iterations with the DC2 splitting strategy decreases a little. However, the strategy needs maximum number of iterations in 64 % cases, on average 883.2 iterations.Therefore, the computation time for the four component mixtures is significantly higher. Consequently, for more component mixtures, the number of iterations without the stopping criteria 3 (maximum number of iterations) will be even higher. Comparing our method with the modified Newton–Raphson method from Section 3.2.3 results in no differences. In all cases, both methods found identical solutions. In the last figure, the number of iterations with an additional condition `UBD` $< -10^{-4}$ is presented. In Figure 3.22a, the number of iterations with this condition is depicted. In Figure 3.22b, the difference in the number of iteration is presented. Similarly to the previous examples, in most cases, only one iteration of the Branch and Bound strategy is needed in the unstable area. In the proximity of the phase boundary, two iterations are needed in the unstable area. Concerning the difference in the number of iterations, the decrease is more significant than in the previous examples. In more than one case, the number of iterations decreases from the maximum number of iterations to only one. On average, the number of iterations decreases by 808.9 iterations in the unstable area. In Table 3.11, the computation times are presented. Using the Branch and Bound strategy, the total computation times (i.e., 2601 stability tests) with the DC1 splitting strategy and DC2 splitting strategy are 33039.35

	N_2	C_1	C_3	C_{10}
N_2	0	0.100	0.100	0.100
C_1	0.100	0	0.036	0.052
C_3	0.100	0.036	0	0
C_{10}	0.100	0.052	0	0

Table 3.12: The binary interaction coefficients. Data taken over from Mikyška and Firoozabadi (2012). Example C3: mixture N_2–C_1–C_3–C_{10}.

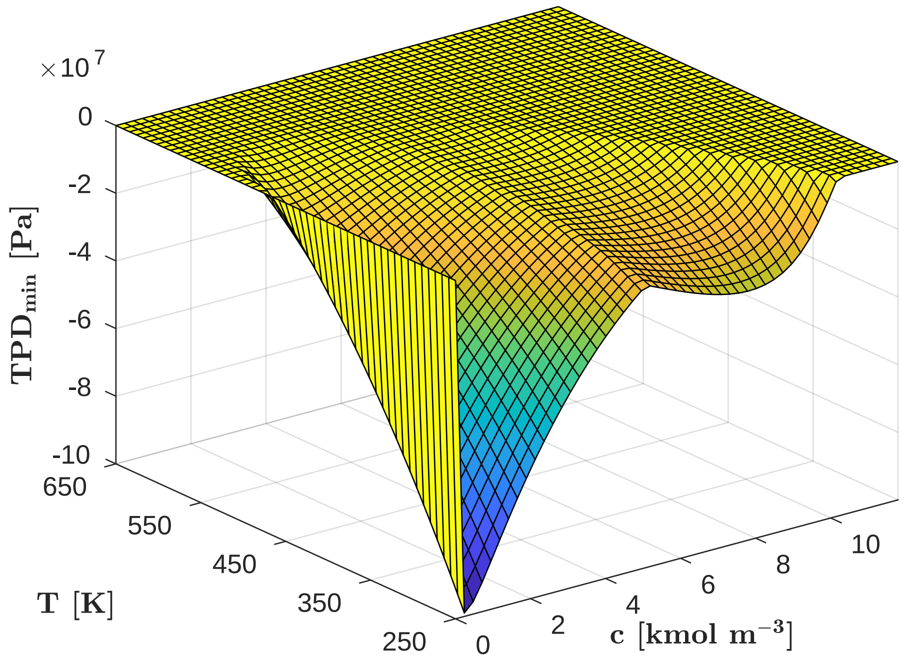
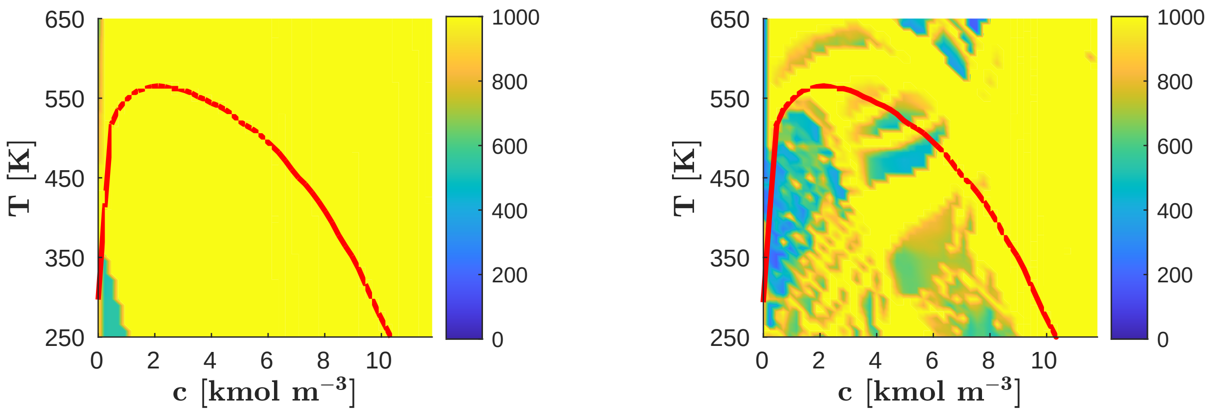


Figure 3.20: The global minima of the TPD function in the cT-space. Example C3: mixture N_2–C_1–C_3–C_{10}.

and 21001.27 seconds (approximately 9 and 6 hours), respectively. Therefore, the computation time for more than two-component mixture rises significantly. With 2601 stability computations, the mean value for one stability test is 12.70 seconds for the DC1 splitting strategy and 8.07 for the DC2 splitting strategy. With an additional stopping criteria `UBD` $< -10^{-4}$ and the DC2 splitting strategy, the computation time can be shortened to 12715.31 seconds. In comparison, the computation time of the stand-alone modified Newton–Raphson method with multiple initial approximations is 2.45 seconds.

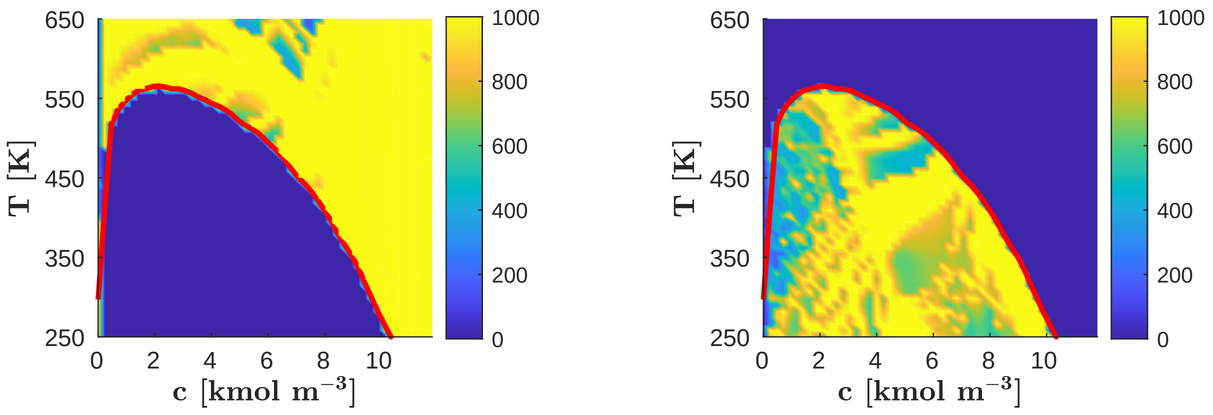
3.3.4 Modern heuristics on solving the phase stability testing problem

In this section, we present numerical results of the heuristical algorithms presented in Section 3.2.6. First, the Newton–Raphson method with line-search was used to compute the solution $\mathbf{x}^{(\mathrm{opt})}$. In all cases, the found minimum $\mathbf{x}^{(\mathrm{opt})}$ was the correct global minimum. This was proven by the global optimization method based on the Branch and Bound strategy from Section 3.2.5. Having the correct solution at hand, the evolution algorithms are stopped with success if the



(a) DC1 splitting (b) DC2 splitting

Figure 3.21: The number of iterations of the Branch and Bound strategy using different convex-concave splitting strategies of the objective function. The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example C3: mixture N_2–C_1–C_3–C_{10}.



(a) number of iterations (b) difference

Figure 3.22: The number of iterations of the Branch and Bound strategy with an additional stopping condition `UBD` $< -10^{-4}$ using the DC2 splitting (left) and the difference in iterations of the Branch and Bound strategy without this condition (right). The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example C3: mixture N_2–C_1–C_3–C_{10}.

found solution $\mathbf{y}$ satisfies

$$\left\|\mathbf{y}-\mathbf{x}^{(\mathrm{opt})}\right\|_{pn} < 10^{-5} \text{ or } \frac{\left|\mathrm{TPD}(\mathbf{y};\mathbf{x}^{*})-\mathrm{TPD}(\mathbf{x}^{(\mathrm{opt})};\mathbf{x}^{*})\right|}{\left|\mathrm{TPD}(\mathbf{x}^{(\mathrm{opt})};\mathbf{x}^{*})\right|+1} < 10^{-5}, \tag{3.226}$$

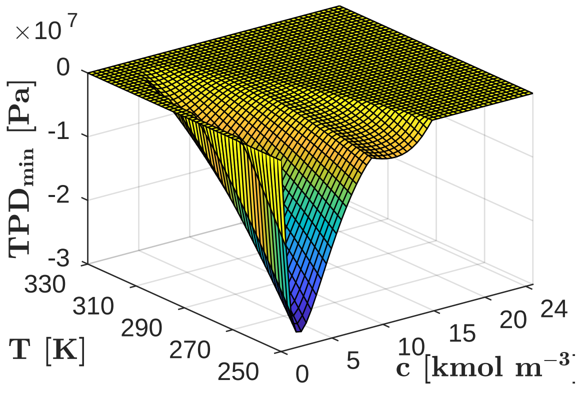
where the $\|\cdot\|_{pn}$ in the VTN-specification is defined by equation (3.84). The maximum number of function evaluations for each evolution algorithm was set to $FE_{max} = 5 \times 10^4$, and each algorithm was run 100 times. Physical parameters of the equation of state of all used components are presented in Appendix in Table A.1. A computer with Intel(R) Core(TM) i7-8700 (3.20 GHz) processor was used.

Example D1: mixture C_1–C_3

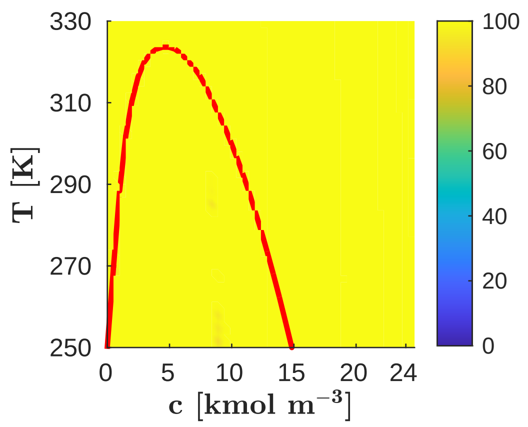
In the first example, we investigate a binary mixture of methane (C_1) and propane (C_3) with mole fractions $x^*_{C_1} = 0.547413$, $x^*_{C_3} = 0.452587$, temperature in range $T^* \in [250, 350]$ K, and a whole range of possible total concentrations c^*. The binary interaction coefficient is $\delta_{C_1-C_3} = 0.0365$. In Figure 3.23a, the minima of the TPD function in the cT space are presented. A grid with 51×51 points was used, therefore, a total of 2601 phase stability testing computations (minimization) were performed. In Figure 3.23a, the red line represents the phase boundary. Above this line, the state is stable, and the global minimum has value TPD = 0. In Figures 3.23b–3.23f, the numbers of successful runs for each evolution algorithm are presented. The most successful algorithm was the Differential Evolution, which found the correct solution in almost all cases. The CMA-ES correctly found solutions in most of the cases. However, the algorithm had problems at the phase boundary and in an area with the total concentration $c^* = 10^4$ mol m^{-3}. In this area, the other heuristics also had problems finding the correct solution. The Cuckoo Search was able to find the correct solution almost everywhere in the unstable area. However, in the stable area, the algorithm was able to find the correct solution in one part of the stable area and only with a probability of success around 70 %. The performance of the last two algorithms (HS and EHO) was not that satisfactory. Neither of them finds the correct solution in the stable area, and in the unstable area, they find the correct solution at best at 20 % of the runs. In our opinion, this was caused by the value of the maximum number of iteration as the HS and EHO algorithms have slower convergence.

Example D2: mixture N_2–CO_2–C_1–PC_i

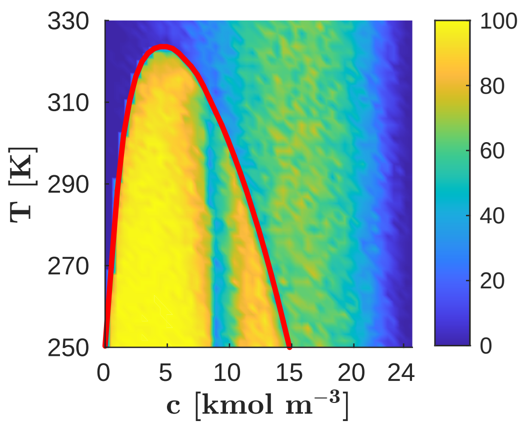
In the second example, we investigate a seven component mixture of nitrogen (N_2), carbon dioxide (CO_2), methane (C_1), and four hydrocarbon pseudo-components denoted as PC_1, PC_2, PC_3, and C_{12+} with mole fractions $x^*_{N_2} = 0.466905$, $x^*_{CO_2} = 0.007466$, $x^*_{C_1} = 0.300435$, $x^*_{PC_1} = 0.105051$, $x^*_{PC_2} = 0.041061$, $x^*_{PC_3} = 0.045060$, and $x^*_{C_{12+}} = 0.034021$. The binary interaction coefficients between all components are presented in Table 3.13. We investigate the phase stability in temperature range $T^* \in [250, 650]$ K and the whole range of feasible total concentrations c^*. In Figure 3.24a, the minima of the TPD function in the cT space are presented. As in Example D1, a grid with 51×51 points was used, which resulted in 2601 phase stability computations. In Figures 3.24b–3.24f, the numbers of successful runs for each evolution algorithm are presented. Similarly to the first example, the Harmony Search, Cuckoo Search, and Elephant Herding Optimization did not perform satisfactorily. The Differential Evolution and CMA-ES find the correct solution in most cases. However, both algorithms had problems in some areas. The Differential Evolution encountered problems when the total concentration c^* was higher than 1.7×10^4 mol m^{-3}, the CMA-ES algorithm if the temperature was lower than 300 K and the total concentration was approximately 1.5×10^4 mol m^{-3}.



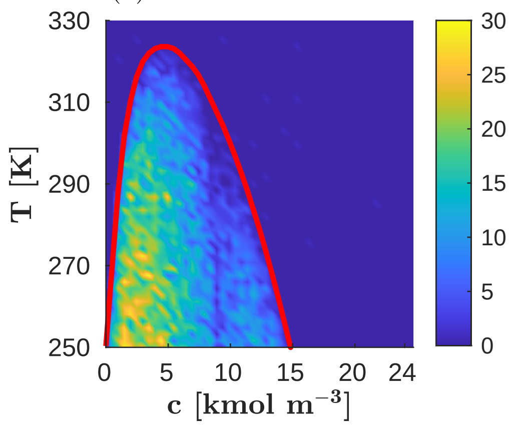
(a) global minima of the TPD



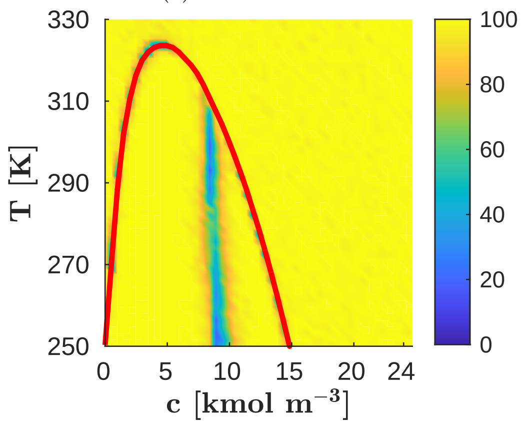
(b) Differential Evolution



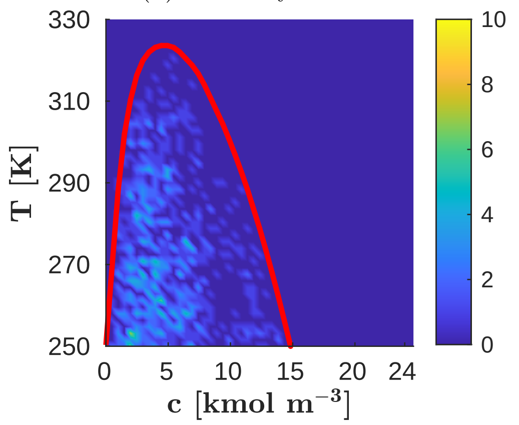
(c) Cuckoo Search



(d) Harmony Search

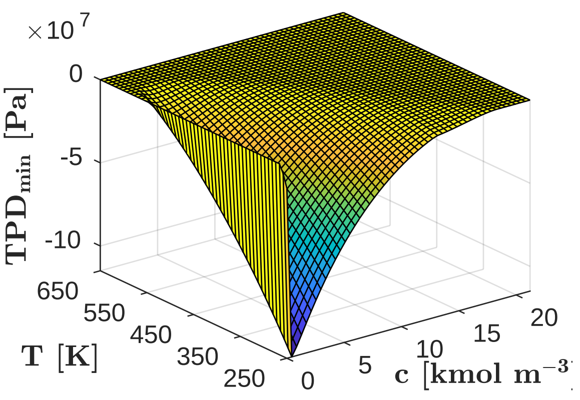


(e) CMAES

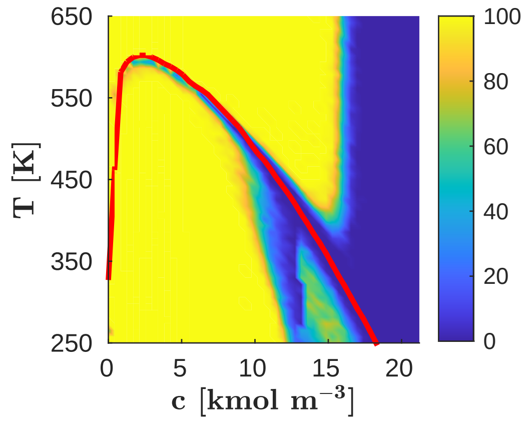


(f) Elephant Herding Optimization

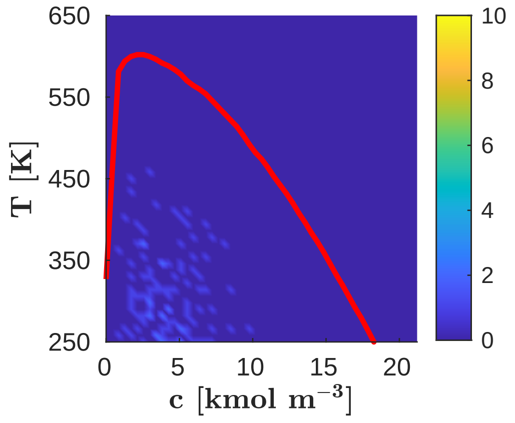
Figure 3.23: Global minima of the TPD function in the cT space and the number of successful runs of each evolution heuristics, i.e., when the given heuristic found the correct global minimum. The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example D1: mixture C_1–C_3.



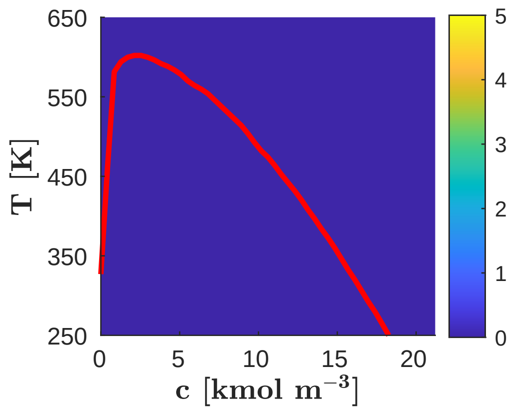
(a) global minima of the TPD



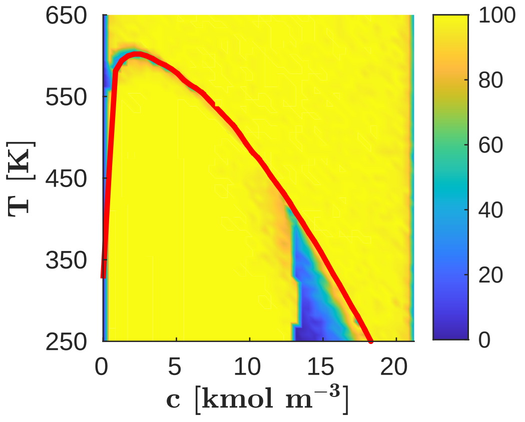
(b) Differential Evolution



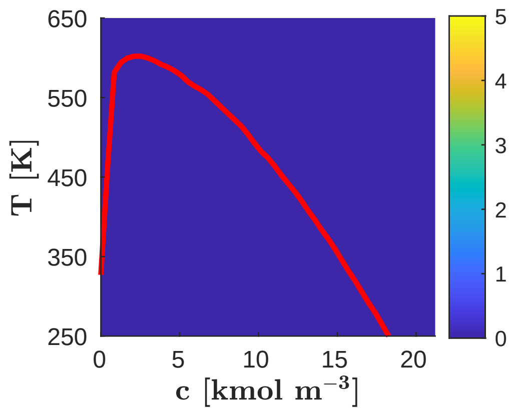
(c) Cuckoo Search



(d) Harmony Search



(e) CMAES



(f) Elephant Herding Optimization

Figure 3.24: Global minima of the TPD function in the cT space and the number of successful runs of each evolution heuristics, i.e., when the given heuristic found the correct global minimum. The red line represents the approximate boundary between the stable and unstable area (below the line the states are unstable). Example D2: mixture $\mathrm{N_2{-}CO_2{-}C_1{-}PC_i}$.

	N_2	CO_2	C_1	PC_1	PC_2	PC_3	C_{12+}
N_2	0	0	0.100	0.100	0.100	0.100	0.100
CO_2	0	0	0.150	0.150	0.150	0.150	0.150
C_1	0.100	0.150	0	0.035	0.040	0.049	0.069
PC_1	0.100	0.150	0.035	0	0	0	0
PC_2	0.100	0.150	0.040	0	0	0	0
PC_3	0.100	0.150	0.049	0	0	0	0
C_{12+}	0.100	0.150	0.069	0	0	0	0

Table 3.13: The binary interaction coefficients between all components. Data taken over from Mikyška and Firoozabadi (2012). Example D2: mixture $N_2{-}CO_2{-}C_1{-}PC_i$.

algorithm	Example D1	Example D2
Newton–Raphson	0.99	11.55
Differential Evolution	35.87	995.03
Cuckoo Search	78.55	394.63
Harmony Search	210.72	862.48
CMA-ES	26.48	408.91
Elephant Herding Optimization	500.30	1777.72

Table 3.14: Computation times in seconds for Examples D1–D2.

Comparison using the Wilcoxon test

To compare the evolution strategies, the Wilcoxon signed-rank test (see Kruskal (1957)) was performed with the criterion based on Auger and Hansen (2005); Kukal and Mojzeš (2018)

$$CRIT = \left(\frac{1-p_s}{p_s}\right) FE_{\max} + \mathbb{E}(n_e), \tag{3.227}$$

where p_s is the probability of the success, $FE_{\max}$ is the maximum number of the function evaluations, and $\mathbb{E}(n_e)$ is the mean value of the function evaluation of the successful runs. This criterion is calculated for both previous examples, therefore, we have vectors of size 5202 for each evolution algorithm. Then, a pairwise Wilcoxon test was performed using Matlab software. All hypotheses about the median equality have been rejected on the significance level $\alpha = 0.05$. In all ten tests, the calculated p-values have been lower than 10^{-9}. In addition, we obtained the same results with the Bonferroni correction (see Dunn (1961)). Moreover, in Table 3.14, the computation times for all algorithms, including the Newton–Raphson method, are presented. The computation times of the evolution algorithms are the mean value for the 100 runs. From this point of view, the Newton–Raphson method is the best method as the computation time is about twenty times faster than the fastest evolution heuristic. However, as the computations are stopped when the correct solution has been found, the Harmony Search and Elephant Herding Optimization algorithms have to do more iterations (evaluations) than more successful algorithms (Differential Evolution or CMA-ES). Therefore, the comparison of the computation times of Harmony Search or Elephant Herding Optimization with Differential Evolution or CMA-ES is not legitimate. The comparison of Differential Evolution or CMA-ES with the Newton–Raphson method is valid as all algorithms found the correct solution in most cases.

3.3.5 Comparison of SSI method with Newton–Raphson

In this section, we provide a critical comparison of the SSI method with the Newton–Raphson method. First, we investigate the PTN-specification, where the SSI method is a standard. Second, the VTN- and UVN-specification will be studied. In these specifications, the SSI method is not commonly used. A computer with Intel(R) Core(TM) i7-9700K (3.60GHz) processor was used.

Example E1: mixture Y14

First, we investigate the SSI iteration in the PTN-specification. In this example, we test the algorithm on a 14 component mixture from Hoteit and Firoozabadi (2006b). The mixture consists of nitrogen (N_2), carbon dioxide (CO_2), methane (C_1), ethane (C_2), propane (C_3), iso-butane (iC_4), butane (C_4), iso-pentane (iC_5), pentane (C_5), and five pseudo-components denoted $PC_{6\text{-}9}$, $PC_{10\text{-}14}$, $PC_{15\text{-}19}$, $PC_{20\text{-}24}$, C_{25+}. The initial mole fractions are $x^*_{N_2} = 0.0019$, $x^*_{CO_2} = 0.0101$, $x^*_{C_1} = 0.8649$, $x^*_{C_2} = 0.0248$, $x^*_{C_3} = 0.0128$, $x^*_{iC_4} = 0.0072$, $x^*_{C_4} = 0.0037$, $x^*_{iC_5} = 0.0022$, $x^*_{C_5} = 0.0014$, $x^*_{PC_{6-9}} = 0.009978$, $x^*_{PC_{10-14}} = 0.012590$, $x^*_{PC_{15-19}} = 0.012321$, $x^*_{PC_{20-24}} = 0.009024$, $x^*_{C_{25+}} = 0.027087$. We investigate the phase stability in the PT space with $P^* \in [5, 2400]$ bar, and $T^* \in [300, 800]$ K. The computation domain was discretised with 100×100 points, i.e., total 10000 phase stability computation were performed. In Figure 3.25, the minima of TPD function found using the Newton–Raphson method in the PT space are depicted. In this example, we test four settings. First, we run the Newton–Raphson method presented in Section 3.2.3. Then, the SSI method from Section 3.2.1 is used. In Figure 3.26, the phase boundaries are presented. One can observe that the SSI method does not correctly detect all unstable states. Therefore, we run the SSI method in two other settings. First, we used more initial approximations. Following Li and Firoozabadi (2012), we used $n + 4$ initial approximations. The phase boundary with this setting is shown in Figure 3.26c. Even with more initial approximations, the SSI method does not find every unstable state as the Newton–Raphson method. The last setting we tested is the combination of SSI with the Newton–Raphson method. First, the SSI method is used. When the norm of the increment of the solution is less than 10^{-4}, the computation is switched to the Newton–Raphson method. The phase boundary for this setting is depicted in Figure 3.26d. Using this setting, we observe identical phase boundary as with the Newton–Raphson method. However, this setting occasionally had problems with the convergence of the Newton–Raphson method. In some cases, the resulting state of the SSI iterations leads to an ill-condition Hessian matrix of the TPD function. Lastly, we compare the methods based on the computation times, which are presented in Table 3.15. In each setting, the computation was stopped when a negative value of the TPD function was found. Therefore, in the unstable area, not all initial approximations were used. On the other hand, in the stable area, all initial approximations have to be used. According to Table 3.15, the fastest is the SSI method. However, this method does not find the correct solution in all cases. The computation times of the Newton–Raphson method and SSI+Newton–Raphson method are comparable. However, since we experience problems with initial guesses from SSI iteration to the Newton–Raphson method, the Newton–Raphson method is preferable in our opinion.

Example E2: mixture Y10

Next, we tested the SSI method in the VTN-specification. In this example, mixture Y10 is used, which has been already studied in Section 3.3.2. The mixture Y10 is a ten component mixture consisting of normal alkanes (denoted as C_i) from methane C_1 to octane C_8, decane C_{10}, and tetradecane C_{14}. The initial mole fractions are $x^*_{C_1} = 0.35$, $x^*_{C_2} = 0.03$, $x^*_{C_3} = 0.04$, $x^*_{C_4} = 0.06$, $x^*_{C_5} = 0.04$, $x^*_{C_6} = 0.03$, $x^*_{C_7} = 0.05$, $x^*_{C_8} = 0.05$, $x^*_{C_{10}} = 0.30$, and $x^*_{C_{14}} = 0.05$. The only non-zero

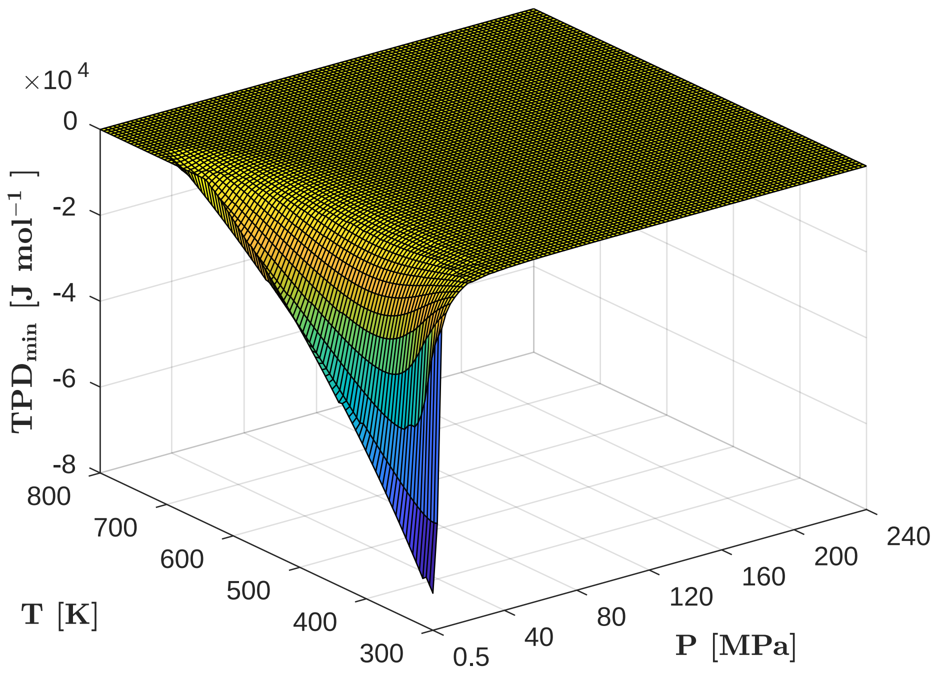
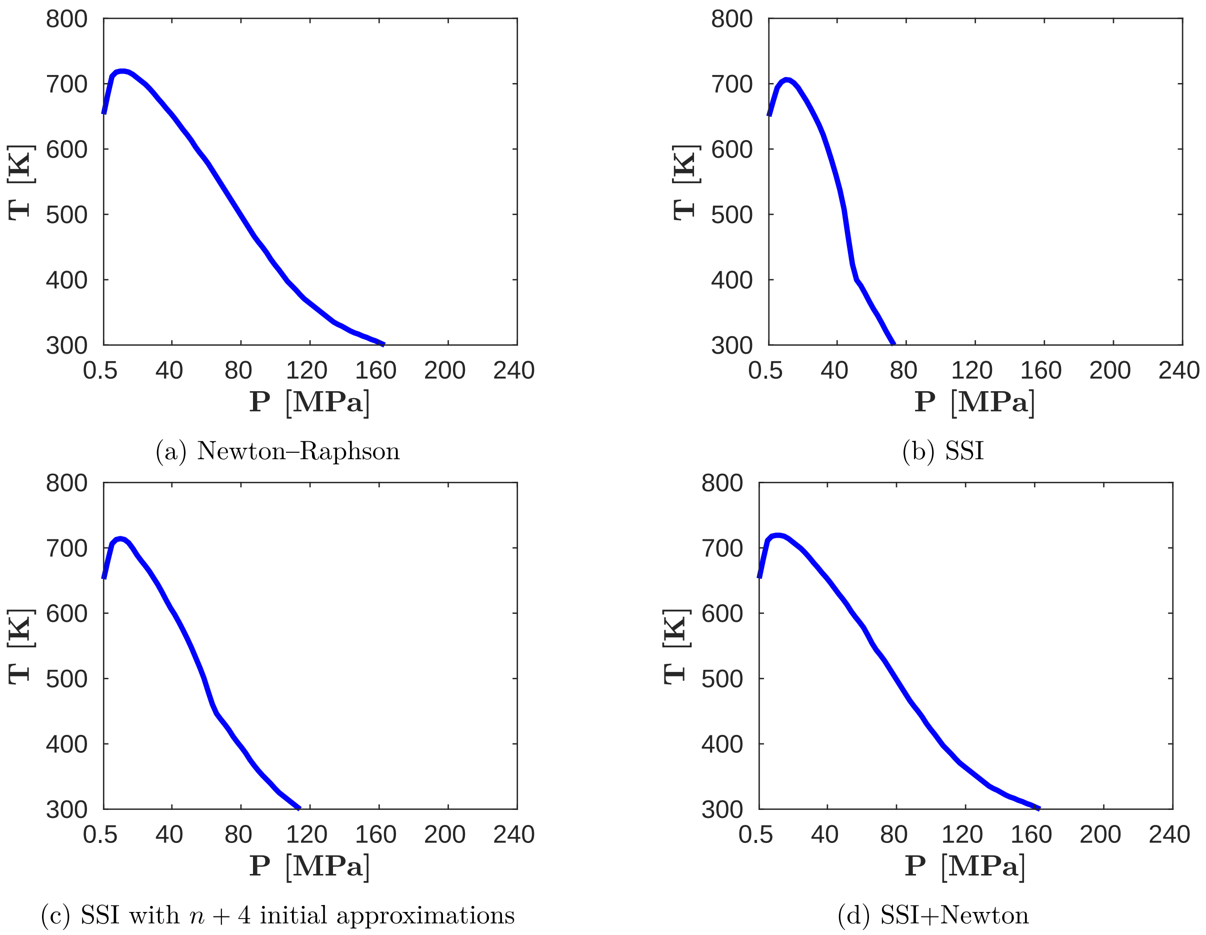


Figure 3.25: The minima of the TPD function in the PT-space. Example E1: mixture Y14.



(a) Newton–Raphson

(b) SSI

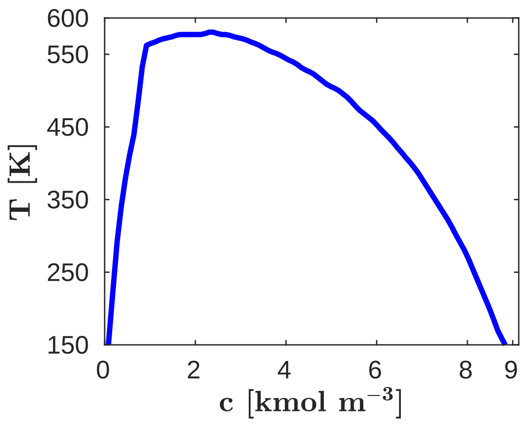
(c) SSI with $n + 4$ initial approximations

(d) SSI+Newton

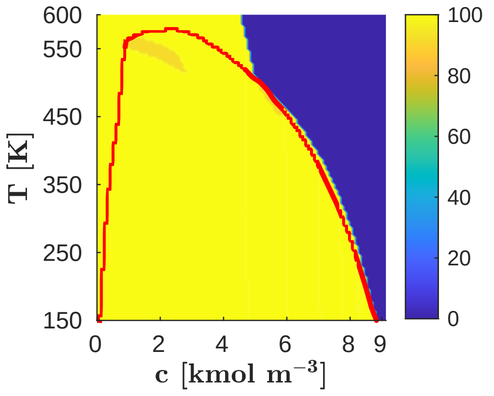
Figure 3.26: The computed approximate phase boundaries using different methods. Below the line the states are unstable (according to the given method). Example E1: mixture Y14.

	Newton–Raphson		SSI		SSI+		SSI+Newton–Raphson	
	t_c [s]	t_{avg} [s]	t_c [s]	t_{avg} [s]	t_c [s]	t_{avg} [s]	t_c [s]	t_{avg} [s]
Example E1	157.70	0.0158	13.68	0.0015	135.78	0.0136	146.31	0.0146

Table 3.15: The overall computation time of 10000 phase stability testing is denoted t_c. The average time per one stability computation is denoted t_{avg}. Moreover, SSI+ denotes for $n + 4$ initial approximations. Example E1: mixture Y14.



(a) phase boundary



(b) percentage of initial approximations where the SSI method converge

Figure 3.27: The red line represents phase boundary using the SSI method. Example E2: mixture Y10.

binary interaction coefficients are between C_1 and other components, and are presented in Table 3.2. The initiation approximations for the SSI method are based on Bi et al. (2020b) where the UVN-specification phase stability testing is discussed. The initial approximations are created by covering the feasible simple $\mathcal{D}$ by $n + 2$ initial approximations. Therefore, for a 10 component mixture there are 12 initial approximations.

In Figure 3.27a, the results are presented. First, one can observe that the phase boundary is not correct. The correct phase boundary for this mixture was presented in Figure 3.9. For lower total concentration, the SSI algorithm falsely predicts a stable phase in an unstable area. Moreover, in Figure 3.27b, the number of initial approximations where the SSI method has convergence is presented. The values are in percent, e.g., the value 50 represents fact that half of the initial approximations did not converge. One can observe that in the stable area for higher total concentrations, the algorithm does not converge for any initial approximations. Therefore, the usage of the SSI method in the VTN-specification is not recommenced.

Example E3: mixture C_1–H_2S

In the UVN -pecification, we test the SSI on two component mixture of methane (C_1) and hydrogen sulfide (H_2S) from Castier (2009). The initial phase composition of Problems 1 and 4 is presented in Table 3.16. We use identical notation as Castier (2009); therefore, we denoted the Problems 1 and 4. Both initial states are unstable and the two-phase state is stable.

The initial approximations are defined by covering the feasible domain $\mathcal{D}$ with 4 initial approximations in the same fashion as in Bi et al. (2020b). The result of Problem 1 is presented in Table 3.17. We present the resulting TPD value with the number of iterations from each of the initial approximations. One can observe that both methods correctly declare the state as unstable.

property	[unit]	Problem 1	Problem 4
U	[J]	−756500.80	−636468.00
V	[cm^3]	52869.00	9926.71
N_{C_1}	[mol]	10.00	10.00
N_{H_2S}	[mol]	90.00	90.00

Table 3.16: Specifications of Problems 1 and 4. The reference state for internal energy U is described in Section 2.7.3. Example E3: mixture C_1–H_2S.

	SSI		Newton–Raphson	
initial approximation	TPD$_{\min}$	iterations	TPD$_{\min}$	iterations
1	−875.549	40	−1555279.350	12
2	−875.549	29	−1555279.350	12
3	−875.549	29	−1555279.350	10
4	−875.549	31	−1555279.350	12
total time [s]	0.0165		0.1052	

Table 3.17: Result of Problem 1. Numbers of iterations needed to achieve convergence in the stability testing and the minimum values of function TPD for each of initial approximation for the SSI and Newton–Raphson method. Example E3: mixture C_1–H_2S.

	SSI		Newton–Raphson	
initial approximation	TPD$_{\min}$	iterations	TPD$_{\min}$	iterations
1	−26708.511	17	3.645×10^{-14}	7
2	−26708.511	14	2.230×10^{-13}	9
3	−26708.511	13	6.667×10^{-14}	9
4	−26708.511	17	−26708.511	10
total time [s]	0.0060		0.0444	

Table 3.18: Result of Problem 4. Numbers of iterations needed to achieve convergence in the stability testing and the minimum values of function TPD for each of initial approximation for the SSI and Newton–Raphson method. Example E3: mixture C_1–H_2S.

However, the SSI method did not find the correct global minimum. The computation time of the SSI is approximately 10 times faster. Second, we tested the algorithm on Problem 4. The result is presented in Table 3.18. Here, the Newton–Raphson method had a problem finding the minimum. In three out of four initial approximations, the Newton–Raphson method converges toward the trivial solution. However, from the last initial approximation, the correct solution is found. Therefore, we do not report the same problem with the Newton–Raphson method as Bi et al. (2020b). The SSI method finds correctly the minimum in all four initial approximations. Concerning the time, the SSI method was again 10 times faster.

Summary

We believe that the use of the SSI method is not ideal in the VTN- and UVN-specification. Multiple times, the SSI method encountered a problem with convergence in the stable area. In the PTN-specification, the combination of the Newton–Raphson and SSI seems to be a robust option.

Phase equilibrium computation 4

The phase equilibrium computation is another basic problem of chemical engineering closely related to the phase stability testing problem. Consider a mixture of n components with temperature T^*, pressure P^*, and mole fractions $x_1^*, \dots, x_n^*$. In Chapter 3, our goal was to predict whether a given state remains stable or splitting will occur. This problem is called the phase stability testing and was discussed in Chapter 3. In the phase equilibrium computation, we extend the question. Now, we are interested not only if the phase is stable, but also in the composition of the equilibrium phases. In Figure 4.1, the diagram of the problem is depicted. Therefore, in the phase equilibrium computation, we have two goals:

1. Find the number of phases of the equilibrium state.

2. Find the chemical compositions and the amounts of all phases.

This problem has broad applications in the industry, e.g., the enhanced oil recovery (Hobson (1975); Mosavat and Torabi (2013); Walsh (2003)), or the closely related carbon dioxide sequestration (Gaspar Ravagnani et al. (2009); Holt et al. (1995); Kaya (1995)).

Similarly to the phase stability testing problem, there exist many formulations/specifications of the phase equilibrium computation. The one presented above is known as the PTN-phase equilibrium computation. Likewise, we define the VTN-phase equilibrium computation, UVN-phase equilibrium computation, and others.

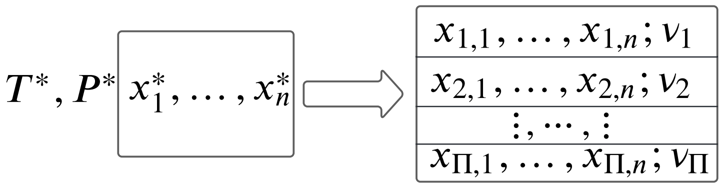
Remark 4.1. As in the phase stability testing, other specifications are studied in the literature. Similarly to the phase stability testing, the HPN-specification (Michelsen (1987)) or SPN-specification (Zhu and Okuno (2016)) were investigated.

Remark 4.2. For shortening the name 'phase equilibrium computation', the term *flash* is often used.

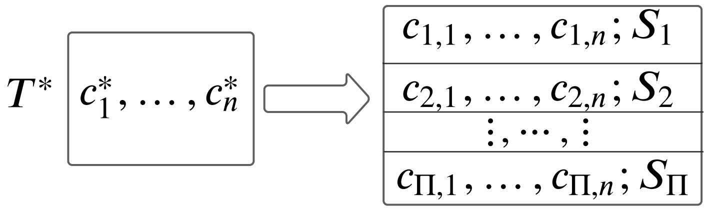
In this chapter, the phase equilibrium computation will be discussed in details. First, a mathematical formulation of the problem will be given. A unified formulation will be presented. Then, a numerical solution based on the elimination of the constraints and the Newton–Raphson method will be given. Lastly, numerical examples showing the performance of the numerical scheme will be presented.

4.1 Mathematical formulation

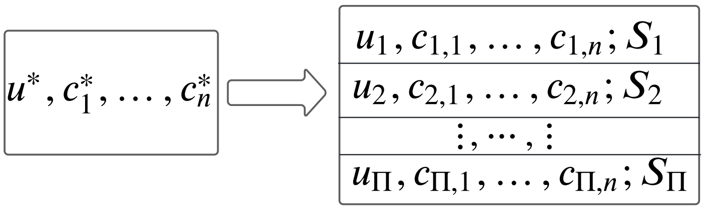
Using the laws of thermodynamics, the equilibrium state is the state with the lowest value of an appropriate potential. In the case of the PTN-specification, we can use Theorem 2.10, the


T^*, P^*
$x_1^*, \dots, x_n^*$
$x_{1,1}, \dots, x_{1,n}; \nu_1$
$x_{2,1}, \dots, x_{2,n}; \nu_2$
$\vdots, \cdots, \vdots$
$x_{\Pi,1}, \dots, x_{\Pi,n}; \nu_\Pi$

(a) PTN-specification


T^*
$c_1^*, \dots, c_n^*$
$c_{1,1}, \dots, c_{1,n}; S_1$
$c_{2,1}, \dots, c_{2,n}; S_2$
$\vdots, \cdots, \vdots$
$c_{\Pi,1}, \dots, c_{\Pi,n}; S_\Pi$

(b) VTN-specification


$u^*, c_1^*, \dots, c_n^*$
$u_1, c_{1,1}, \dots, c_{1,n}; S_1$
$u_2, c_{2,1}, \dots, c_{2,n}; S_2$
$\vdots, \cdots, \vdots$
$u_\Pi, c_{\Pi,1}, \dots, c_{\Pi,n}; S_\Pi$

(c) UVN-specification

Figure 4.1: Diagram of the phase equilibrium computation under different specifications.

appropriate potential is the Gibbs free energy, and the equilibrium state has to satisfy

$$x_i^* = \sum_{k=1}^{\Pi} \nu_k x_{k,i}, \quad i \in \widehat{n}, \tag{4.1}$$

$$1 = \sum_{k=1}^{\Pi} \nu_k, \tag{4.2}$$

$$1 = \sum_{i=1}^{n} x_{k,i}, \quad k \in \widehat{\Pi}, \tag{4.3}$$

$$\sum_{k=1}^{\Pi} \nu_k g(T^*, P^*, x_{k,1}, \dots, x_{k,n}) \to \min . \tag{4.4}$$

See Figure 4.1a for the diagram of the PTN-specification. Similarly, other phase equilibrium specifications can be defined. However, in the next section, we present a unified formulation of the phase equilibrium problem. Then, the three chosen formulations will be presented. We show that the unified formulation covers these specifications.

4.1.1 Unified formulation of the Π-phase equilibrium computation

In Smejkal and Mikyška (2018), we defined the unified formulation as:

Let $f : \mathbb{R}^m \to \mathbb{R}$ be defined on a convex set $\mathcal{D} \subset \mathbb{R}^m$, $\mathbf{x}^* \in \mathcal{D}$, and a natural number $\Pi > 1$ be given. The task is to find affine independent vectors $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi)} \in \mathbb{R}^m$ and coefficients

$\alpha_1, \dots, \alpha_\Pi > 0$ minimizing

$$\mathrm{F}\left(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi)}, \alpha_1, \dots, \alpha_\Pi\right) = \sum_{k=1}^{\Pi} \alpha_k f\left(\mathbf{x}^{(k)}\right) \tag{4.5}$$

subject to

$$\sum_{k=1}^{\Pi} \alpha_k \mathbf{x}^{(k)} = \mathbf{x}^*, \tag{4.6a}$$

$$\sum_{k=1}^{\Pi} \alpha_k = 1. \tag{4.6b}$$

Let us verify that for suitably chosen functions f and vectors $\mathbf{x}^*$ the general formulation (4.5)–(4.6) represents commonly used flash formulations.

4.1.2 *VTN*-specification

Consider a mixture of n components with mole numbers $N_1^*, \dots, N_n^*$ occupying volume V^* at temperature T^*. Let us assume that the mixture occurs in a Π-phase state. Then, using Theorem 2.9, the equilibrium state is the state with the minimum value of the total Helmholtz free energy

$$A^{(\Pi)}\left(\mathbf{N}^{(1)}, \mathbf{N}^{(2)}, \dots, \mathbf{N}^{(\Pi)}, \mathbf{V}\right) = \sum_{k=1}^{\Pi} A\left(T^*, V_k, N_{k,1}, \dots, N_{k,n}\right) \tag{4.7}$$

subject to

$$\sum_{k=1}^{\Pi} V_k = V^*, \tag{4.8a}$$

$$\sum_{k=1}^{\Pi} N_{k,i} = N_i^*, \quad i \in \widehat{n}, \tag{4.8b}$$

where $\mathbf{V} = (V_1, \dots, V_k)^{\mathrm{T}}$ are volumes of each phase, $\mathbf{N}^{(k)} = (N_{k,1}, \dots, N_{k,n})^{\mathrm{T}}$ for $k \in \widehat{\Pi}$ are mole numbers of all components in phase k. According to equation (2.55), the Helmholtz free energy A of phase k reads as

$$A\left(T^*, V_k, N_{k,1}, \dots, N_{k,n}\right) = \sum_{i=1}^{n} N_{k,i}\mu_i\left(T^*, V_k, N_{k,1}, \dots, N_{k,n}\right) - P\left(T^*, V_k, N_{k,1}, \dots, N_{k,n}\right) V_k. \tag{4.9}$$

Introducing the saturations $S_k = \frac{V_k}{V^*}$ for $k \in \widehat{\Pi}$, concentrations $c_{k,i} = \frac{N_{k,i}}{V_k}$ for $k \in \widehat{\Pi}$ and $i \in \widehat{n}$, and Helmholtz free energy density

$$a\left(c_1, \dots, c_n\right) = A\left(T^*, 1, c_1, \dots, c_n\right), \tag{4.10}$$

the VTN-flash problem can be transformed into minimizing

$$a^{(\Pi)}\left(\mathbf{c}^{(1)}, \dots, \mathbf{c}^{(\Pi)}, \mathbf{S}\right) = \sum_{k=1}^{\Pi} S_k a\left(c_{k,1}, \dots, c_{k,n}\right) \tag{4.11}$$

subject to

$$\sum_{k=1}^{\Pi} S_k = 1, \tag{4.12a}$$

$$\sum_{k=1}^{\Pi} S_k c_{k,i} = c_i^*, \qquad i \in \widehat{n}, \tag{4.12b}$$

where $\mathbf{S} = (S_1, \dots, S_\Pi)^{\mathrm{T}}$ and $\mathbf{c}^{(k)} = (c_{k,1}, \dots, c_{k,n})^{\mathrm{T}}$ for $k \in \widehat{\Pi}$. Equations (4.11) and (4.12) represent the general formulation (4.5)–(4.6) with

$$D = \left\{ (c_1, \dots, c_n)^{\mathrm{T}} ; (\forall i \in \widehat{n})\,(c_i \geqslant 0) \wedge \sum_{i=1}^{n} c_i b_i < 1 \right\}, \tag{4.13a}$$

$$\mathbf{x}^* = (c_1^*, \dots, c_n^*)^{\mathrm{T}}, \tag{4.13b}$$

$$\mathbf{x}^{(k)} = (c_{k,1}, \dots, c_{k,n})^{\mathrm{T}}, \qquad k \in \widehat{\Pi}, \tag{4.13c}$$

$$\alpha_k = S_k, \qquad k \in \widehat{\Pi}, \tag{4.13d}$$

$$f\left(\mathbf{x}^{(k)}\right) = a\left(\mathbf{x}^{(k)}\right). \tag{4.13e}$$

In equation (4.13a), b_i is the co-volume parameter of component i from the Peng–Robinson equation of state (see Section 2.4.2).

4.1.3 *UVN*-specification

Next, consider a mixture of n components with mole numbers $N_1^*, \dots, N_n^*$ occupying volume V^* and having internal energy U^*. Let us assume that the mixture occurs in a Π-phase state. Then, using Theorem 2.6, the equilibrium state is the state with the maximum value of the total entropy

$$S^{(\Pi)}\left(\mathbf{U}, \mathbf{N}^{(1)}, \dots, \mathbf{N}^{(\Pi)}, \mathbf{V}\right) = \sum_{k=1}^{\Pi} S\left(U_k, V_k, N_{k,1}, \dots, N_{k,n}\right) \tag{4.14}$$

subject to

$$\sum_{k=1}^{\Pi} U_k = U^*, \tag{4.15a}$$

$$\sum_{k=1}^{\Pi} V_k = V^*, \tag{4.15b}$$

$$\sum_{k=1}^{\Pi} N_{k,i} = N_i^*, \qquad i \in \widehat{n}, \tag{4.15c}$$

where $\mathbf{U} = (U_1, \dots, U_\Pi)^{\mathrm{T}}$ are the internal energies of each phase. The entropy S of phase k reads as

$$S\left(U_k, V_k, N_{k,1}, \dots, N_{k,n}\right) = \frac{U_k}{T} + \frac{PV_k}{T} - \frac{1}{T}\sum_{i=1}^{n} N_{k,i}\mu_i, \tag{4.16}$$

where T, P, μ_i for $i \in \widehat{n}$ are functions of variables $U_k, V_k, N_{k,1}, \ldots, N_{k,n}$. Introducing the saturations S_k and concentrations $c_{k,i}$ as above, energy density $u_k = \frac{U_k}{V_k}$, and entropy density

$$s\left(u, c_1, \ldots, c_n\right) = S\left(u, 1, c_1, \ldots, c_n\right), \tag{4.17}$$

the UVN-flash problem can be transformed into maximizing

$$s^{(\Pi)}\left(\mathbf{u}, \mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(\Pi)}, \mathbf{S}\right) = \sum_{k=1}^{\Pi} S_k s\left(u_k, c_{k,1}, \ldots, c_{k,n}\right) \tag{4.18}$$

subject to

$$\sum_{k=1}^{\Pi} S_k = 1, \tag{4.19a}$$

$$\sum_{k=1}^{\Pi} S_k u_k = u^*, \tag{4.19b}$$

$$\sum_{k=1}^{\Pi} S_k c_{k,i} = c_i^*, \qquad i \in \widehat{n}, \tag{4.19c}$$

where $\mathbf{u} = \left(u_1, \ldots, u_\Pi\right)^{\mathrm{T}}$, which is the general formulation (4.5)–(4.6) with

$$\begin{aligned} D = \Bigg\{ &\left(u, c_1, \ldots, c_n\right)^{\mathrm{T}}; \left(\forall i \in \widehat{n}\right)\left(c_i \geqslant 0\right) \wedge \sum_{i=1}^{n} c_i b_i < 1 \wedge \\ &\left(\exists T > 0\right)\left(u = U^{(EOS)}\left(T, 1, c_1, \ldots, c_n\right)\right) \Bigg\}, \end{aligned} \tag{4.20a}$$

$$\mathbf{x}^* = \left(u^*, c_1^*, \ldots, c_n^*\right)^{\mathrm{T}}, \tag{4.20b}$$

$$\mathbf{x}^{(k)} = \left(u_k, c_{k,1}, \ldots, c_{k,n}\right)^{\mathrm{T}}, \qquad k \in \widehat{\Pi}, \tag{4.20c}$$

$$\alpha_k = S_k, \qquad k \in \widehat{\Pi}, \tag{4.20d}$$

$$f\left(\mathbf{x}^{(k)}\right) = -s\left(\mathbf{x}^{(k)}\right), \tag{4.20e}$$

In equation (4.20a), $U^{(EOS)}$ denotes the thermal equation that was discussed in Section 2.7.3.

4.1.4 *PTN*-specification

Finally, consider a mixture of n components with mole fractions $N_1^*, \ldots, N_n^*$ at temperature T^* and pressure P^*. Let us assume that the mixture occurs in a Π-phase state. Then, using Theorem 2.10, the equilibrium state is the state with the minimum value of the total Gibbs energy

$$G^{(\Pi)}\left(\mathbf{N}^{(1)}, \ldots, \mathbf{N}^{(\Pi)}\right) = \sum_{k=1}^{\Pi} G\left(P^*, T^*, N_{k,1}, \ldots, N_{k,n}\right) \tag{4.21}$$

subject to

$$\sum_{k=1}^{\Pi} N_{k,i} = N_i^*, \qquad i \in \widehat{n}, \tag{4.22}$$

where the Gibbs energy G of phase k reads as

$$G\left(P^*, T^*, N_{k,1}, \ldots, N_{k,n}\right) = \sum_{i=1}^{n} N_{k,i}\mu_i\left(P^*, T^*, N_{k,1}, \ldots, N_{k,n}\right). \tag{4.23}$$

Introducing the mole fractions of the phases

$$\nu_k = \frac{\sum_{i=1}^{n} N_{k,i}}{\sum_{i=1}^{n} N_i^*}, \tag{4.24}$$

mole fractions of component i in phase k

$$x_{k,i} = \frac{N_{k,i}}{\sum_{i=1}^{n} N_{k,i}}, \tag{4.25}$$

and the Gibbs energy per one mol

$$\widetilde{g}\left(x_1, \ldots, x_n\right) = \frac{G(T, P, N_1, \ldots, N_n)}{\sum_{i=1}^{n} N_i}, \tag{4.26}$$

the PTN-flash problem can be transformed into minimizing

$$g^{(\Pi)}\left(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(\Pi)}, \mathbf{v}\right) = \sum_{k=1}^{\Pi} \nu_k \widetilde{g}\left(x_{k,1}, \ldots, x_{k,n}\right) \tag{4.27}$$

subject to

$$\sum_{k=1}^{\Pi} \nu_k = 1, \tag{4.28a}$$

$$\sum_{k=1}^{\Pi} \nu_k x_{k,i} = x_i^*, \qquad i \in \widehat{n}, \tag{4.28b}$$

$$\sum_{i=1}^{n} x_{k,i} = 1, \qquad k \in \widehat{\Pi}, \tag{4.28c}$$

where $\mathbf{v} = (\nu_1, \ldots, \nu_\Pi)^{\mathrm{T}}$, and $\mathbf{X}^{(k)} = (x_{k,1}, \ldots, x_{k,n})^{\mathrm{T}}$ for $k \in \widehat{\Pi}$ are the mole fractions of all components in phase k. Eliminating variables $x_{k,n}$ for $k \in \widehat{\Pi}$ using the constraint (4.28c) and denoting

$$\breve{g}\left(x_{k,1}, \ldots, x_{k,n-1}\right) = \widetilde{g}\left(x_{k,1}, \ldots, x_{k,n-1}, 1 - \sum_{q=1}^{n} x_{k,q}\right),$$

we get a problem of minimizing

$$g_{\mathrm{red}}^{(\Pi)}\left(\widetilde{\mathbf{X}}^{(1)}, \ldots, \widetilde{\mathbf{X}}^{(\Pi)}, \mathbf{v}\right) = \sum_{k=1}^{\Pi} \nu_k \breve{g}\left(x_{k,1}, \ldots, x_{k,n-1}\right), \tag{4.29}$$

where $\widetilde{\mathbf{X}}^{(k)} = (x_{k,1}, \dots, x_{k,n-1})^{\mathrm{T}}$ for $k \in \widehat{\Pi}$, subject to

$$\sum_{k=1}^{\Pi} \nu_k = 1, \tag{4.30a}$$

$$\sum_{k=1}^{\Pi} \nu_k x_{k,i} = x_i^*, \qquad i \in \widehat{n-1}, \tag{4.30b}$$

which is the general formulation (4.5)–(4.6) with

$$D = \left\{ (x_1, \dots, x_{n-1})^{\mathrm{T}} ; \sum_{i=1}^{n-1} x_i \leqslant 1 \wedge \left(\forall i \in \widehat{n-1} \right) (x_i \geqslant 0) \right\}, \tag{4.31a}$$

$$\mathbf{x}^* = \left(x_1^*, \dots, x_{n-1}^* \right)^{\mathrm{T}}, \tag{4.31b}$$

$$\mathbf{x}^{(k)} = (x_{k,1}, \dots, x_{k,n-1})^{\mathrm{T}}, \qquad k \in \widehat{\Pi}, \tag{4.31c}$$

$$\alpha_k = \nu_k, \qquad k \in \widehat{\Pi}, \tag{4.31d}$$

$$f\left(\mathbf{x}^{(k)}\right) = \breve{g}\left(\mathbf{x}^{(k)}\right). \tag{4.31e}$$

4.1.5 Complexities of the individual flash formulations

Although we have developed a single framework for all three flash formulations discussed above, this is by no means to say that all the formulations are the same. All the formulations are theoretically equivalent expressions of the second law of thermodynamics, each of them describing the equilibrium state under different conditions. Nevertheless, the numerical behaviour of the individual formulations can be very different, which will be demonstrated in Section 4.3.2. Here, we will discuss several details related to the three flash formulations and reveal some differences between the formulations.

We will start the discussion with the VTN-flash. The VTN-flash is the simplest flash among the three formulations because the flash specification variables are the same as the natural variables of the commonly used equation of state. When this type of flash is reformulated in the variables temperature and concentrations using the Helmholtz free energy density as the objective function, it fits the general formulation (4.5)–(4.6) perfectly. The objective function of this flash is the Helmholtz free energy density, which is a smooth (at least twice continuously differentiable, i.e., C^2) function of the flash unknowns (concentrations of all components in all phases and phase saturations). The gradient and the Hessian of the objective function are thus at least continuous, and we can expect the Newton–Raphson method to work.

Next, we discuss the UVN-flash. Compared to the VTN-flash, this flash formulation is slightly more complex. The new complexity stems from the fact that the specified flash variables are internal energy density and overall molar concentrations of all components, but the equations of state for pressure and entropy are formulated in terms of temperature and concentrations (see Section 2.7.3). Therefore, for any specified set of variables $u, c_1, \dots, c_n$, we need to find temperature T corresponding to this state by inverting the equation (2.224) written in terms of densities as $u = U^{(EOS)}(T, 1, c_1, \dots, c_n)$. Fortunately, the dependence between u and T in this equation on its domain of validity is one-to-one and smooth, so this complication is rather technical and does not change the properties of the objective function of the flash formulation. Indeed, the entropy density is a twice continuously differentiable function of the internal energy density and molar concentrations of all components and we can expect the Newton–Raphson method to work.

Finally, we discuss the PTN-flash, which appears to be the most complex among the three formulations. In this case, the flash specification variables are the pressure, temperature, and overall molar fractions of all components (or more precisely, $n-1$ independent components), but the equation of state is formulated in terms of temperature and concentrations of all components. For any specified pressure P, temperature T and overall molar fractions x_i, we need to find the overall concentration c such that $P = P^{(EOS)}(T, 1, cx_1, \dots, cx_n)$ holds. Unlike previously, the equation of state for pressure may not be invertible with respect to c and may provide several roots corresponding to different phases. Typically, there are up to three roots in cubic equations of state, the lowest one corresponding to the gas, the highest one corresponding to the liquid and the middle one corresponding to the unstable phase, which is disregarded. From these candidates, we need to select the root with the lowest value of the Gibbs free energy. For small changes of pressure, the concentration may jump abruptly, as it happens in the case of a single component when the pressure is close to the saturation pressure. Consequently, the Gibbs free energy per one mole is just continuous but may have discontinuities in the first derivatives at the points corresponding to the phase change (it is a C^0 function that is not C^1 in its variables). Consequently, we can expect convergence problems in the Newton–Raphson method, which will be confirmed in Section 4.3.2.

4.2 Numerical algorithm for the phase equilibrium computation

In this section, we present a numerical algorithm for solving the optimization problem given by equation (4.5)–(4.6). In the literature, there is a broad variety of numerical algorithms for solving the phase equilibrium, e.g., Gorucu and Johns (2014); Haugen et al. (2010); Li and Firoozabadi (2012); Michelsen et al. (2013); Pan and Firoozabadi (1998). However, these algorithms are mostly designed to solve the PTN-specification and the total number of phases is restricted to $\Pi = 2$ or $\Pi = 3$. In this section, we present a unified algorithm that can be used to solve any specification that can be written in the form given by equations (4.5)–(4.6), and an arbitrary number of phases Π. Our numerical algorithm will be based on the elimination of the constraints. Then, the constrained optimization problem given by equations (4.5)–(4.6) can be transformed to an unconstrained one. The unconstrained optimization problem will be solved using the Newton–Raphson method.

4.2.1 Modified Newton–Raphson method

Eliminating the variables $\mathbf{x}^{(\Pi)}$ and α_Π using the constraints (4.6), we obtain

$$\mathrm{F}_{\mathrm{red}}\left(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi-1)}, \alpha_1, \dots, \alpha_{\Pi-1}\right) = \sum_{k=1}^{\Pi-1} \alpha_k f(\mathbf{x}^{(k)}) + \left(1 - \sum_{k=1}^{\Pi-1} \alpha_k\right) f\left(\check{\mathbf{x}}^{(\Pi)}\right), \tag{4.32}$$

where we denote

$$\check{\mathbf{x}}^{(\Pi)} = \frac{\mathbf{x}^* - \sum\limits_{k=1}^{\Pi-1} \alpha_k \mathbf{x}^{(k)}}{1 - \sum\limits_{k=1}^{\Pi-1} \alpha_k}. \tag{4.33}$$

To determine the equilibrium state, one must solve the unconstrained minimization problem with the objective function $\mathrm{F}_{\mathrm{red}}$ of $(\Pi-1)(m+1)$ unknown variables $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi-1)}, \alpha_1, \dots, \alpha_{\Pi-1}$. This problem can be solved using the modified Newton–Raphson method. Denoting the vectors of unknowns for the reduced unconstrained problem as well as for the original constrained problem as

$$\mathbf{Z} = \left(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi)}, \alpha_1, \dots, \alpha_\Pi\right)^{\mathrm{T}}, \tag{4.34}$$

$$\mathbf{z} = \left(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi-1)}, \alpha_1, \dots, \alpha_{\Pi-1}\right)^{\mathrm{T}}, \tag{4.35}$$

the modified Newton–Raphson method iterates the approximations as

$$\mathbf{Z}^{(k+1)} = \mathbf{Z}^{(k)} + \lambda^{(k)} \boldsymbol{\Delta}\mathbf{Z}^{(k)}, \tag{4.36}$$

where $\lambda^{(k)} \in (0,1]$ is a damping factor determine using the line-search strategy, and $\boldsymbol{\Delta}\mathbf{Z}^{(k)} = \mathbf{E}\boldsymbol{\Delta}\mathbf{z}^{(k)}$ is the solution increment, which is obtained by solving the system of linear algebraic equations

$$\mathbf{H}\left(\mathbf{z}^{(k)}\right) \boldsymbol{\Delta}\mathbf{z}^{(k)} = -\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}\left(\mathbf{z}^{(k)}\right), \tag{4.37}$$

where $\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}\left(\mathbf{z}^{(k)}\right)$ is the gradient of function $\mathrm{F}_{\mathrm{red}}$ and $\mathbf{H}\left(\mathbf{z}^{(k)}\right)$ is the Hessian matrix of the function $\mathrm{F}_{\mathrm{red}}$. The linear mapping $\mathbf{E} \in \mathbb{R}^{2(\Pi-1)} \to \mathbb{R}^{2\Pi}$, which is used to extend the solution of the reduced problem to the full set of variables, is defined as

$$\mathbf{E}\mathbf{z} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \frac{x^*}{1-\sum\limits_{k=1}^{\Pi-1}\alpha_k} \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} \mathbf{z}, \tag{4.38}$$

where the term $\frac{x^*}{1-\sum\limits_{k=1}^{\Pi-1}\alpha_k}$ starts on the $(\Pi-1)m$-th row, and the coefficients A, B, C, D are defined as

$$\begin{aligned} A &\in \mathbb{R}^{\Pi-1,2(\Pi-1)},\ A = \left(\mathbf{I}_{\Pi-1}, \mathbf{O}_{\Pi-1}\right), \\ B &\in \mathbb{R}^{1,2(\Pi-1)},\ B = \left(\frac{-\alpha_1}{1-\sum_{k=1}^{\Pi-1}\alpha_k}, \dots, \frac{-\alpha_{\Pi-1}}{1-\sum_{k=1}^{\Pi-1}\alpha_k}, 0, \dots, 0\right), \\ C &\in \mathbb{R}^{\Pi-1,2(\Pi-1)},\ C = \left(\mathbf{O}_{\Pi-1}, \mathbf{I}_{\Pi-1}\right), \\ D &\in \mathbb{R}^{1,2(\Pi-1)},\ D = \left(\underbrace{0, \dots, 0}_{(\Pi-1)}, \underbrace{-1, \dots, -1}_{(\Pi-1)}\right). \end{aligned}$$

In the equations above, $\mathbf{I}_{\Pi-1}$ is the identity matrix of order $\Pi-1$, and $\mathbf{O}_{\Pi-1}$ is the null matrix of order $\Pi-1$. The gradient $\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}\left(\mathbf{z}^{(k)}\right)$ can be written as

$$\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}\left(\mathbf{z}^{(k)}\right) = \left(\frac{\partial \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(1)}}\left(\mathbf{z}^{(k)}\right), \dots, \frac{\partial \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(\Pi-1)}}\left(\mathbf{z}^{(k)}\right), \frac{\partial \mathrm{F}_{\mathrm{red}}}{\partial \alpha_1}\left(\mathbf{z}^{(k)}\right), \dots, \frac{\partial \mathrm{F}_{\mathrm{red}}}{\partial \alpha_{\Pi-1}}\left(\mathbf{z}^{(k)}\right)\right)^{\mathrm{T}}, \tag{4.39}$$

where the elements can be evaluated using equation (4.32) as

$$\frac{\partial \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(j)}}\left(\mathbf{z}^{(k)}\right) = \alpha_j \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}\left(\mathbf{x}^{(j)}\right) - \alpha_j \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}\left(\breve{\mathbf{x}}^{(\Pi)}\right), \tag{4.40}$$

$$\frac{\partial \mathrm{F}_{\mathrm{red}}}{\partial \alpha_j}\left(\mathbf{z}^{(k)}\right) = f\left(\mathbf{x}^{(j)}\right) - f\left(\mathbf{x}^{(\Pi)}\right) + \frac{1}{\breve{\alpha}_\Pi}\frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}\left(\breve{\mathbf{x}}^{(\Pi)}\right)\left(\mathbf{x}^* - \mathbf{x}^{(j)} + \sum_{k=1}^{\Pi-1}\alpha_k\left(\mathbf{x}^{(j)} - \mathbf{x}^{(k)}\right)\right), \tag{4.41}$$

for $j \in \widehat{\Pi - 1}$, where we denoted $\breve{\alpha}_\Pi = 1 - \sum_{k=1}^{\Pi-1}\alpha_k$. The Hessian matrix $\mathbf{H}\left(\mathbf{z}^{(k)}\right)$ can be written as

$$\mathbf{H}\left(\mathbf{z}^{(k)}\right) = \begin{pmatrix} \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(1)}\partial \mathbf{x}^{(1)}}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(1)}\partial \mathbf{x}^{(\Pi-1)}}, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(1)}\partial \alpha_1}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(1)}\partial \alpha_{\Pi-1}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(\Pi-1)}\partial \mathbf{x}^{(1)}}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(\Pi-1)}\partial \mathbf{x}^{(\Pi-1)}}, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(\Pi-1)}\partial \alpha_1}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(\Pi-1)}\partial \alpha_{\Pi-1}} \\ \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_1\partial \mathbf{x}^{(1)}}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_1\partial \mathbf{x}^{(\Pi-1)}}, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_1\partial \alpha_1}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_1\partial \alpha_{\Pi-1}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_{\Pi-1}\partial \mathbf{x}^{(1)}}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_{\Pi-1}\partial \mathbf{x}^{(\Pi-1)}}, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_{\Pi-1}\partial \alpha_1}, & \dots, & \frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_{\Pi-1}\partial \alpha_{\Pi-1}} \end{pmatrix}, \tag{4.42}$$

where all partial derivatives are evaluated in $\mathbf{z}^{(k)}$. Furthermore, using equation (4.32), we obtain

$$\frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \mathbf{x}^{(i)}\partial \mathbf{x}^{(j)}}\left(\mathbf{z}^{(k)}\right) = \alpha_j \frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{x}^2}\left(\mathbf{x}^{(j)}\right)\delta_{i,j} + \frac{\alpha_i\alpha_j}{\breve{\alpha}_\Pi}\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{x}^2}\left(\breve{\mathbf{x}}^{(\Pi)}\right), \tag{4.43}$$

$$\begin{aligned}\frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_i\partial \mathbf{x}^{(j)}}\left(\mathbf{z}^{(k)}\right) &= \left(\frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}\left(\mathbf{x}^{(j)}\right) - \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}\left(\breve{\mathbf{x}}^{(\Pi)}\right)\right)\delta_{i,j} \\ &\quad - \frac{\alpha_j}{\left(\breve{\alpha}_\Pi\right)^2}\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{x}^2}\left(\breve{\mathbf{x}}^{(\Pi)}\right)\left(\mathbf{x}^* - \mathbf{x}^{(i)} + \sum_{k=1}^{\Pi-1}\alpha_k\left(\mathbf{x}^{(i)} - \mathbf{x}^{(k)}\right)\right),\end{aligned} \tag{4.44}$$

$$\begin{aligned}&\frac{\partial^2 \mathrm{F}_{\mathrm{red}}}{\partial \alpha_i\partial \alpha_j}\left(\mathbf{z}^{(k)}\right) = \\ &\frac{1}{\left(\breve{\alpha}_\Pi\right)^3}\left(\mathbf{x}^* - \mathbf{x}^{(i)} + \sum_{k=1}^{\Pi-1}\alpha_k\left(\mathbf{x}^{(i)} - \mathbf{x}^{(k)}\right)\right)^{\mathrm{T}}\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{x}^2}\left(\breve{\mathbf{x}}^{(\Pi)}\right)\left(\mathbf{x}^* - \mathbf{x}^{(j)} + \sum_{k=1}^{\Pi-1}\alpha_k\left(\mathbf{x}^{(j)} - \mathbf{x}^{(k)}\right)\right),\end{aligned} \tag{4.45}$$

for $i, j \in \widehat{\Pi - 1}$ and where $\delta_{i,j}$ is the Kronecker delta.

Preconditioning of the system $\mathbf{H\Delta z} = -\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}$

To determine $\mathbf{\Delta z}$, we have to solve the system

$$\mathbf{H\Delta z} = -\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}. \tag{4.46}$$

This system is solved using the modified Cholesky decomposition, which was described in Section 3.2.3. This method guarantees that the value of $\mathrm{F}_{\mathrm{red}}$ will decrease in each iteration. Therefore, the modified Newton method will converge to at least a local minimum of function $\mathrm{F}_{\mathrm{red}}$. Numerical experiments indicate that it is advantageous to use the symmetric diagonal preconditioning of matrix $\mathbf{H}$ to equilibrate the diagonal elements of $\mathbf{H}$. Instead of solving (4.46), we solve an equivalent system

$$\mathbf{PHP}^{\mathrm{T}}\mathbf{y} = -\mathbf{P}\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}, \tag{4.47}$$

where $\Delta\mathbf{z} = \mathbf{P}^{\mathrm{T}}\mathbf{y}$ and $\mathbf{P} \in \mathbb{R}^{(\Pi-1)m,(\Pi-1)m}$ is a diagonal matrix with nonzero diagonal elements which are chosen such that the diagonal elements of the preconditioned matrix $\mathbf{PHP}^{\mathrm{T}}$ are equal to ± 1, where the sign is identical to the sign of the original element of matrix $\mathbf{H}$. This procedure significantly improves the convergence of the Newton–Raphson method in the UVN-specification. In the other specifications, the convergence is identical or slightly improved. At the same time, the condition number of matrix $\mathbf{PHP}^{\mathrm{T}}$ is much lower compared to $\mathbf{H}$, and we are thus solving a better-conditioned problem.

The stopping criteria

Following the discussion in Gill et al. (1981), we propose the stopping criterion based on three parameters – a decrease of function F, size of the increment of the solution, and the size of the gradient of the function F_{red}. The modified Newton–Raphson iterations are terminated when all the following conditions hold

$$F\left(\mathbf{Z}^{(j)}\right) - F\left(\mathbf{Z}^{(j-1)}\right) < \Theta_j, \tag{4.48}$$

$$\left\|\mathbf{Z}^{(j)} - \mathbf{Z}^{(j-1)}\right\| < \sqrt{\tau}\left(1 + \left\|\mathbf{Z}^{(j)}\right\|\right), \tag{4.49}$$

$$\left\|\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}(\mathbf{z}^{(j)})\right\|_2 < \sqrt[3]{\tau}\left\|\nabla F\left(\mathbf{Z}^{(j)}\right)\right\|_2, \tag{4.50}$$

where

$$\Theta_j = \tau\left(1 + \left|F\left(\mathbf{Z}^{(j)}\right)\right|\right). \tag{4.51}$$

Parameter τ is the prescribed tolerance. In this thesis, we use $\tau = 10^{-15}$. The norm $\|\cdot\|_2$ in equations (4.50) is the standard Euclidean norm. In equation (4.49), $\|\cdot\|$ is a formulation dependent norm

$$\left\|\mathbf{Z}^{(j)}\right\|^2 = \sum_{k=1}^{\Pi}\left(\left\|\mathbf{x}^{(k)}\right\|_{pn}^2 + \alpha_k^2\right), \tag{4.52}$$

where $\mathbf{Z}^{(j)} = \left(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\Pi)}, \alpha_1, \dots, \alpha_\Pi\right)^{\mathrm{T}}$. The norm $\|\cdot\|_{pn}$ in the three main specifications was defined in equations (3.84)–(3.86).

Summary of the numerical algorithm for the general flash compitation with an a priori known number of phases

In Algorithm 15, we summarize the essential steps of the algorithm for calculating the equilibrium state of a Π-phase system.

Phase addition and removal

In Sections 3.1.1, we have described an algorithm for the general single-phase stability testing. As the TPD function depends only on the values of the intensive thermodynamical variables (which in the equilibrium system are the same in all phases), the same algorithm can be used for testing the phase stability of a general Π-phase equilibrium system. In this situation, it is necessary to test the stability of only one (arbitrarily selected) phase from the equilibrium phase split. As already described at the end of Section 3.1.1, the phase stability testing provides a way to introduce a new phase in an unstable equilibrium system so that the value of the objective

Algorithm 15: Modified Newton–Raphson method for the phase equilibrium computation with an a priori known number of phases.

```
   Input  : $f : \mathbb{R}^m \to \mathbb{R}$, $\mathcal{D} \subset \mathbb{R}^m$, $\Pi > 1$, and $\mathbf{x}^* \in \mathcal{D}$
   Output : composition of the equilibrium state
 1 assume an initial solution $\mathbf{Z}^{(0)}$ in the form
       $$\mathbf{Z}^{(0)} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(\Pi)}, \alpha_1, \ldots, \alpha_\Pi\right)^{\mathrm{T}}$$
    set iteration counter $j = 0$
 2 while $j <$ maximum number of iterations do
 3     assemble the Hessian matrix $\mathbf{H}$ and the gradient $\boldsymbol{\nabla}\mathrm{F}_{\mathrm{red}}$ using equations (4.42) and
        (4.39)
 4     evaluate the increment of the solution $\boldsymbol{\Delta}\mathbf{z}^{(j)} \in \mathbb{R}^{(\Pi-1)(m+1)}$ by solving the linear
        algebraic system (4.37) using the modified Cholesky decomposition from
        Section 3.2.3
 5     evaluate the increment $\boldsymbol{\Delta}\mathbf{Z}^{(\mathbf{j})} \in \mathbb{R}^{\Pi(m+1)}$ using equation (4.38)
 6     determine $\lambda^{(j)} \in (0, 1]$ using the line-search strategy from Section 3.2.3
 7     update the solution using equation (4.36)
 8     if convergence conditions (4.48)–(4.50) hold then
 9         break
10     end
11     set $j := j + 1$
12 end
13 return $\mathbf{Z}^{(j+1)}$
```

function is lower than the value of the equilibrium state. The phase stability testing thus provides an excellent way for the initialization of the phase equilibrium computation.

During the computation of the phase equilibrium in systems with $\Pi \geqslant 3$ phases, it may happen that one of the phases disappears. This may be a consequence of the fact that during the computation of the phase equilibrium in a $(\Pi - 1)$-phase system, the algorithm has converged to a local minimum only, and therefore, there exists an $(\Pi - 1)$-phase state with a lower value of the objective function than the currently computed state. For this reason, the algorithm must involve a test of whether some of the current phases should be removed. We use the following criterion for the phase removal. If $\alpha_k \leqslant 10^{-6}$ for some $k \in \widehat{\Pi}$, and if after this phase is removed from the system, the value of the objective function (4.5) decreases, then the phase k is removed, and its extensive variables uniformly split to the remaining phases, and continue the computation using the $(\Pi - 1)$-phase flash.

The proposed algorithm ensures a decrease of the objective function in each iteration (including the steps in which the number of phases is changing). Thanks to this property, the convergence towards the trivial solution is avoided, which is not the case of some other methods available in the literature (see, e.g., Firoozabadi (1999); Michelsen (1982a,b)).

4.2.2 General equilibrium computation strategy

As we do not know the number of phases a priori, we start with $\Pi = 1$, and we consecutively add and remove phases until the phase stability test indicates the stable phase split. In Algorithm 16, we present a general equilibrium computation strategy. Moreover, the flowchart is depicted in

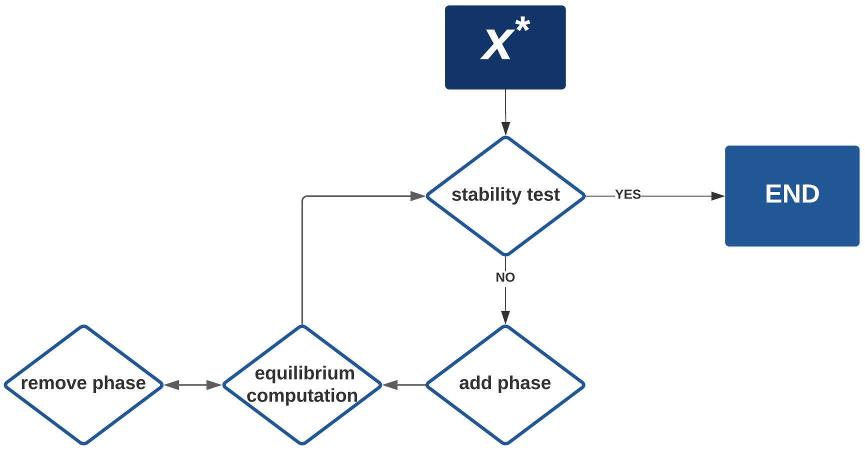


Figure 4.2: Flowchart of the general equilibrium computation strategy.

Figure 4.2. In every step of the algorithm (phase stability or phase equilibrium), the value of the objective function is decreased. Therefore, the loop between phase equilibrium and phase stability (see Figure 4.2) is finite, and the algorithm has to end in a finite number of steps.

Algorithm 16: General equilibrium computation strategy.

```
   Input  : $f : \mathbb{R}^m \to \mathbb{R}$, $\mathcal{D} \subset \mathbb{R}^m$, and $\mathbf{x}^* \in \mathcal{D}$
   Output : composition of the equilibrium state and number of phases
 1 Set number of phases $\Pi = 1$.
 2 while true do
 3     test stability of $\Pi$-phase state using any Algorithm presented in Section 3.2.
 4     if state is stable then
 5         break
 6     end
 7     increase the number of phases by one
 8     introduce a new phase using the strategy described in Section 3.1.1
 9     evaluate the equilibrium state in the system with $\Pi$ phases using the algorithm from
         Section 4.2.1
10     in each iteration, test if for some phase the condition for the phase removal is fulfilled;
         if yes, remove the pertinent phase and decrease the number of phases by one
11 end
12 return equilibrium state
```

4.3 Examples

In this section, we present examples showing the performance of the presented phase equilibrium computation algorithms. First, in Section 4.3.1, we present numerical examples from the UVN-specification using the modified Newton–Raphson method. Then, in Section 4.3.2, a comparison of the individual specifications using the Newton–Raphson method will be presented. Comparison with the standard PTN-specification solver, which uses the natural logarithms of the equilibrium K-values and the gas-phase mole fractions as primary variables presented by Haugen et al. (2010);

property	[unit]	Problem 1	Problem 2	Problem 3	Problem 4
U	[J]	-756500.80	-1511407.60	-331083.70	-636468.00
V	[cm^3]	52869.00	4268.10	80258.10	9926.71
N_{C_1}	[mol]	10.00	0.95	15.10	10.00
N_{H_2S}	[mol]	90.00	99.05	84.90	90.00

Table 4.1: Specifications of Problems 1–4. The reference state for internal energy U is described in Section 2.7.3. Example F1: mixture C_1-H_2S.

Li and Firoozabadi (2012) will be given. In all examples, the Peng–Robinson equation of state from Section 2.4.2 will be used.

4.3.1 UVN examples

In this section, we present examples from Smejkal and Mikyška (2017). Using the initial guess in the UVN-flash from the UVN-stability analysis allows to avoid the need for estimates of the pressure and temperature of the system which were required in the previous works by Castier (2009) and Saha and Carroll (1997). Compared to the previous works, the computational algorithm is much simplified, treats both single-component and multi-component mixtures in the same way, and can be performed in real arithmetics only. The numerical difficulties mentioned by Castier (2009), which required some parts of the algorithm to be performed in the complex arithmetics, are thus avoided.

Example F1: mixture C_1-H_2S

In the first set of examples, we consider a binary mixture of methane (C_1) and hydrogen sulfide (H_2S). The binary interaction parameter between C_1 and H_2S is $\delta_{C_1-H_2S} = 0.083$. Four different specifications are given in Table 4.1. According to Castier (2009), these specifications were chosen so that large amounts of both (vapour and liquid) phases are present in equilibrium in Problem 1, specifications of Problems 2 and 3 lead to states close to the bubble and dew points, respectively, while the solution of Problem 4 is close to the critical point. All four problems have been solved using our method. Results are given in Tables 4.2–4.5. We report the phase split properties together with values of pressure, temperature, and chemical potentials of all components in each phase. These data allow checking whether the iterations have converged towards the equilibrium state. We also provide values of entropy of the hypothetical single-phase state, the total entropy of the phase split, and the numbers of iterations needed for convergence.

The match between our results and the results given in Castier (2009) is quite satisfactory. The numbers of iterations needed for convergence are either the same or lower in our method than those reported in Castier (2009), but these numbers depend on the stopping criterion. Interestingly, in Problem 4, which is deemed to be close-critical, convergence is achieved in 5 iterations only. Unlike in Castier (2009), our method does not need any estimates of the pressure and temperature. Instead, it constructs an initial guess for two-phase flash computation using the UVN-phase stability analysis. In Problems 1–4, stability detects the two-phase state using one initial guess only and provides an initial guess for the UVN-phase equilibrium computation. Iterations reported in Tables 4.2–4.5 count iterations of the modified Newton–Raphson method in the UVN-flash computation using this single initial guess from the UVN-stability. Finally, let us point out that our algorithm is fully performed in real arithmetics. On the other hand,

		Problem 1	
property	[unit]	phase 1	phase 2
U	[J]	-544956.214319	-211544.585681
V	[cm^3]	1502.361229	51366.638771
N_{C_1}	[mol]	0.335680	9.664320
N_{H_2S}	[mol]	35.684022	54.315978
T	[K]	297.997716	297.997716
P	[Pa]	2500170.787203	2500170.787153
μ_{C_1}	[J mol^{-1}]	3303.806129	3303.806129
μ_{H_2S}	[J mol^{-1}]	7051.238967	7051.238967
S^I	[J K^{-1}]	-4847.824318	
S^{II}	[J K^{-1}]	-4335.499136	
iterations	[-]	9	

Table 4.2: Result of Problem 1. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F1: mixture C_1-H_2S.

		Problem 2	
property	[unit]	phase 1	phase 2
U	[J]	-1510985.753624	-421.846376
V	[cm^3]	4165.673900	102.426100
N_{C_1}	[mol]	0.930730	0.019270
N_{H_2S}	[mol]	98.941685	0.108315
T	[K]	298.000861	298.000856
P	[Pa]	2500317.847486	2500317.776275
μ_{C_1}	[J mol^{-1}]	3303.775622	3303.775371
μ_{H_2S}	[J mol^{-1}]	7051.480304	7051.480059
S^I	[J K^{-1}]	-7391.709463	
S^{II}	[J K^{-1}]	-7390.326639	
iterations	[-]	3	

Table 4.3: Result of Problem 2. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F1: mixture C_1-H_2S.

		Problem 3	
property	[unit]	phase 1	phase 2
U	[J]	−566.777015	−330516.922985
V	[cm^3]	1.562506	80256.537494
N_{C_1}	[mol]	0.000349	15.099651
N_{H_2S}	[mol]	0.037113	84.862887
T	[K]	297.996887	297.996887
P	[Pa]	2500125.243552	2500124.858262
μ_{C_1}	[J mol^{-1}]	3303.775698	3303.775686
μ_{H_2S}	[J mol^{-1}]	7051.175471	7051.175450
S^I	[J K^{-1}]	−2613.988230	
S^{II}	[J K^{-1}]	−2613.987835	
iterations	[-]	3	

Table 4.4: Result of Problem 3. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F1: mixture C_1–H_2S.

		Problem 4	
property	[unit]	phase 1	phase 2
U	[J]	−245807.965175	−390660.034825
V	[cm^3]	3512.626019	6414.083981
N_{C_1}	[mol]	3.551418	6.448582
N_{H_2S}	[mol]	33.609473	56.390527
T	[K]	361.997885	361.997885
P	[Pa]	10130505.626170	10130505.626049
μ_{C_1}	[J mol^{-1}]	8352.778379	8352.778379
μ_{H_2S}	[J mol^{-1}]	11536.674427	11536.674427
S^I	[J K^{-1}]	−4579.402758	
S^{II}	[J K^{-1}]	−4579.402147	
iterations	[-]	5	

Table 4.5: Result of Problem 4. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F1: mixture C_1–H_2S.

property	[unit]	Problem 5	Problem 6
U	[J]	−16272506.4	24858.2
V	[cm^3]	479845.0	289380.3
N_{C_2}	[mol]	10.8	10.8
$N_{C_3H_6}$	[mol]	360.8	360.8
N_{C_3}	[mol]	146.5	146.5
N_{iC_4}	[mol]	233.0	233.0
N_{C_4}	[mol]	233.0	233.0
N_{C_5}	[mol]	15.9	15.9

Table 4.6: Specifications of Problems 5 and 6. The reference state for internal energy U is described in Section 2.7.3. Example F2: liquified petroleum gas (LPG) mixture.

Castier (2009) reported that in some cases it is necessary to perform some parts of the algorithm in complex arithmetics.

Example F2: liquified petroleum gas (LPG) mixture

Another two examples from Castier (2009) deal with a six-component LPG mixture. As in Castier (2009), the binary interaction coefficients between all components are set to zero. The specifications of Problems 5 and 6 are given in Table 4.6. Castier (2009) reports that in Problem 5, because of the appearance of the negative pressure in one part of the computation, his algorithm has to be modified to use the nested loops. Using the UVN-phase stability analysis for the construction of the initial phase split, our method converges directly in 10 iterations in Problem 5, and in 5 iterations in Problem 6. These numbers are the same or lower than those reported in Castier (2009), but they depend on the formulation of the stopping criterion. The resulting phase splits together with values of pressure, temperature, chemical potentials of all components in all phases, values of the entropy in both single-phase and two-phase systems, and the numbers of iterations are given in Tables 4.7 and 4.8.

Example F3: LPG gas mixture with water

The last three problems from Castier (2009) (denoted as Problems 7–9) deal with the 6 component LPG mixture from the last subsection mixed with water (H_2O). In agreement with Castier (2009), all binary interaction coefficients are set equal to zero. The specifications of the problems are given in Table 4.9. According to Castier (2009), the specifications are chosen so that Problem 7 leads to a three-phase vapour-liquid-liquid equilibrium, Problem 8 is a two-phase liquid-liquid equilibrium, and Problem 9 represents a high-pressure three-phase equilibrium computation. The resulting phase splits together with values of pressure, temperature, chemical potentials of all components in all phases, entropies, and the numbers of iterations of the modified Newton–Raphson method in the two-phase and three-phase UVN-flash equilibrium computations are summarized in Tables 4.10, 4.11, and 4.12. The number of iterations in Problems 8 and 9 are similar (same or lower in our method) to those reported in Castier (2009). For Problem 7, Castier (2009) reports 3 iterations in three-phase computation together with 24 outer loop iterations for initial estimates, while we have 15 iterations in two-phase and 13 iterations in three-phase UVN-flash equilibrium computation using a single initial guess provided by the UVN-stability testing algorithm. Let us point out again that unlike in Castier (2009), our approach does not

		Problem 5	
property	[unit]	phase 1	phase 2
U	[J]	−15892619.468615	−379886.931385
V	[cm^3]	78647.609580	401197.390420
N_{C_2}	[mol]	6.596564	4.203436
$N_{C_3H_6}$	[mol]	292.574168	68.225832
N_{C_3}	[mol]	122.083040	24.416960
N_{iC_4}	[mol]	214.470841	18.529159
N_{C_4}	[mol]	219.114563	13.885437
N_{C_5}	[mol]	15.574400	0.325600
T	[K]	299.999735	299.999735
P	[Pa]	700082.833469	700082.833469
μ_{C_2}	[J mol^{-1}]	−3805.672092	−3805.672092
$\mu_{C_3H_6}$	[J mol^{-1}]	2997.221501	2997.221501
μ_{C_3}	[J mol^{-1}]	397.265640	397.265640
μ_{iC_4}	[J mol^{-1}]	−445.138790	−445.138790
μ_{C_4}	[J mol^{-1}]	−1196.477103	−1196.477103
μ_{C_5}	[J mol^{-1}]	−10746.440440	−10746.440440
S^I	[J K^{-1}]	−73647.697512	
S^{II}	[J K^{-1}]	−54939.068244	
iterations	[-]	10	

Table 4.7: Results of Problems 5. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F2: liquified petroleum gas (LPG) mixture.

require any initial estimates of the equilibrium pressure and temperature and the whole algorithm proceeds in a straightforward way using the real arithmetics only.

Detailed computation of Problem 7

To demonstrate the main features of the UVN-stability analysis and its application in the UVN-phase equilibrium computation, we present a detailed description of computation for Problem 7 from the previous paragraph. The procedure starts by testing the single-phase stability of the proposed UVN-specification. The stability test indicates the unstable single-phase using only one initial guess (the first guess indicated the phase as unstable, so the other initial guesses are not tested) and needed 124 iterations to converge. The result of stability testing is summarized in Table 4.13. This result is used for the construction of a two-phase split with higher entropy than that of the single-phase state using the procedure described at the end of Section 3.1.1. The initial phase split for two-phase equilibrium computation is presented in Table 4.14. Then, the two-phase UVN-flash is performed. In 15 iterations, the algorithm converges towards the two-phase split presented in Table 4.15. This split is tested for stability (the second phase is tested). The stability test indicates the unstable two-phase split using the first available initial guess (therefore, other initial guesses are not tested) and needed 23 iterations to converge. The result of the two-phase stability testing is summarized in Table 4.16. This result is used for the

property	[unit]	Problem 6 phase 1	Problem 6 phase 2
U	[J]	−150012.775415	174870.975415
V	[cm^3]	16232.876572	273147.423428
N_{C_2}	[mol]	0.735307	10.064693
$N_{C_3H_6}$	[mol]	27.089302	333.710698
N_{C_3}	[mol]	11.174346	135.325654
N_{iC_4}	[mol]	19.334487	213.665513
N_{C_4}	[mol]	19.881086	213.118914
N_{C_5}	[mol]	1.508810	14.391190
T	[K]	394.998501	394.998501
P	[Pa]	4230233.608414	4230233.576530
μ_{C_2}	[J mol^{-1}]	−3002.464669	−3002.464675
$\mu_{C_3H_6}$	[J mol^{-1}]	7239.889856	7239.889846
μ_{C_3}	[J mol^{-1}]	3882.999701	3882.999692
μ_{iC_4}	[J mol^{-1}]	3964.389004	3964.388995
μ_{C_4}	[J mol^{-1}]	3693.302135	3693.302127
μ_{C_5}	[J mol^{-1}]	−6800.367303	−6800.367304
S^I	[J K^{-1}]	−9052.552759	
S^{II}	[J K^{-1}]	−9052.431373	
iterations	[-]	5	

Table 4.8: Result of Problems 6. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F2: liquified petroleum gas (LPG) mixture.

property	[unit]	Problem 7	Problem 8	Problem 9
U	[J]	−17008802.6	−4575454.3	−7088052.5
V	[cm^3]	401916.6	2209.9	265831.3
N_{C_2}	[mol]	10.8	0.0108	10.8
$N_{C_3H_6}$	[mol]	360.8	0.3608	360.8
N_{C_3}	[mol]	146.5	0.1465	146.5
N_{iC_4}	[mol]	233.0	0.233	233.0
N_{C_4}	[mol]	233.0	0.233	233.0
N_{C_5}	[mol]	15.9	0.0159	15.9
N_{H_2O}	[mol]	14.0	100.0	200.0

Table 4.9: Specifications of Problems 7–9. The reference state for internal energy U is described in Section 2.7.3. Example F3: LPG gas mixture with water.

property	[unit]	Problem 7 phase 1	phase 2	phase 3
U	[J]	−13481.947036	−16692030.289355	−303290.363609
V	[cm^3]	6.272850	81021.288073	320889.039078
N_{C_2}	[mol]	0.000000	7.247817	3.552183
$N_{C_3H_6}$	[mol]	0.000000	306.177159	54.622840
N_{C_3}	[mol]	0.000000	127.045435	19.454565
N_{iC_4}	[mol]	0.000000	218.558557	14.441443
N_{C_4}	[mol]	0.000000	222.262356	10.737644
N_{C_5}	[mol]	0.000000	15.651320	0.248680
N_{H_2O}	[mol]	0.295804	13.205980	0.498216
T	[K]	299.999610	299.999610	299.999610
P	[Pa]	700079.813661	700079.562268	700079.562267
μ_{C_2}	[J mol^{-1}]	−3666.963414	−3666.963422	−3666.963422
$\mu_{C_3H_6}$	[J mol^{-1}]	3001.678870	3001.678922	3001.678922
μ_{C_3}	[J mol^{-1}]	383.559360	389.822378	389.822378
μ_{iC_4}	[J mol^{-1}]	−507.024620	−507.024327	−507.024327
μ_{C_4}	[J mol^{-1}]	−1278.467315	−1277.812907	−1277.812907
μ_{C_5}	[J mol^{-1}]	−10963.269047	−10858.073503	−10858.073503
μ_{H_2O}	[J mol^{-1}]	−8731.106730	−8731.106738	−8731.106738
S^I	[J K^{-1}]		−75123.865978	
S^{III}	[J K^{-1}]		−57057.389544	
iterations	[-]		15+13	

Table 4.10: Result of Problem 7. Iterations $X + Y$ means X iterations in the two-phase and Y iterations in the three-phase UVN-flash computations. S^I denotes the entropy of the hypothetical single-phase state, while S^{III} denotes the equilibrium entropy of the stable three-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F3: LPG gas mixture with water.

construction of an initial three-phase split with higher entropy than that of the two-phase state. The initial phase split for the three-phase equilibrium computation is presented in Table 4.17. Then, the three-phase UVN-flash is performed. In 13 iterations the algorithm converges towards the final result presented previously in Table 4.10. To confirm the stability of this three-phase split, the UVN-phase stability test is performed. The numbers of iterations needed to achieve convergence in the UVN-stability testing and the minimum values of function TPD for each initial guess are summarized in Table 4.18. As all minimum values of TPD are positive, the three-phase split is deemed to be stable and the computation is terminated.

Example F4: mixture CO_2–C_1

To demonstrate the robustness of our method, we consider a binary mixture of carbon dioxide (CO_2) and methane (C_1) with overall mole fractions $x^*_{CO_2} = 0.452587$ and $x^*_{C_1} = 0.547413$. Parameters for these components are presented in Appendix Table A.1.The binary interaction coefficient between CO_2 and C_1 is $\delta_{CO_2-C_1} = 0.15$. In Figure 4.3, we present the number of equilibrium phases and the boundaries between the single-phase, two-phase, and three-phase subdomains in the u–c domain (i.e., as a function of the internal energy density $u = U/V$ and

		Problem 8	
property	[unit]	phase 1	phase 2
U	[J]	−4556984.999158	−18469.300842
V	[cm^3]	2120.250219	89.649781
N_{C_2}	[mol]	0.000032	0.010768
$N_{C_3H_6}$	[mol]	0.000173	0.360627
N_{C_3}	[mol]	0.000014	0.146486
N_{iC_4}	[mol]	0.000000	0.233000
N_{C_4}	[mol]	0.000001	0.232999
N_{C_5}	[mol]	0.000000	0.015900
N_{H_2O}	[mol]	99.985323	0.014677
T	[K]	300.024831	300.024831
P	[Pa]	1018719.106609	1018719.108927
μ_{C_2}	[J mol^{-1}]	−2928.755308	−2928.755272
$\mu_{C_3H_6}$	[J mol^{-1}]	3165.466979	3165.466979
μ_{C_3}	[J mol^{-1}]	501.933729	501.933729
μ_{iC_4}	[J mol^{-1}]	−584.801381	−584.801380
μ_{C_4}	[J mol^{-1}]	−1397.224869	−1397.224869
μ_{C_5}	[J mol^{-1}]	−11050.066041	−11050.066040
μ_{H_2O}	[J mol^{-1}]	−8721.253771	−8721.253897
S^I	[J K^{-1}]	−12420.400838	
S^{II}	[J K^{-1}]	−12337.725969	
iterations	[-]	17	

Table 4.11: Result of Problem 8. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable two-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F3: LPG gas mixture with water.

overall molar concentration of the mixture c assuming that the overall composition of the mixture remains the same). Note that for sufficiently low values of the internal energy U, there may be no temperature corresponding to the given values U, V, and $N_1, \ldots, N_n$. The set of physically reasonable values of U is thus bounded from below. In Figure 4.3, the nonphysical domain is denoted as 0-phase domain.

Next, we investigate two cases, which are indicated in Figure 4.3. First, we study compression of the mixture in a closed vessel while keeping the constant value of the internal energy density $u^* = -2.5 \times 10^8$ J m^{-3}. In Figures 4.4 and 4.5, we plot the equilibrium pressure of the mixture, equilibrium temperature of the mixture, volume fractions of the phases, and molar fractions of all components in all phases, respectively, as functions of the overall molar density of the mixture. Second, we study the heating of the mixture in a closed vessel of constant volume. The total molar density of the mixture is kept constant with $c^* = 26000$ mol m^{-3}. In Figures 4.6 and 4.7, we plot the equilibrium pressure of the mixture, equilibrium temperature of the mixture, volume fractions of the phases, and molar fractions of all components in all phases, respectively, as functions of the internal energy density u.

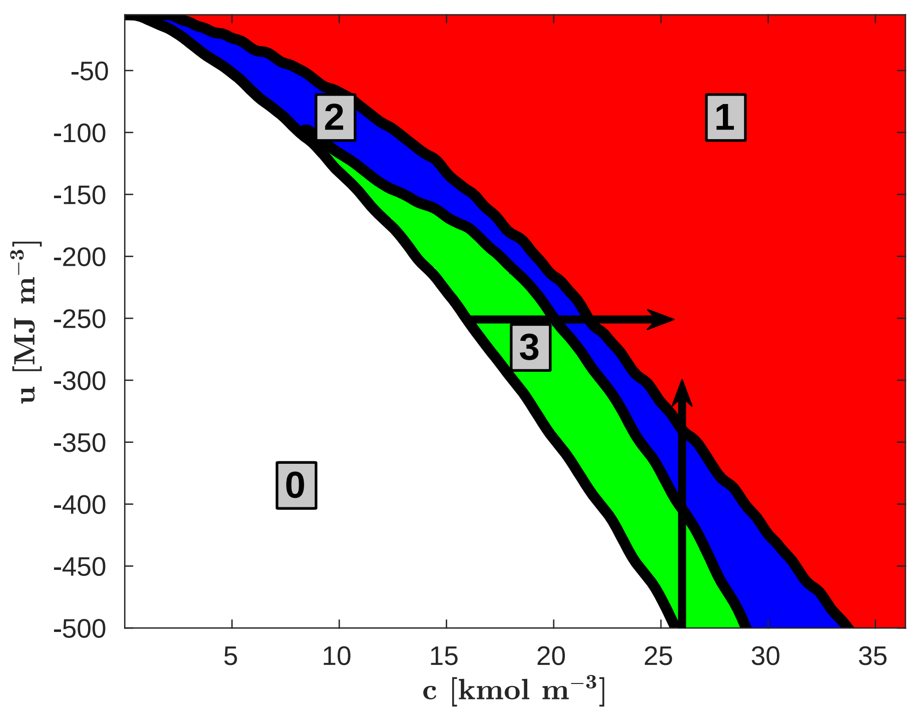
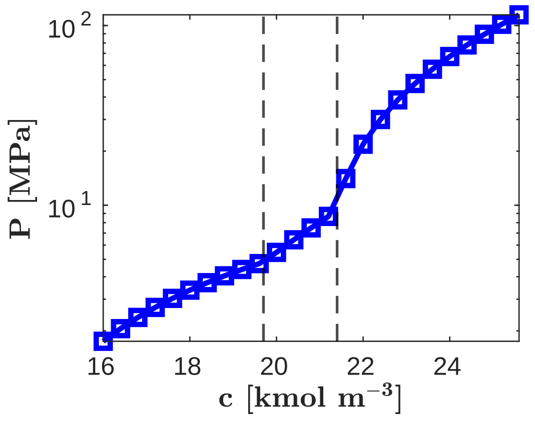
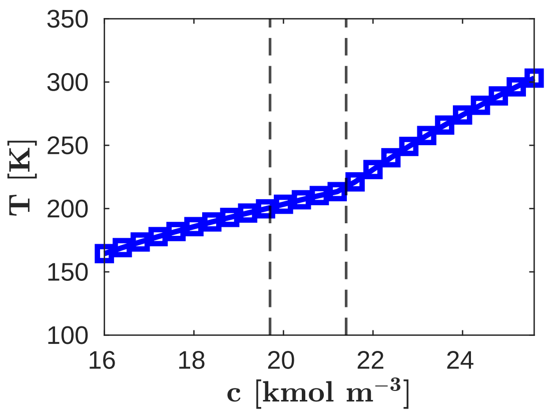


Figure 4.3: Approximate boundaries between the single-phase, two-phase, and three-phase domains. Example F4: mixture $CO_2{-}C_1$.

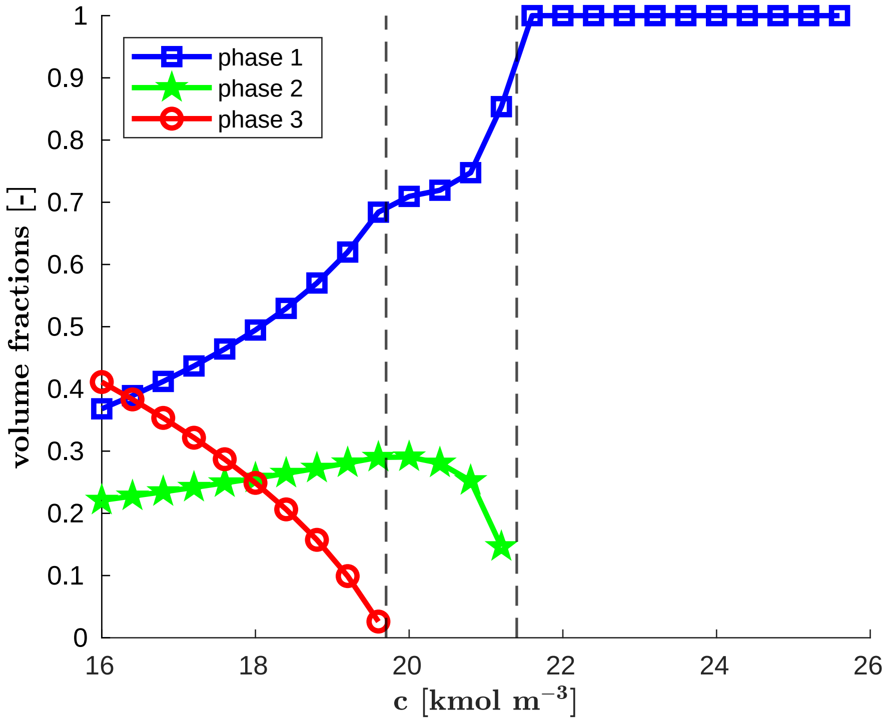


(a) equilibrium pressure

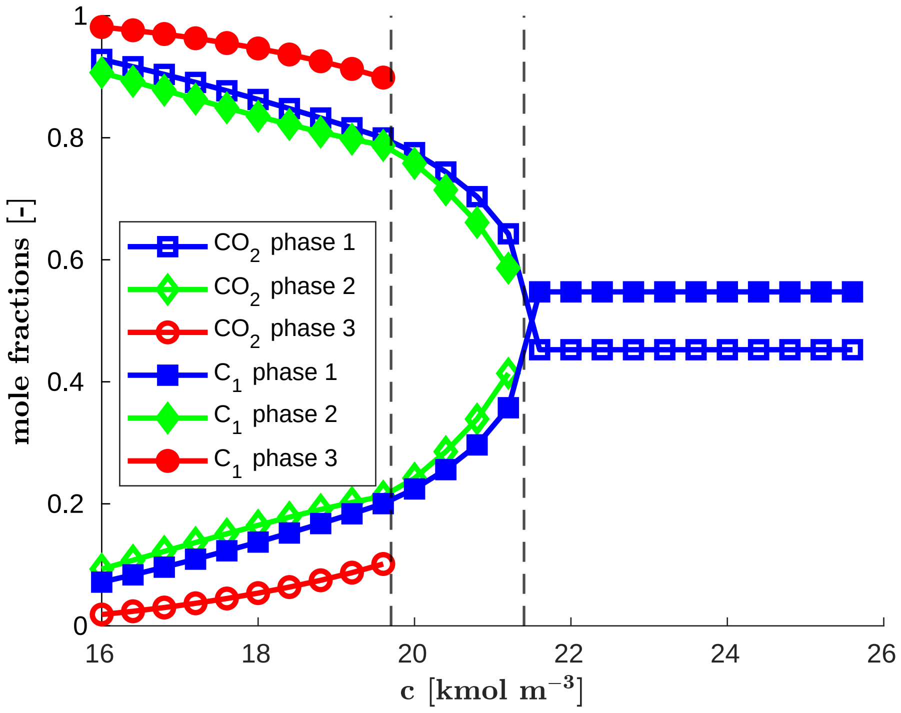


(b) equilibrium temperature

Figure 4.4: Equilibrium pressure and temperature at $u^* = -2.5 \times 10^8$ J m^{-3} as functions of the overall molar concentration c. The dashed lines represent the change of the number of phases in equilibrium. Example F4: mixture $CO_2{-}C_1$.



(a) equilibrium volume fractions

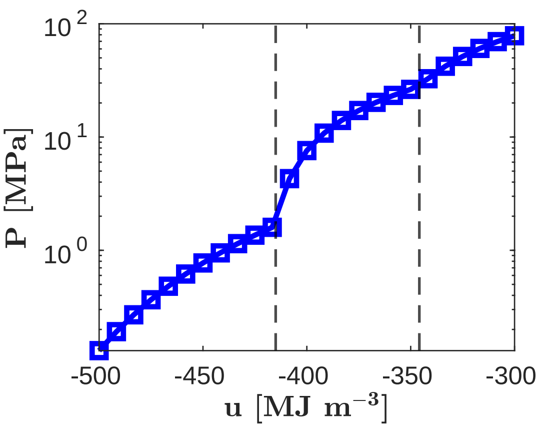


(b) equilibrium mole fractions

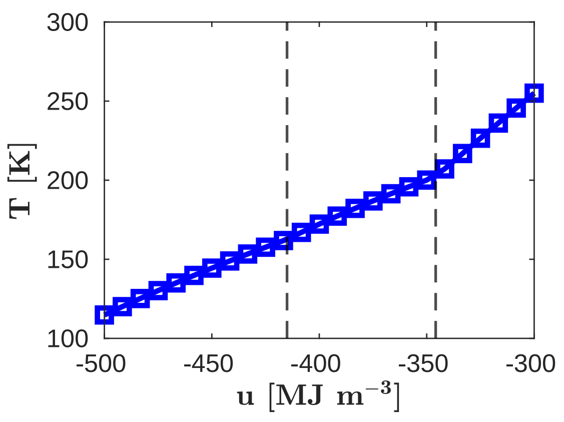
Figure 4.5: Volume fractions and mole fractions of the equilibrium phases at $u^* = -2.5\times10^8$ J m^{-3} as functions of the overall molar concentration c. The dashed lines represent the change of the number of phases in equilibrium. Example F4: mixture CO_2–C_1.

			Problem 9	
property	[unit]	phase 1	phase 2	phase 3
U	[J]	−4248079.288176	−3197022.030237	357048.818413
V	[cm^3]	2558.556768	99659.564416	163613.178816
N_{C_2}	[mol]	0.000813	5.516386	5.282801
$N_{C_3H_6}$	[mol]	0.013817	209.103028	151.683155
N_{C_3}	[mol]	0.002294	86.413985	60.083721
N_{iC_4}	[mol]	0.000395	150.396122	82.603483
N_{C_4}	[mol]	0.000684	154.757385	78.241932
N_{C_5}	[mol]	0.000005	11.577650	4.322345
N_{H_2O}	[mol]	111.866010	59.314485	28.819505
T	[K]	392.998062	392.998062	392.998062
P	[Pa]	4000181.828793	4000181.828829	4000181.828791
μ_{C_2}	[J mol^{-1}]	−2911.962428	−2911.962428	−2911.962428
$\mu_{C_3H_6}$	[J mol^{-1}]	7042.451019	7042.451019	7042.451019
μ_{C_3}	[J mol^{-1}]	3674.467867	3674.467867	3674.467867
μ_{iC_4}	[J mol^{-1}]	3526.187438	3526.187438	3526.187438
μ_{C_4}	[J mol^{-1}]	3147.780938	3147.780938	3147.780938
μ_{C_5}	[J mol^{-1}]	−7673.646488	−7673.646488	−7673.646488
μ_{H_2O}	[J mol^{-1}]	1691.295413	1691.295413	1691.295413
S^I	[J K^{-1}]		−28761.584090	
S^{III}	[J K^{-1}]		−27592.345637	
iterations	[-]		8+6	

Table 4.12: Result of Problem 9. Iterations $X + Y$ means X iterations in the two-phase and Y iterations in the three-phase UVN-flash computations. S^I denotes the entropy of the hypothetical single-phase state, while S^{II} denotes the equilibrium entropy of the stable three-phase system. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F3: LPG gas mixture with water.



(a) equilibrium pressure



(b) equilibrium temperature

Figure 4.6: Equilibrium pressure and temperature at $c^* = 26000$ mol m^{-3} as functions of the internal energy density u. The dashed lines represent the change of the number of phases in equilibrium. Example F4: mixture CO_2−C_1.

property	[unit]	Problem 7
c_{C_2}	[mol m^{-3}]	0.000003
$c_{C_3H_6}$	[mol m^{-3}]	0.000011
c_{C_3}	[mol m^{-3}]	0.000000
c_{iC_4}	[mol m^{-3}]	0.000000
c_{C_4}	[mol m^{-3}]	0.000000
c_{C_5}	[mol m^{-3}]	0.000000
c_{H_2O}	[mol m^{-3}]	50790.652384
u	[J m^{-3}]	−3029929171.784120
TPD	[Pa K^{-1}]	-9790660.167058
iterations	[-]	124

Table 4.13: Problem 7: Result of the single-phase stability testing – values presented are the concentrations and internal energy density of a trial phase that minimizes the function TPD, the value of TPD at this state, and the number of iterations in the UVN-stability test. The reference state for internal energy U is described in Section 2.7.3. Example F3: LPG gas mixture with water.

		Problem 7	
property	[unit]	phase 1	phase 2
U	[J]	−594618.569807	−16414184.030193
V	[cm^3]	196.248340	401720.351660
N_{C_2}	[mol]	0.000000	10.800000
$N_{C_3H_6}$	[mol]	0.000000	360.800000
N_{C_3}	[mol]	0.000000	146.500000
N_{iC_4}	[mol]	0.000000	233.000000
N_{C_4}	[mol]	0.000000	233.000000
N_{C_5}	[mol]	0.000000	15.900000
N_{H_2O}	[mol]	9.967581	4.032419
S^I	[J K^{-1}]	−75123.865978	
S^{II}	[J K^{-1}]	−73383.490865	

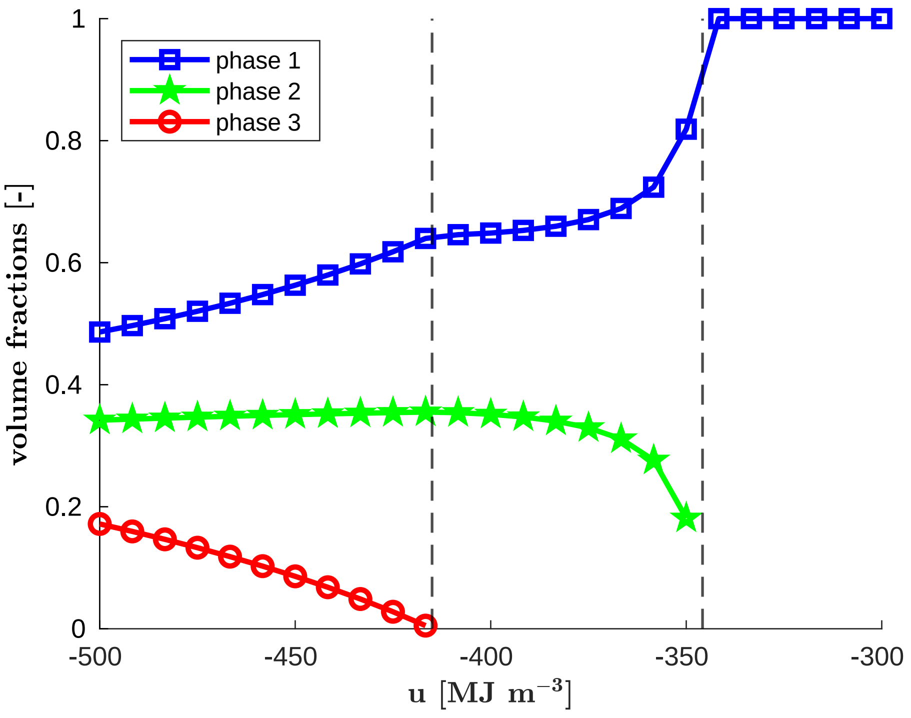
Table 4.14: Problem 7: The initial two-phase split for the two-phase UVN-phase equilibrium computation constructed using the single-phase stability analysis. S^{II} denotes the entropy of the two-phase split, while S^I is the entropy of the hypothetical (unstable) single-phase state. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F3: LPG gas mixture with water.

property	[unit]	Problem 7 phase 1	phase 2
U	[J]	−819369.896249	−16189432.703751
V	[cm^3]	275.324119	401641.275881
N_{C_2}	[mol]	0.000000	10.800000
$N_{C_3H_6}$	[mol]	0.000000	360.800000
N_{C_3}	[mol]	0.000000	146.500000
N_{iC_4}	[mol]	0.000000	233.000000
N_{C_4}	[mol]	0.000000	233.000000
N_{C_5}	[mol]	0.000000	15.900000
N_{H_2O}	[mol]	14.000000	0.000000
T	[K]	145.637031	145.637031
P	[Pa]	−5338578.032320	−5338578.032331
μ_{C_2}	[J mol^{-1}]	−8212.719344	−8212.719344
$\mu_{C_3H_6}$	[J mol^{-1}]	−5846.099219	−5846.099170
μ_{C_3}	[J mol^{-1}]	−7640.732298	−7640.731359
μ_{iC_4}	[J mol^{-1}]	−9531.520073	−9531.520028
μ_{C_4}	[J mol^{-1}]	−9937.110132	−9937.109580
μ_{C_5}	[J mol^{-1}]	−15845.701333	−15845.701333
μ_{H_2O}	[J mol^{-1}]	−31867.132701	−31867.132707
S^{II}	[J K^{-1}]	−72803.265597	
iterations	[-]	15	

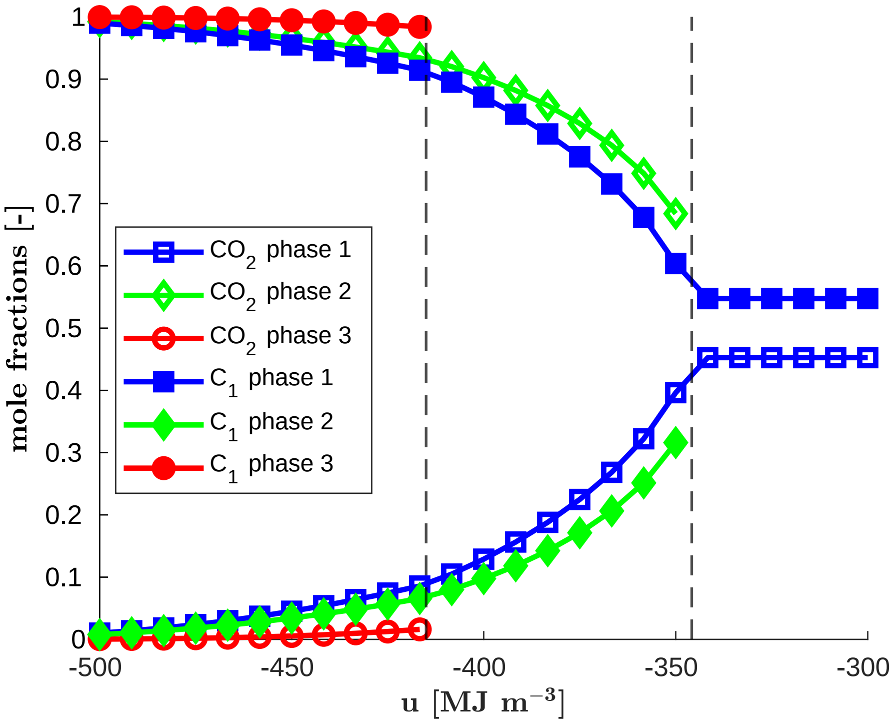
Table 4.15: Problem 7: The two-phase split obtained from the two-phase UVN-flash equilibrium computation. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F3: LPG gas mixture with water.

property	[unit]	Problem 7
c_{C_2}	[mol m^{-3}]	44.606046
$c_{C_3H_6}$	[mol m^{-3}]	4819.073335
c_{C_3}	[mol m^{-3}]	1489.109655
c_{iC_4}	[mol m^{-3}]	2262.318729
c_{C_4}	[mol m^{-3}]	5356.898847
c_{C_5}	[mol m^{-3}]	636.576083
c_{H_2O}	[mol m^{-3}]	0.007316
u	[J m^{-3}]	−513216740.034418
TPD	[Pa K^{-1}]	-669527.524959
iterations	[-]	23

Table 4.16: Problem 7: Result of the two-phase stability testing – values presented are the concentrations and internal energy density of a trial phase that minimizes function TPD, the value of TPD at this state, and the number of iterations in the UVN-stability test. The reference state for internal energy U is described in Section 2.7.3. Example F3: LPG gas mixture with water.



(a) equilibrium volume fractions



(b) equilibrium mole fractions

Figure 4.7: Volume fraction and mole fractions of equilibrium phases at $c^* = 26000$ mol m^{-3} as functions of the internal energy density u. The dashed lines represent the change of the number of phases in equilibrium. Example F4: mixture CO_2–C_1.

property	[unit]	phase 1	phase 2	phase 3
U	[J]	−819369.896249	−3145.279332	−16186287.424419
V	[cm^3]	275.324119	6.128560	401635.147322
N_{C_2}	[mol]	0.000000	0.000273	10.799727
$N_{C_3H_6}$	[mol]	0.000000	0.029534	360.770466
N_{C_3}	[mol]	0.000000	0.009126	146.490874
N_{iC_4}	[mol]	0.000000	0.013865	232.986135
N_{C_4}	[mol]	0.000000	0.032830	232.967170
N_{C_5}	[mol]	0.000000	0.003901	15.896099
N_{H_2O}	[mol]	14.000000	0.000000	0.000000
S^{III}	[J K^{-1}]		−72799.163197	

Table 4.17: Problem 7: The initial three-phase split for the three-phase UVN-phase equilibrium computation constructed using the UVN-stability analysis. S^{III} denotes the entropy of the three-phase split. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F3: LPG gas mixture with water.

initial approximation	iterations	TPD$_{\min}$
1	18	106.400525
2	19	106.430255
3	19	105.906830
4	18	106.464686
5	18	106.296797
6	18	106.369733
7	18	106.108887
8	30	0.547776
9	21	106.098904

Table 4.18: Problem 7: Numbers of iterations needed to achieve convergence in the UVN-phase stability testing and the minimum values of function TPD for each of 9 initial approximation in testing stability of the final three-phase equilibrium split.

Example F5: A single-component mixture CO_2

In the last example, we investigate the phase equilibrium of pure CO_2. We consider volume $V = 1$ m^3 containing $N_{CO_2} = 10^4$ mol of CO_2 with internal energy $U = -87211375.744478$ J. The single-phase stability test indicates that CO_2 is unstable using the first initial guess, and the test needed 71 iterations to converge with the final value of TPD = 4608.218797 Pa K^{-1} attained for $c' = 19469.178481$ mol m^{-3}, and $u' = -249975359.270471$ J m^{-3}. Then, an initial phase split with higher entropy than that of the single-phase state is constructed, see Table 4.19. This two-phase split is then used as an initial guess in two-phase flash equilibrium computation, which converges in 8 iterations to the final two-phase equilibrium state that is presented in Table 4.20. Note that the computation of the UVN-phase equilibrium for the single-component fluid proceeds without problems in the same way as for the multi-component mixture. This is not the case of the UVN-flash presented in Saha and Carroll (1997), where the single-component case required special treatment.

property	[unit]	phase 1	phase 2
U	[J]	-31246919.908809	-55964455.835669
V	[cm^3]	125000.000000	875000.000000
N_{CO_2}	[mol]	2433.647310	7566.352690
S^{II}	[J K^{-1}]	-584220.924005	

Table 4.19: Suggested two-phase split with higher entropy than that of single-phase state for pure CO_2 at $U^* = -87211375.744478$ J, $V^* = 1$ m^3, and $N^*_{\text{CO}_2} = 10^4$ mol. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F5: A single-component mixture CO_2.

property	[unit]	phase 1	phase 2
U	[J]	-70337586.354061	-16873789.390417
V	[cm^3]	518716.380364	481283.619636
N_{CO_2}	[mol]	7181.961116	2818.038884
T	[K]	299.040785	299.040785
P	[Pa]	6570486.596964	6570486.595448
μ_{CO_2}	[J mol^{-1}]	9384.232798	9384.232798
S^I	[J K^{-1}]	-584388.217059	
S^{II}	[J K^{-1}]	-583476.321606	
iterations	[-]	8	

Table 4.20: Equilibrium phase split for pure CO_2 at $U = -87211375.744478$ J, $V = 1$ m^3, and $N_{\text{CO}_2} = 10^4$ mol. The reference states for the internal energy U and entropy S are described in Section 2.7.3. Example F5: A single-component mixture CO_2.

4.3.2 Comparison of individual flash formulations

In the following section, we report the results of the Newton–Raphson method algorithm presented in Section 4.2.1. Further, we will always consider a specific physical situation and compare how different flash algorithms (different in terms of the specification variables) solve the same problem. In these examples, we will confirm our hypothesis that the flash computations are not equivalent, and their numerical performance may vary significantly. Especially, the VTN-flash computations has the same or better convergence than the PTN-flash, which may be caused by the fact that in the PTN-flash the objective function is not differentiable everywhere as discussed in Section 4.1.5 or by the choice of the primary variables (see Section 4.3.2). In Examples G1–G3, we start from an initial PTN-specification (initial temperature, pressure, and mole fractions are provided) and from the solution of the PTN-flash we calculate the specification for the VTN- and UVN-flash computations. The initial concentrations c_i^* for $i \in \widehat{n}$ are calculated using

$$c_i^* = \sum_{k=1}^{\Pi} \nu_k c_k x_{k,i}, \qquad i \in \widehat{n}, \tag{4.53}$$

where c_k is the molar concentration of phase k. The energy density u^* is calculated using

$$u^* = \sum_{k=1}^{\Pi} c^* \nu_k c_k U^{(EOS)}\left(T^*, 1, c_{k,1}, \ldots, c_{k,n}\right), \tag{4.54}$$

property	[unit]	phase 1	phase 2
x_{C_1}	[-]	0.041775	0.873807
x_{C_5}	[-]	0.958225	0.126193
ν_k	[-]	0.461805	0.538195
μ_{C_1}	[J mol^{-1}]	17583.692411	17583.692519
μ_{C_5}	[J mol^{-1}]	12107.947856	12107.947852

Table 4.21: Resulting phase split with chemical potentials of both components. Example G1: mixture C_1–C_5.

where c^* is the initial molar concentration, and $U^{(EOS)}$ is the thermal equation of state from Section 2.7.3. In Examples G4 and G5, we start with an initial VTN-specification (initial temperature and concentration are provided). From the solution of the VTN-flash, the initial specifications for the PTN- and UVN-flashes can be computed. The initial energy density is calculated using

$$u^* = \sum_{k=1}^{\Pi} S_k U^{(EOS)} \left(T^*, 1, c_{k,1}, \ldots, c_{k,n}\right), \tag{4.55}$$

and the initial pressure P^* is the equilibrium pressure resulting from the VTN-flash.

Example G1: mixture C_1–C_5

In the first example, we investigate a binary mixture of methane (C_1) and normal pentane (C_5) with initial mole fractions $x^*_{C_1} = 0.48957$ and $x^*_{C_5} = 0.51043$. The initial pressure is $P^* = 9.93516$ bar and the initial temperature is $T^* = 310.95$ K. The binary interaction parameter between C_1 and C_5 is $\delta_{C_1-C_5} = 0.041$. The PTN-flash algorithm has converged in 9 iterations. The resulting phase split together with the chemical potentials of all components in both phases are summarized in Table 4.21. From the solution, we can calculate the initial specifications for the VTN- and UVN-flash, which are presented in Table 4.22. The VTN-flash has converged in 6 iterations, and the UVN-flash has converged in 8 iterations. The progress of convergence is presented in Figure 4.8. In these three figures, we investigate the three stopping criteria (4.48)–(4.50), namely decrease of the objective function, the norm of the restricted gradient, and norm of the solution increment as functions of the number of iterations. For this purpose, we have denoted the left-hand sides of equations (4.48)–(4.50) as

$$CRIT_1 = F\left(\mathbf{Z}^{(j)}\right) - F\left(\mathbf{Z}^{(j-1)}\right), \tag{4.56}$$

$$CRIT_2 = \left\|\mathbf{Z}^{(j)} - \mathbf{Z}^{(j-1)}\right\|, \tag{4.57}$$

$$CRIT_3 = \left\|\boldsymbol{\nabla}\left(\mathbf{z}^{(j)}\right)\right\|_2. \tag{4.58}$$

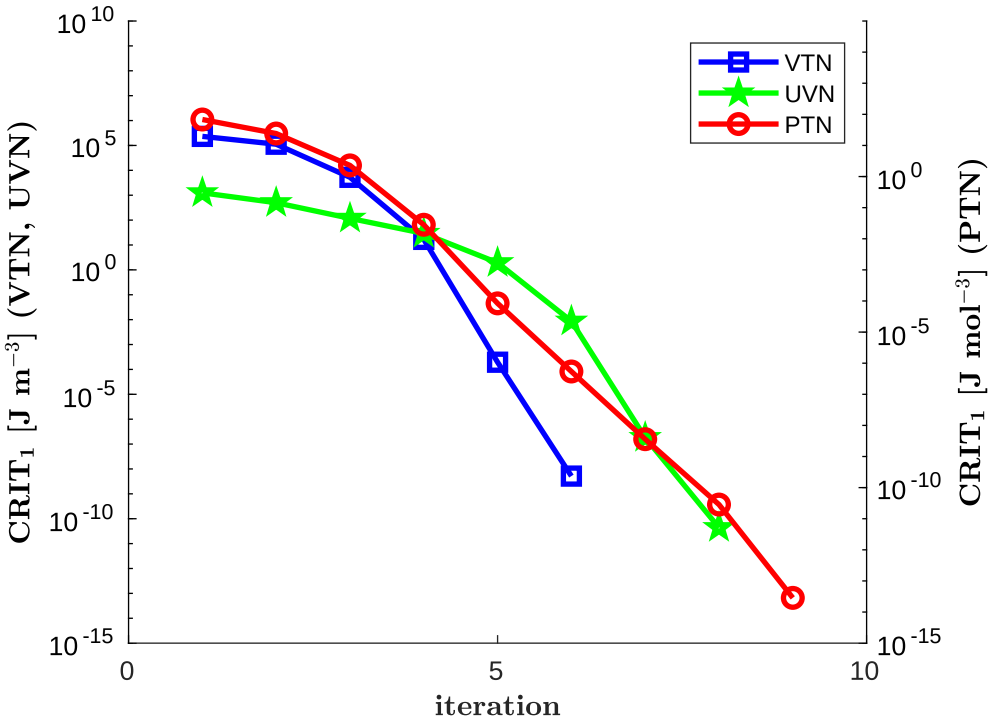
In Figure 4.8, we can see that all three computations have converged with almost identical slopes.

Example G2: mixture C_1–CO_2–C_{16}

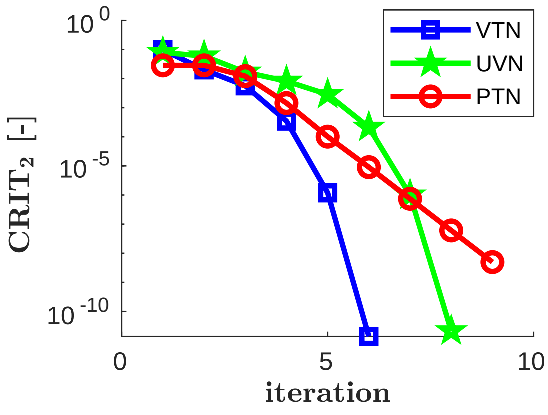
In the second example, we investigate a ternary mixture of methane (C_1), carbon dioxide (CO_2), and normal hexadecane (C_{16}) with mole fractions $x^*_{C_1} = 0.05$, $x^*_{CO_2} = 0.90$, and $x^*_{C_{16}} = 0.05$. The initial pressure is $P^* = 67$ bar and the initial temperature is $T^* = 294$ K. The binary interaction parameters are $\delta_{C_1-CO_2} = 0.15$, $\delta_{C_1-C_{16}} = 0.065$, and $\delta_{CO_2-C_{16}} = 0$. The PTN-flash algorithm

property	[unit]	VTN-flash	UVN-flash
$c^*_{C_1}$	[mol m^{-3}]	359.164316	
$c^*_{C_5}$	[mol m^{-3}]	3955.202609	
T^*	[K]	310.95	–
u^*	[J m^{-3}]	–	−97236416.256943

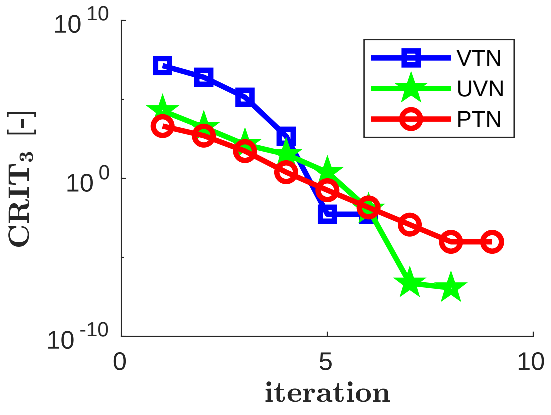
Table 4.22: Specifications of the VTN- and UVN-flash for the same physical situation. Example G1: mixture C_1–C_5.



(a) decrease of the objective function



(b) norm of the solution increment



(c) Euclidean norm of the restricted gradient

Figure 4.8: Progress of the convergence during the flash computation. Example G1: mixture C_1–C_5.

property	[unit]	phase 1	phase 2
x_{C_1}	[-]	0.138929	0.046580
x_{CO_2}	[-]	0.861058	0.901498
$x_{C_{16}}$	[-]	0.000013	0.051922
ν_k	[-]	0.037034	0.962966
μ_{C_1}	[J mol^{-1}]	17136.266476	17136.266651
μ_{CO_2}	[J mol^{-1}]	20351.567761	20351.567717
$\mu_{C_{16}}$	[J mol^{-1}]	−24196.389036	−24196.390543

Table 4.23: Resulting phase split with chemical potentials of all components. Example G2: mixture C_1−CO_2−C_{16}.

property	[unit]	phase 1	phase 2
x_{C_1}	[-]	0.141623	0.049599
x_{CO_2}	[-]	0.858349	0.900182
$x_{C_{16}}$	[-]	0.000028	0.050219
ν_k	[-]	0.004350	0.995650
μ_{C_1}	[J mol^{-1}]	17176.731680	17272.909313
μ_{CO_2}	[J mol^{-1}]	20344.923885	20347.209260
$\mu_{C_{16}}$	[J mol^{-1}]	−22376.514270	−24248.128053

Table 4.24: Resulting phase split using the conventional PTN-flash solver together with chemical potentials of all components in both phases. Example G2: mixture C_1−CO_2−C_{16}.

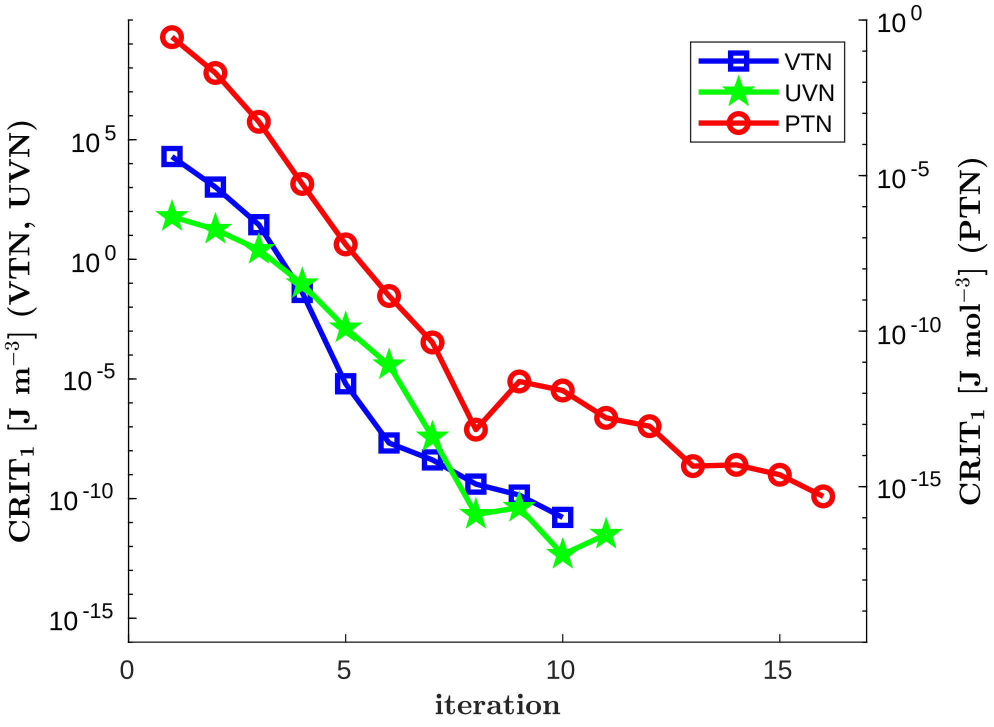
has converged in 18 iterations. The resulting phase split together with chemical potentials of all component in both phases are summarized in Table 4.23. The initial specifications for the VTN- and UVN-flash computation are presented in Table 4.25. The VTN-flash has converged in 12 iterations and the UVN-flash in 13 iterations. The progress of decrease of the objective function, the norm of the restricted gradient, and the norm of the solution increment are presented in Figure 4.9. From these figures, we can see that in the UVN- and PTN-flash computations the criteria for the norm of the solution increment (CRIT_2) and of the restricted gradient (CRIT_3) are fulfilled already in iterations 8 and 11, respectively. However, the last condition (CRIT_1) for the decrease of the objective function is still not satisfied, and the computation continues. This example shows that all three stopping criteria are needed to obtain accurate results. Moreover, in Table 4.24, the resulting phase split using a conventional solver is presented. The comparison between the solutions is presented at the end of Section 4.3.2.

Example G3: mixture $\mathbf{CO_2}$−$\mathbf{N_2}$−$\mathbf{C_i}$

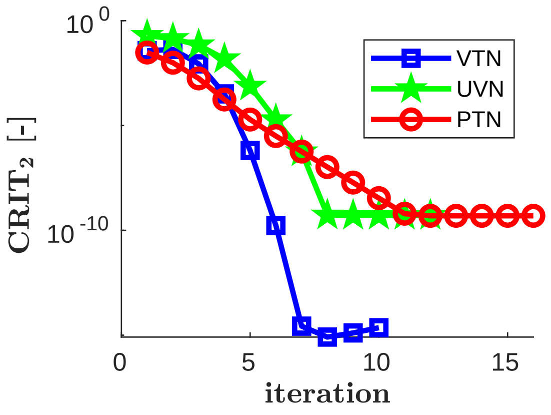
In the third example, we investigate an 11 component mixture of carbon dioxide (CO_2), nitrogen (N_2), and nine alkanes, more precisely of methane (C_1), ethane (C_2), propane (C_3), iso-butane (iC_4), normal butane (C_4), iso-pentane (iC_5), normal pentane (C_5), hexane (C_6), and a pseudocomponent C_{7+} with molar fractions $x^*_{CO_2} = 0.02980$, $x^*_{N_2} = 0.00120$, $x^*_{C_1} = 0.66870$, $x^*_{C_2} = 0.06860$, $x^*_{C_3} = 0.03960$, $x^*_{iC_4} = 0.00730$, $x^*_{C_4} = 0.01820$, $x^*_{iC_5} = 0.00830$, $x^*_{C_5} = 0.01030$, $x^*_{C_6} = 0.01400$, and $x^*_{C_{7+}} = 0.13400$. The initial temperature is $T^* = 295.9$ K and the initial pressure is $P^* = 156.8557$ bar. The binary interaction parameters between all components are shown in Table 4.26. In this example the PTN-flash has converged in 206 iterations. The

property	[unit]	VTN-flash	UVN-flash
$c^*_{C_1}$	[mol m^{-3}]	657.809193	
$c^*_{CO_2}$	[mol m^{-3}]	12405.774622	
$c^*_{C_{16}}$	[mol m^{-3}]	705.694805	
T^*	[K]	294	–
u^*	[J m^{-3}]	–	−200605469.872798

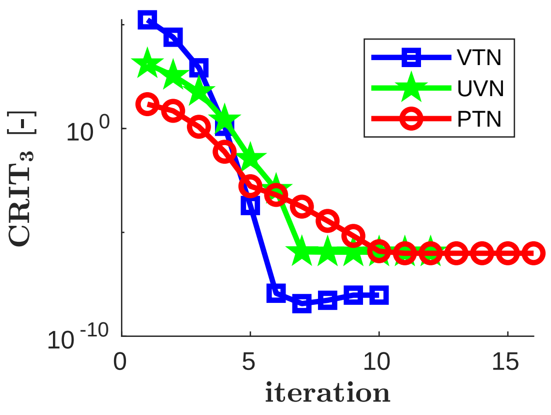
Table 4.25: Specifications of the VTN- and UVN-flash for the same physical situation. Example G2: mixture $C_1-CO_2-C_{16}$.



(a) decrease of the objective function



(b) norm of the solution increment



(c) Euclidean norm of the restricted gradient

Figure 4.9: Progress of the convergence during the flash computation. Example G2: mixture $C_1-CO_2-C_{16}$.

	CO_2	N_2	C_1	C_2	C_3	iC_4	C_4	iC_5	C_5	C_6	C_{7+}
CO_2	0	0	0.150	0.150	0.150	0.150	0.150	0.150	0.150	0.150	0.150
N_2	0	0	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100
C_1	0.150	0.100	0	0.034	0.036	0.038	0.038	0.041	0.041	0.043	0.056
C_2	0.150	0.100	0.034	0	0	0	0	0	0	0	0
C_3	0.150	0.100	0.036	0	0	0	0	0	0	0	0
iC_4	0.150	0.100	0.038	0	0	0	0	0	0	0	0
C_4	0.150	0.100	0.038	0	0	0	0	0	0	0	0
iC_5	0.150	0.100	0.041	0	0	0	0	0	0	0	0
n_5	0.150	0.100	0.041	0	0	0	0	0	0	0	0
C_6	0.150	0.100	0.043	0	0	0	0	0	0	0	0
C_{7+}	0.150	0.100	0.056	0	0	0	0	0	0	0	0

Table 4.26: The binary interaction parameters between all components. Data taken over from Firoozabadi (2016). Example G3: mixture CO_2–N_2–C_i.

resulting phase split together with the chemical potentials of all components in both phases are presented in Table 4.27. The initial specification for the VTN- and UVN-flash computation are presented in Table 4.29. The VTN-flash has converged in 7 iterations, the UVN-flash in 19 iterations. The progress of decrease of the objective function, the norm of the solution increment, and the norm of the restricted gradient are presented in Figure 4.10. This example clearly shows that our implementation of the PTN-flash computation with the Newton–Raphson iterations is not as fast as the VTN-flash computation. From all three figures, it can be seen that the PTN-flash convergence is only linear, and the number of iterations needed for convergence is much higher compared to the other two formulations. Moreover, in Table 4.28, the resulting phase split using a conventional solver is presented. The comparison between the solutions is presented later at the end of Section 4.3.2.

Example G4: mixture C_1–CO_2

In the fourth example, we investigate a binary mixture of methane (C_1) and carbon dioxide (CO_2) with molar fractions $x^*_{C_1} = 0.547413$ and $x^*_{CO_2} = 0.452587$. The binary interaction parameter between C_1 and CO_2 is $\delta_{C_1-CO_2} = 0.15$. First, as in Jindrová and Mikyška (2015), the VTN-phase stability test was performed in the temperature range $T^* \in [180, 260]$ K and the whole range of feasible molar concentrations. Boundaries between the single-phase, two-phase, and three-phase region are shown in Figure 4.11a. A grid with 200×200 points was used. Then, we compared different flash algorithms in physical situations of $T^* = 220$ K and molar concentrations in the range $c^* \in [3000, 17000]$ mol m^{-3}. In Figure 4.11b, numbers of iterations of the VTN-, UVN- and PTN-flash solver are presented. In this figure, we also include data from a conventional PTN-flash, see the end of Section 4.3.2 for more information. From this figure, one can see that with a higher molar concentration, when the phase boundary is being approached, the PTN-flash computation needs more iterations for convergence. However, in the VTN- and UVN-flash computations, the number of iterations is almost constant for every molar concentration.

Example G5: mixture C_1–CO_2–H_2S

In the fifth example, we investigate a ternary mixture of methane (C_1), carbon dioxide (CO_2), and hydrogen disulfide (H_2S) with mole fractions $x^*_{C_1} = 0.4023$, $x^*_{CO_2} = 0.0988$, and $x^*_{H_2S} = 0.4989$.

property	[unit]	phase 1	phase 2
$x_{\mathrm{CO_2}}$	[-]	0.032681	0.024743
$x_{\mathrm{N_2}}$	[-]	0.001689	0.000342
$x_{\mathrm{C_1}}$	[-]	0.857059	0.338079
$x_{\mathrm{C_2}}$	[-]	0.062048	0.080100
$x_{\mathrm{C_3}}$	[-]	0.026078	0.063334
$x_{\mathrm{iC_4}}$	[-]	0.003851	0.013354
$x_{\mathrm{C_4}}$	[-]	0.008013	0.036081
$x_{\mathrm{iC_5}}$	[-]	0.002712	0.018108
$x_{\mathrm{C_5}}$	[-]	0.002941	0.023217
$x_{\mathrm{C_6}}$	[-]	0.002604	0.034003
$x_{\mathrm{C_7+}}$	[-]	0.000324	0.368639
ν_k	[-]	0.637059	0.362941
$\mu_{\mathrm{CO_2}}$	[J mol^{-1}]	14160.356303	14160.356619
$\mu_{\mathrm{N_2}}$	[J mol^{-1}]	8852.593725	8852.594027
$\mu_{\mathrm{C_1}}$	[J mol^{-1}]	22951.153205	22951.153508
$\mu_{\mathrm{C_2}}$	[J mol^{-1}]	14675.131950	14675.132203
$\mu_{\mathrm{C_3}}$	[J mol^{-1}]	11077.596731	11077.596914
$\mu_{\mathrm{iC_4}}$	[J mol^{-1}]	5239.010667	5239.010769
$\mu_{\mathrm{C_4}}$	[J mol^{-1}]	6723.790309	6723.790420
$\mu_{\mathrm{iC_5}}$	[J mol^{-1}]	2944.546598	2944.546635
$\mu_{\mathrm{C_5}}$	[J mol^{-1}]	2840.684984	2840.685014
$\mu_{\mathrm{C_6}}$	[J mol^{-1}]	1156.026651	1156.026593
$\mu_{\mathrm{C_7+}}$	[J mol^{-1}]	−12671.659672	−12671.660058

Table 4.27: Resulting phase split together with chemical potential of all components in both phases. Example G3: mixture $\mathrm{CO_2}$−$\mathrm{N_2}$−$\mathrm{C_i}$.

The binary interaction parameters are $\delta_{\mathrm{C_1-CO_2}} = 0.13$, $\delta_{\mathrm{C_1-H_2S}} = 0.095$, and $\delta_{\mathrm{CO_2-H_2S}} = 0.097$. First, the VTN-phase stability test was performed in the temperature range $T^* \in [100, 350]$ K and the whole range of feasible molar concentration. Boundaries between the single-phase, two-phase, three-phase, and four-phase regions are shown in Figure 4.12a. A grid with 200×200 points was used. Then, we compared different flash algorithms for physical situations of $T^* = 250$ K and molar concentrations in the range $c^* \in [4000, 20000]$ mol m^{-3}. In Figure 4.12b, numbers of iterations of the VTN-, UVN- and PTN-flash solver are presented. As in Example G4, data from the conventional PTN-flash are included, see the end of Section 4.3.2 for more information. As in Example G4, with a higher molar concentration, the PTN-flash computation needs more iterations for convergence. Different is the behaviour of the UVN-flash computation. For low molar concentrations, the number of iterations is around 25, but when increasing the molar concentration above 7000 mol m^{-3} the number of iterations decreases to 12 and remains around that value till the end of the computation.

Comparison of the PTN-flash with the conventional formulation

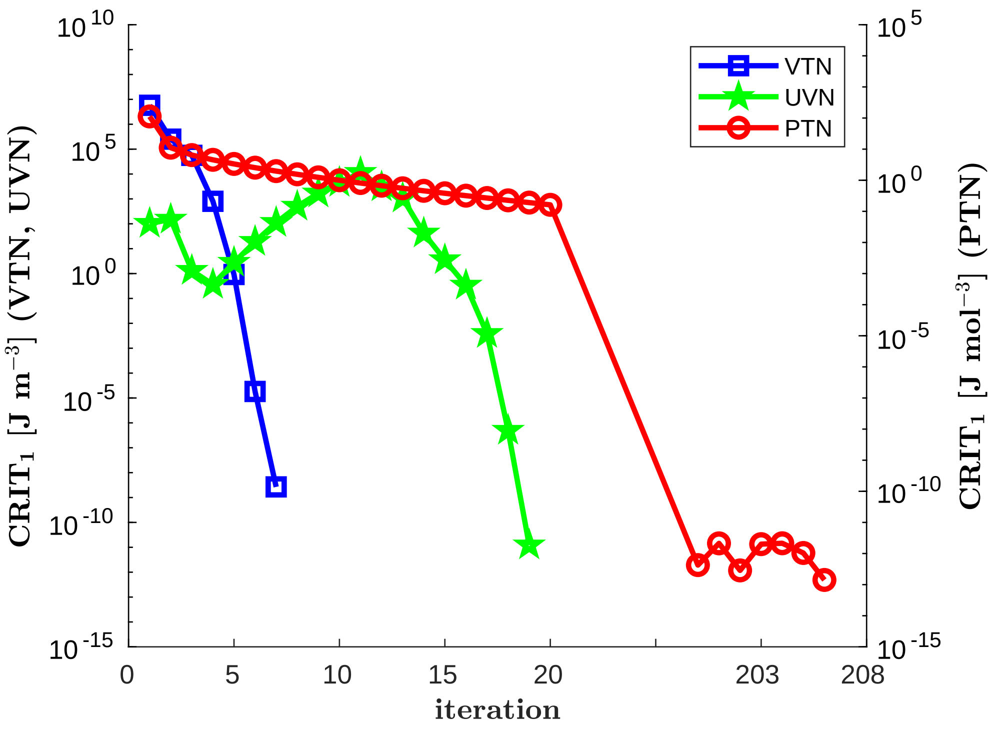
Our PTN-flash formulation presented above uses $\Pi - 1$ phase mole fractions and $(\Pi - 1) \times (n - 1)$ components' mole fractions as primary variables. This set of independent variables is exactly that presented for the first time in Fussell and Yanosik (1978); Fussell (1979). Results reported in the previous section indicate that this implementation of the PTN-flash needs much more

property	[unit]	phase 1	phase 2
x_{CO_2}	[-]	0.032406	0.025802
x_{N_2}	[-]	0.001723	0.000397
x_{C_1}	[-]	0.861537	0.372836
x_{C_2}	[-]	0.060308	0.081323
x_{C_3}	[-]	0.024783	0.062333
x_{iC_4}	[-]	0.003581	0.013006
x_{C_4}	[-]	0.007481	0.034646
x_{iC_5}	[-]	0.002490	0.017214
x_{C_5}	[-]	0.002704	0.021954
x_{C_6}	[-]	0.002359	0.031861
x_{C_7+}	[-]	0.000628	0.338628
ν_k	[-]	0.605410	0.394590
μ_{CO_2}	[J mol^{-1}]	14143.188989	14219.743729
μ_{N_2}	[J mol^{-1}]	8892.253125	9123.313691
μ_{C_1}	[J mol^{-1}]	22961.512188	23115.194361
μ_{C_2}	[J mol^{-1}]	14616.715573	14680.674694
μ_{C_3}	[J mol^{-1}]	10973.893296	11015.018484
μ_{iC_4}	[J mol^{-1}]	5089.314460	5145.138482
μ_{C_4}	[J mol^{-1}]	6586.461648	6607.955629
μ_{iC_5}	[J mol^{-1}]	2774.135731	2802.161068
μ_{C_5}	[J mol^{-1}]	2675.697104	2690.383824
μ_{C_6}	[J mol^{-1}]	963.722977	982.363635
$\mu_{C_{7+}}$	[J mol^{-1}]	−10924.119746	−12782.958426

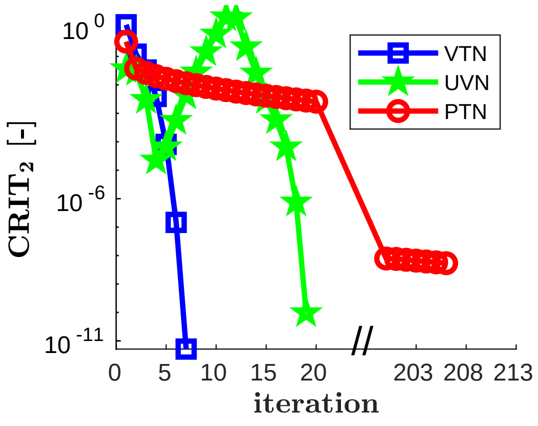
Table 4.28: Resulting phase split using the conventional PTN-flash solver together with chemical potentials of all components in both phases. Example G3: mixture CO_2–N_2–C_i.

property	[unit]	VTN-flash	UVN-flash
$c^*_{CO_2}$	[mol m^{-3}]	265.666841	
$c^*_{N_2}$	[mol m^{-3}]	10.740300	
$c^*_{C_1}$	[mol m^{-3}]	5975.497098	
$c^*_{C_2}$	[mol m^{-3}]	610.072314	
$c^*_{C_3}$	[mol m^{-3}]	351.064668	
$c^*_{iC_4}$	[mol m^{-3}]	64.607924	
$c^*_{C_4}$	[mol m^{-3}]	160.897042	
$c^*_{iC_5}$	[mol m^{-3}]	73.269184	
$c^*_{C_5}$	[mol m^{-3}]	90.876259	
$c^*_{C_6}$	[mol m^{-3}]	123.362969	
$c^*_{C_{7+}}$	[mol m^{-3}]	1177.967621	
T^*	[K]	295.9	–
u^*	[J m^{-3}]	–	−139163012.871018

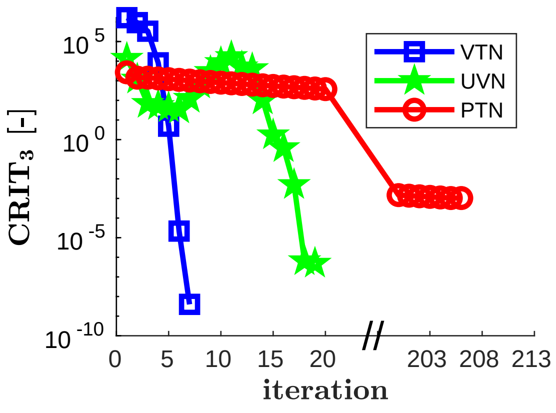
Table 4.29: Specifications of the VTN- and UVN-flash for the same physical situation. Example G3: mixture CO_2–N_2–C_i.



(a) decrease of the objective function



(b) norm of the solution increment

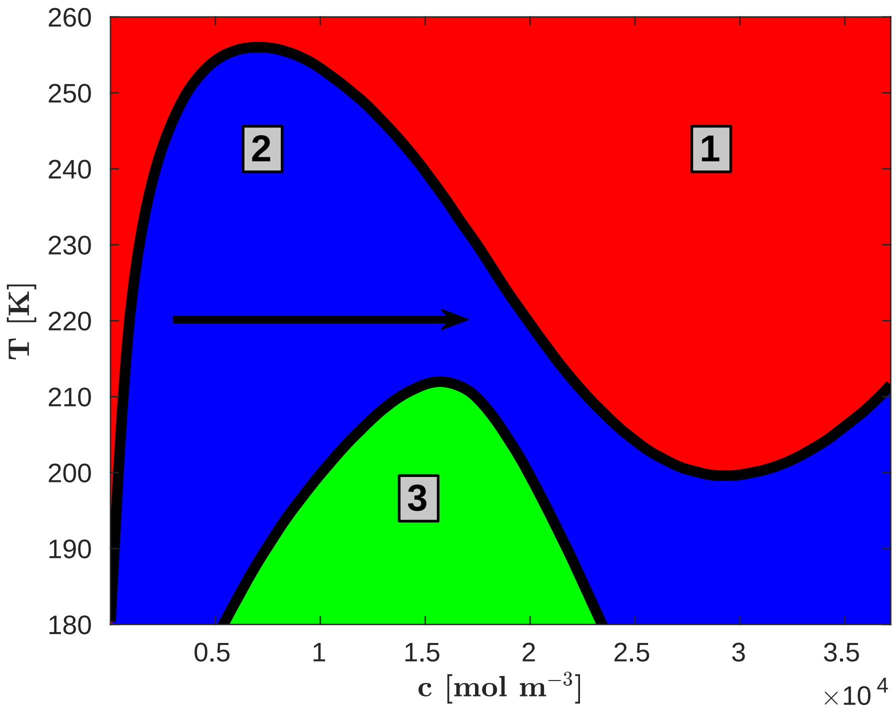


(c) Euclidean norm of the restricted gradient

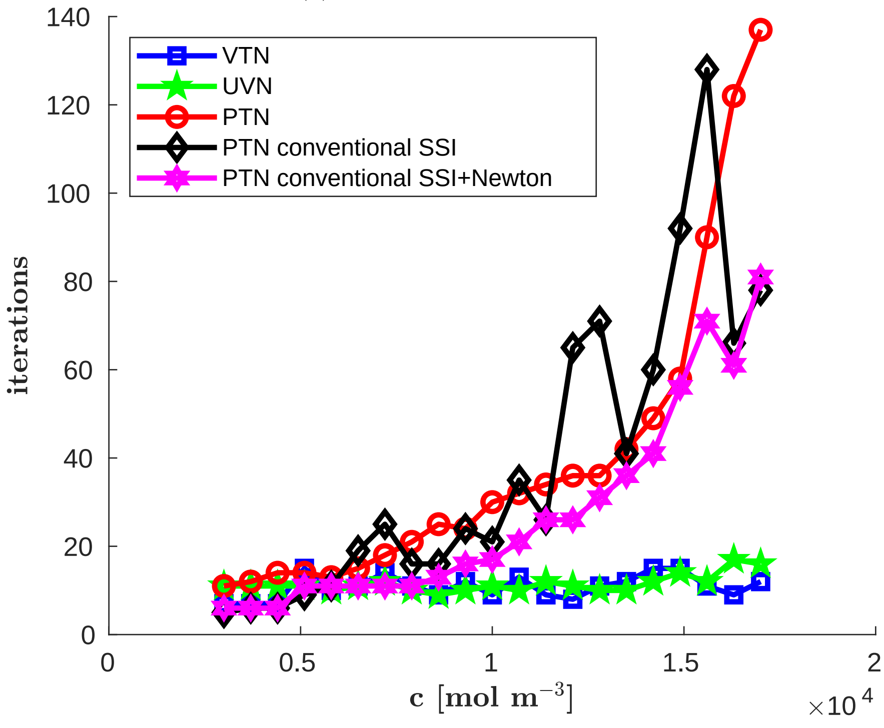
Figure 4.10: Progress of the convergence during the flash computation. The uninteresting area between iteration 20 and 200 is omitted. Example G3: mixture $CO_2-N_2-C_i$.

iterations compared to the other two flash formulations, especially close to the phase boundary. Contemporary formulations of the PTN-flash may not suffer from this deficiency. Therefore, we would like to discuss the conventional formulation of the PTN-flash in this thesis as well.

Let us note that the increase of the number of iterations close to the phase boundary has also been reported for the volume-based PTN-flash by Nichita (2018b) when the flash computation was initialized using the ideal equilibrium constants. According to Petitfrere and Nichita (2016), it is recommended to initialize the volume based PTN-flash computation using the results of stability testing (Nichita (2018c)). At the same time, the volume-based PTN-flash formulation can be derived readily from the VTN-flash. It is; therefore, in principle, easy to transform the VTN-flash code to the volume-based PTN-formulation (see Michelsen (1999b, 2012); Nichita (2018b)). However, the volume based PTN-formulation does not fit our general framework given

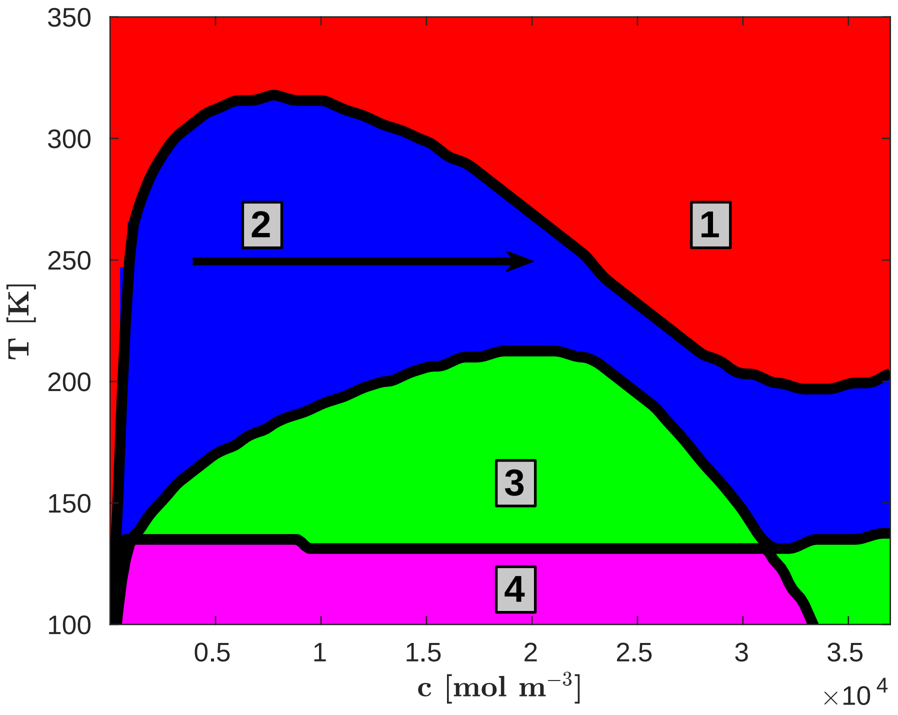


(a) approximate boundaries

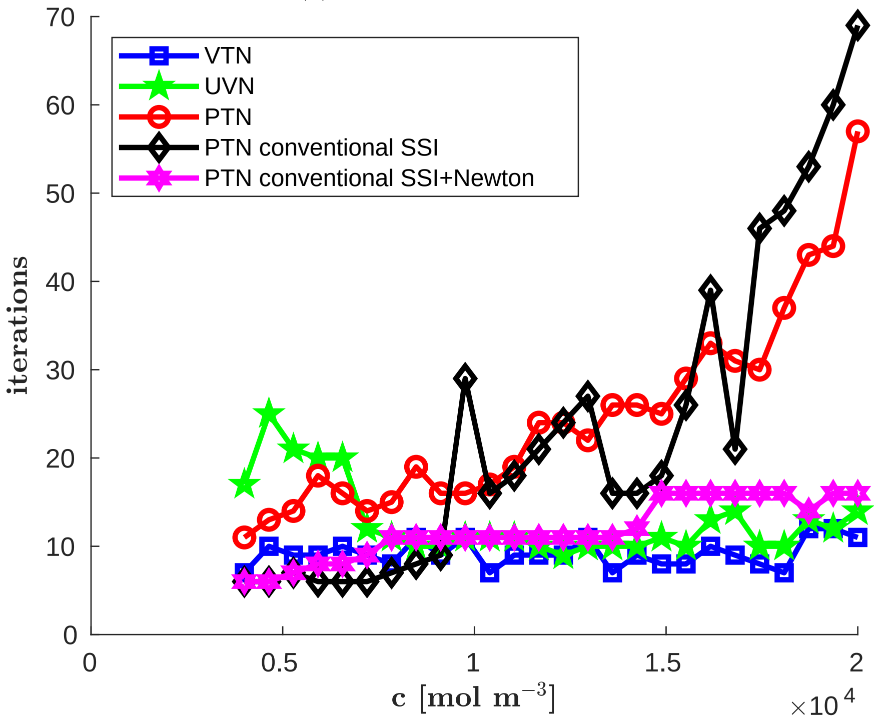


(b) numbers of iterations

Figure 4.11: Approximate boundaries between the single-phase, two-phase and three-phase regions in the cT space (top). Numbers of iterations needed for convergence in the flash computation. In this case the number of iterations is the sum of all computations (bottom). Example G4: mixture $C_1{-}CO_2$.



(a) approximate boundaries



(b) numbers of iterations

Figure 4.12: Approximate boundaries between the single-phase, two-phase and three-phase regions in the cT space (top). Numbers of iterations needed for convergence in the flash computation. In this case the number of iterations is the sum of all computations (bottom). Example G5: mixture $C_1{-}CO_2{-}H_2S$.

by equations (4.5)–(4.6). Similarly, other approaches to the PTN-stability such as volume-based formulation (Nagarajan et al. (1991); Nichita (2018c); Nichita et al. (2006b); Souza et al. (2006); Xu et al. (2002)), where the molar densities or other volume-based variables are the choice of primary variables, do not fit our general stability formulation (see Section 3.1.1). To get working implementations of the volume-based PTN-stability and volume-based PTN-flash, we would have to develop a stand-alone code. For this reason, the volume-based PTN-stability and volume-based PTN-flash formulations are not discussed in this thesis any more.

Instead, in this section, we compare our implementation of the PTN-flash with a conventional PTN-flash solver, which uses natural logarithms of equilibrium K-values and the gas-phase mole fractions as primary variables (see Firoozabadi (2016); Haugen et al. (2010); Li and Firoozabadi (2012); Petitfrere and Nichita (2016)). Neither the volume-based PTN-flash, nor the conventional PTN-flash formulation fit the unifying framework given by equations (4.5) and (4.6). However, for the conventional PTN-formulation, we have a stand-alone code available, which can be used for the comparison.

As the conventional PTN-flash solver we have used an implementation developed in RERI (Reservoir Engineering Research Institute) provided to us by prof. Abbas Firoozabadi. In the conventional PTN-stability testing, a quasi-Newton method with BFGS (Broyden–Fletcher–Goldfarb–Shanno, see Broyden (1970); Fletcher (1970); Goldfarb (1970); Shanno (1970)) update is used to minimize the modified TPD function of Michelsen (1982a), which leads to the unconstrained optimization. In the PTN-flash computation the Successive Substitution Iteration (SSI) and Newton–Raphson methods are adopted. The SSI is accelerated with the General-Dominant-Eigenvalue method, which is applied after every five iterations. The natural logarithms of equilibrium K-values are chosen as primary variables. The computation starts with the SSI. When a switching criterion is met, the computation switches to the Newton–Raphson method. If a Newton–Raphson iteration is not convergent, the computation is switched back to the SSI. Such a combination of a slow but robust method (SSI) and the fast solving method (Newton–Raphson), which requires a good initial guess, is known to be a fast and robust solver of the PTN-flash problem. The details of implementation can be found elsewhere, e.g., Firoozabadi (2016); Haugen et al. (2010); Li and Firoozabadi (2012).

Let us emphasize that the stopping criterion used in the conventional solver differs significantly from the one used in our unified implementation. The available conventional PTN-flash implementation does not allow us to get reports on the progress of convergence in terms of the criteria that we have used in the first three examples. In Examples G4 and G5, the number of iterations can be evaluated, but the comparison of the individual numbers of iterations with those obtained with the conventional PTN-code must be done with care, as the accuracy of the results can be very different. In some cases, the two implementations of the PTN-flash lead to different results because the too loose stopping criterion can lead to the premature stop of iterations before convergence. On the other hand, we can study the behaviour of the conventional PTN-flash solver when the phase boundary is being approached. The conventional PTN-flash solver will be used in two settings. The first option is the combined SSI and Newton–Raphson iterations. The other option is to use the SSI method only for the PTN-flash computation. Below, we summarize the performance of the conventional PTN-flash solver on the five examples from the previous section.

In Example G1: mixture C_1-C_5, the conventional PTN-flash solver has converged to the same solution as our code after 4 SSI and 2 Newton–Raphson iterations. With SSI iterations only, the solver has converged after 6 iterations.

In Example G2: mixture $C_1-CO_2-C_{16}$, the conventional PTN-flash solver has converged after 5 SSI and 2 Newton–Raphson iterations or with 6 SSI iterations only. However, the resulting phase split, which is presented in Table 4.24 is different from ours reported in Table

4.23. Comparing the values of the objective function F, which are 17963.659835 Jmol^{-1} for the result from the conventional PTN-flash and 17963.404751 Jmol^{-1} for our result, we can state that our results are more precise. Moreover, in Table 4.24, the chemical potentials of all component of the resulting phase split from the conventional PTN-flash solver are presented. We see that the necessary condition of phase equilibrium is not satisfied, as the chemical potentials of components in both phases are not equal.

In Example G3: mixture $CO_2{-}N_2{-}C_i$, using the conventional PTN-flash solver, the phase split from Table 4.28 is obtained after 7 SSI and 3 Newton–Raphson iterations or with 11 SSI iterations only. As in Example G2, our result and the results obtained using the conventional PTN-flash are not the same. The values of the objective function are 15760.288044 Jmol^{-1} for the phase split from the conventional PTN-flash solver and 15757.923293 Jmol^{-1} for our implementation. Comparing these values, we again can state that our phase split result is more precise. The chemical potentials, which are provided together with the resulting phase split from the conventional PTN-flash solver in Table 4.28, clearly show that the necessary condition of phase equilibrium is not satisfied. Most probably, the conventional PTN-flash solver was stopped too early. This example shows the importance of stopping criteria for the evaluation of different flash formulations. The stopping criteria that were presented in this thesis within the unified framework lead to better results in comparison with the conventional approaches.

In Example G4: mixture $C_1{-}CO_2$, we provide in Figure 4.11b the numbers of iterations obtained by the conventional PTN-flash in the two settings mentioned above. In Figure 4.11b, we can observe almost identical behaviour of our PTN-flash and the conventional PTN-flash solver when using the combined SSI and Newton–Raphson iterations. In each computation, the number of iterations of the conventional PTN-flash is lower than ours. However, when approaching the phase boundary, the number of iterations of both solvers increase. The computation with the SSI method only is more disorganized, but the number of iterations again increases when the phase boundary is being approached.

In Example G5: mixture $C_1{-}CO_2{-}H_2S$, we use the same settings of the conventional PTN-flash solver as in Example G4. In this example, we have observed (see Figure 4.12b) different behaviour of the conventional PTN-flash solver with SSI and Newton–Raphson iterations. The number of iterations in this settings remains constant around 18 even if the phase boundary is being approached. In our implementation of the PTN-flash, the number of iterations increases up to 57. However, the number of iterations of the VTN- and UVN-flash are lower than the number of iterations of the conventional PTN-flash in all computation except the first 6 computation, where the UVN-flash needed more iterations. The conventional PTN-flash with SSI iterations only shows similar behaviour as in Example G4. As the phase boundary is being approached, the number of iterations increases and is more or less comparable to our PTN-flash solver.

5 Application of phase equilibrium computation in multi-phase compositional flow in porous medium

In Chapters 3 and 4, we presented several numerical algorithms for the computation of the phase equilibrium states of a multi-component mixture. In the examples, we showed that the most robust, reliable, and efficient are the algorithms based on the Newton–Raphson method (defined in Sections 3.2.3 and 4.2.1). Now, we present one of the possible applications of these algorithms in the area of the numerical modelling of the fluid flow in a porous medium. Consider a porous medium (e.g., a hydrocarbon reservoir) filled with a mixture of n components at a constant temperature T^*. Based on the injection and the boundary conditions, we want to study the compositional flow of this multi-component fluid through the porous medium. These simulations of the compositional flow are applicable in environmental sciences (carbon dioxide sequestration) or the energy sector (enhanced oil recovery).

First, we present few basic assumptions:

- Diffusion is neglected.
- The porous medium has a fixed position during the simulation, i.e., it is incompressible.
- The velocity is described by Darcy flow.
- The temperature is constant during the simulation.
- The multi-component mixture is considered compressible, multi-phase, and fully miscible.

With respect to the given assumptions, our mathematical model will be given. Then, a numerical solution based on the mixed-hybrid finite element method (MHFEM) will be derived. Lastly, examples showing the performance of the numerical algorithm will be presented.

5.1 Mathematical model

In this section, we present a compact form of our mathematical model. For the full derivation of the equations, see, e.g., Kolev (2005) or Slattery (1999). Our mathematical model is designed to describe the multi-phase multi-component compressible miscible flow in a porous medium. The mathematical model consists of the mass conservation equation of each component, extended Darcy's law for each phase, and an appropriate set of the initial and boundary conditions. The phase split is computed using the phase equilibrium computation in the VTN-specification. Classically, the PTN-specification is used to determine the phase equilibrium state, see, e.g., Hoteit and Firoozabadi (2006a) or Moortgat et al. (2011). However, we choose to use the VTN-specification based on our experience with the flash computation.

5.1.1 Transport equations

The mass balance equation for component $i \in \widehat{n}$ is

$$\frac{\partial(\phi c_i)}{\partial t} + \boldsymbol{\nabla} \cdot \mathbf{q}_i = f_i, \tag{5.1}$$

where ϕ [-] is the porosity, c_i [mol m^{-3}] is the molar concentration (density) of the i-th component, $\mathbf{q}_i$ [mol m^{-2} s^{-1}] is the flux of the i-th component, and f_i [mol m^{-3} s^{-1}] is the source/sink of the i-th component. Since the porous medium is incompressible, i.e., $\frac{\partial \phi}{\partial t} = 0$, equation (5.1) can be written as

$$\phi \frac{\partial c_i}{\partial t} + \boldsymbol{\nabla} \cdot \mathbf{q}_i = f_i. \tag{5.2}$$

For a multi-phase system without diffusion, the flux $\mathbf{q}_i$ can be expressed as

$$\mathbf{q}_i = \sum_{\alpha=1}^{\Pi} c_{\alpha,i} \mathbf{u}_\alpha, \tag{5.3}$$

where $c_{\alpha,i}$ [mol m^{-3}] is the concentration of the i-th component in phase α, Π is the number of phases present in the phase split, and $\mathbf{u}_\alpha$ [m s^{-1}] is the velocity of phase α. The relation between concentrations c_i and $c_{\alpha,i}$ is given by the phase equilibrium computation and presented in Section 4.1.2. Note, here for clarity, we omit the superscript $*$ denoting the initial concentrations. For the derivation of equation (5.1), the reader is referred to, e.g., Slattery (1999).

5.1.2 Extended Darcy's law

In fluid dynamics, the velocity field $\mathbf{u}$ is generally given by the Navier-Stokes equations

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \boldsymbol{\nabla} \cdot \left(\rho \mathbf{u}\mathbf{u}^{\mathrm{T}}\right) = -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot \sigma + \rho \mathbf{g}, \tag{5.4}$$

where ρ is the mass density, σ is the stress tensor, and $\mathbf{g}$ is the gravity vector. For the derivation of the Navier-Stokes equations, see, e.g., Drazin (2006). In our model, we replace the Navier-Stokes equations by Darcy's law which can be derived from the Navier-Stokes equations using the averaging procedure, see Neuman (1977) or Hassanizadeh (1986). In the literature, there is consent that Darcy's law is valid for the Reynolds number Re up to the range 1 to 10. See, e.g., Bejan (2013), Kaviany (1991), or Bear (1988). Therefore, the velocity of each phase is modeled using Darcy's law

$$\mathbf{u}_\alpha = -\lambda_\alpha \mathbf{K} \left(\boldsymbol{\nabla} p - \rho_\alpha \mathbf{g}\right), \tag{5.5}$$

where λ_α [kg^{-1} m s] is the mobility of phase α, $\mathbf{K}$ [m^2] is the intrinsic permeability tensor, p [Pa] is the pressure, ρ_α [kg m^{-3}] is the mass density of the phase α, and $\mathbf{g}$ [m s^{-2}] is the gravity acceleration. The mobility and the density are calculated using

$$\lambda_\alpha = \frac{k_{\mathrm{r},\alpha}(S_\alpha)}{\eta_\alpha\left(T, c_{\alpha,1}, \ldots, c_{\alpha,n}\right)}, \tag{5.6}$$

$$\rho_\alpha = \sum_{i=1}^{n} c_{\alpha,i} M_i, \tag{5.7}$$

where $k_{\mathrm{r},\alpha}$ [-] is the relative permeability of phase α, S_α [-] is the saturation of phase α, η_α [kg m^{-1} s^{-1}] is the dynamic viscosity of phase α, and M_i [kg mol^{-1}] is the molar weight of the i-th component. In this thesis, we are using two models to compute the relative permeability. The first option is the linear model

$$k_{\mathrm{r},\alpha}(S_\alpha) = S_\alpha. \tag{5.8}$$

The second option is the quadratic model

$$k_{\mathrm{r},\alpha}(S_\alpha) = S_\alpha^2. \tag{5.9}$$

The dynamic viscosity η_α is calculated using the Lohrenz, Bray, and Clark model presented by Lohrenz et al. (1964). See Appendix B.1 for details.

5.1.3 Pressure equation

Let the pressure be a function of total concentrations

$$p(t,\mathbf{x}) = p^{(\mathrm{eq})}\left(c_1\left(t,\mathbf{x}\right), \ldots, c_n\left(t,\mathbf{x}\right)\right). \tag{5.10}$$

If the state is in one phase ($\Pi = 1$), the equilibrium pressure $p^{(\mathrm{eq})}$ (5.10) is given by the equation of state

$$p^{(\mathrm{eq})}(c_1, \ldots, c_n) = p^{(EOS)}(c_1, \ldots, c_n). \tag{5.11}$$

The possible equations of state are listed in Section 2.4. On the other hand, if the equilibrium state is in $\Pi > 1$ phases, the equilibrium pressure $p^{(\mathrm{eq})}$ is given by

$$p^{(\mathrm{eq})}\left(c_1, \ldots, c_n\right) = p^{(EOS)}\left(c_{\alpha,1}, \ldots, c_{\alpha,n}\right), \tag{5.12}$$

for an arbitrary $\alpha \in \widehat{\Pi}$ since the pressures of each phase in the phase equilibrium are equal (see equation (2.47)).

Taking derivative with respect to time in equation (5.10) results in

$$\frac{\partial p}{\partial t}\left(t,\mathbf{x}\right) = \sum_{i=1}^{n} \frac{\partial p^{(\mathrm{eq})}}{\partial c_i}\left(c_1, \ldots, c_n\right) \frac{\partial c_i}{\partial t}\left(t,\mathbf{x}\right), \tag{5.13}$$

where we use the chain rule. Therefore, we assume that $p^{(\mathrm{eq})}$ is a smooth function. From the mass balance equation (5.2), the terms $\frac{\partial c_i}{\partial t}$ read as

$$\frac{\partial c_i}{\partial t}\left(t,\mathbf{x}\right) = \frac{1}{\phi}\left(f_i\left(t,\mathbf{x}\right) - \boldsymbol{\nabla}\cdot\mathbf{q}_i\left(t,\mathbf{x}\right)\right), \quad i \in \hat{n}. \tag{5.14}$$

Combining equations (5.13) and (5.14) results in

$$\phi\frac{\partial p}{\partial t}\left(t,\mathbf{x}\right) + \sum_{i=1}^{n} \Theta_i\left(\boldsymbol{\nabla}\cdot\mathbf{q}_i\left(t,\mathbf{x}\right) - f_i\left(t,\mathbf{x}\right)\right) = 0, \tag{5.15}$$

where we denoted

$$\Theta_i = \Theta_i\left(c_1, \ldots, c_n\right) = \frac{\partial p^{(\mathrm{eq})}}{\partial c_i}\left(c_1, \ldots, c_n\right). \tag{5.16}$$

The previous equation is called pressure equation. The derivatives $\frac{\partial p}{\partial c_i}\left(c_1, \ldots, c_n\right)$ are calculated from the phase equilibrium computation, see Appendix B.2.

5.1.4 Fluxes definition

In this section, we define fluxes needed for the description of the numerical scheme. Let

$$\mathbf{q}_{\alpha,i} = c_{\alpha,i}\mathbf{u}_\alpha \tag{5.17}$$

be the flux of the i-component of α-phase. Then, the flux of phase α is

$$\mathbf{q}_\alpha = \sum_{i=1}^{n} \mathbf{q}_{\alpha,i} = c_\alpha \mathbf{u}_\alpha, \tag{5.18}$$

where $c_\alpha = \sum\limits_{i=1}^{n} c_{\alpha,i}$ is the total concentration of phase α. Lastly, the total flux $\mathbf{q}$ is defined as

$$\mathbf{q} = \sum_{\alpha=1}^{\Pi} \mathbf{q}_\alpha = \sum_{\alpha=1}^{\Pi} c_\alpha \mathbf{u}_\alpha, \tag{5.19}$$

and the total velocity $\mathbf{u}$ as

$$\mathbf{u} = \sum_{\alpha=1}^{\Pi} \mathbf{u}_\alpha. \tag{5.20}$$

Inserting equation (5.5) into the previous equation results in

$$\mathbf{u} = -\lambda \mathbf{K}\left(\boldsymbol{\nabla} p - \rho^{(\mathrm{avg})}\mathbf{g}\right), \tag{5.21}$$

where the total mobility λ and the average density $\rho^{(\mathrm{avg})}$ are defined as

$$\lambda = \sum_{\alpha=1}^{\Pi} \lambda_\alpha, \tag{5.22}$$

$$\rho^{(\mathrm{avg})} = \frac{\sum\limits_{\alpha=1}^{\Pi} \lambda_\alpha \rho_\alpha}{\lambda}. \tag{5.23}$$

If the tensor $\mathbf{K}$ is positive definite, its inverse exists, and the gradient $\boldsymbol{\nabla} p$ can be expressed from equation (5.21) as

$$\boldsymbol{\nabla} p = -\lambda^{-1}\mathbf{K}^{-1}\mathbf{u} + \rho^{(\mathrm{avg})}\mathbf{g}. \tag{5.24}$$

Inserting the previous equation into Darcy's law (5.5) results in

$$\mathbf{u}_\alpha = \frac{\lambda_\alpha}{\lambda}\left(\mathbf{u} - \sum_{\beta=1}^{\Pi} \lambda_\beta \left(\rho_\beta - \rho_\alpha\right)\mathbf{K}\mathbf{g}\right). \tag{5.25}$$

Therefore, the flux of the i-th component in phase α is

$$\mathbf{q}_{\alpha,i} = c_{\alpha,i}\frac{\lambda_\alpha}{\lambda}\left(\mathbf{u} - \sum_{\beta=1}^{\Pi} \lambda_\beta \left(\rho_\beta - \rho_\alpha\right)\mathbf{K}\mathbf{g}\right). \tag{5.26}$$

5.1.5 Initial and boundary conditions

Now, let us summarize the equations and define the initial and boundary conditions. Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and J be a time interval. In $J \times \Omega$, we solve equations (5.2) and (5.15)

$$\frac{\partial(\phi c_i)}{\partial t} + \boldsymbol{\nabla} \cdot \mathbf{q}_i = f_i,$$

$$\phi\frac{\partial p}{\partial t} + \sum_{i=1}^{n} \Theta_i(\boldsymbol{\nabla} \cdot \mathbf{q}_i - f_i) = 0,$$

for $p = p(t, \mathbf{x})$ and $c_i = c_i(t, \mathbf{x})$, $i \in \widehat{n}$. The fluxes $\mathbf{q}_i$ are given by equation (5.3) and the velocities $\mathbf{u}_\alpha$ are computed using Darcy's law (5.5). The composition of the multi-phase state is determined by solving the optimization problem given by (4.11) and (4.12). Note, here we omit the superscript $*$ that indicates the initial state. The mathematical model has to be equipped with initial conditions and an appropriate set of boundary conditions. The initial conditions read as

$$c_i(0, \mathbf{x}) = c_i^{(0)}, \quad \forall \mathbf{x} \in \Omega, i \in \widehat{n}, \tag{5.27}$$

$$p(0, \mathbf{x}) = p^{(\mathrm{eq})}\left(c_1^{(0)}, \ldots, c_n^{(0)}\right), \quad \forall \mathbf{x} \in \Omega. \tag{5.28}$$

Moreover, we impose the following boundary conditions

$$p(t, \mathbf{x}) = p^{(D)}(t, \mathbf{x}), \quad \mathbf{x} \in \Gamma_p, t \in J, \tag{5.29}$$

$$\mathbf{q}_i(t, \mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = 0, \mathbf{x} \in \Gamma_q, t \in J, \tag{5.30}$$

where $\mathbf{n}$ is the unit outward normal vector to the boundary $\partial\Omega$. The Dirichlet part $\Gamma_p \subset \partial\Omega$ and Neumann part $\Gamma_q \subset \partial\Omega$ of the boundary have to satisfy

$$\overline{\Gamma_p \cup \Gamma_q} = \partial\Omega, \tag{5.31}$$

$$\Gamma_p \cap \Gamma_q = \varnothing. \tag{5.32}$$

5.2 Numerical solution

In this work, we assume that the domain Ω is a 2D rectangular domain. We consider a spatial discretization $\tau_\Omega = \left\{K_i; i \in \widehat{N_{el}}\right\}$ consisting of triangles, where N_{el} is the number of elements. We assume that the mesh is conforming. Moreover, we denote Υ_Ω the set of all sides of τ_Ω, and $\Upsilon_\Omega^{(\mathrm{int})}$ and $\Upsilon_\Omega^{(\mathrm{ext})}$ the set of interior and exterior (boundary) sides of τ_Ω, respectively. The total number of sides in the mesh is denoted by N_{si}. By Υ_K, we denote the set of all sides of an element $K \in \tau_\Omega$. Lastly, we denote Υ^{Γ_p} and Υ^{Γ_q} the set of the Dirichlet and Neumann boundary sides, respectively.

5.2.1 Discretization of Darcy's law

On each element $K \in \tau_\Omega$, we shall approximate $\mathbf{u}$ in the lowest order Raviar-Thomas-Nédélec space $\mathbf{RTN}_0(K)$ (see Brezzi and Fortin (2012); Nedelec (1980); Raviart and Thomas (1977))

$$\mathbf{u}(t, \mathbf{x}) = \sum_{E \in \Upsilon_K} u_{K,E}(t)\mathbf{w}_{K,E}(\mathbf{x}), \tag{5.33}$$

where $\mathbf{w}_{K,E} \in \mathbf{RTN}_0(K)$ are the basis functions and $u_{K,E}(t)$ is the velocity across the side E in the outward direction with respect to K at time t. The basis functions $\mathbf{w}_{K,E}$ are chosen such that for all $E, F \in \Upsilon_K$

$$\mathbf{w}_{K,E} \cdot \mathbf{n}_{K,E'} = \frac{\delta_{E,E'}}{|E|_1}, \tag{5.34}$$

$$\boldsymbol{\nabla} \cdot \mathbf{w}_{K,E} = \frac{1}{|K|}, \tag{5.35}$$

where $|K|$ and $|E|_1$ denote the Lebesgue measures of element K in $\mathbb{R}^2$ and side E in $\mathbb{R}^1$, respectively. Multiplying equation (5.24) with function $\mathbf{w}_{K,E'}$ and integrating over $K \in \mathcal{T}_\Omega$ results in

$$\int_K \boldsymbol{\nabla} p \cdot \mathbf{w}_{K,E'} \mathrm{d}\mathbf{x} = \int_K \left(-\lambda^{-1} \mathbf{K}^{-1} \mathbf{u} + \rho^{(\mathrm{avg})} \mathbf{g} \right) \cdot \mathbf{w}_{K,E'} \mathrm{d}\mathbf{x}. \tag{5.36}$$

The left-hand side of the previous equation is

$$\begin{aligned} \int_K \boldsymbol{\nabla} p \cdot \mathbf{w}_{k,E'} \mathrm{d}\mathbf{x} &= \sum_{E \in \Upsilon_K} \int_E p \mathbf{w}_{K,E'} \cdot \mathbf{n}_{K,E} \mathrm{d}S - \int_K p \boldsymbol{\nabla} \cdot \mathbf{w}_{K,E} \mathrm{d}\mathbf{x} \\ &= \frac{1}{|E'|_1} \int_{E'} p \mathrm{d}S - \frac{1}{|K|} \int_K p \mathrm{d}\mathbf{x} \\ &= \breve{p}_{K,E'} - p_K, \end{aligned} \tag{5.37}$$

where we have denoted the average pressures on element K by p_K, and the average traces of the pressures on side E by $\breve{p}_{K,E}$. Next, we will assume that the λ and $\rho^{(\mathrm{avg})}$ are constant on each element K and denote these values by λ_K, and $\rho_K^{(\mathrm{avg})}$, respectively. Therefore, using the previous derivation and equation (5.33), the weak formulation of Darcy's law reads as

$$\begin{aligned} \breve{p}_{K,E'} - p_K = -\lambda_K^{-1} \sum_{E \in \Upsilon_K} u_{K,E} \int_K (\mathbf{K}^{-1} \mathbf{w}_{K,E}) \cdot \mathbf{w}_{K,E'} \mathrm{d}\mathbf{x} \\ + \rho_K^{(\mathrm{avg})} \int_K \mathbf{g} \cdot \mathbf{w}_{K,E'} \mathrm{d}\mathbf{x}, \end{aligned} \tag{5.38}$$

where $\rho_K^{(\mathrm{avg})}$ is the average density on element K. Denoting

$$B_{E,E'}^K = \int_K (\mathbf{K}^{-1} \mathbf{w}_{K,E}) \cdot \mathbf{w}_{K,E'} \mathrm{d}\mathbf{x}, \tag{5.39}$$

$$C_{E'}^K = \int_K \mathbf{g} \cdot \mathbf{w}_{K,E'} \mathrm{d}\mathbf{x}, \tag{5.40}$$

equation (5.38) reads as

$$\sum_{E \in \Upsilon_K} u_{K,E} B_{E,E'}^K = \lambda_K \left(p_K - \breve{p}_{K,E'} + \rho_K^{(\mathrm{avg})} C_{E'}^K \right). \tag{5.41}$$

This equation can be inverted, and the velocities $u_{K,E}$ are expressed as

$$u_{K,E} = \lambda_K \left(D_E^K p_K - \sum_{E' \in \Upsilon_K} \left(B^K\right)_{E,E'}^{-1} \breve{p}_{K,E'} + F_E^K \rho_K^{(\mathrm{avg})} \right), \tag{5.42}$$

where

$$D_E^K = \sum_{E' \in \Upsilon_K} \left(B^K\right)^{-1}_{E,E'}, \tag{5.43}$$

$$F_E^K = \sum_{E' \in \Upsilon_K} \left(B^K\right)^{-1}_{E,E'} C^K_{E'}. \tag{5.44}$$

Now, we will use continuity assumptions and the balance of fluxes without source at a given side E. If E is not on the boundary, i.e, $E \in \Upsilon_\Omega^{(\mathrm{int})}$, then,

$$u_{K',E} + u_{K,E} = 0, \tag{5.45}$$

$$\breve{p}_{K,E} = \breve{p}_{K',E} =: \breve{p}_E, \tag{5.46}$$

where $K' \cap K = E$. If E is a side on the boundary, i.e., $E \in \Upsilon_\Omega^{(\mathrm{ext})}$, then

$$\breve{p}_{K,E} = p_E^{(D)}, \text{ for } E \in \Upsilon^{\Gamma_p}, \tag{5.47}$$

$$u_{K,E} = 0, \text{ for } E \in \Upsilon^{\Gamma_q}. \tag{5.48}$$

Therefore, in equation (5.42), the velocities $u_{K,E}(t)$ can be eliminated, and the only unknowns are $p_K, \breve{p}_E$. In the case E is not the boundary side, i.e., $E \in \Upsilon_\Omega^{(\mathrm{int})}$, equation (5.45) implies

$$0 = \sum_{K \supset E} \lambda_K \left(D_E^K p_K - \sum_{E' \in \Upsilon_K} \left(B^K\right)^{-1}_{E,E'} \breve{p}_{E'} + F_E^K \rho_K^{(\mathrm{avg})} \right). \tag{5.49}$$

If E is a boundary side, i.e., $E \in \Upsilon_\Omega^{(\mathrm{ext})}$, then

$$\breve{p}_E = p_E^{(D)}, \text{ for } E \in \Upsilon^{\Gamma_p}, \tag{5.50}$$

$$-\lambda_K D_E^K p_K + \sum_{E' \in \Upsilon_K} \lambda_K \left(B^K\right)^{-1}_{E,E'} \breve{p}_{E'} = \lambda_K F_E^K \rho_K^{(\mathrm{avg})}, \text{ for } E \in \Upsilon^{\Gamma_q}. \tag{5.51}$$

The previous equations (5.49)–(5.51) form a system of linear equations for the unknowns $\breve{p}_E, p_K$:

$$\mathbf{R}_1 \mathbf{p} + \mathbf{R}_2 \breve{\mathbf{p}} = \mathbf{L}_1, \tag{5.52}$$

where $\mathbf{R}_1 \in \mathbb{R}^{N_{si},N_{el}}$, $\mathbf{R}_2 \in \mathbb{R}^{N_{si},N_{si}}$, $\mathbf{L}_1 \in \mathbb{R}^{N_{si}}$, and

$$\mathbf{p} = \left(p_1, \ldots, p_{N_{el}}\right)^{\mathrm{T}}, \tag{5.53}$$

$$\breve{\mathbf{p}} = \left(\breve{p}_1, \ldots, \breve{p}_{N_{si}}\right)^{\mathrm{T}}. \tag{5.54}$$

5.2.2 Discretization of the pressure equation

Integrating the pressure equation (5.15) over an element $K \in \tau_\Omega$ results in

$$0 = \phi_K \int_K \frac{\partial p}{\partial t} \mathrm{d}\mathbf{x} + \sum_{i=1}^{n} \int_K \Theta_i \left(\boldsymbol{\nabla} \cdot \mathbf{q}_i - f_i \right) \mathrm{d}\mathbf{x}, \tag{5.55}$$

where ϕ_K is the average porosity on element K. Then, we denote the average source term on element K by

$$f_{i,K} = \frac{1}{|K|} \int_K f_i \mathrm{d}\mathbf{x}. \tag{5.56}$$

Moreover, interchanging the integral and partial derivative in the first term, equation (5.55) reads as

$$0 = \phi_K \frac{\mathrm{d}}{\mathrm{d}t} \int_K p \mathrm{d}\mathbf{x} + \sum_{i=1}^{n} \Theta_i \int_K \boldsymbol{\nabla} \cdot \mathbf{q}_i \mathrm{d}\mathbf{x} - \sum_{i=1}^{n} \Theta_i |K| f_{i,K}. \tag{5.57}$$

Using the divergence theorem (see, e.g., Arfken (2005)), we have

$$0 = \phi_K |K| \frac{\mathrm{d}p_K}{\mathrm{d}t} + \sum_{i=1}^{n} \Theta_i \int_{\partial K} \mathbf{q}_i \cdot \mathbf{n} \mathrm{d}S - \sum_{i=1}^{n} \Theta_i |K| f_{i,K}. \tag{5.58}$$

The flux q_i across side E is approximated by numerical flux $q_{i,K,E}$. The form of the approximation $q_{i,K,E}$ will be given later in equation (5.71). Therefore, equation (5.58) is approximated by

$$0 = \phi_K |K| \frac{\mathrm{d}p_K}{\mathrm{d}t} + \sum_{i=1}^{n} \Theta_i \sum_{E \in \Upsilon_K} q_{i,K,E} - \sum_{i=1}^{n} \Theta_i |K| f_{i,K}. \tag{5.59}$$

Using relation

$$q_{i,K,E} = \sum_{\alpha=1}^{\Pi(K)} q_{\alpha,i,K,E}, \tag{5.60}$$

equation (5.26), and the backwards Euler scheme, equation (5.59) can be approximated by

$$\begin{aligned} 0 = \phi_K |K| \frac{p_K^{m+1} - p_K^m}{\Delta t} &- \sum_{i=1}^{n} \Theta_i^{m+1} |K| f_{i,K}^{m+1} \mathrm{d}\mathbf{x} + p_K^{m+1} X_K^{m+1} \\ &+ \sum_{E' \in \Upsilon_K} \breve{p}_{E'}^{m+1} Y_{E'}^{m+1} + Z_K^{m+1}, \end{aligned} \tag{5.61}$$

where the superscript denotes the time level, i.e., $p_K^m = p_K(t_0 + m\Delta t)$, where Δt is the fixed discrete time step. In equation (5.61), we denoted

$$\begin{aligned} X_K &= \sum_{i=1}^{n} \sum_{E \in \Upsilon_K} \sum_{\alpha=1}^{\Pi(K)} \Theta_i c_{\alpha,i,K} \lambda_{\alpha,K} D_E^K \\ Y_{E'} &= \sum_{i=1}^{n} \sum_{E \in \Upsilon_K} \sum_{\alpha=1}^{\Pi(K)} -\Theta_i c_{\alpha,i,K} \lambda_{\alpha,K} \left(B_K\right)_{E,E'}^{-1}, \\ Z_K &= \sum_{i=1}^{n} \sum_{E \in \Upsilon_K} \sum_{\alpha=1}^{\Pi(K)} \lambda_K^{-1} \Theta_i c_{\alpha,i,K} \lambda_{\alpha,K} \Bigg(\lambda_K F_E^K \rho_K^{avg} \\ &\qquad - \sum_{\beta=1}^{\Pi(K)} \lambda_{\beta,K} (\rho_{\beta,K} - \rho_{\alpha,K}) F_E^K \Bigg), \end{aligned} \tag{5.62}$$

where $c_{\alpha,i,K}$ is the average concentration of the i-th component in phase α on element K, $\lambda_{\alpha,K}$ is the average mobility of phase α on element K, and $\Pi(K)$ is the number of phases in the equilibrium on element K. Equation (5.61) forms a system of linear equations for the unknowns p_K^{m+1} and $\breve{p}_{E'}^{m+1}$

$$\mathbf{R}_3 \mathbf{p} + \mathbf{R}_4 \breve{\mathbf{p}} = \mathbf{L}_2, \tag{5.63}$$

where $\mathbf{R}_3 \in \mathbb{R}^{N_{el},N_{el}}$, $\mathbf{R}_4 \in \mathbb{R}^{N_{el},N_{si}}$, and $\mathbf{L}_2 \in \mathbb{R}^{N_{el}}$. To conclude, combining equations (5.52) and (5.63) gives the final system for the pressure field

$$\begin{pmatrix} \mathbf{R}_1 & \mathbf{R}_2 \\ \mathbf{R}_3 & \mathbf{R}_4 \end{pmatrix} \begin{pmatrix} \mathbf{p} \\ \breve{\mathbf{p}} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_1 \\ \mathbf{L}_2 \end{pmatrix}. \tag{5.64}$$

The matrix $\mathbf{R}_3$ is diagonal, therefore, its inverse $\mathbf{R}_3^{-1}$ is readily available. Multiplying equation (5.63) with $\mathbf{R}_3^{-1}$ gives

$$\mathbf{p} = \mathbf{R}_3^{-1}\mathbf{L}_2 - \mathbf{R}_3^{-1}\mathbf{R}_4\breve{\mathbf{p}}. \tag{5.65}$$

Therefore, the unknowns $\mathbf{p}$ can be eliminated from the system (5.64), and only the pressure traces $\breve{\mathbf{p}}$ are computed using

$$(\mathbf{R}_2 - \mathbf{R}_1\mathbf{R}_3^{-1}\mathbf{R}_4)\breve{\mathbf{p}} = \mathbf{L}_1 - \mathbf{R}_1\mathbf{R}_3^{-1}\mathbf{L}_2. \tag{5.66}$$

In this thesis, we are using the `C++` numerical library `Armadillo` (see Eddelbuettel and Sanderson (2014); Sanderson and Curtin (2016)) to solve the system (5.66). Having the pressure traces $\breve{\mathbf{p}}$, the pressures $\mathbf{p}$, and consequently, the discrete velocities $u_{K,E}$ are computed using equations (5.65) and (5.42), respectively.

5.2.3 Solution of transport equations

Having the pressure field, the concentrations are updated using the explicit finite-volume method. Integrating equation (5.2) over $K \in \tau_\Omega$ results in

$$\phi \int_K \frac{\partial c_i}{\partial t} \mathrm{d}\mathbf{x} + \int_K \boldsymbol{\nabla} \cdot \mathbf{q}_i \mathrm{d}\mathbf{x} = \int_K f_i \mathrm{d}\mathbf{x}. \tag{5.67}$$

Interchanging the integral and partial derivative in the first term, using the divergence theorem, and equation (5.56) gives

$$\phi \frac{\mathrm{d}}{\mathrm{d}t} \int_K c_i \mathrm{d}\mathbf{x} + \int_{\partial K} \mathbf{q}_i \cdot \mathbf{n} \mathrm{d}S = |K| f_{i,K}. \tag{5.68}$$

Defining the average total concentration $c_{i,K}$ on element K by

$$c_{i,K} = \frac{1}{|K|} \int_K c_i \mathrm{d}\mathbf{x}, \tag{5.69}$$

and using the Euler forward scheme, equation (5.68) can be approximated by

$$c_{i,K}^{m+1} = c_{i,K}^m + \frac{\Delta t}{\phi |K|} \left(|K| f_{i,K}^m - \sum_{E \in \Upsilon_K} q_{i,K,E}^m \right), \tag{5.70}$$

The numerical fluxes $q_{i,K,E}$ are calculated using the upwind scheme

$$q_{i,K,E} = \begin{cases} \sum\limits_{\alpha \in \Pi^+(K,E)} q_{\alpha,i,K,E} - \sum\limits_{\beta \in \Pi^+(K',E)} q_{\beta,i,K',E}, & \forall E \in \Upsilon_\Omega^{(\mathrm{int})}, E = K \cap K' \\ \sum\limits_{\alpha \in \Pi^+(K,E)} q_{\alpha,i,K,E}, \quad \forall E \in \Upsilon^{\Gamma_p}, \\ 0, \quad \forall E \in \Upsilon^{\Gamma_q}. \end{cases} \tag{5.71}$$

where $\Pi^+(K,E) = \left\{ \alpha \in \widehat{\Pi}(K);\ q_{\alpha,i,K,E} > 0 \right\}$ for $E \in \Upsilon_K$, and

$$q_{\alpha,i,K,E} = c_{\alpha,i,K} \lambda_K^{-1} \lambda_{\alpha,K} \left(u_{K,E} - \sum_{\beta=1}^{\Pi(K)} \lambda_{\beta,K} \left(\rho_\beta - \rho_\alpha\right) F_{K,E} \right), \tag{5.72}$$

where the discrete velocies $u_{K,E}$ are given by equation (5.42).

5.2.4 Algorithm for one time step Δt

Now, we present the full numerical algorithm. This iterative IMPEC algorithm is based on the numerical scheme presented in Chen et al. (2019). Having solution on time-level t_m, the solution on time level t_{m+1} is computed using the following algorithm.

1. Set $l = 0$ and $p_K^{m+1,0} = p_K^m, c_{i.K}^{m+1,0} = c_{i,K}^m$, $\Theta_{i,K}^{m+1,0} = \frac{\partial p^{(\mathrm{eq})}}{\partial c_i}\left(c_{1,K}^m, \dots, c_{n,K}^m\right)$ for $K \in \tau_\Omega, i \in \widehat{n}$.

2. Set $l = l + 1$.

3. On each element $K \in \tau_\Omega$, compute $c_{\alpha,i,K}^{m+1,l-1}$ and $S_{\alpha,K}^{m+1,l-1}$ by solving the phase equilibrium computation given by equations (4.11)–(4.12) with initial concentrations
$$c_{1,K}^{m+1,l-1}, \dots, c_{n,K}^{m+1,l-1}.$$

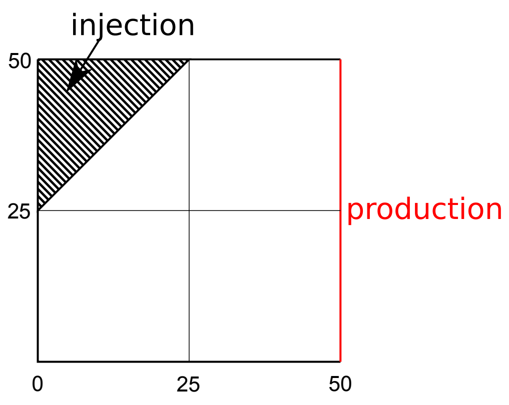
4. On each element $K \in \tau_\Omega$, update $\lambda_K^{m+1,l-1}$ and $\rho_K^{(\mathrm{avg}),m+1,l-1}$ using equations (5.6) and (5.22) with values $c_{\alpha,i,K}^{m+1,l-1}$ and $S_{\alpha,K}^{n+1,l-1}$ computed in the previous step.

5. Find $p_K^{m+1,l}$ and $u_{K,E}^{m+1,l}$ by solving system (5.64) with the concentrations $c_{\alpha,i,K}^{m+1,l-1}$, coefficients $\Theta_{i,K}^{m+1,l-1}$, total mobility $\lambda_K^{m+1,l-1}$, and average density $\rho_K^{(\mathrm{avg}),m+1,l-1}$.

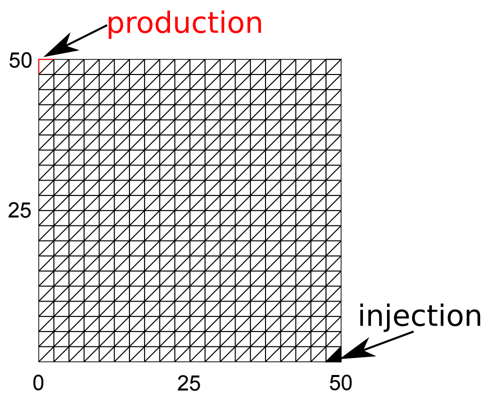
6. On each element $K \in \tau_\Omega$, for all $i \in \widehat{n}$ update $c_{i,K}^{m+1,l}$ explicitly by
$$c_{i,K}^{m+1,l} = c_{i,K}^m + \frac{\Delta t}{\phi|K|}\left(|K| f_{i,K}^m - \sum_{E \in \Upsilon_K} q_{i,K,E}^{m+1,l-1}\right),$$
where the flux $q_{i,K,E}^{m+1,l-1}$ is evaluated using the velocity $u_{K,E}^{m+1,l}$ and concentrations $c_{\alpha,i,K}^{m+1,l-1}$.

7. On each element $K \in \tau_\Omega$, for all $i \in \widehat{n}$ update $\Theta_{i,K}^{m+1,l}$ by
$$\Theta_{i,K}^{m+1,l} = \frac{p^{(\mathrm{eq})}(\mathbf{c}^{(1)}) - p^{(\mathrm{eq})}(\mathbf{c}^{(2)})}{c_{i,K}^{m+1,l} - c_{i,K}^m}, \tag{5.73}$$
where
$$\mathbf{c}^{(1)} = \left(c_{1,K}^{m+1,l}, \dots, c_{i,K}^{m+1,l}, c_{i+1,K}^m, \dots, c_{n,K}^m\right)^{\mathrm{T}}, \tag{5.74}$$
$$\mathbf{c}^{(2)} = \left(c_{1,K}^{m+1,l}, \dots, c_{i-1,K}^{m+1,l}, c_{i,K}^m, \dots, c_{n,K}^m\right)^{\mathrm{T}}. \tag{5.75}$$
To compute the pressures, the phase equilibrium computation is used to determine the number of phases and the equilibrium pressure.

8. Check convergence. If the convergence criteria are met, set
$$p_K^{m+1} = p_K^{m+1,l}, \quad c_{i,K}^{m+1} = c_{i,K}^{m+1,l}, \quad \forall K \in \tau_\Omega, \forall i \in \widehat{n}, \tag{5.76}$$



(a) Example H1: EOC study



(b) Examples H2–H4

Figure 5.1: Depiction of the physical situation in Example H1: EOC study (left), and the structure of the mesh used in Examples H2–H4 (right).

and terminate the Algorithm. Otherwise, go to step 2. In this work, we terminate the algorithm if the maximum number of iterations $l_{\max}$ is reached or the criterion

$$\max\left\{\frac{\left\|p^{m+1,l}-p^{m+1,l-1}\right\|}{\left\|p^{m+1,l}\right\|}, \sum_{i=1}^{n}\frac{\left\|c_i^{m+1,l}-c_i^{m+1,l-1}\right\|}{\left\|c_i^{m+1,l}\right\|}, \sum_{i=1}^{n}\frac{\left\|\Theta_i^{m+1,l}-\Theta_i^{m+1,l-1}\right\|}{\left\|\Theta_i^{m+1,l}\right\|}\right\} < \varepsilon, \tag{5.77}$$

is fulfilled. In the previous equation $\|\cdot\|$ is the $L^2(\Omega)$ norm and ε is a given tolerance.

5.3 Examples

In this section, we provide numerical examples showing the behaviour of the numerical scheme presented in Section 5.2. In all examples, the computation domain Ω is a square domain of size 50×50 meters with porosity $\phi = 0.2$ and isotropic permeability

$$\mathbf{K} = k\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}, \tag{5.78}$$

where $k = 9.87\times10^{-15}$ m^2 = 10 mD. The ε tolerance is set to $\varepsilon = 10^{-5}$, and the maximum number of inner iterations is set to $l_{\max} = 30$.

5.3.1 Example H1: EOC study

In the first example, we simulate the injection of methane (C_1) into a horizontal (i.e., no gravity) reservoir. The reservoir is initially filled with a propane (C_3) at a constant pressure $p = 6.9$ MPa and temperature $T = 311$ K. The binary interaction coefficient is $\delta_{C_1-C_3} = 0.0365$. Methane is injected at the left top corner and the propane is produced at the right side, see Figure 5.1a for the visualization. The rate of the injection is 85 m^2 per day at atmospheric pressure and temperature 293 K. The boundary of the domain is impermeable except for the right side where

mesh	$\left\Vert E_m^{(p)} \right\Vert_1$	$\mathrm{EOC}_1^{(p)}$	$\left\Vert E_m^{(p)} \right\Vert_2$	$\mathrm{EOC}_2^{(p)}$
$m = 2 \times 2 \times 2$	3.1159×10^6		7.5965×10^4	
$m = 2 \times 4 \times 4$	3.2525×10^6	**−0.062**	7.5602×10^4	**0.007**
$m = 2 \times 8 \times 8$	1.2495×10^6	**1.380**	2.9422×10^4	**1.362**
$m = 2 \times 16 \times 16$	7.2306×10^5	**0.789**	1.7179×10^4	**0.776**
$m = 2 \times 32 \times 32$	3.5328×10^5	**1.033**	8.5044×10^3	**1.014**

Table 5.1: The errors and experimental orders of convergence for the pressure p. The pseudo-analytical solution is taken as a solution on a grid with $m = 2 \times 64 \times 64$ elements. Example H1: EOC study.

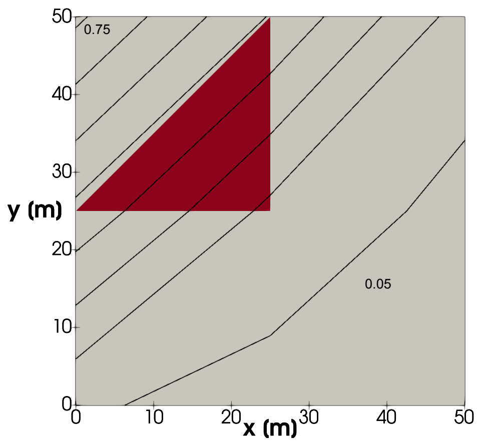
pressure $p = 6.9$ MPa is maintained. The linear model given by equation (5.8) is used to compute the relative permeability. The final time is set to $t_{\text{final}} = 175$ days. On this example, we verify the convergence of the numerical scheme. To be best of our knowledge, there is no analytical solution for the multi-phase compositional flow. Therefore, the numerical solution on the finest mesh with $m = 8192$ elements is taken as a pseudo-analytical solution. The experimental order of convergence (EOC) is defined as

$$\mathrm{EOC}_\xi^{(\alpha)} = \frac{\ln \left\Vert E_{m_1}^{(\alpha)} \right\Vert_\xi - \ln \left\Vert E_{m_2}^{(\alpha)} \right\Vert_\xi}{\ln m_2 - \ln m_1}, \tag{5.79}$$

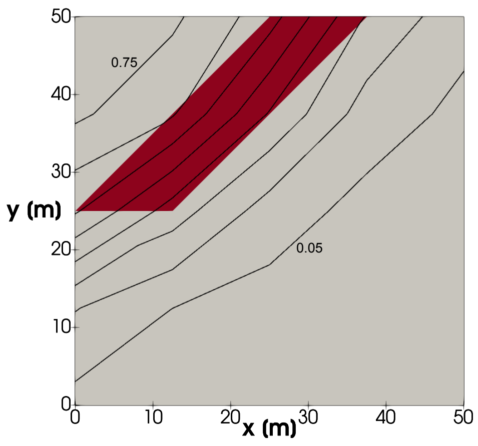
where $\left\Vert E_{m_1}^{(\alpha)} \right\Vert_\xi$ is the error of the α variable using mesh with m_1 elements in norm ξ. The experimental orders of convergence were computed at the final time between neighboring triangular meshes with $m = 2 \times 2 \times 2, 2 \times 4 \times 4, 2 \times 8 \times 8, 2 \times 16 \times 16, 2 \times 32 \times 32$ using the L_1 and L_2 norms. The notation $m = 2 \times x \times y$ represents regular triangular mesh with $x + 1$ nodes on the x-axis and y nodes on the y-axis forming $x \times y$ rectangles that are divided in half to create regular triangular mesh with $2 \times x \times y$ elements. The time step on the coarsest grid ($m = 8$) is $\Delta t = 50000$ s. Then, the time step is four times smaller for each mesh refinement ($\Delta t \sim m^{-1}$). In Figure 5.2, the resulting computed state is presented on different meshes. The iso-lines of methane mole fraction from 0.75 to 0.05 are depicted. The two-phase area is depicted in red color. The resulting errors and EOC for the pressure and molar concentration of methane are presented in Tables 5.1 and 5.2. In the pressure variable, we observe that during the first refinement, the error even increases in the L_1 norm. The explanation could be that the first mesh, which has only eight elements, is too coarse. Therefore, the numerical solution on this mesh is not comparable. The other EOC are around one and at least 0.75. A similar situation is in the second variable - the molar concentration of methane. Here, during the first refinement, the EOC in the L_2 norm is small, around 0.3. However, after the first refinement, the values of EOC increases and are around 0.6, which is expected for the first order upwind on hyperbolic problems with discontinuous solutions (see Leveque (2004)).

5.3.2 Example H2: C_1 injection

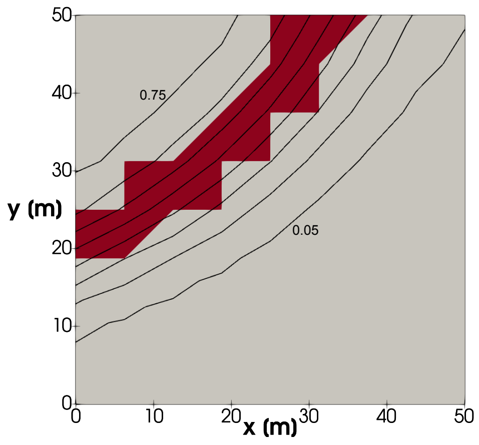
In the second example, we simulate the injection of methane (C_1) into a reservoir. The reservoir is initially filled with a propane (C_3) at a constant pressure $p = 6.9$ MPa and temperature $T = 311$ K. The methane (C_1) is injected at the right bottom corner. The rate of the injection is 85 m^2 per day at atmospheric pressure and temperature 293 K. The binary interaction coefficient is $\delta_{C_1-C_3} = 0.0365$. The boundary of the domain is impermeable except for the outflow corner where pressure $p = 6.9$ MPa is maintained. The linear model given by equation (5.8) is used to



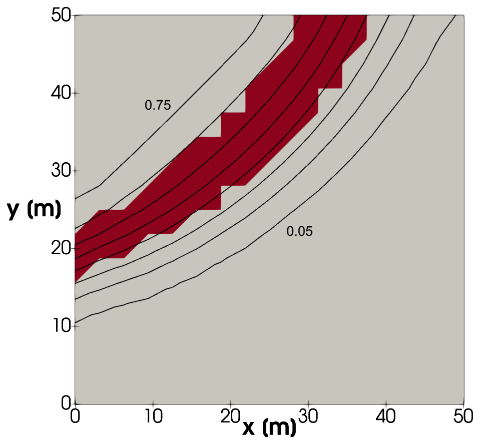
(a) $2 \times 2 \times 2$ elements



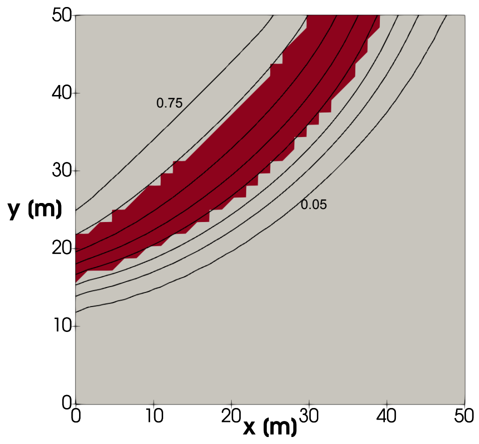
(b) $2 \times 4 \times 4$ elements



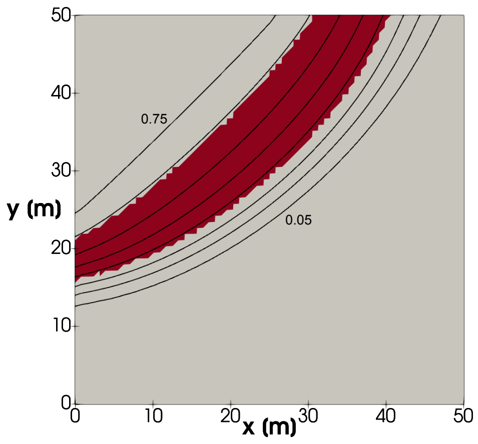
(c) $2 \times 8 \times 8$ elements



(d) $2 \times 16 \times 16$ elements



(e) $2 \times 32 \times 32$ elements



(f) $2 \times 64 \times 64$ elements

Figure 5.2: The iso-lines of methane mole fraction in final time $t_{\text{final}} = 175$ days. The values are from 0.05 to 0.75 with step size 0.1. The two-phase area is depicted in red color. Example H1: EOC study.

mesh	$\left\Vert E_m^{(c_{C_1})}\right\Vert_1$	$\mathrm{EOC}_1^{(c_{C_1})}$	$\left\Vert E_m^{(c_{C_1})}\right\Vert_2$	$\mathrm{EOC}_2^{(c_{C_1})}$
$m = 2 \times 2 \times 2$	1.0677×10^6		3.2790×10^4	
$m = 2 \times 4 \times 4$	6.7335×10^5	**0.665**	2.5486×10^4	**0.364**
$m = 2 \times 8 \times 8$	4.2968×10^5	**0.648**	1.7389×10^4	**0.552**
$m = 2 \times 16 \times 16$	2.5000×10^5	**0.781**	1.0722×10^4	**0.698**
$m = 2 \times 32 \times 32$	1.0935×10^5	**1.193**	5.0671×10^3	**1.081**

Table 5.2: The errors and experimental orders of convergence for the methane concentration c_{C_1}. The pseudo-analytical solution is taken as a solution on a grid with $m = 2 \times 64 \times 64$ elements. Example H1: EOC study.

compute the relative permeability. In this example, we use a triangular mesh with $2 \times 20 \times 20$ elements, i.e., total 400 elements are used. The scheme of the mesh is presented in Figure 5.1b.

First, we simulate a horizontal cut through the reservoir, i.e., the gravity is neglected. The final time is set to $t_{\text{final}} = 365$ days and the time step is $\Delta t = 1000$ seconds. In Figure 5.3, the iso-lines of methane mole fraction at different times are depicted. The values are from 0.05 to 0.95 with a step size 0.1. Moreover, in Figure 5.3, the two-phase region is depicted in red color.

Second, we simulate a vertical cut through the reservoir, i.e., the gravity is considered with the gravity vector $\mathbf{g} = (0, -9.81)$ m s^{-2}. The final time is set to $t_{\text{final}} = 200$ days and the time step is $\Delta t = 1000$ seconds. In Figure 5.4, the iso-lines of methane mole fraction at different times are depicted. The values are from 0.05 to 0.95 with a step size 0.1. Moreover, in Figure 5.4, the two-phase region is depicted in red color.

5.3.3 Example H3: CO_2 injection

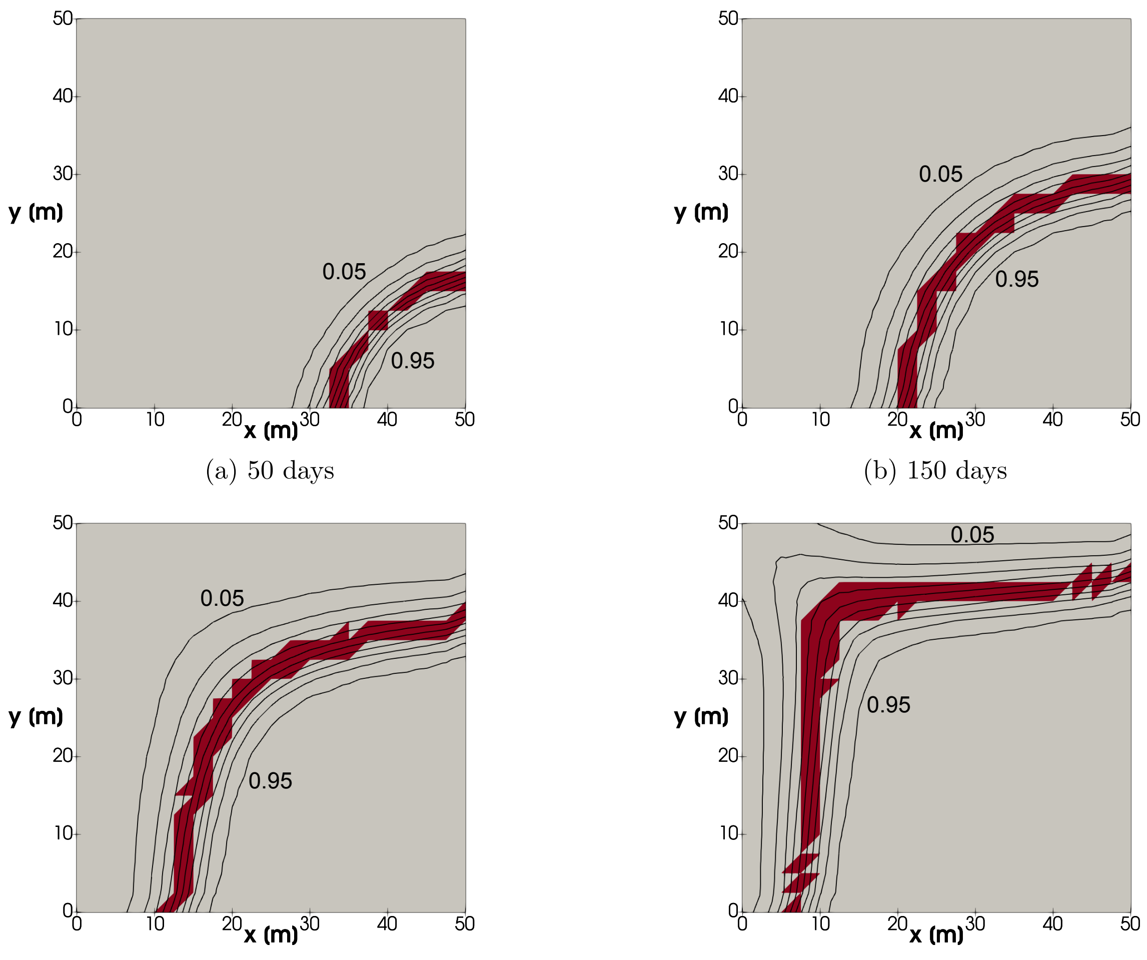
In the third example, we simulate the injection of carbon dioxide (CO_2) into a reservoir. The reservoir is initially filled with pure propane (C_3) at a constant pressure $p = 5$ MPa and temperature $T = 311$ K. The CO_2 is injected at the right bottom corner. The rate of the injection is 85 m^2 per day at atmospheric pressure and temperature 293 K. The binary interaction coefficient is $\delta_{C_1-C_3} = 0.15$. The boundary of the domain is impermeable except for the outflow corner where pressure $p = 5$ MPa is maintained. The linear model given by equation (5.8) is used to compute the relative permeability. In this example, we use a triangular mesh with $2 \times 20 \times 20$ elements, i.e., total 400 elements are used. The scheme of the mesh is presented in Figure 5.1b.

First, we simulate a horizontal cut through the reservoir, i.e., the gravity is neglected. The final time is set to $t_{\text{final}} = 365$ days and the time step is $\Delta t = 500$ seconds. In Figure 5.5, the iso-lines of carbon dioxide mole fraction at different times are depicted. The values are from 0.05 to 0.95 with step size 0.1. Moreover, in Figure 5.5, the two-phase region is depicted in red color.

Second, we simulate a vertical cut through the reservoir, i.e., the gravity is considered with the gravity vector $\mathbf{g} = (0, -9.81)$ m s^{-2}. The final time is set to $t_{\text{final}} = 200$ days and the time step is $\Delta t = 500$ seconds. In Figure 5.6, the iso-lines of methane mole fraction at different times are depicted. The values are from 0.05 to 0.95 with a step size 0.1. Moreover, in Figure 5.6, the two-phase region is depicted in red color.

5.3.4 Example H4: CO_2 injection into oil

Lastly, we again simulate injection of carbon dioxide (CO_2) into a reservoir. However, here, the reservoir is initially filled with a multi-component mixture representing oil at a constant pressure $p = 27.6$ MPa and temperature $T = 403.15$ K. The oil is modelled using two components – nitrogen



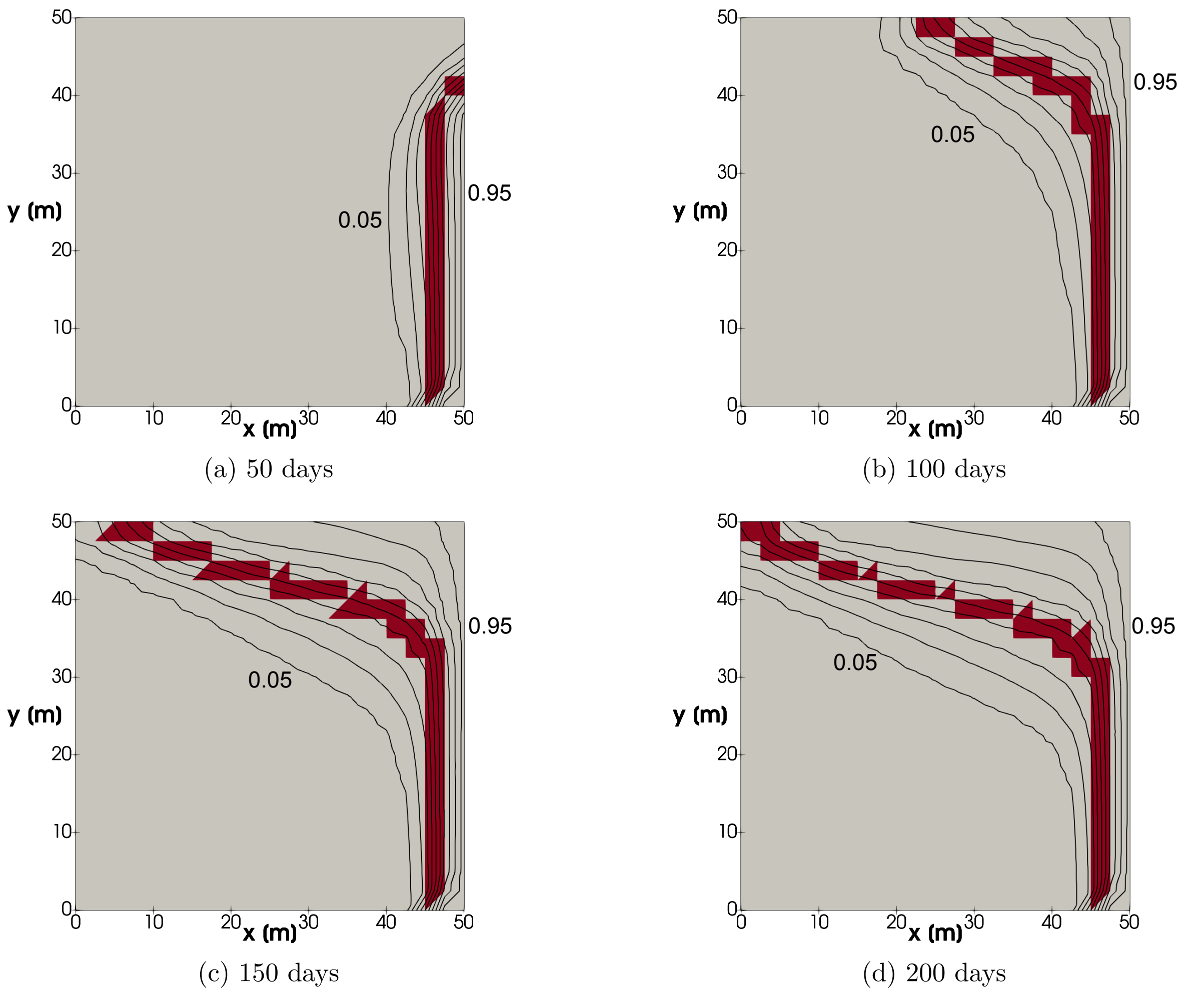
(a) 50 days (b) 150 days

(c) 250 days (d) 365 days

Figure 5.3: The iso-lines of methane mole fraction in different times. The values are from 0.05 to 0.95 with step size 0.1. The two-phase area is depicted in red color. Example H2: C_1 injection without gravity.

(N_2), methane (C_1), and five pseudo-components representing higher alkanes – $PC_{2\text{-}3}$, $PC_{4–5}$, $PC_{6–10}$, $PC_{11–24}$, and PC_{25+}. The initial mole fractions are $x_{CO_2} = 0.0086, x_{N_2} = 0.0028, x_{C_1} = 0.4451, x_{PC_{2-3}} = 0.1207, x_{PC_{4-5}} = 0.0505, x_{PC_{6-10}} = 0.1328, x_{PC_{11-24}} = 0.1660, x_{PC_{25+}} = 0.0735$. The binary interaction coefficients are presented in Table 5.3. The data are taken over from Polívka and Mikyška (2014). The boundary of the domain is impermeable except for the outflow corner where pressure $p = 27.6$ MPa is maintained. The CO_2 is injected at the right bottom corner. The rate of the injection is 266.66 m^2 per day at atmospheric pressure and temperature 293 K. The quadratic model given by equation (5.9) is used to compute the relative permeability. In this example, we use a triangular mesh with $2 \times 20 \times 20$ elements, i.e., total 400 elements are used. The scheme of the mesh is presented in Figure 5.1b.

In this example, we only simulate a horizontal cut through the reservoir, i.e., the gravity is neglected. The final time is set to $t_{\text{final}} = 300$ days and the time step is $\Delta t = 1000$ seconds. In Figure 5.7, the iso-lines of carbon dioxide mole fraction at different times are depicted. The values are from 0.05 to 0.95 with step size 0.1. Moreover, in Figure 5.7, the two-phase region is depicted in red color.



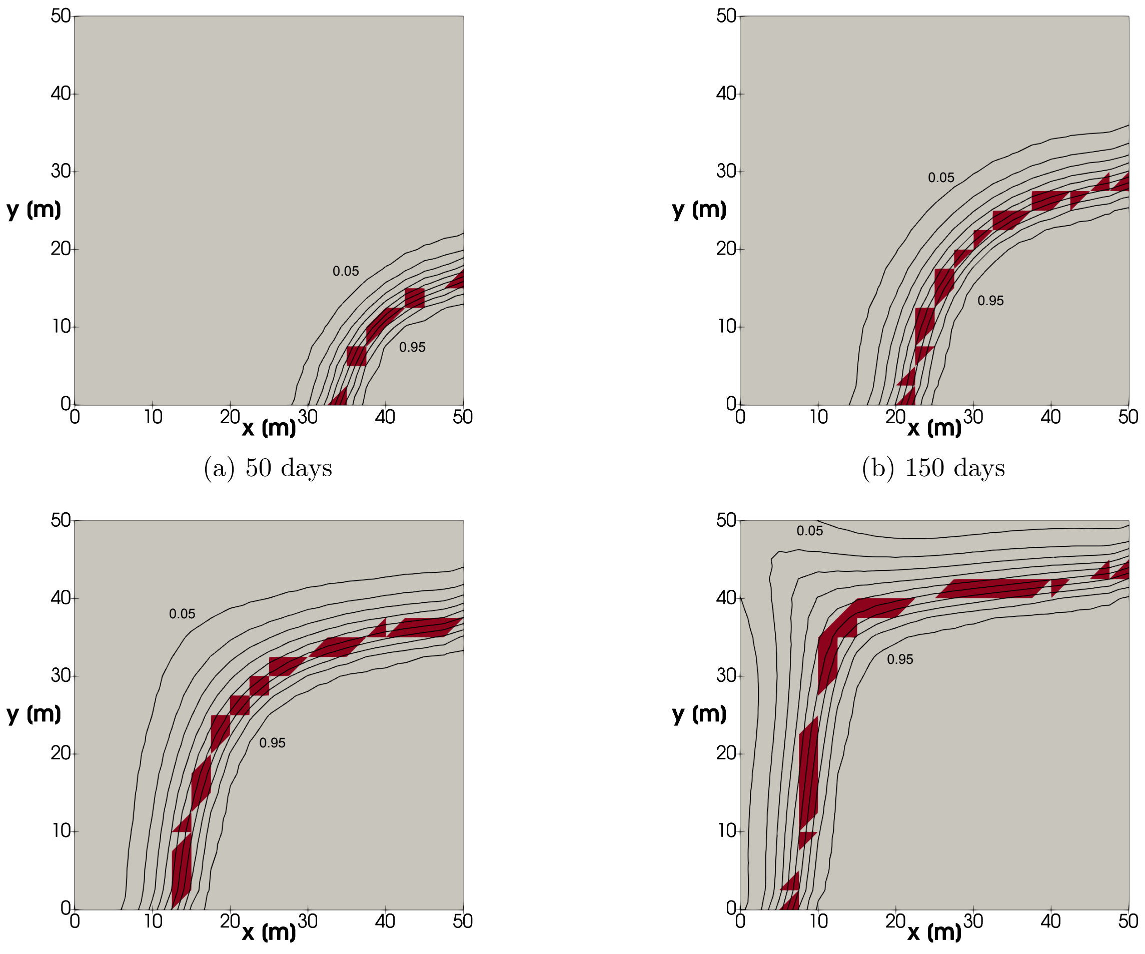
(a) 50 days (b) 100 days

(c) 150 days (d) 200 days

Figure 5.4: The iso-lines of methane mole fraction in different times. The values are from 0.05 to 0.95 with step size 0.1. The two-phase area is depicted in red color. Example H2: C_1 injection with gravity.

	CO_2	N_2	C_1	$PC_{2\text{-}3}$	$PC_{4\text{-}5}$	$PC_{6\text{-}10}$	$PC_{11\text{-}24}$	PC_{25+}
CO_2	0	0	0.15	0.15	0.15	0.15	0.15	0.08
N_2	0	0	0.10	0.10	0.10	0.10	0.10	0.10
C_1	0.15	0.1	0.0	0.0346	0.0392	0.0469	0.0635	0.1052
$PC_{2\text{-}3}$	0.15	0.1	0.0346	0	0	0	0	0
$PC_{4\text{-}5}$	0.15	0.1	0.0392	0	0	0	0	0
$PC_{6\text{-}10}$	0.15	0.1	0.0469	0	0	0	0	0
$PC_{11\text{-}24}$	0.15	0.1	0.0635	0	0	0	0	0
PC_{25+}	0.15	0.1	0.1052	0	0	0	0	0

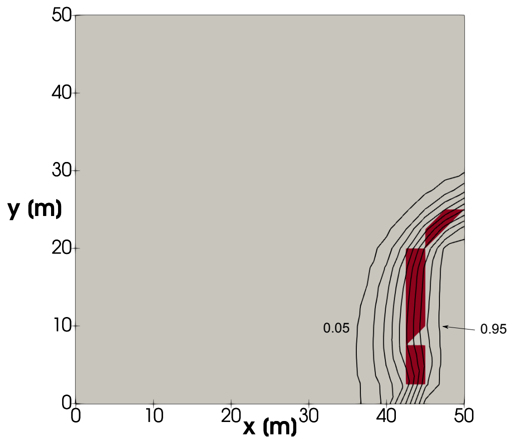
Table 5.3: The binary interaction coefficients. Data taken over from Polívka and Mikyška (2014). Example H4: CO_2 injection into oil.



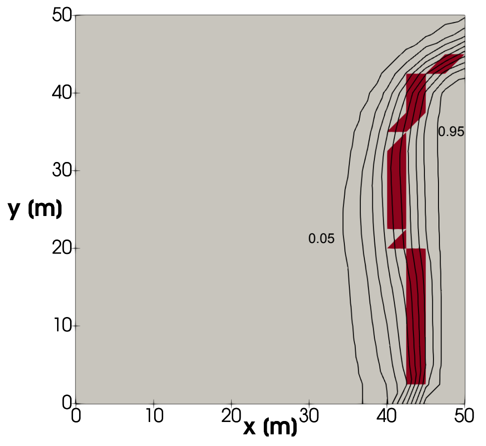
(a) 50 days (b) 150 days

(c) 250 days (d) 365 days

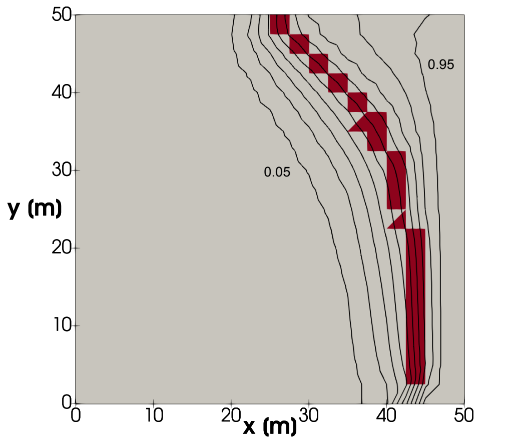
Figure 5.5: The iso-lines of carbon dioxide mole fraction in different times. The values are from 0.05 to 0.95 with step size 0.1. The two-phase area is depicted in red color. Example H3: CO_2 injection without gravity.



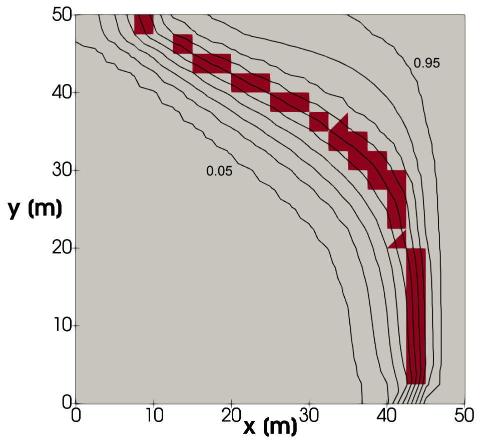
(a) 50 days



(b) 100 days

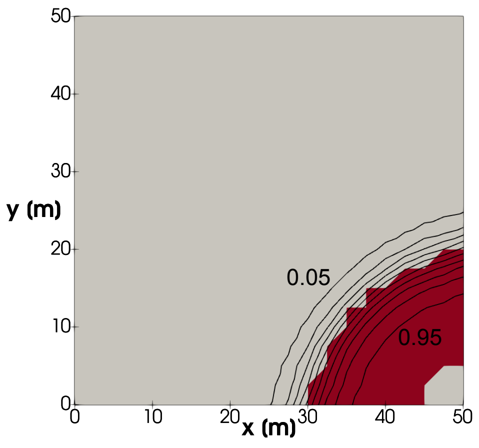


(c) 150 days

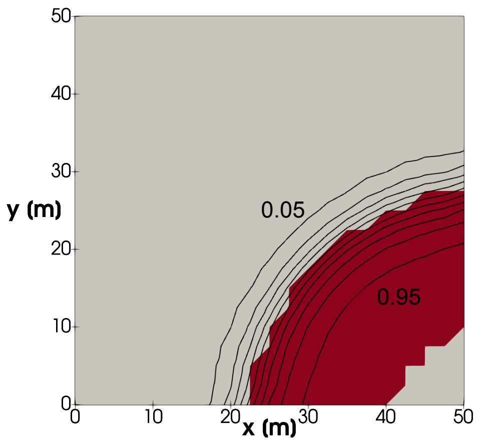


(d) 200 days

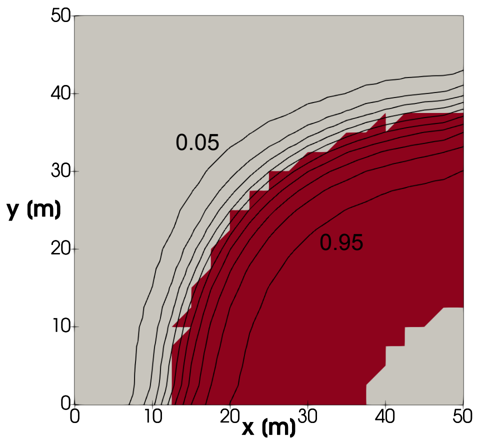
Figure 5.6: The iso-lines of carbon dioxide mole fraction in different times. The values are from 0.05 to 0.95 with step size 0.1. The two-phase area is depicted in red color. Example H3: CO_2 injection with gravity.



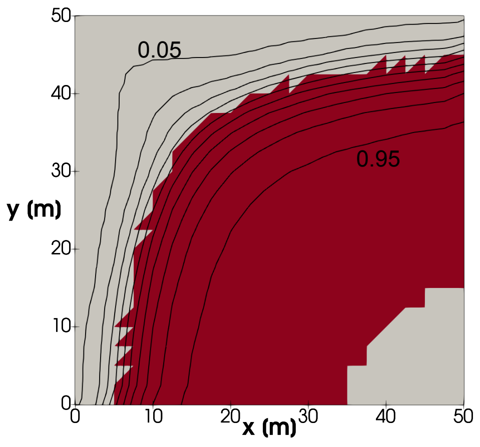
(a) 50 days



(b) 150 days



(c) 250 days



(d) 300 days

Figure 5.7: The iso-lines of carbon dioxide mole fraction in different times. The values are from 0.05 to 0.95 with step size 0.1. The two-phase area is depicted in red color. Example H4: CO_2 injection into oil without gravity.

Conclusion 6

We have presented the mathematical model and possible numerical solutions of the phase stability testing and phase equilibrium computation problems. The mathematical model has been based on the unified framework in which an arbitrary specification or an equation of state can be used. Moreover, we have presented a possible application of the phase equilibrium computation in the modelling of the multi-phase compositional simulations in a porous medium.

In this thesis, we have started with the equilibrium thermodynamics. We have presented the basic principles and derived the essential equations needed in this thesis. Then, the main parts of this thesis has been presented.

Phase stability testing

In Chapter 3, the phase stability testing problem has been discussed. We have proposed a unified formulation of the phase stability testing problems for multi-component mixtures. We have verified that this formulation covers the VTN-, UVN-, and PTN-specifications. A unified condition of stability has been derived. The condition of stability leads to a non-convex optimization problem. We have presented various numerical strategies for solving this global optimization problem. First, we have presented the SSI (Successive Substitution Iteration) method, which is one of the most popular methods. However, this method suffers from slow convergence at the vicinity of the critical points, and the computation time rises unacceptable. Moreover, a unified formulation of the problem could not be used, and each specification had to be solved separably. Therefore, we have proposed a local optimization method based on the Newton–Raphson method with multiple initial approximations. The modified Cholesky decomposition has been used to ensure the convergence toward a minimum. Moreover, we have presented an alternative approach for the computation of the increment in the Newton–Raphson method. Using the structure of the Hessian matrix, we have derived a simple procedure for solving the system of linear equations arising from the linearization of the VTN-phase stability problem. The significant advantage of this method is that the Hessian matrix does not have to be even assembled, and the computation times can be decreased. Then, the system has been solved using the sequential usage of the Sherman–Morrison iterations. However, using the local Newton–Raphson method, there is no guarantee that the algorithm is going to converge toward the global minimum. Therefore, we have proposed a global optimization method that has to converge toward the global minimum. We have presented a global optimization algorithm for the VTN-phase stability testing based on the Branch and Bound strategy. In this strategy, an underestimate convex function of the objective function has to be defined. We have developed a strategy based on the convex-concave splitting of the Helmholtz free energy function. We have presented two different splitting strategies; one our own, the second has been found in the literature. Using a global deterministic method, the global minimum is always found. However, the computation can exceed a reasonable time, and its usage is limited. Therefore, we have presented global minimization methods based on the heuristical approach. In contrast to deterministic, the heuristical approach should solve the problem faster. However, this is only a trade-off of the mathematical certainty of finding the minimum for speed. We have compared the performance of five different evolution strategies: Differential Evolution, Cuckoo Search, Harmony Search, CMA-ES, and Elephant Herding Optimization.

All presented algorithms for solving the phase stability testing problem have been tested on numerous examples from the literature.

First, we have shown the performance of the numerical algorithm based on the Newton–Raphson method using the unified formulation of the phase stability testing. In all examples, the algorithm proceeded without any difficulties in both VTN and PTN specifications. In one example, to correctly model the water, we used the Peng–Robinson equation of state with association. Even with this more complex equation of state, the algorithm found the minima of the TPD function successfully in all cases. Then, the procedure for solving the system of linear equations arising from the linearization of the VTN-phase stability problem using the Sherman–Morrison iterations has been tested. The actual speed-up of the computation has been presented in two computational studies. In both examples, the new algorithm was able to reduce the computation times if the number of components was at least 10. For smaller mixtures, the classical algorithm is preferred. For the mixtures with 45 components, the computation speed-ups for examples C1 and C2 were 4.75 and 3.66, respectively. Moreover, we have shown that in the stable area, where more calculation has to be performed, the speed-up is even higher. Then, we tested the global optimization method based on the Branch and Bound strategy. In our tests, our new splitting strategy was at least ten times faster than the splitting from Kou and Sun (2018) with binary mixtures. In case of a four-component mixture, the strategy from Kou and Sun (2018) needed the maximum number of iterations in almost all cases. Therefore, the difference between the splitting strategies in this example was not that significant. Furthermore, in one example, we managed to find a solution that previously used locally convergent algorithms have not detected. However, the computation times are higher than the times with a stand-alone gradient method, so the usability in large scale algorithms is only possible for mixtures with a small number of components. Then, we have compared the global heuristical algorithms on two examples from the literature. The best performance had the Differential Evolution and CMA-ES algorithm. The other three algorithms had not been able to find the solution in most cases. Moreover, we have performed the Wilcoxon signed-rank test to prove the significant difference between the methods. All pairwise tests resulted in the rejection of the hypothesis about the median equality. The highest p-value was less than 10^{-9}. However, none of the evolution strategies found the solution in 100 percent of cases. Therefore, the usage of these algorithms in the area of chemical engineering is limited. Even more, in comparison with the classical Newton–Raphson method with line-search, the computation times of the evolution strategies were significantly higher. Lastly, we have tested the SSI method in the three main specifications: VTN, UVN, and PTN. The method solves the PTN-specification without any difficulties. Moreover, the combination of the Newton–Raphson and SSI has seemed to be a robust option. However, in the VTN-, and UVN-specification, the method has encountered a problem with convergence in the stable area.

Phase equilibrium computation

In Chapter 4, the phase equilibrium computation for multi-component mixtures has been discussed. We have presented a unified formulation of this problem and have verified that the formulation covers the VTN-, UVN-, and PTN-specification. The unified formulation transforms the phase equilibrium problem into a global non-convex optimization problem with constraints. We have presented a numerical algorithm based on the elimination of the constraints and the Newton–Raphson method. The algorithm has been designed to find the equilibrium state with a priori given number of phases. As the number of phases is in practice a priori unknown, we have presented a strategy for computation of the equilibrium state without a priori given number of phases. This strategy has been based on the repeated phase stability testing and phase split computation until a stable state is found.

The numerical algorithm for solving the phase equilibrium computation has been tested on examples from the literature. First, examples showing the performance in the UVN-specification has been presented. In all cases, the algorithm proceeded without difficulties and has found a sequence of states with increasing value of the total entropy converging towards an equilibrium state consisting of up to three phases. Thanks to this property, convergence towards the trivial solution does not occur. Using the initial guess in the UVN-flash from the UVN-stability analysis allows avoiding the need for estimates of the pressure and temperature of the system, which were required in the previous works. The numerical difficulties mentioned in Castier (2009), which required some parts of the algorithm to be performed in the complex arithmetics, are thus avoided. Then, as we verify that the general formulation of the phase equilibrium problems represent commonly used flash formulations in our own examples, we have compared the behaviour of VTN-, UVN-, and PTN- flash computation using the elimination method in different physical situations. In the presented examples, we have demonstrated that these formulations are not equivalent, and the computation can proceed very differently. Especially in our implementation of the PTN-flash computation, the number of iterations needed for convergence strongly differs from the other two formulations. We have compared our implementation of the PTN-flash with a conventional PTN-flash solver. In Example G2: mixture $\mathrm{C_1{-}CO_2{-}C_{16}}$ and Example G3: mixture $\mathrm{CO_2{-}N_2{-}C_i}$, we observed that the conventional solver found a solution in fewer iterations than ours. However, the results were different, and by comparing the values of the Gibbs free energy, we find that our phase split is more precise. In Example G4: mixture $\mathrm{C_1{-}CO_2}$ and Example G5: mixture $\mathrm{C_1{-}CO_2{-}H_2S}$, in the vicinity of the phase boundary, our PTN solver needed more iterations. We have observed the same behaviour of the conventional solver in one of these examples, while in the other example, the conventional solver performed well even when approaching the phase boundary.

Compositional simulations

In Chapter 5, we have presented a possible application of the phase equilibrium computation in the multi-phase compositional flow in a porous medium. The mathematical model has consisted of the mass conservation equations for each component, extended Darcy's law for the velocity field of each phase, and the pressure equation for the pressure field. The numerical solution has been based on a mixed-hybrid finite element method and a novel iterative IMPEC scheme. Unlike in traditional solvers, the local thermodynamical behaviour has been determined by the phase equilibrium computation in the VTN-specification. We have provided examples showing the performance of the numerical scheme. The converge of the numerical algorithm has been verified using the experimental orders of convergence (EOC).

Appendices

Component database A

Here, we present chemical properties of all component used in examples. The data comes from various research papers and books, mainly Reid et al. (1987) and Firoozabadi (2016). The components are listed by their ID used in this thesis and the values are:

T_c	[K]	critical temperature,
P_c	[MPa]	critical pressure,
ω	[-]	acentric factor,
M	[g mol^{-1}]	molar weight,
α_k	[J mol^{-1} K^{-k-1}]	correlation coefficients for $c_p^{(ig)} = \sum_{k=0}^{3} T^k \alpha_k$.

The correlation coefficients α_i are only listed for components where the UVN-specification was used.

ID	T_c	P_c	ω	M	α_0	$\alpha_1 \times 10^1$	$\alpha_2 \times 10^4$	$\alpha_3 \times 10^8$
components								
N_2	126.21	3.390	0.0390	28.0	31.15	−0.1357	0.2680	0.0293
CO_2	304.14	7.375	0.2390	44.0	19.80	0.7344	−0.5602	−1.715
H_2S	373.20	8.940	0.0810	34.1	31.94	0.01463	0.2432	−1.176
C_1	190.56	4.599	0.0110	16.04	19.25	0.5213	0.1197	−1.132
C_2	305.32	4.872	0.0990	30.10	5.409	1.781	−0.6938	0.8713
C_3	369.83	4.248	0.1530	44.096	−4.224	3.063	−1.586	3.215
C_3H_6	364.9	4.600	0.144	42.08	3.710	2.345	−1.160	2.205
iC_4	408.20	3.650	0.1830	58.1	−1.390	3.847	−1.846	2.895
C_4	425.12	3.796	0.1990	58.1	9.487	3.313	−1.108	−0.2822
iC_5	460.40	3.380	0.2270	72.15	−9.525	5.066	−2.729	5.723
C_5	469.70	3.370	0.2510	72.15	3.626	4.873	−2.580	5.305
C_6	507.40	3.012	0.2960	86.2	−4.413	5.820	−3.119	6.494
C_7	556.45	2.675	0.2940	100.0				
C_8	574.76	2.330	0.4180	114.2				
C_9	593.07	2.155	0.4910	128.0				
C_{10}	617.07	2.110	0.5340	142.0				
C_{11}	638.24	1.974	0.5660	156.0				
C_{12}	657.81	2.167	0.7251	170.0				
C_{13}	675.69	2.044	0.7771	184.0				
C_{14}	691.84	1.925	0.8293	198.0				
C_{15}	702.79	1.824	0.8764	212.0				
C_{16}	715.63	1.725	0.9271	226.0	−69.02	16.54	−9.613	21.43
C_{17}	728.11	1.634	0.9771	240.0				
C_{18}	738.13	1.557	1.0237	254.0				
C_{19}	749.76	1.474	1.0788	268.0				

ID	T_c	P_c	ω	M	α_0	$\alpha_1 \times 10^1$	$\alpha_2 \times 10^4$	$\alpha_3 \times 10^8$
C_{20}	758.61	1.408	1.1280	282.0				
C_{21}	768.11	1.348	1.1761	296.0				
C_{22}	777.55	1.292	1.2254	310.0				
C_{23}	787.12	1.238	1.2781	324.0				
C_{24}	795.55	1.190	1.3315	338.0				
C_{25}	803.69	1.147	1.3834	352.0				
C_{26}	810.22	1.114	1.4295	366.0				
C_{27}	819.52	1.074	1.4859	380.0				
C_{28}	824.86	1.047	1.5342	394.0				
C_{29}	831.36	1.019	1.5849	408.0				
C_{30}	837.94	0.993	1.6385	422.0				
C_{31}	845.34	0.970	1.6877	436.0				
C_{32}	851.87	0.948	1.7432	450.0				
C_{33}	858.49	0.927	1.8019	464.0				
C_{34}	863.52	0.911	1.8543	478.0				
C_{35}	869.83	0.895	1.9092	492.0				
C_{36}	874.79	0.881	1.9629	506.0				
C_{37}	880.83	0.869	2.0137	520.0				
C_{38}	886.00	0.857	2.0742	534.0				
C_{39}	891.18	0.846	2.1365	548.0				
H_2O	647.3	22.12	0.3440	18.015	32.24	0.019 24	0.1055	−0.3596
pseudo-components								
PC_1	333.91	5.329	0.1113	34.64				
PC_2	456.25	3.445	0.2344	69.52				
PC_3	590.76	2.376	0.4470	124.57				
C_{7+}	647.59	2.224	0.7006	164.7	−5.146	6.762	−3.651	7.658
C_{12+}	742.58	1.341	0.9125	248.3				
C_{20+}	793.40	1.202	1.3175	334.0				
C_{25+}	849.61	0.99	1.6043	437.75				
C_{30+}	860.39	0.800	1.7500	550.0				
C_{40+}	896.12	0.837	2.1941	562.0				
$PC_{6\text{-}9}$	547.43	3.03	0.4099	103.56				
$PC_{10\text{-}14}$	643.77	2.29	0.6714	161.99				
$PC_{15\text{-}19}$	724.23	1.70	0.9296	233.97				
$PC_{20\text{-}24}$	777.38	1.34	1.1603	302.66				

Table A.1: Database of components and pseudo-components used in Examples.

Miscellaneous equations B

B.1 Viscosity model

The dynamic viscosity η_α is calculated using the Lohrenz, Bray, and Clark model presented in Lohrenz et al. (1964). According to Lohrenz et al. (1964), the scientific units of measurement have to be used: temperature in the kelvin [K], pressure in the atmosphere [atm], and molar weights in the pound per pound-mole [lb lb-mol^{-1}]. Here, we briefly present the algorithm. First, the zero viscosity η_0 of the i-th component is computed using

$$\eta_{0,i} = \begin{cases} 17.78\frac{(4.580T_{\mathrm{r},i}-1.67)^{\frac{5}{8}}}{\zeta_i} \times 10^{-5}, & T_{\mathrm{r},i} > 1.5, \\ 34.0\frac{T_{\mathrm{r},i}^{0.94}}{\zeta_i} \times 10^{-5}, & T_{\mathrm{r},i} \leqslant 1.5, \end{cases} \tag{B.1}$$

where $T_{\mathrm{r},i} = T/T_{\mathrm{c},i}$ is the reduced temperature, and ζ_i is the viscosity reducing parameter defined as

$$\zeta_i = \frac{T_{\mathrm{c},i}^{\frac{1}{6}}}{\sqrt{M_i} P_{c,i}^{\frac{2}{3}}}. \tag{B.2}$$

Then, the dynamic viscosity at the atmospheric pressure η_α^* is computed using

$$\eta_\alpha^* = \frac{\sum\limits_{i=1}^{n} z_i \eta_{0,i} \sqrt{M_i}}{\sum\limits_{i=1}^{n} z_i \sqrt{M_i}}, \tag{B.3}$$

where z_i is mole fraction of the i-th component. To conclude, the dynamic viscosity of the phase α is computed using

$$\eta_\alpha = \eta_\alpha^* + \xi_m^{-1} \left(0.1023 + 0.023364\rho_r + 0.058533\rho_r^2 - 0.040758\rho_r^3 + 0.0093324\rho_r^4\right)^4 - 10^{-4}, \tag{B.4}$$

where ρ_r is the reduced mass density, and ξ_m is mixture viscosity parameter defined as

$$\xi_m = \frac{\left(\sum\limits_{i=1}^{n} z_i T_{\mathrm{c},i}\right)^{\frac{1}{6}}}{\sqrt{\sum\limits_{i=1}^{n} z_i M_i} \left(\sum\limits_{i=1}^{n} z_i P_{c,i}\right)^{\frac{2}{3}}}. \tag{B.5}$$

B.2 Computation of $\frac{\partial p^{(\mathrm{eq})}}{\partial c_i}$

In this section, we present a method for the computation of the Θ_i coefficients which are defined as

$$\Theta_i\left(c_1,\ldots,c_n\right) = \frac{\partial p^{(\mathrm{eq})}}{\partial c_i}\left(c_1,\ldots,c_n\right). \tag{B.6}$$

In the Π-phase equilibrium, the saturations, pressures, and chemical potentials satisfy

$$\sum_{\alpha=1}^{\Pi} S_\alpha = 1, \tag{B.7}$$

$$\sum_{\alpha=1}^{\Pi} S_\alpha c_{\alpha,i} = c_i, \tag{B.8}$$

$$p^{(\mathrm{EOS})}\left(c_{\alpha,1},\ldots,c_{\alpha,n}\right) = p^{(\mathrm{EOS})}\left(c_{\beta,1},\ldots,c_{\beta,n}\right), \quad \alpha,\beta\in\widehat{\Pi}, \alpha\neq\beta, \tag{B.9}$$

$$\mu_i^{(\mathrm{EOS})}\left(c_{\alpha,1},\ldots,c_{\alpha,n}\right) = \mu_i^{(\mathrm{EOS})}\left(c_{\beta,1},\ldots,c_{\beta,n}\right), \quad \alpha,\beta\in\widehat{\Pi}, \alpha\neq\beta. \tag{B.10}$$

First two equations represent the conservation of volume and mass (see Section 4.1.2), the third and fourth are the necessary equilibrium conditions (see Section 2.1.3). Therefore, we have $2+(1+n)\binom{\Pi}{2}$ equations. Differentiation these equations with respect to c_j (other c_i are kept constant) results in

$$\sum_{\alpha=1}^{\Pi} \frac{\partial S_\alpha}{\partial c_j}\left(c_1,\ldots,c_n\right) = 0, \tag{B.11}$$

$$\sum_{\alpha=1}^{\Pi}\left(\frac{\partial S_\alpha}{\partial c_j}\left(c_1,\ldots,c_n\right)c_{\alpha,i} + S_\alpha\frac{\partial c_{\alpha,i}}{\partial c_j}\left(c_1,\ldots,c_n\right)\right) = \delta_{i,j}, \tag{B.12}$$

$$\begin{aligned}\sum_{k=1}^{n}\frac{\partial p^{(\mathrm{EOS})}}{\partial c_{\alpha,k}}\left(c_{\alpha,1},\ldots,c_{\alpha,n}\right)\frac{\partial c_{\alpha,k}}{\partial c_j}\left(c_1,\ldots,c_n\right)\\ = \sum_{k=1}^{n}\frac{\partial p^{(\mathrm{EOS})}}{\partial c_{\beta,k}}\left(c_{\beta,1},\ldots,c_{\beta,n}\right)\frac{\partial c_{\beta,k}}{\partial c_j}\left(c_1,\ldots,c_n\right),\end{aligned} \tag{B.13}$$

$$\begin{aligned}\sum_{k=1}^{n}\frac{\partial \mu_i^{(\mathrm{EOS})}}{\partial c_{\alpha,k}}\left(c_{\alpha,1},\ldots,c_{\alpha,n}\right)\frac{\partial c_{\alpha,k}}{\partial c_j}\left(c_1,\ldots,c_n\right)\\ = \sum_{k=1}^{n}\frac{\partial \mu_i^{(\mathrm{EOS})}}{\partial c_{\beta,k}}\left(c_{\beta,1},\ldots,c_{\beta,n}\right)\frac{\partial c_{\beta,k}}{\partial c_j}\left(c_1,\ldots,c_n\right)\end{aligned} \tag{B.14}$$

The unknowns in the previous system are

$$\frac{\partial S_\alpha}{\partial c_j}\left(c_1,\ldots,c_n\right), \frac{\partial c_{\alpha,i}}{\partial c_j}\left(c_1,\ldots,c_n\right) \tag{B.15}$$

for $i,j\in\widehat{n}$ and $\alpha\in\widehat{\Pi}$. Therefore, we have $\Pi n+\Pi n^2$ unknowns. Since

$$\frac{\partial p^{(\mathrm{eq})}}{\partial c_i}\left(c_1,\ldots,c_n\right) = \sum_{k=1}^{n}\frac{\partial p^{(\mathrm{EOS})}}{\partial c_{\alpha,k}}\left(c_{\alpha,1},\ldots,c_{\alpha,n}\right)\frac{\partial c_{\alpha,k}}{\partial c_i}, \tag{B.16}$$

the unknown derivatives $\Theta_i = \frac{\partial p^{(\mathrm{eq})}}{\partial c_i}$ are the right or left hand side of the equation (B.13) for an arbitrary α or β in $\widehat{\Pi}$. The solution of the linear system given by equations (B.11)–(B.14) can be solved using LU decomposition.

Bibliography

A

Acs, G., Doleschall, S., and Farkas, E. (1985). General purpose compositional model. *Society of Petroleum Engineers*, 25:543–553.

Androulakis, I. P., Maranas, C. D., and Floudas, C. A. (1995). αBB: a global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7:337–363.

Arfken, G. (2005). *Mathematical methods for physicists.* Elsevier, Boston.

Auger, A. and Hansen, N. (2005). Performance evaluation of an advanced local search evolutionary algorithm. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1777–1784.

B

Baker, L., Pierce, A., and Luks, K. (1982). Gibbs energy analysis of phase equilibria. *Society of Petroleum Engineers Journal*, 22(05):731–742.

Bartlett, M. S. (1951). An inverse matrix adjustment arising in discriminant analysis. *Annals of Mathematical Statistics*, 22(1):107–111.

Bačák, M. and Borwein, J. M. (2011). On difference convexity of locally Lipschitz functions. *Optimization*, 60(8-9):961–978.

Bear, J. (1988). *Dynamics of fluids in porous media.* Dover, New York.

Bejan, A. (2013). *Convection heat transfer.* Wiley, Hoboken, N.J.

Bi, R., Firoozabadi, A., and Myint, P. C. (2020a). Efficient and robust phase-split computations in the internal energy, volume, and moles (UVN) space. *Fluid Phase Equilibria*, 526:112729.

Bi, R., Zidane, A., and Firoozabadi, A. (2020b). Efficient and robust stability analysis in the internal energy, volume, and moles (UVN) space. *Fluid Phase Equilibria*, 512:112468.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization.* Cambridge University press.

Brezzi, F. and Fortin, M. (2012). *Mixed and hybrid finite element methods*, volume 15. Springer Science & Business Media.

Broyden, C. G. (1970). The convergence of a class of double-rank minimization algorithms 1. General considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90.

Buckley, S. and Leverett, M. (1942). Mechanism of fluid displacement in sands. *Transactions of the AIME*, 146(01):107–116.

Butler, R. (1991). *Thermal recovery of oil and bitumen.* Prentice Hall, Englewood Cliffs, N.J.

C

Callen, H. B. (1985). *Thermodynamics and an introduction to thermostatistics; 2nd ed.* Wiley, New York, NY.

Castier, M. (2009). Solution of the isochoric-isoenergetic flash problem by direct entropy maximization. *Fluid Phase Equilibria*, 276:7–17.

Castier, M. (2010). Dynamic simulation of fluids in vessel via entropy maximization. *Journal of Industrial and Engineering Chemistry*, 16:122–129.

Castier, M. (2014). Helmholtz function-based global phase stability test and its link to the isothermal-isochoric flash problem. *Fluid Phase Equilibria*, 379:104–111.

Chapman, W., Gubbins, K., Jackson, G., and Radosz, M. (1989). SAFT: Equation-of-state solution model for associating fluids. *Fluid Phase Equilibria*, 52:31–38.

Chen, H., Fan, X., and Sun, S. (2019). A fully mass-conservative iterative IMPEC method for multicomponent compressible flow in porous media. *Journal of Computational and Applied Mathematics*, 362:1 – 21.

Chen, Z. (2006). *Computational methods for multiphase flows in porous media.* Society for Industrial and Applied Mathematics, Philadelphia.

Chernyavsky, I., Jensen, O., and Leach, L. (2010). A mathematical model of intervillous blood flow in the human placentone. *Placenta*, 31(1):44–52.

Clausius, R. (1857). Über die Art der Bewegung, welche wir Wärme nennen. *Annalen der Physik und Chemie*, 176(3):353–380.

Coats, K. H. (1980). An equation of state compositional model. *Society of Petroleum Engineers Journal*, 20(05):363–376.

D

Dayyani, S., Prasher, S. O., Madani, A., and Madramootoo, C. A. (2010). Development of DRAIN–WARMF model to simulate flow and nitrogen transport in a tile-drained agricultural watershed in eastern canada. *Agricultural Water Management*, 98(1):55–68.

Demirel, Y. (2014). *Nonequilibrium thermodynamics: transport and rate processes in physical, chemical and biological systems.* Elsevier, Amsterdam.

Drazin, P. G. (2006). *The Navier-Stokes Equations: A Classification of Flows and Exact Solutions.* Cambridge University Press, Cambridge.

Dunn, O. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64.

Durlofsky, L., Efendiev, Y., and Ginting, V. (2007). An adaptive local–global multiscale finite volume element method for two-phase flow simulations. *Advances in Water Resources*, 30(3):576–588.

E

Eddelbuettel, D. and Sanderson, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics and Data Analysis*, 71:1054–1063.

Efendiev, Y., Ginting, V., Hou, T., and Ewing, R. (2006). Accurate multiscale finite element methods for two-phase flow simulations. *Journal of Computational Physics*, 220(1):155–174.

Ewing, R. (1983). *The mathematics of reservoir simulation.* Society for Industrial and Applied Mathematics, Philadelphia.

F

Fang, H. and O'Leary, D. P. (2007). Modified Cholesky algorithms: a catalog with new approaches. *Mathematical Programming*, 115(2):319–349.

Firoozabadi, A. (1999). *Thermodynamics of hydrocarbon reservoirs.* McGraw-Hill, New York.

Firoozabadi, A. (2016). *Thermodynamics and applications in hydrocarbon energy production.* McGraw-Hill Education, New York.

Firoozabadi, A. and Myint, P. C. (2010). Prospects for subsurface CO2 sequestration. *AIChE Journal*, 56(6):1398–1405.

Firoozabadi, A. and Pan, H. (2002). Fast and robust algorithm for compositional modeling: Part I - stability analysis testing. *SPE Journal*, 7(1):78–89.

Fletcher, R. (1970). A new approach to variable metric algorithms. *The Computer Journal*, 13(3):317–322.

Fučík, R., Illangasekare, T. H., and Beneš, M. (2016). Multidimensional self-similar analytical solutions of two-phase flow in porous media. *Advances in Water Resources*, 90:51–56.

Fučík, R., Klinkovský, J., Solovský, J., Oberhuber, T., and Mikyška, J. (2019). Multidimensional mixed–hybrid finite element method for compositional two-phase flow in heterogeneous porous media and its parallel implementation on GPU. *Computer Physics Communications*, 238:165–180.

Fussell, D. D. and Yanosik, J. (1978). An iterative sequence for phase equilibrium calculations incorporating the Redlich-Kwong equation of state. *Society of Petroleum Engineers*, 18:173–182.

Fussell, L. T. (1979). A technique for calculating multiphase equilibria. *Society of Petroleum Engineers*, 19:203–210.

Fučík, R. and Mikyška, J. (2011). Mixed-hybrid finite element method for modelling two-phase flow in porous media. *Journal of Math-for-Industry*, 3(2011C-2):9–19.

G

Gaspar Ravagnani, A., Ligero, E., and Suslick, S. (2009). CO2 sequestration through enhanced oil recovery in a mature oil field. *Journal of Petroleum Science and Engineering*, 65(3):129–138.

Geem, Z. (2009). *Music-inspired harmony search algorithm: theory and applications*. Springer-Verlag, Berlin.

Gibbs, J. W. (1873). A method of geometrical representation of the thermodynamic properties of substances by means of surfaces. *Transactions of the Connecticut Academy of Arts and Sciences*, 2:382–404.

Gibbs, J. W. (1876). On the equilibrium of heterogeneous substances. *Transactions of the Connecticut Academy of Arts and Sciences*, 3:108–248.

Gibbs, J. W. (1948). *The collected works of J. Willard Gibbs*. Yale Univ. Press, New Haven.

Gill, P. E. and Murray, W. (1974). Newton-type methods for unconstrained and linearly constrained optimization. *Mathematical Programming*, 7(1):311–350.

Gill, P. E., Murray, W., and Wright, M. H. (1981). *Practical optimization*. Academic Press, London New York.

Goldfarb, D. (1970). A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24(109):23–23.

Gorucu, S. E. and Johns, R. T. (2014). Comparison of reduced and conventional two-phase flash calculations. *SPE Journal*, 20(02):294–305.

Gross, J. and Sadowski, G. (2001). Perturbed-Chain SAFT: An equation of state based on a perturbation theory for chain molecules. *Industrial & Engineering Chemistry Research*, 40(4):1244–1260.

Gupta, A. K., Bishnoi, P., and Kalogerakis, N. (1990). Simultaneous multiphase isothermal/isenthalpic flash and stability calculations for reacting/non-reacting systems. *Gas Separation & Purification*, 4(4):215–222.

H

Hager, W. W. (1989). Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239.

Hansen, N. (2016). The CMA evolution strategy: A tutorial. *ArXiv e-prints*, arXiv:1604.00772.

Hansen, N. and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195.

Hassanizadeh, S. M. (1986). Derivation of basic equations of mass transport in porous media, Part 2. Generalized Darcy's and Fick's laws. *Advances in Water Resources*, 9(4):207–222.

Haugen, K. B. and Beckner, B. (2013). A critical comparison of reduced and conventional EOS algorithms. *Society of Petroleum Engineers*, 18:378–388.

Haugen, K. B., Firoozabadi, A., and Sun, L. (2010). Efficient and robust three-phase split computations. *AIChE Journal*, 57(9):2555–2565.

Henderson, N., de Sá Rego, M., Sacco, W. F., and Rodrigues Jr., R. A. (2015). A new look at the topographical global optimization method and its application to the phase stability analysis of mixtures. *Chemical Engineering Science*, 127:151–174.

Hill, T. (1986). *An introduction to statistical thermodynamics.* Dover Publications, New York.

Hobson, G. (1975). *Introduction to petroleum geology.* Scientific Press, Beaconsfield.

Holt, T., Jensen, J.-I., and Lindeberg, E. (1995). Underground storage of CO2 in aquifers and oil reservoirs. *Energy Conversion and Management*, 36(6):535–538. Proceedings of the Second International Conference on Carbon Dioxide Removal.

Hoteit, H. and Firoozabadi, A. (2005). Multicomponent fluid flow by discontinuous Galerkin and mixed methods in unfractured and fractured media. *Water Resources Research*, 41(11).

Hoteit, H. and Firoozabadi, A. (2006a). Compositional modeling by the combined discontinuous Galerkin and mixed methods. *Society of Petroleum Engineers*, 11:19–34.

Hoteit, H. and Firoozabadi, A. (2006b). Simple phase stability-testing algorithm in the reduction method. *AIChE Journal*, 52(8):2909–2920.

Hoteit, H. and Firoozabadi, A. (2008). Numerical modeling of two-phase flow in heterogeneous permeable media with different capillary pressures. *Advances in Water Resources*, 31:56–73.

Huyakorn, P. S. (1983). *Computational methods in subsurface flow.* Elsevier Science, Oxford.

J

Jensen, B. H. and Fredenslund, A. (1987). A simplified flash procedure for multicomponent mixtures containing hydrocarbons and one non-hydrocarbon using two-parameter cubic equations of state. *Industrial & Engineering Chemistry Research*, 26:2129–2134.

Jensen, J. L. W. V. (1906). Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30:175–193.

Jim Douglas, J. (1983). Finite difference methods for two-phase incompressible flow in porous media. *SIAM Journal on Numerical Analysis*, 20(4):681–696.

Jindrová, T. and Mikyška, J. (2013). Fast and robust algorithm for calculation of two-phase equilibria at given volume, temperature, and moles. *Fluid Phase Equilibria*, 353:101–114.

Jindrová, T. and Mikyška, J. (2015). General algorithm for multiphase equilibria calculation at given volume, temperature, and moles. *Fluid Phase Equilibria*, 393:7–25.

K

Kaviany, M. (1991). *Principles of Heat Transfer in Porous Media.* Springer US, New York, NY.

Kaya, Y. (1995). The role of CO2 removal and disposal. *Energy Conversion and Management*, 36(6):375–380. Proceedings of the Second International Conference on Carbon Dioxide Removal.

Kolev, N. (2005). *Multiphase flow dynamics.* Springer, Berlin New York.

Kontogeorgis, G. (2010). *Thermodynamic models for industrial applications: from classical and advanced mixing rules to association theories.* Wiley, Hoboken, N.J.

Kontogeorgis, G. M., Voutsas, E. C., Yakoumis, I. V., and Tassios, D. P. (1996). An equation of state for associating fluids. *Industrial & Engineering Chemistry Research*, 35(11):4310–4318.

Kontogeorgis, G. M., Yakoumis, I. V., Meijer, H., Hendriks, E., and Moorwood, T. (1999). Multicomponent phase equilibrium calculations for water–methanol–alkane mixtures. *Fluid Phase Equilibria*, 158-160:201–209.

Kou, J. and Sun, S. (2018). Thermodynamically consistent modeling and simulation of multi-component two-phase flow model with partial miscibility. *Computer Methods in Applied Mechanics and Engineering*, 331:623–649.

Krönig, A. (1856). Grundzüge einer Theorie der Gase. *Annalen der Physik und Chemie*, 175(10):315–322.

Kruskal, W. H. (1957). Historical notes on the Wilcoxon unpaired two-sample test. *Journal of the American Statistical Association*, 52(279):356–360.

Kukal, J. and Mojzeš, M. (2018). Quantile and mean value measures of search process complexity. *Journal of Combinatorial Optimization*, 35(4):1261–1285.

L

Leveque, R. J. (2004). *Finite Volume Methods for Hyperbolic Problems.* Cambridge University Press.

Li, Y., Qiao, Z., Sun, S., and Zhang, T. (2020). Thermodynamic modeling of CO2 solubility in saline water using NVT flash with the cubic-plus-association equation of state. *Fluid Phase Equilibria*, 520:112657.

Li, Z. and Firoozabadi, A. (2012). General strategy for stability testing and phase-split calculation in two and three phases. *Society of Petroleum Engineers*, 17:1096–1107.

Lohrenz, J., Bray, B. G., and Clark, C. R. (1964). Calculating viscosities of reservoir fluids from their compositions. *Journal of Petroleum Technology*, 16(10):1171–1176.

M

Marchi, E. (2010). When is the product of two concave functions concave? *International Journal of Mathematics, Game Theory, and Algebra*, 19(3):165–172.

McWhorter, D. B. and Sunada, D. K. (1990). Exact integral solutions for two-phase flow. *Water Resources Research*, 26(3):399–413.

Michelsen, M. (1993). Phase equilibrium calculations. What is easy and what is difficult? *Computers & Chemical Engineering*, 17(5-6):431–439.

Michelsen, M. L. (1982a). The isothermal flash problem, Part 1. Stability. *Fluid Phase Equilibria*, 9:1–19.

Michelsen, M. L. (1982b). The isothermal flash problem, Part 2. Phase-split computation. *Fluid Phase Equilibria*, 9:21–40.

Michelsen, M. L. (1986a). Simplified flash calculations for cubic equations of state. *Industrial & Engineering Chemistry Process Design and Development*, 25(1):184–188.

Michelsen, M. L. (1986b). Simplified flash calculations for cubic equations of state. *Industrial & Engineering Chemistry Process Design and Development*, 25:184–188.

Michelsen, M. L. (1987). Multiphase isenthalpic and isentropic flash algorithms. *Fluid Phase Equilibria*, 33(1):13–27.

Michelsen, M. L. (1999a). State function based flash specifications. *Fluid Phase Equilibria*, 158-160:617–626.

Michelsen, M. L. (1999b). State function based flash specifications. *Fluid Phase Equilibria*, 158-160:617–626.

Michelsen, M. L. (2006). Robust and efficient solution procedures for association models. *Industrial & Engineering Chemistry Research*, 45(25):8449–8453.

Michelsen, M. L. (2012). Computation of phase equilibria: Status and future perspectives. In *IX Iberoamerican Conference on Phase Equilibria and Fluid Properties for Process Design Equiface 8-12 Octorber 2012, Puerto Varas, Chile.*

Michelsen, M. L. and Mollerup, J. M. (2007). *Thermodynamic Models: Fundamentals and Computational Aspects*. Tie-Line Publications, Holte, Denmark.

Michelsen, M. L., Yan, W., and Stenby, E. H. (2013). A comparative study of reduced-variables-based flash and conventional flash. *SPE Journal*, 18(5):952–959.

Mikyška, J. and Firoozabadi, A. (2012). Investigation of mixture stability at given volume, temperature, and moles. *Fluid Phase Equilibria*, 321:1–9.

Mikyška, J. and Firoozabadi, A. (2011). A new thermodynamic function for phase-splitting at constant temperature, moles, and volume. *AIChE Journal*, 57(7):1897–1904.

Mohamed, A. W., Sabry, H. Z., and Khorshid, M. (2012). An alternative differential evolution algorithm for global optimization. *Journal of Advanced Research*, 3:149–165.

Moortgat, J., Sun, S., and Firoozabadi, A. (2011). Compositional modeling of three-phase flow with gravity using higher-order finite element methods. *Water Resources Research*, 47(5).

Mosavat, N. and Torabi, F. (2013). Performance of secondary carbonated water injection in light oil systems. *Industrial & Engineering Chemistry Research*, 53(3):1262–1273.

N

Nagarajan, N. R., Cullik, A. S., and Griewank, A. (1991). New strategy for phase equilibrium and critical point calculations by thermodynamic energy analysis. Part 1. Stability analysis and flash. *Fluid Phase Equilibria*, 62:191–210.

Nayagum, D., Schäfer, G., and Mosé, R. (2004). Modelling two-phase incompressible flow in porous media using mixed hybrid and discontinuous finite elements. *Computational Geosciences*, 8(1):49–73.

Nedelec, J. C. (1980). Mixed finite elements in R3. *Numerische Mathematik*, 35(3):315–341.

Neuman, S. P. (1977). Theoretical derivation of Darcy's law. *Acta Mechanica*, 25(3):153–170.

Nichita, D. V. (2017). Fast and robust phase stability testing at isothermal-isochoric conditions. *Fluid Phase Equilibria*, 447:107–124.

Nichita, D. V. (2018a). Density-based phase envelope construction. *Fluid Phase Equilibria*, 478:100–113.

Nichita, D. V. (2018b). A volume-based approach to phase equilibrium calculations at pressure and temperature specifications. *Fluid Phase Equilibria*, 461:70–83.

Nichita, D. V. (2018c). Volume-based phase stability testing at pressure and temperature specifications. *Fluid Phase Equilibria*, 458:123–141.

Nichita, D. V., Broseta, D., and de Hemptinne, J.-C. (2006a). Multiphase equilibrium calculation using reduced variables. *Fluid Phase Equilibria*, 246(1):15–27.

Nichita, D. V., Broseta, D., and Montel, F. (2007). Calculation of convergence pressure/temperature and stability test limit loci of mixtures with cubic equations of state. *Fluid Phase Equilibria*, 261(1-2):176–184.

Nichita, D. V., de Hemptinne, J.-C., and Gomez, S. (2009). Isochoric phase stability testing for hydrocarbon mixtures. *Petroleum Science and Technology*, 27:2177–2191.

Nichita, D. V., de los Angeles Duran Valencia, C., and Gomez, S. (2006b). Volume-based thermodynamics global phase stability analysis. *Chemical Engineering Communications*, 193:1194–1216.

Nichita, D. V., Gomez, S., and Luna, E. (2002). Multiphase equilibria calculation by direct minimization of Gibbs free energy with global optimization method. *Computers & Chemical Engineering*, 26(12):1703–1724.

Nichita, D. V. and Petitfrere, M. (2013). Phase stability analysis using a reduction method. *Fluid Phase Equilibria*, 358:27–39.

O

Olajire, A. A. (2014). Review of ASP EOR (alkaline surfactant polymer enhanced oil recovery) technology in the petroleum industry: Prospects and challenges. *Energy*, 77:963–982.

Ostermeier, A., Gawelczyk, A., and Hansen, N. (1994). A derandomized approach to self-adaptation of evolution strategies. *Evolutionary Computation*, 2(4):369–380.

P

Pan, H. and Firoozabadi, A. (1998). Complex multiphase equilibrium calculations by direct minimization of Gibbs free energy by use of simulated annealing. *SPE Reservoir Evaluation & Engineering*, 1(1):36–42.

Pan, H. and Firoozabadi, A. (2003). Fast and robust algorithm for compositional modeling: Part II - two-phase flash computations. *Society of Petroleum Engineers*, 8:380–391.

Paterson, D., Michelsen, M. L., Yan, W., and Stenby, E. H. (2018). Extension of modified RAND to multiphase flash specifications based on state functions other than (T,P). *Fluid Phase Equilibria*, 458:288–299.

Pavlyukevich, I. (2007a). Cooling down Lévy flights. *Journal of Physics A: Mathematical and Theoretical*, 40(41):12299–12313.

Pavlyukevich, I. (2007b). Lévy flights, non-local search and simulated annealing. *Journal of Computational Physics*, 226(2):1830–1844.

Peng, D.-Y. and Robinson, D. B. (1976). A new two-constant equation of state. *Industrial & Engineering Chemistry Fundamentals*, 15(1):59–64.

Pereira, F., Jackson, G., Galindo, A., and Adjiman, C. S. (2010). Robust algorithms for the calculation of phase equilibrium. In *20th European Symposium on Computer Aided Process Engineering*.

Petitfrere, M. and Nichita, D. V. (2015a). A comparison of conventional and reduction approaches for phase equilibrium calculations. *Fluid Phase Equilibria*, 386:30–46.

Petitfrere, M. and Nichita, D. V. (2015b). Multiphase equilibrium calculations using a reduction method. *Fluid Phase Equilibria*, 401:110–126.

Petitfrere, M. and Nichita, D. V. (2016). On a choice of independent variables in Newton iterations for multiphase flash calculations. *Fluid Phase Equilibria*, 427:147–151.

Pétrowski, A. (2017). *Evolutionary algorithms*. ISTE, London.

Polívka, O. and Mikyška, J. (2011). Numerical simulation of multicomponent compressible flow in porous medium. *Journal of Math-for-Industry*, 3(2011C-7):53–60.

Polívka, O. and Mikyška, J. (2014). Compositional modeling in porous media using constant volume flash and flux computation without the need for phase identification. *Journal of Computational Physics*, 272:149–179.

Polívka, O. and Mikyška, J. (2015). Compositional modeling of two-phase flow in porous media using semi-implicit scheme. *IAENG International Journal of Applied Mathematics*, 45(3):218–226.

Pozo, R. (1997). Template numerical toolkit for linear algebra: High performance programming with C++ and the standard template library. *The International Journal of Supercomputer Applications and High Performance Computing*, 11(3):251–263.

Prausnitz, J. M., Lichtenthaler, R. N., and de Azevedo, E. G. (1999). *Molecular thermodynamics of fluid-phase equilibria*. Prentice Hall PTR, Upper Saddle River, N.J, third edition.

Price, K., Storn, R., and Lampinen, J. (2005). *Differential evolution: a practical approach to global optimization.* Springer, Berlin New York.

Q

Qin, A., Huang, V., and Suganthan, P. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 13(2):398–417.

Qiu, L., Wang, Y., and Reitz, R. D. (2014). Multiphase dynamic flash simulations using entropy maximization and application to compressible flow with phase change. *AIChE Journal*, 60(8):3013–3024.

R

Rachford, H. and Rice, J. D. (1952). Procedure for use of electronic digital computers in calculating flash vaporization hydrocarbon equilibrium. *Journal of Petroleum Technology*, 4(10):189–196.

Raphson, J. (1697). *Analysis aequationum universalis seu ad aequationes algebraicas resolvendas methodus generalis, & expedita, ex nova infinitarum serierum methodo, deducta ac demonstrata: cui annexum est de spatio reali, seu ente infinito conamen mathematico-metaphysicum.* Londini: typis Tho. Braddyll, prostant venales apud Iohannem Taylor. ETH-Bibliothek Zürich, Rar 1459, https://doi.org/10.3931/e-rara-13516/.

Raviart, P.-A. and Thomas, J. (1977). *Mathematical Aspects of Finite Element Methods.* Springer.

Reid, R. C., Prausnitz, J. M., and Poling, B. E. (1987). *The Properties of Gases and Liquids.* McGraw-Hill, fourth edition.

Reynolds, A. M. and Frye, M. A. (2007). Free-flight odor tracking in drosophila is consistent with an optimal intermittent scale-free search. *PloS ONE*, 2(4):e354.

Rudin, W. (1976). *Principles of mathematical analysis.* McGraw-Hill, New York.

S

Saha, S. and Carroll, J. J. (1997). The isoenergetic-isochoric flash. *Fluid Phase Equilibria*, 138:23–41.

Sanderson, C. and Curtin, R. (2016). Armadillo: a template-based C++ library for linear algebra. *Journal of Open Source Software*, 1(2):26.

Sarker, R. (2002). *Evolutionary optimization.* Kluwer Academic Publishers, Boston.

Schnabel, R. B. and Eskow, E. (1990). A new modified Cholesky factorization. *SIAM Journal on Scientific and Statistical Computing*, 11(6):1136–1158.

Schnabel, R. B. and Eskow, E. (1999). A revised modified Cholesky factorization algorithm. *SIAM Journal on Optimization*, 9(4):1135–1148.

Sewell, M. J. (1987). *Maximum and Minimum Principles: A Unified Approach with Applications*. Cambridge Texts in Applied Mathematics. Cambridge University Press.

Shanno, D. F. (1970). Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24(111):647–647.

Sherafati, M. and Jessen, K. (2017). Stability analysis for multicomponent mixtures including capillary pressure. *Fluid Phase Equilibria*, 433:56–66.

Sherman, A. H. (1978). On Newton-iterative methods for the solution of systems of nonlinear equations. *SIAM Journal on Numerical Analysis*, 15(4):755–771.

Siccuan, J. H. (2001). Mémoire sur la puissance motrice de la chaleur. *Journal de l'École Polytechnique (in French)*, XIV:153–90.

Simon, D. (2013). *Evolutionary optimization algorithms*. Wiley-Blackwell, Chichester.

Slattery, J. (1999). *Advanced transport phenomena*. Cambridge University Press, New York.

Smejkal, T. and Mikyška, J. (2021). Multi-phase compressible compositional simulations with phase equilibrium computation in the VTN specification. In Paszynski, M., Kranzlmüller, D., Krzhizhanovskaya, V. V., Dongarra, J. J., and Sloot, P. M. A., editors, *Computational Science – ICCS 2021*, pages 159–172, Cham. Springer International Publishing.

Smejkal, T., Mikyška, J., and Kukal, J. (2021). Comparison of modern heuristics on solving the phase stability testing problem. *Discrete & Continuous Dynamical Systems - S*, 14(3):1161–1180.

Smejkal, T. and Mikyška, J. (2017). Phase stability testing and phase equilibrium calculation at specified internal energy, volume, and moles. *Fluid Phase Equilibria*, 431:82–96.

Smejkal, T. and Mikyška, J. (2018). Unified presentation and comparison of various formulations of the phase stability and phase equilibrium calculation problems. *Fluid Phase Equilibria*, 476:61–88.

Smejkal, T. and Mikyška, J. (2020). VTN-phase stability testing using the Branch and Bound strategy and the convex-concave splitting of the Helmholtz free energy density. *Fluid Phase Equilibria*, 503:112323.

Smejkal, T. and Mikyška, J. (2021). Efficient solution of linear systems arising in the linearization of the VTN-phase stability problem using the Sherman-Morrison iterations. *Fluid Phase Equilibria*, 527:112832.

Soave, G. (1972). Equilibrium constants from a modified Redlich–Kwong equation of state. *Chemical Engineering Science*, 27(6):1197–1203.

Souza, A. T., Cardozo-Filho, L., Wolff, F., and Guirardello, R. (2006). Application of interval analysis for Gibbs and Helmholtz free energy global minimization in phase stability analysis. *Brazilian Journal of Chemical Engineering*, 23:117–124.

Stadtherr, M. A., Xu, G., Solorzano, G. I. B., and Haynes, W. D. (2007). Reliable computation of phase stability and equilibrium using interval methods. *International Journal of Reliability and Safety*, 1(4):465–488.

Storn, R. and Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359.

Strauss, W. (2008). *Partial differential equations: an introduction.* Wiley, New York.

Sun, L., Kontogeorgis, G. M., von Solms, N., and Liang, X. (2019). Modeling of gas solubility using the electrolyte cubic plus association equation of state. *Industrial & Engineering Chemistry Research*, 58(37):17555–17567.

Sun, S., Firoozabadi, A., and Kou, J. (2012). Numerical modeling of two-phase binary fluid mixing using mixed finite elements. *Computational Geosciences*, 16:1101–1124.

T

Tisza, L. (1966). *Generalized thermodynamics.* M.I.T. Press, Cambridge, Mass.

Trévisan, D. and Periáñez, R. (2016). Coupling catchment hydrology and transient storage to model the fate of solutes during low-flow conditions of an upland river. *Journal of Hydrology*, 534:317–325.

V

van Duijn, C. J., Peletier, L. A., and Pop, I. S. (2007). A new class of entropy solutions of the Buckley–Leverett equation. *SIAM Journal on Mathematical Analysis*, 39(2):507–536.

Veselý, L. and Zajíček, L. (2009). On composition of DC functions and mapping. *Journal of Convex Analysis*, 16:423–439.

W

Walsh, M. (2003). *A generalized approach to primary hydrocarbon recovery.* Elsevier, Amsterdam Boston.

Wilson, G. M. (1969). A modified Redlich-Kwong equation of state, application to general physical data calculations, paper no. 15c. In *65th National AIChE Meeting, Cleveland, OH.*

Woodbury, M. A. (1950). Inverting modified matrices. Technical report, Statistical Research Group, Memo. Rep. no. 42, Princeton University, Princeton.

Wu, Z., Phillips, G. N., Tapia, R., and Zhang, Y. (2001). A fast Newton algorithm for entropy maximization in phase determination. *SIAM Review*, 43(4):623–642.

X

Xu, G., Vrennecke, J. F., and Stadtherr, M. A. (2002). Reliable computation of phase stability and equilibrium from the SAFT equation of state. *Industrial Engineering Chemistry*, 41:938–952.

Y

Yang, X.-S. (2009). Cuckoo search via Levy flights. In *World Congress on Nature Biologically Inspired Computing*, pages 210–214.

Yu, X. and Gen, M. (2010). *Introduction to evolutionary algorithms.* Springer, London.

Z

Zhang, H., Bonilla-Petriciolet, A., and Rangaiah, G. P. (2011). A review on global optimization methods for phase equilibrium modeling and calculations. *The Open Thermodynamics Journal*, 5(S1):71–92.

Zhang, Z., Pan, S.-Y., Li, H., Cai, J., Olabi, A. G., Anthony, E. J., and Manovic, V. (2020). Recent advances in carbon dioxide utilization. *Renewable and Sustainable Energy Reviews*, 125:109799.

Zhu, D. and Okuno, R. (2016). Multiphase isenthalpic flash integrated with stability analysis. *Fluid Phase Equilibria*, 423:203–219.

Zidane, A. and Firoozabadi, A. (2019). Two-phase compositional flow simulation in complex fractured media by 3D unstructured gridding with horizontal and deviated wells. *SPE Reservoir Evaluation & Engineering*, 23(2):498–517.

Zidane, A. and Firoozabadi, A. (2020). Higher-order simulation of two-phase compositional flow in 3D with non-planar fractures. *Journal of Computational Physics*, 402:108896.