Volker Siepmann

Process modelling on a canonical basis

Doctoral thesis
for the degree of doktor ingeniør

Trondheim, July 2006

Norwegian University of
Science and Technology
Faculty of Natural Sciences and Technology
Department of Chemical Engineering

NTNU
Innovation and Creativity

NTNU
Norwegian University of Science and Technology

Doctoral thesis
for the degree of doktor ingeniør

Faculty of Natural Sciences and Technology
Department of Chemical Engineering

©Volker Siepmann

ISBN 82-471-7892-3 (printed ver.)
ISBN 82-471-7891-5 (electronic ver.)
ISSN 1503-8181

Doctoral Theses at NTNU, 2006:71

Printed by Tapir Uttrykk

Process modelling on a canonical basis

by

V S

Thesis submitted for partial fulfilment
of the requirements for the degree of

Doktor Ingeniør

July 2006

NTNU
Innovation and Creativity

Faculty of Natural Sciences and Technology
Department of Chemical Engineering
N-7491 Trondheim

Dr. ing. thesis 2006:71
ISBN 82-471-7892-3 (printed version)
ISBN 82-471-7891-5 (electronic version)

Preface

When I graduated at the University of Dortmund, Germany, I had favoured fluid dynamics and process control during my course of study. I clearly preferred these subjects, as their problems are well-structured and formulated logically in a clean mathematical manner.

During a summer job in 1998 at Norsk Hydro ASA in Porsgrunn, Norway, I came in contact with Tore Haug-Warberg, who later became my main doctoral advisor. Impressed by his well structured view towards equilibrium thermodynamics, I began to understand the background of what was so difficult to assimilate during my undergraduate study.

I liked the challenge of utilising this structure in practical problems, and to combine this effort with my affection for software development. Being a student on a Norwegian university, I spent my first semester as a 'foreign' student at *Lehrstuhl für Proßesstechnik*, RWTH Aachen, Germany. There, I was introduced to the European CAPE-OPEN project. With insight into the software design of process modelling tools, it was a good starting point for my investigations.

The initial objective was to develop methods and tools for the energy and exergy efficiency analysis of industrial processes. Although this direction disclosed a different aspect of process modelling to me, the focus on second law thermodynamics moved increasingly into the background, while process modelling on a canonical basis became the main subject of my research. Having developed a fully functional process simulator called *Yasim*, a consistent and natural exergy analysis method falls naturally into place, easy to integrate into this environment.

Acknowledgement

I would like to thank my supervisors Associate Prof. Dr. Ing. Tore Haug-Warberg (NTNU) and Dr. Ing. Knut Wiig Mathisen (Yara International). Whenever needed, I received technical and administrative support from both of them. They furthermore involved me into a number of attractive industrial projects, which often served as an opportunity to apply my theoretical insight from this work.

Furthermore, I would like to thank my colleague Dipl. Ing. Geir Arne Evjen, not only for numerous technical discussions, but also for outstanding cooperation, which made it possible to offer some functionality of methods derived in this work through a user-friendly graphical user interface in *Microsoft Visio*.

Financial support for this project was provided through Norsk Hydro ASA, former section for *Modelling and Process Safety* at the Corporate Research Centre in Porsgrunn. I spent the first years of the project at Telemark University College, Faculty of Technology in Porsgrunn, before moving my office location to the research centre.

It is my pleasure to thank Prof. Dr. Dipl. Ing. Wolfgang Marquardt and his group of phd students, in particular Dr. Dipl. Ing. Lars v. Wedel for their hospitality and cooperation during my stay at LPT, RWTH Aachen, Germany.

Abstract

Based on an equation oriented solving strategy, this thesis investigates a new approach to process modelling. Homogeneous thermodynamic state functions represent consistent mathematical models of thermodynamic properties. Such state functions of solely extensive canonical state variables are the basis of this work, as they are natural objective functions in optimisation nodes to calculate thermodynamic equilibrium regarding phase-interaction and chemical reactions. Analytical state function derivatives are utilised within the solution process as well as interpreted as physical properties.

By this approach, only a limited range of imaginable process constraints are considered, namely linear balance equations of state variables. A second-order update of source contributions to these balance equations is obtained by an additional constitutive equation system. These equations are general dependent on state variables and first-order sensitivities, and cover therefore practically all potential process constraints. Symbolic computation technology efficiently provides sparsity and derivative information of active equations to avoid performance problems regarding robustness and computational effort.

A benefit of detaching the constitutive equation system is that the structure of the main equation system remains unaffected by these constraints, and *a priori* information allows to implement an efficient solving strategy and a concise error diagnosis. A tailor-made linear algebra library handles the sparse recursive block structures efficiently.

The optimisation principle for single modules of thermodynamic equilibrium is extended to host entire process models. State variables of different modules interact through balance equations, representing material flows from one module to the other. To account for reusability and encapsulation of process module details, modular process modelling is supported by a recursive module structure.

The second-order solving algorithm makes it possible to retrieve symbolically obtained derivatives of arbitrary process properties with respect to process parameters efficiently as a post calculation. The approach is therefore perfectly suitable to perform advanced process systems engineering tasks, such as sensitivity analysis, process optimisation, and data reconciliation.

The concept of canonical modelling yields a natural definition of a general exergy state function for second law analysis. By partitioning of exergy into latent, mechanical, and chemical contributions, irreversible effects can be identified specifically, even for black-box models.

The calculation core of a new process simulator called *Yasim* is developed and implemented. The software design follows the concepts described in the theoretical part of this thesis. Numerous exemplary process models are presented to address various subtopics of canonical modelling.

Contents

Nomenclature

Syntax

symbol	description (page of first occurence)	unit of measurement

Latin symbols

A	H energy (22)	J
$\underset{\approx}{A}$	Constraint matrix in a reacting system (29)	-
$\underset{\approx}{B}$	Coefficient matrix of a canonical equation system (26)	
$\underset{\approx}{C}$	Coupling matrix between two FMs (36)	
$\underset{\approx}{D}$	Diagonal matrix of child FM coefficient matrices $\underset{\approx}{B}_i$ (63)	
$\underset{\approx}{E}$	Selection matrix (permutation matrix with removed columns) (32)	-
E	Exergy (82)	W
F	Area (surface or cross-section) (12)	m^2
G	G free energy (22)	J
H	Enthalpy (12)	J
$\underset{\approx}{H}$	H matrix (26)	
$\underset{\approx}{I}$	Identity matrix $\underset{\approx}{I} = \sum_i \underset{\sim}{e}_i \underset{\sim}{e}_i$ (26)	-
$\underset{\approx}{J}$	J matrix (20)	
$\underset{\approx}{L}$	Lower triangular decomposition matrix (27)	
$\underset{\approx}{N}$	Stoichiometric matrix, null space of formula matrix $\underset{\approx}{R}$ (44)	-
N	Total molar quantity: $N = \sum_i n_i$ (54)	mol
O, o	Landau symbols (145)	
$\bar{O}, \bar{o}$	Inverse Landau symbols (145)	
P	Thermodynamic state function (21)	
$\underset{\approx}{P}$	Permutation matrix (27)	-
Q	Heat flow (12)	W
$\underset{\approx}{R}$	Formula matrix as part of the constraint matrix of reactor FMs (41)	-
R	Universal gas constant (71)	J/molK
S	Entropy (22)	J/K
$\underset{\approx}{S}$	Scaling matrix (112)	
T	Temperature (12)	K
U	Internal energy (22)	J

$\underset{\approx}{U}$	Upper triangular decomposition matrix (27)	
V	Volume (12)	m^3
W	Work duty (12)	W
$\underset{\approx}{W}$	Weight matrix (14)	
$\mathbb{W}$	State function $\mathbb{W}(H, V/T, \underset{\sim}{n}) = S + pV/T$ (23)	J/K
$\underset{\sim}{a}$	Constraint vector (48)	
$\underset{\sim}{b}$	Righthand side of a canonical equation system (27)	
c	Thermodynamic parameter (11)	
c_p, c_V	Molar heat capacity at constant pressure (c_p) or volume (c_V) (35)	J/molK
$\underset{\sim}{e}_i$	Unity vector in direction i (22)	-
f	Safety factor (57)	-
g	Gradient of state function with respect to a canonical variable (26)	
$\underset{\sim}{h}(\underset{\sim}{x}, \underset{\sim}{\lambda})$	Set of constitutive equations (31)	
$\Delta_f h^{\mathrm{ref}}$	Molar reference state enthalpy of formation (71)	J/mol
k	Heat transfer coefficient (54)	W/m^2K
$\underset{\sim}{l}$	Gradient of objective function Λ (26)	
m	Mass (12)	kg
n	Molar quantity of a chemical species (12)	mol
p	Pressure (12)	Pa
q	Number of iterations (20)	-
r	Residual equation of a process model (11)	
s^{ref}	Molar reference state entropy (71)	J/molK
t	Time (135)	s
u	Process model parameter (11)	
v	Velocity (53)	m/s
x	Canonical state variable (11)	
y	Calculated property (11)	
z	Valve opening (12)	-

Greek symbols

Ω	Number of equilibrium phases (28)	-
$\underset{\sim}{\alpha}$	Modification of righthand side to obey constitutive equations (32)	
β	Vapour fraction (59)	-
δ	Residual of constraint equation (26)	
ε_T	Thermal expansion coefficient (54)	1/K
ε_p	Compressibility (54)	1/Pa
η	Efficiency or efficiency-like value (51)	-
γ	Relaxation factor for step size restriction (57)	-
κ	Adiabatic exponent: $\kappa = c_p/c_V$ (52)	-
λ	L -multiplier in constrained optimisation (26)	
μ	Chemical potential (27)	J/mol
ν	Stoichiometric coefficient (41)	-
ψ	Arbitrary function or variable (22)	

ϱ	Density (12)	kg/m^3
ξ	General function argument (68)	
$\underset{\sim}{\zeta}$	$\underset{\sim}{\zeta} = (\underset{\sim}{x}, \underset{\sim}{\lambda})$ (70)	

Objects

C	A coupling between two flowsheet modules (17)
M	A flowsheet module (FM) (17)
R	Relaxation object (57)

Sets

$\mathcal{C}$	Domain of thermodynamic parameters (11)
$\vec{\mathcal{C}}$	A set of couplings (17)
$\mathcal{E}$	Set of extensive canonical variables (22)
$\bar{\mathcal{E}}$	Set of intensive canonical variables (22)
$\mathcal{M}$	Set of flowsheet modules (17)
$\mathcal{P}$	A path in the directed graph representing a process model (18)
$\mathcal{R}$	A circle in the directed graph representing a process model (17)
$\mathcal{U}$	Set of possible process model parameterisations (11)
$\mathcal{X}$	Domain of feasible states (11)

Accents, sub- and superscripts supplementing an arbitrary quantity Ψ

$\Delta\psi$	Discrete change of a quantity (12)
$\dot{\psi}$	Flow of an extensive quantity (11)
ψ_{eq}	Property at thermodynamic equilibrium (71)
ψ_{exp}	Experimental value (14)
$\psi^{[\mathrm{g}]}$	Hierarchy level of flowsheet modules (17)
$\hat{\psi}$	Altered or modified variable (22)
ψ_i, ψ_j	Index of elements in a set (17)
ψ_{in}	Symbol related to the input of a FM (40)
ψ_{initial}	Initial quantity (26)
$\psi^{(k)}$	A quantity calculated in iteration k (27)
$\psi^{(\infty)}$	A quantity calculated in the solution point (in theoretically ∞ iterations) (35)
$\psi_{(\mathrm{l})}$	A quantity regarding the liquid phase (26)
ψ_{main}	A symbol related to a main phase in a saturation node (49)
ψ_{max}	Quantity at maximised conditions (58)
ψ_{meas}	Measured value (13)
ψ_{open}	Quantity related to an open valve (53)
ψ_{opt}	Optimised quantity (78)
ψ_{out}	Symbol related to the output of a FM (53)
ψ_{poly}	Quantity related to the definition of polytropic efficiency (52)
ψ_{raw}	Object prior to further treatment (42)
ψ_{ref}	Reference state of a thermodynamic model (84)
ψ_{rev}	Reversible contribution (51)

ψ_{row}	Symbol related to rows of a matrix (27)
ψ_{col}	Symbol related to columns of a matrix (27)
ψ_{sonic}	A quantity related to sonic conditions, i.e. a fluid moving at sonic speed (53)
ψ_{spec}	Specified quantity, instance of a process parameter u (31)
ψ_{split}	Symbol related to flow splitter DOF to determine split behaviour (48)
ψ_{sub}	Subset (14)
ψ_{trial}	A symbol related to a trial phase in a saturation node (48)
$\psi_{(\mathrm{v})}$	A quantity regarding the vapour phase (26)
$\underset{\sim}{\psi}$	Vector of variables ψ_i, $\underset{\sim}{\psi} \in \mathbb{R}^d$ with $d = \dim \underset{\sim}{\psi}$ (18)

Transformations

$\mathfrak{L}_j[\psi(\underset{\sim}{x})]$	L transformation of ψ with respect to x_j (22)
$\mathfrak{M}_j[\psi(\underset{\sim}{x})]$	M transformation of ψ with respect to x_j (23)

Abbreviations

CAS	Computer Algebra System (69)
DOF	Degree of freedom (2)
FM	Flowsheet module (15)
NRTL	Non-Random-Two-Liquid – G excess model for polar liquid phases (94)
PID	Proportional-Integral-Derivative (controller) (138)
PSE	Process Systems Engineering (67)
RPN	Reverse polish notation (21)
SRK	S -R -K equation of state (127)
UML	Unified Modelling Language (16)
s.t.	Subject to (14)

Chapter 1

Introduction

1.1 Basics of process modelling

During the last four decades, computer aided process modelling has evolved into a broad, indispensable and ever extending discipline of process engineering. The range of applications has expanded into process design, control, optimisation and safety. Each of them is of significant importance to industry, and increasingly sophisticated models must be developed to be competitive in process plant operation. Today's process engineering software must provide the engineer with a wide range of functionality, but at the same time enable an efficient work flow. Berger and Perris (1979) have formulated the following criterion for the design of a process simulator:

The minimum total expenditure of manpower and computing resources to derive a satisfactory solution to the problem, within the timescale dictated by the project.

This criterion involves three main aspects to guide the development of process modelling tools, namely technology, scope and paradigm.

1. The technological aspect covers the user interface and data handling, but most importantly the way of solving the mathematical model of the process. Different solution strategies are discussed in Section 1.2.

2. The scope defines the range of applications that is handled or addressed. The solution of a problem must be within scope of the software tool used. The scopes of all existing tools are limited, and these limits must be accepted by both users and developers. The challenge is to cover a wide scope, but provide the functionality as efficiently as it would be possible within tailor-made tools. In many cases, flexibility is hard to combine with usability and computational efficiency.

3. The paradigm defines the structural mapping of the real or hypothetical process equipment towards a computer model. Early models were hard-coded in

existing programming languages, such as FORTRAN, and hence followed a procedural paradigm. Before graphical interfaces became available, input languages were invented to describe process models. For MASSBAL (Shewchuk, 1987), this language is mainly logic-based, i.e. the user defines a set of rules, which define the problem. A rule can be an equation, but also a topological specification, such as a material coupling. At the same time, the MASSBAL input language includes aspects of a module-based paradigm. Parallel to the evolution of software design paradigms, process modelling paradigms are further developed. Marquardt (1996) identifies the challenge of modelling non-standard process equipment and maintenance of models. His object oriented paradigm of modelling methodology yields a clean hierarchically defined topological structure and a breakdown of mathematical models into reusable building blocks.

Object oriented modelling tools must define an interface language, which defines the functionality and available information of user-defined objects. If the elements of this language are mainly equations and variables as in gPROMS (Oh and Pantelides, 1996), the tool offers a very flexible scope, and virtually any physical system can be described. With a more specialised interface language, including thermodynamic models and material ports as basic data types, process models can be established more effectively. A process model can be understood as a mathematical model of a chemical process. Terms are introduced more precisely in Section 2.3. Additional structural knowledge can then be utilised for efficient solving and informative error diagnosis.

1.2 Fundamental solving strategies

There are two fundamental strategies to solve process model equations (Biegler *et al.*, 1997): (i) sequential modular, and (ii) equation-based. In the sequential modular approach, each unit operation is solved sequentially, based on given input streams. Outer iterations are inevitable to handle process models with recycle streams. Most common equation-based solvers collect the linearised equations of each unit operation and the connecting streams. These equations are then solved simultaneously, and iterations are performed for non-linear process models.

As indicated in Table 1.1, both methods have their advantages and drawbacks. It needs to be noted that this table is a general comparison, and that individual software tools might overcome some of the drawbacks of the applied solution method. The term *coupled equation* is used to describe equations that cause state variables of a calculation unit to be influenced by changes (e.g. of specifications) downstream to this unit. The effect of such a coupled equation is similar to that of a recycle stream.

As early as 1979, Evans *et al.* recognised the potential of equation-based methods, but followed the sequential modular approach in their tool ASPEN (Advanced

Table 1.1: *General comparison of sequential modular and equation-based solving strategies.*

Sequential modular solver	Equation-based solver
+ The calculation path follows material streams. → An intuitive error analysis is possible. The failing calculation unit is often clearly identified. Tailor-made solution methods for individual calculation units allow for a detailed error diagnostics.	+ The solution method is robust with recycles and coupled equations.
+ The solution method is efficient with few recycles and coupled equations.	+ Second-order equation solvers converge quadratically close to the solution. → The approach is more suitable for dynamic simulation. → The approach is more suitable for all kinds of optimisation.
+ Initial values are only required for a small fraction of all state variables.	+ A linearly specified model is solved exactly if it contains recycle streams or coupled equations.
+ Tailor-made calculation methods for each unit operation can be applied.	
	– A global DOF (degree of freedom) analysis creates more problems to balance equations and variables.
	– A general equation solver is inefficient for large process models
	– The initialisation of every state variable is essential.
– The approach is inefficient for strongly coupled process models.	– Highly non-linear thermodynamic equations cause problems, if solved simultaneously with the process model equations. For instance, a sequential-modular approach uses specialised solution methods to calculate phase equilibrium.
– Process optimisation is dependent on derivatives that, using this approach, are not analytically available. The common use of numerical approximations reduces the usability of sequential-modular solvers for such tasks.	– An error analysis difficult to carry out, if the solving step is performed by a general equation solver that either fails or succeeds.
– A linearly specified model is not solved exactly, if it contains recycles or coupled equations.	

System for Process Engineering) for legacy reasons and a general lack of experience with equation-based solvers regarding industrial systems. Still today, the disadvantages of equation-based solvers inhibit their range of application. Aspen Plus® (Evans *et al.*, 1979) and Hysys® (Mahoney and Santollani, 1994), the most successful commercial process systems engineering tools, are based on the sequential modular approach.

1.3 Concept of canonical modelling

The concept of canonical variables is defined by the natural variable set of a thermodynamic state function. Primarily, this is temperature and mole-numbers, furthermore volume for H -models (or residual models), or pressure for G -models (or excess models). Transformations can be utilised to reach other sets of canonical variables, such as entropy, volume and mole numbers. The canonical modelling approach is based on thermodynamic models transformed to state functions with suitable sets of canonical variables. The entire process model can then be based on constrained optimisation programs for thermodynamic state functions.

The flexibility of this method applied to single-stage flash calculations was discovered by Dluzniewski and Adler (1972), but restricted to G coordinates, hence restricted to material balance at constant temperature and pressure. By use of L and M state function transformations (Callen, 1985), Brendsdal (1999) extended the set of possible constraints.

Balance equation sets describe the constraints for energy, volume, and material flow between and within the unit operations. By selecting transformations towards a set of solely extensive canonical variables, these constraints form a well defined structure, which can be exploited efficiently in a canonical flowsheet solver.

This way of solving process models has certain technical advantages compared to the traditional approaches of sequential modular and equation-based solution strategies:

- Though basically equation-based, this approach allows for *a priori* partitioning of equations according to process topology, thus allowing for more specific error diagnostics and in many cases a better performance of the solver.

- The thermodynamic state function represents a common framework for all thermodynamic models. This allows for a clean interface between a process model and underlying thermodynamic models. There is no problem to exchange the thermodynamic model used by a process model, or to reuse a thermodynamic model for different process models.

- Complex thermodynamic models do not affect the size or structure of the equation system. The state function and its derivatives are evaluated at given state variables, and the result serves as input to the equation system of the process model. The computational effort to evaluate the thermodynamic model is subject to its complexity, but in general small compared to the necessary effort to solve the process model.

While the most dominant available process modelling software today has already existed for many decades, the opportunity to build a prototype for a new tool from scratch raises further topics regarding scope and paradigm:

- How to account for the wide range of requirements to a modern process modelling tool, i.e. how to minimise the effort and maintenance to provide the functionality required by modern engineering problems, such as optimisation, data reconciliation and parameter fitting in a steady-state or dynamic context.
- How to achieve maximal reusability of the developed process models to avoid redundant modelling efforts.
- How to preserve the amount of knowledge for increased performance, but – even more importantly – for an engineer to pick up previously started work or a project of a colleague. In particular, equation-based models tend to be difficult to maintain, since equations and variables are defined in one large system without or with little human-readable meta information.

The canonical approach in its pure form yields a large equation system, and solving this with conventional methods would require excessive computational effort. Identification of various matrix types within the sparse block-structure of the coefficient matrix, as well as an advanced block-pivoting algorithm can clearly enhance performance towards a level that is comparable with available process modelling tools.

1.4 Thesis overview

This work explores the potential of canonical modelling to a wide range of process modelling applications. The approach is to combine the use of topological information as in sequential modular methods within an equation-based solving strategy. The objective is to combine the advantages of both of the standard methods, while eliminating their drawbacks. The main focus is placed on steady-state process simulation, but aspects of optimisation, data reconciliation, model parameterisation, and dynamic simulation and control are addressed as well.

Chapter 2 is an introduction to the field of process modelling. In particular, a number of terms are defined as a basis for subsequent chapters. Following a short overview over various process systems engineering disciplines, the concept of process models and the two most common solving strategies for such models are described. This work is strongly based on a uniform representation of thermodynamic models. A section about thermodynamic state functions and mathematical transformations on these state functions completes this chapter.

Chapter 3 explains the mathematical models for the smallest possible building blocks, which then are assembled into composite modules and entire process models. Various combinations of the canonical and well structured equation system with a second system of constitutive equations are discussed.

In Chapter 4, the scope is extended from steady-state process simulation to advanced process systems engineering disciplines. With the help of symbolically obtained derivatives of constitutive equations, the subjects of sensitivity analysis, process optimisation and data reconciliation are addressed. Exergy analysis is another discipline, which is easily embraced in terms of the canonical modelling approach.

The process simulation tool *Yasim* has been developed and implemented in this work. The main aspects of software design are described in Chapter 5. The subsequent chapter discusses performance issues, such as convergence properties, quality of symbolically obtained derivatives, and the condition number of coefficient matrices.

1.5 Contribution of this work

In many cases, a research project is based directly on the results of recent advances in the particular field. The basis of such work is somehow naturally limited in scope, and there is often a well-defined goal to achieve. However, that kind of foundation was not available as such in this case, even though Haug-Warberg (1988) and Brendsdal (1999) provided a solid basis from a thermodynamic viewpoint.

The subject of steady-state process modelling received no particular attention for the last 20 years, and no specific goal guided the direction of research in this work. The abstract goal however is to develop and investigate the potential of canonical modelling in various fields of process systems engineering.

Basis for this work is the previously known approach to perform calculations on phase equilibria and equilibrium reactions by optimisation on the basis of extensive canonical thermodynamic state variables. An algorithm is developed to extend this concept to handle arbitrary process constraint equations. The solution scheme is based on the N -R method, and second-order convergence is preserved in the overall algorithm. Two equation systems are used, namely a well-structured canonical equation system to perform the original optimisation, and an equation system consisting of constitutive equations, which defines the source contributions of selected constraint equations in the canonical system. Such modified constraint equations are from now on denoted as *released*.

The concept is then extended to be applied on entire process models. A process model is defined by a hierarchical structure of local optimisation nodes, which are linked by balance equations. A library of basic optimisation nodes is defined in the framework of canonical modelling. These nodes describe the most common operations in chemical engineering and build therefore a solid basis to establish a wide range of process models.

This approach provides clear advantages to the existing process modelling techniques known by the author:

- The structure of the coefficient matrix of the canonical equation system is a direct mapping of the process topology. Each diagonal block in the matrix is associated with one module in the process model, and each off-diagonal block represents one material stream.
- All diagonal blocks of the canonical coefficient matrix are invertible and, on the lowest hierarchical level, minimal in size. With the available structural information, a new tailor-made equation solver can be developed. Such a solver will potentially be more efficient than any other solver, which does not use this *a priori* structural input.
- The non-ambiguous association between constitutive equations and released constraint equations eliminates the common user problems regarding degree of freedom analysis. The number of active equations is always balanced to the number of independent variables, and the interconnection between a particular constitutive equation and a released constraint equation conserves valuable information to maintain larger process models.
- The association between constitutive equations and released constraint equations is observed to initiate a gain of thermodynamic understanding to the users of the prototype implementation (*Yasim*) of this concept. This educational aspect allows a novice user to work efficiently with the process modelling tool after a short period of familiarisation.

It is shown that the canonical process modelling approach is a solid basis for advanced process system engineering disciplines, such as sensitivity analysis, process optimisation, and data reconciliation. Reliable derivative information can easily be generated on the basis of symbolic algebra. A detailed exergy analysis can be performed and combined with the previously named tasks. A brief study of the feasibility to calculate on dynamic process models is carried out with positive results. The concept of process modelling on a canonical basis is easily extensible towards dynamic process simulation.

A software implementation of the concept is completed, resulting in the prototype of a new steady-state process modelling tool: *Yasim*. *Yasim* provides the functionality to nearly all concepts described in this work, and has been tested by conducting process simulation, optimisation, and data reconciliation of several medium-sized processes.

Chapter 2

Process systems engineering

2.1 Introduction

2.1.1 Hierarchical modelling approach

In order to establish a detailed process model of an entire plant, it is a strong requirement to structure this model into smaller units. Process models based on an entirely flat approach are not maintainable, and reuse of model parts in other process models is virtually impossible. Traditional tools, like e.g. Aspen Plus® (Evans *et al.*, 1979) define one layer of pre-defined process units, which then can be instantiated and supplemented by process topology information into a process model.

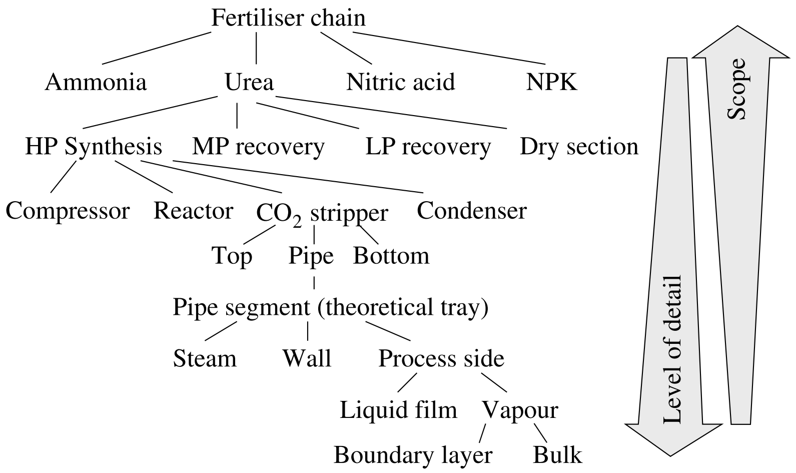


Figure 2.1: *The fertiliser process chain represented in the context of hierarchical process modelling.*

A more flexible approach is described by Marquardt (1996). A process flowsheet model can be decomposed into modules and interconnections. As shown in Figure 2.1, a module can be a part of a unit operation (e.g. column tray, heat exchanger shell side or a discrete volume in a plug-flow reactor), a unit operation itself, or a collection of interconnected unit operations (e.g. a plant section or an entire plant), hence a sub process flowsheet. An interconnection can be a flow of material, or any other physical interaction such as heat exchange, or a pure mathematical dependency such as product quality specifications.

A process model can be defined as the stand-alone flowsheet module on the top-level. Any flowsheet module shown in Figure 2.1 can assume this role. The CO_2 stripper interacts with other flowsheet modules within the high pressure synthesis part of the urea process. As a stand-alone module with fixed input flows and given environmental conditions, it represents a process model in itself, and can be used to investigate the stripping process in detail. The bulk phase of the vapour is a primitive, but valid process model. Its purpose can be to determine the properties of the stripping gas at a given state.

2.1.2 Paradigms in process modelling

A flowsheet solver is the executive instance to generate results of a given problem. On this level, the process model, as part of the problem definition, is represented by sets of equations. But a process model is established at a more abstract level by the engineer. For example, a flowsheet module is defined as a reacting two-phase equilibrium between given sets of chemical species in both phases. Predefined thermodynamic models are applied for the calculation of properties in each phase. The process modelling tool must translate these specifications into a suitable mathematical model to be taken care of by the solver. Figure 2.2 shows a possible categorisation

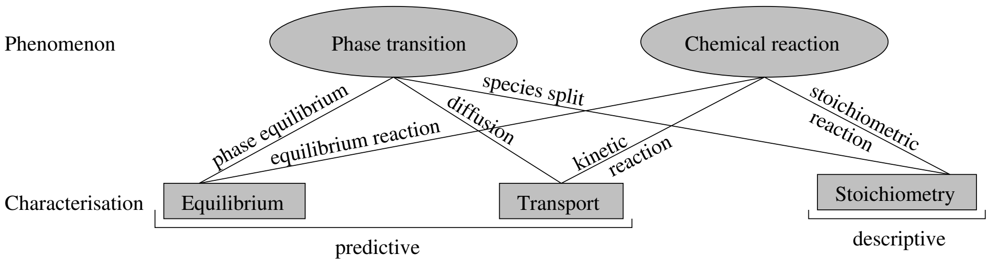


Figure 2.2: *Building blocks as a basis for atomic flowsheet modules.*

of suitable building blocks, which describe a flowsheet module. The physical phenomena *phase transition* and *chemical reaction* can be characterised by three main approaches, namely *equilibrium*, *transport* and *stoichiometry*. In general, stoichiometric characterisations tend to be of descriptive nature, while equilibrium and transport based characterisations are predictive, with a wide field of research dedicated to each of them.

Numerous methods to calculate phase equilibrium properties have been developed with emphasis on isothermal and isobaric conditions, reviewed recently by

(Wakeham and Stateva, 2004). The main approaches are the direct substitution method by Boston and Britt (1978), improved by Michelsen (1982), and minimisation of G energy, first utilised by White *et al.* (1958). Michelsen (1994) formulates a minimisation approach, which also considers chemical reactions.

Constraints other than isothermal and isobaric are addressed by Michelsen in 1987 and 1999. Methods to exploit the mathematical structure of thermodynamic state functions are investigated by Haug-Warberg (1988) and Brendsdal (1999). In this work, the basis for process modelling is the utilisation of L and M transformations (Callen, 1985) in order to obtain a suitable set of canonical variables.

2.2 Process systems engineering disciplines

The scope of process systems engineering disciplines increases proportionally to the available calculation capacity of modern computers. This section gives a definition of the main branches. Sensitivity analysis, data reconciliation, fit of thermodynamic parameters, and process optimisation are disciplines, which built on process simulation. Process simulation is the task of solving a mathematical model of a process.

A vector of state variables $\underset{\sim}{x}$ is an unambiguous description of the state. In general, two classes of state variables are distinguished: Accumulated states $\underset{\sim}{x}$ (e.g. as the content of a tank), and flows $\underset{\sim}{\dot{x}}$ (e.g. the water flow rate through a heat exchanger). More specifically, only extensive state variables are subdivided into flows and accumulated states. A similar grouping of intensive variables, like pressure, temperature, or concentration, is not preferable.

The objective in this work is to examine the principles and potentials of canonical modelling, and emphasis is put on steady-state problems, for which no accumulated states are considered.

Let $\mathcal{X}$ be the domain of feasible states $\underset{\sim}{\dot{x}} \in \mathcal{X}$ of a steady-state process model. The state vector $\underset{\sim}{\dot{x}}$ represents a unique description of the state, for instance in terms of molar flows, enthalpies, and pressures. $\mathcal{U}$ is the set of possible model parameters $\underset{\sim}{u} \in \mathcal{U}$, typically a specified valve position, compressor heat duty, or a heat exchanger surface. C is the domain of thermodynamic parameters $\underset{\sim}{c} \in C$, as for example a critical temperature or heat of formation of a pure species, or binary interaction coefficients. Generally, the mathematical representation of a steady-state process model can be described as

$$\underset{\sim}{r}(\underset{\sim}{\dot{x}}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\sim}{0} \quad \text{and} \quad \underset{\sim}{y} = \underset{\sim}{y}(\underset{\sim}{\dot{x}}, \underset{\sim}{u}), \tag{2.1}$$

where $\underset{\sim}{y}$ represents process properties as a function of $\underset{\sim}{\dot{x}}$ and $\underset{\sim}{u}$. Examples are a phase split fraction in a thermal separator, a heat transfer value in a heat exchanger, or the calculated isentropic efficiency of a turbine. The following sections describe process system engineering disciplines in a steady-state context with small examples illustrated in Figure 2.3.

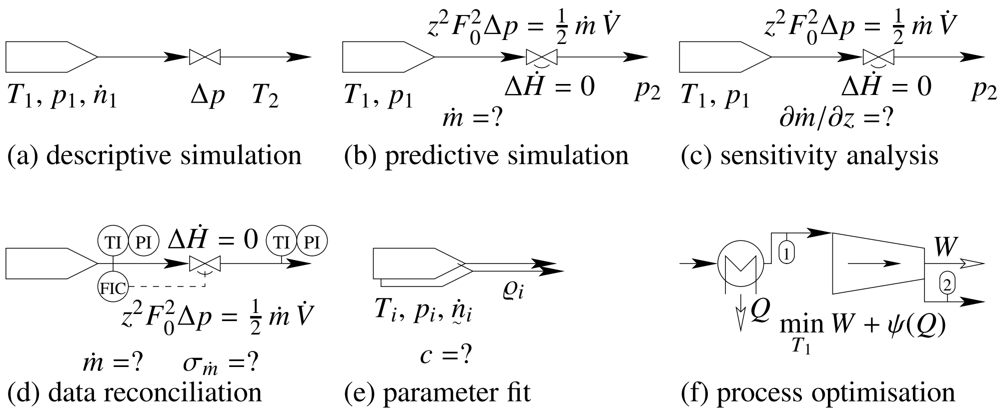

$T_1, p_1, \dot{n}_1$
Δp
T_2
(a) descriptive simulation
$z^2 F_0^2 \Delta p = \frac{1}{2} \dot{m} \dot{V}$
T_1, p_1
$\Delta \dot{H} = 0$
p_2
$\dot{m} = ?$
(b) predictive simulation
$z^2 F_0^2 \Delta p = \frac{1}{2} \dot{m} \dot{V}$
T_1, p_1
$\Delta \dot{H} = 0$
p_2
$\partial \dot{m} / \partial z = ?$
(c) sensitivity analysis
TI
PI
$\Delta \dot{H} = 0$
TI
PI
FIC
$z^2 F_0^2 \Delta p = \frac{1}{2} \dot{m} \dot{V}$
$\dot{m} = ?$ $\sigma_{\dot{m}} = ?$
(d) data reconciliation
ϱ_i
$T_i, p_i, \underset{\sim}{\dot{n}}_i$
$c = ?$
(e) parameter fit
1
W
2
Q
$\min_{T_1} W + \psi(Q)$
(f) process optimisation

Figure 2.3: *Concise overview of applications of various process modelling disciplines in a steady-state context.*

At this point, a number of symbols are introduced: Here, the intensive variables pressure p and temperature T are flow properties. The differences of enthalpy flows $\Delta \dot{H}$ and pressures Δp are derived flow properties. As they are in this case defined on input and output flows of a specific FM (the valve), these variables can as well be interpreted as flowsheet module properties. Flows are defined on the basis of molar quantities ($\dot{n}$), mass ($\dot{m}$), or volume ($\dot{V}$). F_0 is the cross-section of an open valve, and z the valve position, here defined as a linear characteristics to determine the cross-section at valve position z: $F = z F_0$. Because work W and heat Q are always defined as flows, the dotted notation is omitted in this case.

All the disciplines invoke the sub-task to obtain one or more solutions of the process simulation problem. Therefore, solving $\underset{\sim}{r}(\underset{\sim}{\dot{x}}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\sim}{0}$ efficiently is essential for all disciplines.

Though there is no sharp definition, it is possible to characterise process models as *descriptive* (Figure 2.3a) or *predictive* (Figure 2.3b). The purpose of a purely *descriptive* process model is to back-calculate an observed state with a minimum of process knowledge included into the model. Typically, one would formulate the mass balance equations, and directly specify pressures, temperatures, and enough streams to obtain a unique solution. As a rule of thumb, the calculated state is not affected by thermodynamic models. The process parameters $\underset{\sim}{u}$ do not reflect the degrees of freedom (DOFs) and process constraints of the real process. As shown in Figure 2.3a, the molar flow is specified, though in the real process, the amount is the consequence of the valve equation (cf. Figure 2.3b).

A *predictive* process model contains a maximum amount of process knowledge. Suitable thermodynamic models are applied to determine fluid properties, phase equilibria, and the extent of chemical reactions. Detailed performance characteristics of process equipment are included, such as compressor curves, valve equations, and heat transfer laws. The process parameters $\underset{\sim}{u}$ reflect the actual parameters of the real process.

In practice, a process model is never completely predictive, but always includes descriptive parts.

A *descriptive* process model can be the starting point for the refinement towards a *predictive* process model.

2.2.1 Process simulation

$\underset{\sim}{r}(\underset{\sim}{\dot{x}}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\sim}{0}$ is solved for $\underset{\sim}{\dot{x}}$ at constant thermodynamic parameters $\underset{\sim}{c}$ and constant process parameters $\underset{\sim}{u}$. With regard to the different types of process models, the attributes *descriptive* and *predictive* can be assigned to the simulation as well. *Descriptive* process models contain fewer or none non-linear equations, such that a *descriptive* process simulation is robust, and a solution can be obtained efficiently. *Predictive* process models contain a high number of non-linear equations, potentially even non-differentiable or discontinuous. Subsequently, there might exist multiple or no solutions of Equation (2.1), or it can be difficult to obtain the desired solution numerically. Results of a *descriptive* simulation are suitable starting values to simulate a *predictive* version of the process model.

2.2.2 Sensitivity analysis

Equation (2.1) can formally be written as a function $\underset{\sim}{y} = \underset{\sim}{y}(\underset{\sim}{u}, \underset{\sim}{c})$, i.e. each vector of process parameters and thermodynamic parameters is assigned a vector of calculated properties. Sensitivity analysis describes the process of discussing the effect of process parameters $\underset{\sim}{u}$ on the process properties $\underset{\sim}{y}$, in particular the derivative $\partial \underset{\sim}{y}/\partial \underset{\sim}{u}$ at constant c. The effect of the valve opening z to the mass flow $\dot{m}$ is the question of interest in Figure 2.3c.

Sensitivity analysis is an excellent tool to align the results of a predictive process model qualitatively with the results expected by the engineer. The explanation of any discrepancy either improves the understanding of the process, or it reveals a weakness of the process model, if the predicted effect was not physical.

An alternative to focus on process parameters is to investigate the effect of thermodynamic parameters $\underset{\sim}{c}$ to process properties $\underset{\sim}{y}$ at constant $\underset{\sim}{u}$. The limitation of accuracy of process simulation results due to uncertainty of thermodynamic parameters can be revealed through such a study.

2.2.3 Data reconciliation

The purpose of data reconciliation is to minimise a defined norm of deviation between redundant measurements $\underset{\sim}{y}_{\mathsf{meas}}$ and calculated properties $\underset{\sim}{y}$. One approach is to remove some the constraints represented by Equation (2.1), such that some state variables represent the independent variables in the minimisation problem.

A more concise approach is to include the entire process model represented by Equation (2.1), but select a subset of process parameters $\underset{\sim}{u} \in \mathcal{U}_{\mathsf{sub}} \subseteq \mathcal{U}$ as independent variables, hence solve the program

$$\min_{\underset{\sim}{u}} \Lambda(\underset{\sim}{y}, \underset{\sim}{y}_{\mathsf{meas}}) \quad \text{s.t.} \quad \underset{\sim}{r}(\underset{\sim}{x}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\sim}{0}\,. \tag{2.2}$$

Here, Λ is a general objective function, approaching its global minimum at $\underset{\sim}{y} = \underset{\sim}{y}_{\mathsf{meas}}$. The most common definition of Λ yields the least squares method:

$$\Lambda(\underset{\sim}{y}, \underset{\sim}{y}_{\mathsf{meas}}) = (\underset{\sim}{y} - \underset{\sim}{y}_{\mathsf{meas}})\, \underset{\approx}{W}\, (\underset{\sim}{y} - \underset{\sim}{y}_{\mathsf{meas}})\,. \tag{2.3}$$

The diagonal matrix $\underset{\approx}{W}$ contains weight factors to compensate for different scaling of elements of $\underset{\sim}{y}$, and to give room to incorporate the expected standard deviations of individual measurements.

The advantages and drawbacks of various objective functions are described by Özyurt and Pike (2004). A main aspect here is *gross error detection*, the process of filtering out faulty measurement values of non-statistical distribution, such as defect measuring equipment or interrupted signals.

The example of Figure 2.3d provides like the base case (b) 5 DOFs, of which only 2 (namely $\Delta H = 0$ and the pressure-flow relation) are to be fulfilled exactly. The deviation of 5 measurements plus valve position z to calculated process properties is minimised on the three remaining DOFs. The independent process parameters $\underset{\sim}{u}$ in equation (2.2) can be selected e.g. as $\dot{m}$, T_1, and p_1.

2.2.4 Fit of thermodynamic parameters

A common task in the field of thermodynamic modelling is to determine the set of thermodynamic parameters $\underset{\sim}{c}$ to obtain an optimal agreement between experimental data y_{exp} and calculated properties $\underset{\sim}{y}$ of a process model. One part of the experimental data is used as process parameters $\underset{\sim}{u}_{\mathsf{exp}}$, the other to be compared with calculated properties $\underset{\sim}{y}_{\mathsf{exp}}$. The problem can be formulated as

$$\min_{\underset{\sim}{c}} \Lambda(\underset{\sim}{y}, \underset{\sim}{y}_{\mathsf{exp}}) \quad \text{subject to (s.t.)} \quad \underset{\sim}{r}(\underset{\sim}{x}, \underset{\sim}{u}_{\mathsf{exp}}, \underset{\sim}{c}) = \underset{\sim}{0} \quad \text{with} \quad \underset{\sim}{c} \in C_{\mathsf{sub}} \subseteq C\,. \tag{2.4}$$

As for data reconciliation, the most common objective function $\Lambda(\underset{\sim}{y}, \underset{\sim}{y}_{\mathsf{exp}})$ is the geometric sum as given in Equation (2.3).

Typically, a large number of experimental data sets are utilised, and each set adds a contribution to the overall objective function. The process model itself is kept simple, as e.g. shown in Figure 2.3e with a single material flow from a reservoir. For each data set, the density $\underset{\sim}{\varrho}$ is measured for a given T, p, and $\underset{\sim}{n}$. Thermodynamic parameters related to the prediction of molar volumes might represent the independent variables to minimise the deviation of measured and calculated density.

2.2.5 Process optimisation

Given a predictive process model, the task of finding an optimal set of process parameters $\underset{\sim}{u}$ by minimising an objective function $\Lambda(\underset{\sim}{y})$, here solely as a function of calculated properties $\underset{\sim}{y}$, is called process optimisation:

$$\min_{\underset{\sim}{u}} \Lambda(\underset{\sim}{y}) \quad \text{s.t.} \quad \underset{\sim}{r}(\dot{\underset{\sim}{x}}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\sim}{0} \quad \text{with} \quad \underset{\sim}{u} \in \mathcal{U}_{\mathsf{sub}} \subseteq \mathcal{U}\,. \tag{2.5}$$

In Figure 2.3f, the temperature of stream 1, T_1, represents the independent variable in the optimisation of the total energy required to achieve a specified outlet pressure p_2. The lower T_1, the more cooling effort is necessary to reach this temperature, but the less energy is required to compress the gas to p_2.

In practice, the result of a process optimisation is often influenced or even determined by additional inequality constraints $\underset{\sim}{\psi}(\underset{\sim}{u}, \underset{\sim}{y}) \geq \underset{\sim}{0}$. In the example above, T_1 might have a lower constraint to avoid icing problems. In other cases, the material properties of process equipment pose upper constraints in temperature and pressure.

Inequality constraints represent a major challenge in process optimisation, and the development of general and tailor-made methods to solve specific problems represent a major field of research today. An introduction to this field is given by Nocedal and Wright (1999). With focus on the subject of canonical modelling, however, the scope of this work regarding process optimisation is limited to the discussion of Equation (2.5).

2.3 Concept of process models

The structure of mathematical models in process systems engineering can be defined in many ways with respect to various aspects. So long in this work, the concept of a process model has been used on a rather abstract level (cf. Equation (2.1)). The following terms and collaborations give a refined definition of a process model within the scope of this work.

Terms and definitions 2.1

Flowsheet module (FM) A self-contained mathematical model of a process or a part of a process. Self-contained means in this context that given all incoming material flows, there is a configuration and parameterisation of the model, which is sufficient to calculate the outgoing material flows.

Composite flowsheet module A FM, which can be further decomposed into a set of child FMs. The CO_2-stripper as shown in Figure 2.1 is an example, as it can be decomposed into the pipes, the top, and the bottom, each represented by another FM.

Atomic flowsheet module Any FM, which is not a *composite* FM. Assuming the bottom of the CO_2-stripper to be represented by an ordinary two-phase flash, this is an example for an atomic flowsheet module.

Input port The interface of a FM representing a distinguishable incoming material flow. Examples of different input ports of a FM are feed flows to a column on different trays. Multiple flows into one common control volume (e.g. a tank) are regarded as entering through one single input port.

Output port The interface of a FM representing an outgoing material flow.

Coupling A material flow between two FMs. The start-point is the *output port* of the upstream FM, and the end-point is the *input port* of the downstream FM. From a composite FM point of view, couplings represent the topology of the described process. Next to child FMs, couplings are therefore a part of a composite FM.

Process model A *composite* FM, which is no child of another *composite* FM in the current context. The FM called *HP Synthesis* in Figure 2.1 is a process model, if the high pressure synthesis part of the urea production is investigated as an isolated model. Any process model can be degraded to a composite FM, if it is used within a wider context (in the given example the complete urea production process).

Figure 2.4 gives an overview over the concepts introduced at this point. A short introduction to UML (Unified Modelling Language) according to OMG (2003) is given in Appendix F.2.

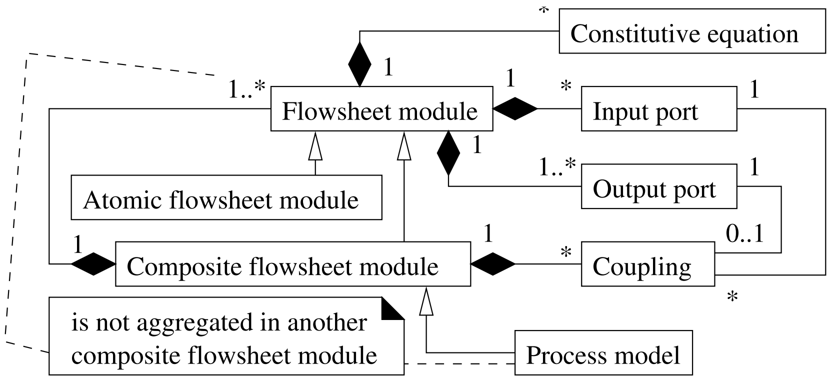


Figure 2.4: *UML static structure diagram of the general flowsheeting concept.*

2.3.1 Process topology

A process model consist of FMs and couplings, and can be represented by a directed graph[1]. Let $\mathcal{M}$ be a set of FMs representing the vertices of the graph, and $\vec{C}$ the set of couplings representing the edges. The edge C_{ij} is part of the graph, if there is a material flow from $M_i \in \mathcal{M}$ to $M_j \in \mathcal{M}$.

In the context of the hierarchical modelling approach, the entire graph represents not necessarily the process model, but possibly a FM in the parent context, hence a

[1]For an introduction in graph theory see Appendix F.3 and the book by Trudeau (1993)

single vertex in a super graph. On the hierarchy level $[g]$, This vertex is then defined as

$$M^{[g]} = (\mathcal{M}^{[g]}, \vec{\mathcal{C}}^{[g]}). \tag{2.6}$$

Here, $\mathcal{M}^{[g]}$ is the set of child FMs $M_i^{[h<g]}$, and $\vec{\mathcal{C}}^{[g]}$ the set of couplings $\vec{C}_{ij}^{[g]}$.

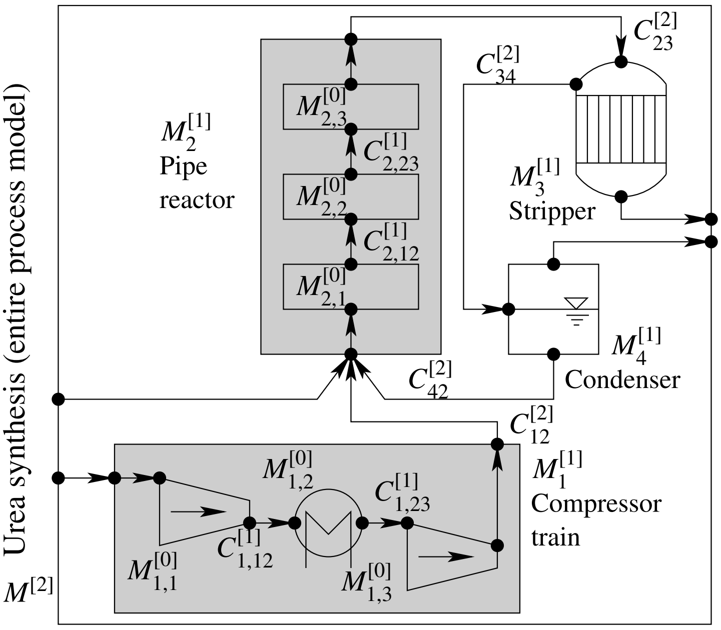


Figure 2.5: *Hierarchical topology graph of a simplified urea synthesis process.*

Figure 2.5 shows an example of such a topology graph. In this case:

$$\begin{aligned} \mathcal{M}^{[2]} &= \left\{M_1^{[1]}, M_2^{[1]}, M_3^{[1]}, M_4^{[1]}\right\}, \\ \mathcal{M}_1^{[1]} &= \left\{M_{1,1}^{[0]}, M_{1,2}^{[0]}, M_{1,3}^{[0]}\right\}, \quad \text{and} \quad \mathcal{M}_2^{[1]} = \left\{M_{2,1}^{[0]} M_{2,2}^{[0]}, M_{2,3}^{[0]}\right\}. \end{aligned} \tag{2.7}$$

Furthermore

$$\begin{aligned} \vec{\mathcal{C}}^{[2]} &= \left\{C_{12}^{[2]}, C_{23}^{[2]}, C_{34}^{[2]}, C_{42}^{[2]}\right\}, \\ \vec{\mathcal{C}}_1^{[1]} &= \left\{C_{1,12}^{[1]}, C_{1,23}^{[1]}\right\}, \quad \text{and} \quad \vec{\mathcal{C}}_2^{[1]} = \left\{C_{2,12}^{[1]}, C_{2,23}^{[1]}\right\}. \end{aligned} \tag{2.8}$$

Two important phenomena can be described on the basis of this type of graph:

Terms and definitions 2.2

Recycle Any circle[2] $\mathcal{R} \subseteq M^{[g]}$. Physically, a *recycle* **allows** material to flow in a circle.

[2]In terms of graph theory. See Appendix F.3

Circulation A *recycle* $\mathcal{R} \in M^{[g]}$, such that no path $\mathcal{P} = \{M_i, \ldots, M_j\}$ exists with $M_i \notin \mathcal{R}$, $M_j \notin \mathcal{R}$, but $\mathcal{P} \cap \mathcal{R} \neq \emptyset$. Physically, a *circulation* **forces** material to flow in a circle.

The effects of recycles on the complexity of process models is a crucial decision factor when selecting the solving strategy. Basically, a recycle prevents the system from being partitioned, yielding bigger sub-systems to solve simultaneously.

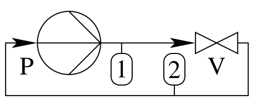


Figure 2.6: *Common case of a circulation.*

Little attention, in particular related to steady-state process models, has been paid to the numerical treatment of a circulation, which can be desired e.g. in cooling systems, but as well occur as part of a design or modelling fault, e.g. if material is locked in a circle. In both cases, the engineer and the program ought to identify the phenomenon. Engineers often prefer dynamic process models in this case. A hold-up volume combined with a bleed stream avoids the linear dependency of the balance equations. However, if the process dynamics are not of major interest, the effort to establish and maintain a dynamic process model is hardly justified.

The simple case of interconnected valve and pump shown in Figure 2.6 represents a typical case of a circulation. Both the pump and the valve provide the same balance equations, namely $\underset{\sim}{\dot{n}}_1 = \underset{\sim}{\dot{n}}_2$ and $\underset{\sim}{\dot{n}}_2 = \underset{\sim}{\dot{n}}_1$, which are obviously linear dependent. Furthermore, there are no DOFs left to define the actual flow conditions, such as composition. The approach chosen for the canonical solver is described in Section 3.9.2.

2.4 Steady-state process simulation solvers

To solve a process model efficiently, one has to exploit the structural information of the equation system. There are two distinct approaches to do this: (i) partitioning of the system on the basis of topological information, and sequentially to solve each partition (Biegler *et al.*, 1997), or (ii) application of methods for solving sparse matrices on the linear algebra level (Stadtherr and Wood, 1984; Zitney and Stadtherr, 1988). These approaches correspond to the sequential-modular and equation-based solution strategies respectively.

There are numerous approaches to enhance robustness and performance of the solution process, some of them on a higher level, such that they can be applied to both strategies. As an example, material balance equations can be relaxed during the first iterations. This yields a pseudo-dynamic simulation, iterating along a physically meaningful path, which is more likely to stay within the domain of involved equations.

2.4.1 Sequential-modular approach

The sequential-modular approach is strongly based on the topology graph $M^{[k]}$ defined in Equation (2.6). Hernandez and Sargent (1979) describe the strategy of *partitioning* and *tearing*:

Terms and definitions 2.3

Partitioning The program to determine a sorted list of the k smallest possible disjoint subsets $\mathcal{M}_i \subseteq \mathcal{M}$, $i \in \{1, \ldots, k\}$, with $i_1 > i_2 \Rightarrow (\mathcal{M}_{i_1} \times \mathcal{M}_{i_2}) \cap \vec{C} = \emptyset$. The result of this operation is a set of sub-graphs $M_i^{[k]} = (\mathcal{M}_i, \vec{C}_i)$ with $\vec{C}_i = \vec{C} \cap (\mathcal{M}_i \times \mathcal{M}_i)$.

Tearing The program to determine a subset $\vec{C}_t \subset \vec{C}_i$ for each Partition i, such that the modified graph $M_i'^{[k]} = (\mathcal{M}_i, \vec{C}_i \setminus \vec{C}_t)$ is free of circles, and an objective $\Lambda(\vec{C}_t)$ is minimised. A common choice is $\Lambda(\vec{C}_t) := |\vec{C}_t|$. The couplings in $\vec{C}_t$ are commonly referred to as *tear streams*.

The partitioning step splits the tearing problem into smaller sub-problems, which practically removes the problem due to the complexity of the subsequent tearing algorithm, which is exponential in problem size. The result of these two algorithms is a recursive structuring as shown in Figure 2.7. After $M_{-1}^{[k]}$ is pre-calculated, the tear streams are estimated, and an iteration is conducted on the calculation of $M_0^{[k]}$, before $M_1^{[k]}$ can be treated in a post-calculation. $M_i^{[k+1]}$ might be the process model itself, or it is part of the same structure on level $k+1$ with $i \in \{-1, 0, 1\}$. The representation of the structure shown in Figure 2.7 allows for a straightforward complexity analysis.

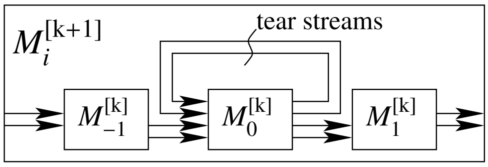


Figure 2.7: *A Process model structured for sequential solving.*

Let the complexity denoted by $\text{cmp}(M^{[0]})$ represent a metric for the effort to evaluate $M_i^{[0]}$ with $M_i^{[0]} \sim \text{cmp}(M^{[0]})\ \forall_i$ according to Appendix F.1.

The number of necessary iterations to converge a group of tear streams at level k is assumed to be constant and described by the symbol q. The recursion

$$\text{cmp}(M^{[k+1]}) \sim (2+q)\ \text{cmp}(M^{[k]}) \tag{2.9}$$

then yields the explicite equation

$$\text{cmp}(M^{[k]}) \sim (2+q)^k\ \text{cmp}(M^{[0]}) \tag{2.10}$$

for a process model of size $|M^{[k]}| \sim 3^k |M^{[0]}|$. Hence

$$\text{cmp}(M^{[k]}) \sim \left(\frac{2+q}{3}\right)^k \frac{\text{cmp}(M^{[0]})}{|M^{[0]}|}\ |M^{[k]}|. \tag{2.11}$$

Thus, a major advantage of the sequential modular approach is that the computational effort grows only linearly[3] in problem size $|M^{[k]}|$. But in particular for the common case $q > 1$, the computational effort is exponential to the number of nested recycles, which is the main drawback of this approach. External constitutive equations add further complexity to the model, but Perkins (1979) developed a method to solve those equations simultaneously with the tear stream equations, so that the effective overhead is minimised.

2.4.2 Equation-based approach

The pure equation-based approach is solely based on the mathematical solution of Equation system (2.1). This is a very efficient approach for linear systems described by

$$\underset{\sim}{r}(\underset{\sim}{\dot{x}}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\approx}{J}(\underset{\sim}{u}, \underset{\sim}{c})\, \underset{\sim}{\dot{x}} + \underset{\sim}{r}_0(\underset{\sim}{u}, \underset{\sim}{c}) = \underset{\sim}{0} \;\Rightarrow\; \underset{\sim}{\dot{x}} = -\underset{\approx}{J}^{-1}(\underset{\sim}{u}, \underset{\sim}{c})\, \underset{\sim}{r}_0(\underset{\sim}{u}, \underset{\sim}{c})\,. \qquad (2.12)$$

Solving the general equation system representing a process model M is of complexity $O(|M|^3)$ (Golub and Loan, 1996), even though recycles and external constitutive equations have no further impact. The essential need to utilise the sparse structure of $\underset{\approx}{J}$ was soon recognised. Markowitz (1957) presented a pivoting sequence to obtain a kind of LU-decomposition under the objective to preserve sparsity in this operation. Various improvements have been developed regarding different objectives:

- Integration of stability criteria into the objective to find an optimal pivoting sequence (Zlatev, 1980)
- Guarantee to not let the pivoting problem dominate the computational effort (Gilbert and Peierls, 1988)
- Prevention of time-consuming dynamic memory allocation (George and Ng, 1985)
- Utilisation particular hardware architectures, like e.g. vector processing (Zitney and Stadtherr, 1993)
- Handling of model hierarchy to presort variables and equations (Abbott *et al.*, 1997)

Today's process models are rarely linear. Non-linear equations result from even primitive thermodynamic models such as the ideal gas law, and constitutive equations such as even the simplest description of heat transfer. With Equation (2.12) no longer valid, a linearisation can be conducted as follows:

$$\underset{\sim}{r}(\underset{\sim}{\dot{x}}, \underset{\sim}{u}, \underset{\sim}{c}) = \underset{\approx}{J}(\underset{\sim}{x}_0, \underset{\sim}{u}, \underset{\sim}{c})\,(\underset{\sim}{\dot{x}} - \underset{\sim}{\dot{x}}_0) + \underset{\sim}{r}_0(\underset{\sim}{x}_0, \underset{\sim}{u}, \underset{\sim}{c}) + O((\underset{\sim}{\dot{x}} - \underset{\sim}{\dot{x}}_0)^2) = \underset{\sim}{0}\,. \qquad (2.13)$$

This results in three new challenges in process simulation arise:

[3]This is optimal, as no program can exhaustively process data in less time than proportional to its size.

Terms and definitions 2.4

Initialisation A scheme to provide a feasible state $x_0 \in \mathcal{X}$ as close to the solution as possible. Zitney and Stadtherr (1988) review schemes of different complexity.

Differentiation A method to obtain the non-zero elements of $\underset{\approx}{J}(\underset{\sim}{x}_0, \underset{\sim}{u}, \underset{\sim}{c})$. The different approaches such as hand-coded derivatives, finite-difference approximation of derivatives, symbolic differentiation, reverse polish notation (RPN) evaluation of derivatives and automatic differentiation are exemplified by Tolsma and Barton (1998). Appendix A.1 describes the design of a slim data type, utilised among other things for symbolic differentiation in this work. Another aspect discussed by Tolsma *et al.* (2002) and Li *et al.* (2004) is the smooth integration of external models into a simulation environment.

Solving Strategy An iteration scheme to improve the state vector towards the fulfilment of Equation (2.1). Zitney and Stadtherr (1988) point out three aspects, namely the correction step formulation (Chen and Stadtherr, 1981; Bogle and Perkins, 1988; Cofer and Stadtherr, 1996), sparse J evaluation, and hybrid J methods. Wilhelm and Swaney (1994) present a robust algorithm that prevents violation of domain boundaries and implements a back-tracking mechanism.

The following chapter addresses these items in the context of canonical process modelling, but rather than accepting the equation system (2.1) as is, the main focus is put on the formulation of the mathematical model. The objective is to reduce the required effort on the items above. For instance, a major part of the required derivatives can be provided by the implementation of the thermodynamic models. Furthermore, the equation system is generated with a large amount of *a priori* structural information. This reduces the problems in the solution process encountered by less structured model equations.

This work deliberately does not engage in the research of robust methods for the solution of general equation systems. The application of algorithms, such as that by Wilhelm and Swaney (1994), is likely to improve the robustness significantly, but, at this stage, it is important to use a straightforward solution method in order to judge the properties of the equation system.

2.5 Representation of thermodynamic models

2.5.1 Thermodynamic state functions

A thermodynamic state function P is a property of a system, which depends only on the current state of the system. The synonym *thermodynamic potential* for energy functions emphasises the attribute of path-independence and the necessity of a reference state for each independent argument. This work utilises homogeneous first-order state functions of the extensive parameters $x_{\mathcal{E}}$ and intensive parameters $x_{\bar{\mathcal{E}}}$ (Callen, 1985). As shown in Appendix C.1, the property of first-order homogeneity expressed by

$$P(\psi\, \underset{\sim}{x}_{\mathcal{E}}, \underset{\sim}{x}_{\bar{\mathcal{E}}}) = \psi\, P(\underset{\sim}{x}_{\mathcal{E}}, \underset{\sim}{x}_{\bar{\mathcal{E}}})\ ,\ \psi \in \mathbb{R} \tag{2.14}$$

yields E 's 1st and 2nd theorem:

$$P(\underset{\sim}{x}_{\mathcal{E}}, \underset{\sim}{x}_{\bar{\mathcal{E}}}) = \frac{\partial P}{\partial \underset{\sim}{x}_{\mathcal{E}}}\, \underset{\sim}{x}_{\mathcal{E}}\ , \text{ and subsequently } \quad \frac{\partial^2 P}{\partial \underset{\sim}{x}_{\mathcal{E}}\, \partial \underset{\sim}{x}_{\mathcal{E}}}\, \underset{\sim}{x}_{\mathcal{E}} = \underset{\sim}{0}. \tag{2.15}$$

The homogeneity of thermodynamic state functions has never been proven[4], but observed and postulated. The further work is therefore based on the following postulate (Callen, 1985; Brendsdal, 1999):

> *The internal energy U of a homogeneous phase is a first-order homogeneous function of its entropy S, volume V and mole numbers $\underset{\sim}{n}$.*

Note the unrelated concepts of homogeneity regarding mathematical functions as defined in Equation (2.14), and physical phases. Both concepts appear in this postulate.

2.5.2 State function transformations

The approach in canonical modelling in general is to utilise a state function with canonical variables natural to the constraints of the given system. For instance the G energy $G(T, p, \underset{\sim}{n})$ is suitable to describe configurations at specified T and p, while a dynamic tank constrained in U and V is described by the entropy function: $S(U, V, \underset{\sim}{n})$.

A typical thermodynamic model can be represented analytically by one, at most by two different state functions, namely H energy $A(T, V, \underset{\sim}{n})$ and G energy $G(T, p, \underset{\sim}{n})$. Other state functions are obtained applying two transformations, namely the L and the M transformations, which both are described by Callen (1985) and Brendsdal (1999). The L transformation of a state function P with respect to the variable x_j is defined as

$$\hat{P}(\underset{\sim}{\hat{x}}) = \mathfrak{L}_j[P(\underset{\sim}{x})] := P(\underset{\sim}{x}) - \left.\frac{\partial P(\underset{\sim}{x})}{\partial x_j}\right|_{x_{i\neq j}} x_j \quad \text{with} \quad \underset{\sim}{\hat{x}} = \sum_{i\neq j} x_i\, \underset{\sim}{e}_i + \frac{\partial P(\underset{\sim}{x})}{\partial x_j}\, \underset{\sim}{e}_j. \tag{2.16}$$

Hence, the L transformation exchanges information between the state and the gradient vector. The variables x_j and $\hat{x}_j$ are called *conjugated variables*.

In terms of group theory, the L transformation is a permutation of fourth order:

$$\mathfrak{L}_j[\mathfrak{L}_j[P(\underset{\sim}{x})]] = P(\underset{\sim}{\hat{x}}) \quad \text{with} \quad \hat{x}_j = -x_j \quad \text{and} \quad \mathfrak{L}_j[\mathfrak{L}_j[\mathfrak{L}_j[\mathfrak{L}_j[P(\underset{\sim}{x})]]]] = P(\underset{\sim}{x})\,. \tag{2.17}$$

[4] A disproof however would invalidate the first law of thermodynamics with all its conclusions, therefore solving all world's energy problems

It is therefore practical to define the inverse L -transformation

$$P(\underset{\sim}{x}) = \mathfrak{L}_j^{-1}[\hat{P}(\underset{\sim}{\hat{x}})] := \hat{P}(\underset{\sim}{\hat{x}}) - \left.\frac{\partial \hat{P}(\underset{\sim}{\hat{x}})}{\partial \hat{x}_j}\right|_{\hat{x}_{i\neq j}} \hat{x}_j \quad \text{with} \quad \underset{\sim}{x} = \sum_{i\neq j} \hat{x}_i \underset{\sim}{e}_i - \frac{\partial \hat{P}(\underset{\sim}{\hat{x}})}{\partial \hat{x}_j} \underset{\sim}{e}_j \tag{2.18}$$

instead of applying $P(\underset{\sim}{x}) = \mathfrak{L}_j[\mathfrak{L}_j[\mathfrak{L}_j[\hat{P}(\underset{\sim}{\hat{x}})]]]$.

The M transformation swaps an extensive canonical variable x_j and the state function. With the subspace $\mathcal{E} \subseteq \mathbb{R}^{\dim \underset{\sim}{x}}$ containing the extensive components of $\underset{\sim}{x}$, the definition is given as

$$x_j = \mathfrak{M}_j[P(\underset{\sim}{x})] := \left(P(\underset{\sim}{x}) - \sum_{i\in\mathcal{E}\setminus\{j\}} \frac{\partial P(\underset{\sim}{x})}{\partial x_i} x_i\right) \Big/ \frac{\partial P(\underset{\sim}{x})}{\partial x_j} \text{ with } \underset{\sim}{\hat{x}} = \sum_{i\in\mathcal{E}\setminus\{j\}} x_i \underset{\sim}{e}_i + P(\underset{\sim}{x}) \underset{\sim}{e}_j. \tag{2.19}$$

The M transformation is self-inverse, i.e. $\mathfrak{M}_j[\mathfrak{M}_j[P(\underset{\sim}{x})]] = P(\underset{\sim}{x})$ or $\mathfrak{M}_j^{-1} = \mathfrak{M}_j$.

All state functions used in this work can be obtained through (inverse) L and M transformations originating in $U(S, V, \underset{\sim}{n})$ as follows:

$$\begin{tikzcd} U(S,V,\underset{\sim}{n}) \arrow[r, "\mathfrak{L}_V^{-1}"] \arrow[d, "\mathfrak{M}_S"] & H(S,p,\underset{\sim}{n}) \arrow[r, "\mathfrak{L}_S"] \arrow[d, "\mathfrak{M}_S"] & G(T,p,\underset{\sim}{n}) \arrow[r, "\mathfrak{L}_p"] & A(T,V,\underset{\sim}{n}) \\ S(U,V,\underset{\sim}{n}) & S(H,p,\underset{\sim}{n}) \arrow[r, "\mathfrak{L}_p^{-1}"] & \mathbb{W}(H,V/T,\underset{\sim}{n}) & \end{tikzcd} \tag{2.20}$$

Postulated only for U, a simple proof for the preservation of homogeneity through these transformations is given in Appendix C.2. The property of homogeneity is therefore ensured for all state functions used in this work. Furthermore, Callen (1985) and Tester and Modell (1997) prove the preservation of the extremum principle for selected systems and state functions.

State functions with only extensive canonical variables ($U(S, V, \underset{\sim}{n})$, $S(U, V, \underset{\sim}{n})$ and $\mathbb{W}(H, V/T, \underset{\sim}{n})$) are of special interest in this work, as they allow one to map the topological structure of the model towards the structure of the equation system. Two basic examples are given at the beginning of the next chapter.

The equations of thermodynamic models are originally represented in H or G coordinates. The transformations are then used to obtain the desired state functions. This approach plays an essential role in the implementation of the canonical process modelling tool *Yasim*, which is described in detail in Chapter 5.

Chapter 3

Canonical process modelling

3.1 Introduction

The concept of canonical process modelling is to base the mathematical description of a process model on the natural state variables of thermodynamic state functions as introduced in Section 2.5.1. Each flowsheet module (FM) consists of building blocks. These blocks are formulated as local optimisation nodes, and sets of balance equations represent couplings between these blocks. The well defined structure of the resulting equation system directly reflects the process topology. This way, *a priori* structural knowledge can be exploited to achieve efficient equation solving. Furthermore, this equation system only contains stoichiometric constraints and thermodynamic information, but no coefficients that depend on geometric information or any other process parameter. From now on, this will be referred to as the *canonical equation system*.

In practice, some canonical balance equations are not actually used in the actual process model, as for instance the enthalpy balance over an isothermal storage tank. These balance equations are therefore released, in other words: a constitutive equation provides a source term to this balance equation. The modified balance equation then yields a solution, which fulfils the constitutive equation. This additional *constitutive equation system* is rather unstructured. All process parameters are part of this equation system. Additionally, the L multipliers of the optimisation nodes can be interpreted as canonical conjugated variables, and therefore be included.

The overall problem formulation is large in size and can easily exceed 1000 variables for a process model with 30 FM and 10 chemical species in each flow. But the well-structured canonical system can be solved efficiently, while the unstructured constitutive equation system is typically by a factor of 10 smaller, and therefore does not require significant calculation time. Several ways to formulate an algorithm to solve these two systems are discussed in Section 3.3.

A priori process topology knowledge is used to gain performance of the solution process. The framework also gives full control to associate degrees of freedom

(DOFs) and constitutive equations for maintainability of process models. This happens automatically by releasing canonical constraints in favour of constitutive equations in a one-to-one relationship. The occurrence of singular matrices can at any point be assigned to one particular FM for efficient error diagnosis (see Section 3.9).

The concept of canonical modelling applies to both material flows and accumulated states. The description of building blocks in the following section is based on accumulated states. Material flows are introduced in Section 3.4. However, this work focuses on steady-state process models, and the interaction between flows and accumulated states is therefore not considered. In a dynamic context however, most dynamic behaviour is contained within the modelling of this kind of interaction. A brief discussion of the extension to dynamic simulation is given in Appendix D.

3.2 Building blocks

3.2.1 Calculation of phase equilibria

Consider an insulated storage tank, constrained by U, V and $\underset{\sim}{n}$, the contained medium being split in a liquid $_{(\mathrm{l})}$ and a vapour $_{(\mathrm{v})}$ phase. With $\underset{\sim}{x} = (U, V, \underset{\sim}{n})$ as the canonical state vector of the entropy function, the program to solve is

$$\max_{\underset{\sim}{x}_{(\mathrm{l})},\underset{\sim}{x}_{(\mathrm{v})}} S = S_{(\mathrm{l})}(\underset{\sim}{x}_{(\mathrm{l})}) + S_{(\mathrm{v})}(\underset{\sim}{x}_{(\mathrm{v})}) \quad \text{s.t.} \quad \underset{\sim}{\delta} = \underset{\sim}{x}_{\text{initial}} - (\underset{\sim}{x}_{(\mathrm{l})} + \underset{\sim}{x}_{(\mathrm{v})}) = \underset{\sim}{0}\,. \tag{3.1}$$

The residual expression $\underset{\sim}{\delta}$ vanishes, if the total tank content $\underset{\sim}{x}_{(\mathrm{l})} + \underset{\sim}{x}_{(\mathrm{v})}$ is equal to the initial feed $\underset{\sim}{x}_{\text{initial}}$.

A standard solving method described by Jungnickel (1999) and Biegler *et al.* (1997) is to formulate a L function

$$\Lambda(\underset{\sim}{x}_{(\mathrm{l})}, \underset{\sim}{x}_{(\mathrm{v})}, \underset{\sim}{\lambda}) = S_{(\mathrm{l})} + S_{(\mathrm{v})} - \underset{\sim}{\lambda} \cdot \underset{\sim}{\delta} \tag{3.2}$$

and find the stationary point of Λ. With

$$\underset{\sim}{g}_{(\mathrm{i})} = \frac{\partial S_{(\mathrm{i})}}{\partial \underset{\sim}{x}_{(\mathrm{i})}} \quad \text{and} \quad \underset{\approx}{H}_{(\mathrm{i})} = \frac{\partial^2 S_{(\mathrm{i})}}{\partial \underset{\sim}{x}_{(\mathrm{i})} \partial \underset{\sim}{x}_{(\mathrm{i})}}\,, \; i \in \{\mathrm{l}, \mathrm{v}\}\,, \tag{3.3}$$

further symbols can be defined, namely the gradient $\underset{\sim}{l}$ and the H matrix $\underset{\approx}{B}$ of the L function:

$$\underset{\sim}{l} = \frac{\partial \Lambda}{\partial(\underset{\sim}{x}_{(\mathrm{l})}, \underset{\sim}{x}_{(\mathrm{v})}, \underset{\sim}{\lambda})} = \begin{pmatrix} \underset{\sim}{g}_{(\mathrm{l})} + \underset{\sim}{\lambda} \\ \underset{\sim}{g}_{(\mathrm{v})} + \underset{\sim}{\lambda} \\ -\underset{\sim}{\delta} \end{pmatrix} \quad \text{and} \quad \underset{\approx}{B} = \frac{\partial^2 \Lambda}{\partial(\underset{\sim}{x}_{(\mathrm{l})}, \underset{\sim}{x}_{(\mathrm{v})}, \underset{\sim}{\lambda})^2} = \begin{pmatrix} \underset{\approx}{H}_{(\mathrm{l})} & \underset{\approx}{0} & \underset{\approx}{I} \\ \underset{\approx}{0} & \underset{\approx}{H}_{(\mathrm{v})} & \underset{\approx}{I} \\ \underset{\approx}{I} & \underset{\approx}{I} & \underset{\approx}{0} \end{pmatrix}. \tag{3.4}$$

Note that a stationary point is found, if $\underset{\sim}{\delta} = \underset{\sim}{0}$ and $\underset{\sim}{g}_{(\mathrm{l})} = \underset{\sim}{g}_{(\mathrm{v})} = -\underset{\sim}{\lambda}$. The condition $\underset{\sim}{g}_{(\mathrm{l})} = \underset{\sim}{g}_{(\mathrm{v})}$ can be interpreted physically as the equality of the intensive state (temperature T, pressure p, and chemical potential $\underset{\sim}{\mu}$):

$$T_{(\mathrm{l})} = T_{(\mathrm{v})}, \quad p_{(\mathrm{l})} = p_{(\mathrm{v})}, \text{ and } \quad \underset{\sim}{\mu}_{(\mathrm{l})} = \underset{\sim}{\mu}_{(\mathrm{v})}\,. \tag{3.5}$$

In an updating scheme, the update-vector $\hat{\underset{\sim}{\Delta}}$ at iteration k is introduced as follows:

$$\hat{\underset{\sim}{\Delta}} = \begin{pmatrix}\Delta\underset{\sim}{x}_{(\mathrm{l})}\\ \Delta\underset{\sim}{x}_{(\mathrm{v})}\\ \Delta\underset{\sim}{\lambda}\end{pmatrix} \quad\text{with}\quad \begin{array}{rcl} \Delta\underset{\sim}{x}_{(\mathrm{l})} & = & \underset{\sim}{x}^{(k+1)}_{(\mathrm{l})} - \underset{\sim}{x}^{(k)}_{(\mathrm{l})}\\ \Delta\underset{\sim}{x}_{(\mathrm{v})} & = & \underset{\sim}{x}^{(k+1)}_{(\mathrm{v})} - \underset{\sim}{x}^{(k)}_{(\mathrm{v})}\\ \Delta\underset{\sim}{\lambda} & = & \underset{\sim}{\lambda}^{(k+1)} - \underset{\sim}{\lambda}^{(k)} \end{array} . \tag{3.6}$$

The N -R method suggests $\underset{\approx}{B}\,\hat{\underset{\sim}{\Delta}} = -\underset{\sim}{l}$:

$$\begin{pmatrix}\underset{\approx}{H}_{(\mathrm{l})} & & \underset{\approx}{I}\\ & \underset{\approx}{H}_{(\mathrm{v})} & \underset{\approx}{I}\\ \underset{\approx}{I} & \underset{\approx}{I} & \end{pmatrix}\begin{pmatrix}\Delta\underset{\sim}{x}_{(\mathrm{l})}\\ \Delta\underset{\sim}{x}_{(\mathrm{v})}\\ \Delta\underset{\sim}{\lambda}\end{pmatrix} = -\begin{pmatrix}\underset{\sim}{g}_{(\mathrm{l})} + \underset{\sim}{\lambda}\\ \underset{\sim}{g}_{(\mathrm{v})} + \underset{\sim}{\lambda}\\ -\underset{\sim}{\delta}\end{pmatrix} . \tag{3.7}$$

By adding $\underset{\sim}{\lambda}$ to the first and second block-rows of the equation system, the result is (zero-blocks $\underset{\approx}{0}$ omitted)

$$\begin{pmatrix}\underset{\approx}{H}_{(\mathrm{l})} & & \boxed{\underset{\approx}{I}}_{2}\\ & \boxed{\underset{\approx}{H}_{(\mathrm{v})}}_{3} & \underset{\approx}{I}\\ \boxed{\underset{\approx}{I}}_{1} & \underset{\approx}{I} & \end{pmatrix}\begin{pmatrix}\Delta\underset{\sim}{x}_{(\mathrm{l})}\\ \Delta\underset{\sim}{x}_{(\mathrm{v})}\\ \underset{\sim}{\lambda}^{(k+1)}\end{pmatrix} = \begin{pmatrix}-\underset{\sim}{g}_{(\mathrm{l})}\\ -\underset{\sim}{g}_{(\mathrm{v})}\\ \underset{\sim}{\delta}\end{pmatrix} , \tag{3.8}$$

or, after introducing $\underset{\sim}{\Delta}$ and $\underset{\sim}{b}$ as abbreviations,

$$\underset{\approx}{B}\cdot\underset{\sim}{\Delta} = \underset{\sim}{b} . \tag{3.9}$$

The matrix $\underset{\approx}{B}$ is block-invertible, i.e. for a given block-structure, there is at least one complete sequence of invertible pivoting blocks, which can be utilised in a block-inversion by G elimination.

A row and column pivoted LU-decomposition of $\underset{\approx}{B}$ with pivot elements as marked in Equation (3.9) yields

$$\underset{\approx}{B} = (\underset{\approx}{P}_{\mathsf{row}}\,\underset{\approx}{L})\,(\underset{\approx}{U}\,\underset{\approx}{P}_{\mathsf{col}}) = \begin{pmatrix}\underset{\approx}{H}_{(\mathrm{l})} & \underset{\approx}{I} & \\ & \underset{\approx}{I} & \underset{\approx}{I}\\ \underset{\approx}{I} & & \end{pmatrix}\cdot\begin{pmatrix}\underset{\approx}{I} & \underset{\approx}{I} & \\ & -\underset{\approx}{H}_{(\mathrm{l})} & \underset{\approx}{I}\\ & \underset{\approx}{H}_{(\mathrm{l})} + \underset{\approx}{H}_{(\mathrm{v})} & \end{pmatrix} . \tag{3.10}$$

Hence, identity pivot blocks can be found almost through the whole solution process. The only exception is the block $\underset{\approx}{H}_{(\mathrm{l})} + \underset{\approx}{H}_{(\mathrm{v})}$ that requires the solution of a non-trivial subsystem. The update $\underset{\sim}{\Delta}$ is therefore obtainable, iff $\underset{\approx}{H}_{(\mathrm{l})} + \underset{\approx}{H}_{(\mathrm{v})}$ is non-singular.

A trivial solution emerges, if the state vectors of the two phases differ only by a scaling factor: $\underset{\sim}{x}_{(\mathrm{l})} = \psi\,\underset{\sim}{x}_{(\mathrm{v})}$. As a consequence of Equation (2.15), the singular directions of $\underset{\approx}{H}_{(\mathrm{l})}$ and $\underset{\approx}{H}_{(\mathrm{v})}$ fall together, and $\underset{\approx}{H}_{(\mathrm{l})} + \underset{\approx}{H}_{(\mathrm{v})}$ becomes singular.

From this point of view, critical points are special cases of trivial solutions, because only one phase actually exists at the critical point. The attempt to solve for the conditions of a critical point by a phase equilibrium calculation can not succeed. Specialised techniques have been developed to solve the task of critical point calculations (Michelsen and Mollerup, 2004).

Azeotropic conditions do not yield a singular coefficient matrix. Even though the chemical composition is equal in both phases, entropy and volume assume distinct values. The calculation of phase equilibria for an azeotropic mixture is similar to an equilibrium calculation of a pure substance. The equation system becomes singular, if only intensive variables are specified (e.g. temperature and pressure).

Multiphase equilibria

for a system of Ω phases, Equation (3.1) can be generalised to

$$\max_{\underset{\sim}{x}_{(\mathrm{i})}} S = \sum_{i=1}^{\Omega} S_{(\mathrm{i})} \quad \text{s.t.} \quad \underset{\sim}{\delta} = \underset{\sim}{x}_{\text{initial}} - \sum_{i=1}^{\Omega} \underset{\sim}{x}_{(\mathrm{i})} = \underset{\sim}{0}\,, \tag{3.11}$$

and the L function takes the form

$$\Lambda(\underset{\sim}{x}_{(1)}, \ldots, \underset{\sim}{x}_{(\mathrm{i})}, \ldots, \underset{\sim}{x}_{(\Omega)}) = \sum_{i=1}^{\Omega} S_i - \underset{\sim}{\lambda} \cdot \underset{\sim}{\delta}\,. \tag{3.12}$$

The equation system $\underset{\approx}{B} \cdot \underset{\sim}{\Delta} = \underset{\sim}{b}$ results to

$$\underset{\approx}{B} = \begin{pmatrix} \ddots & & & \vdots \\ & \underset{\approx}{H}_{(\mathrm{i})} & & \underset{\approx}{I} \\ & & \ddots & \vdots \\ \cdots & \underset{\approx}{I} & \cdots & \underset{\approx}{0} \end{pmatrix} \begin{pmatrix} \vdots \\ \Delta \underset{\sim}{x}_{(i)} \\ \vdots \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} \vdots \\ \underset{\sim}{g}_{(i)} \\ \vdots \\ \underset{\sim}{\delta} \end{pmatrix}. \tag{3.13}$$

Again, it is possible to decompose $\underset{\approx}{B}$ as $(\underset{\approx}{P}_{\text{row}}\,\underset{\approx}{L})\,(\underset{\approx}{U}\,\underset{\approx}{P}_{\text{col}})$, using the boxed blocks as pivot elements for back-substitution:

$$\underset{\approx}{B} = \begin{pmatrix} \underset{\approx}{H}_{(1)} & \boxed{\underset{\approx}{I}} & & & \\ & \vdots & \boxed{\underset{\approx}{I}} & & \\ & \vdots & & \ddots & \\ & \underset{\approx}{I} & & & \boxed{\underset{\approx}{I}} \\ \boxed{\underset{\approx}{I}} & & & & \end{pmatrix} \cdot \begin{pmatrix} \boxed{\underset{\approx}{I}} & \underset{\approx}{I} & & \underset{\approx}{I} & \cdots & \underset{\approx}{I} \\ & -\underset{\approx}{H}_{(1)} & \boxed{\underset{\approx}{I}} & -\underset{\approx}{H}_{(1)} & \cdots & -\underset{\approx}{H}_{(1)} \\ & & & \boxed{\underset{\approx}{H}_{(1)} + \underset{\approx}{H}_{(2)}} & & \\ & & & & \ddots & \\ & & & & & \boxed{\underset{\approx}{H}_{(1)} + \underset{\approx}{H}_{(\Omega-1)}} \\ & \boxed{\underset{\approx}{H}_{(1)} + \underset{\approx}{H}_{(\Omega)}} & & & & \end{pmatrix}. \tag{3.14}$$

The solution of the total system (3.9) is obtained by solving $\Omega - 1$ subsystems of the size of one phase each. The computation time of a multiphase-flash is therefore linear in the number of phases. For the special case $\Omega = 1$, the system becomes linear and the solution $\underset{\sim}{x} = \underset{\sim}{x}_{\text{initial}}$ and $\underset{\sim}{\lambda} = -\underset{\sim}{g}$.

A necessary requirement for a converged solution of Equation (3.9) is that $\Delta\underset{\sim}{x}_{(\mathrm{i})} = \underset{\sim}{0}$, $\forall_{\mathrm{i}}$. Hence $\underset{\sim}{g}_{(\mathrm{i})} = -\underset{\sim}{\lambda}\ \forall_{\mathrm{i}}$ demonstrates that the conjugated variables at the converged solution are represented by the L -multipliers.

3.2.2 Calculation of chemical equilibria

Consider the same storage tank as in the previous section, but this time filled with a reacting phase, for instance a mixture of NO_2 and N_2O_4. The complete set of species balance equations would disallow any chemical reaction, but a chemical reaction still fulfils the balance equations of energy, volume, and chemical elements. In a reacting system, the number of elements is lower than the number of species, such that the constraint matrix $\underset{\approx}{A}$ is no longer square and invertible. For one phase, the program is

$$\max_{\underset{\sim}{x}} S(\underset{\sim}{x}) \quad \text{s.t.} \quad \underset{\sim}{\delta} = \underset{\approx}{A}\,(\underset{\sim}{x}_{\text{initial}} - \underset{\sim}{x}) = \underset{\sim}{0}. \tag{3.15}$$

For the system $NO_2 - N_2O_4$, the state vector is given as $\underset{\sim}{x} = (U, V, n_{NO_2}, n_{N_2O_4})$. The balance equations for oxygen and nitrogen are linear dependent in this case. The row-reduced constraint matrix becomes

$$\underset{\approx}{A} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & 2 \end{pmatrix} \begin{array}{cr} \leftarrow & U\text{-balance} \\ \leftarrow & V\text{-balance}\,. \\ \leftarrow & \text{N/O-balance} \end{array} \tag{3.16}$$

Equation (3.9) now is modified to

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} \\ \underset{\approx}{A} & \end{pmatrix} \begin{pmatrix} \Delta\underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} \\ \underset{\sim}{\delta} \end{pmatrix}. \tag{3.17}$$

At the converged solution, the condition $\Delta\underset{\sim}{x} = \underset{\sim}{0}$ yields $\underset{\sim}{g} = -\underset{\approx}{A}^{\mathrm{T}}\,\underset{\sim}{\lambda}$, implying that the equilibrium condition for the chemical potentials is $\mu_{NO_2} = 2\,\mu_{N_2O_4}$.

Chemical equilibrium in a multi-phase system

The generalisation to reactive systems with many phases does not require a common $\underset{\approx}{A}$ over all phases, since different species sets might occur in different phases. The program is

$$\max_{\underset{\sim}{x}_{(i)}} \sum_i S_{(i)} \quad \text{s.t.} \quad \underset{\sim}{\delta} = \underset{\approx}{A}_{\text{initial}}\,\underset{\sim}{x}_{\text{initial}} - \sum_i \underset{\approx}{A}_{(i)}\underset{\sim}{x}_{(i)} = \underset{\sim}{0}. \tag{3.18}$$

The matrix $\underset{\approx}{A}_{\text{initial}}$ projects the set of initial species into the space of elements. The equation system becomes

$$\begin{pmatrix} \ddots & & \vdots \\ & \underset{\approx}{H}_{(i)} & \underset{\approx}{A}^{\mathrm{T}}_{(i)} \\ & \ddots & \vdots \\ \cdots\ \underset{\approx}{A}_{(i)} & \cdots & \underset{\approx}{0} \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ \Delta \underset{\sim}{x}_{(i)} \\ \vdots \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} \vdots \\ -\underset{\sim}{g}_{(i)} \\ \vdots \\ \underset{\sim}{\delta} \end{pmatrix} . \tag{3.19}$$

In spite of each single $\underset{\approx}{A}_{(\mathrm{i})}$, the total balance equation system $(\ldots, \underset{\approx}{A}_{(\mathrm{i})}, \ldots)$ must be reduced to full row rank in order to obtain a non-singular matrix $\underset{\approx}{B}$ and therefore a solvable system.

Unfortunately, $\underset{\approx}{B}$ is not block-invertible for a general reacting system, since it contains no single invertible block. Naturally, the computational effort is high in the general case, in which no structural information can be utilised directly, but $\underset{\approx}{B}$ must be decomposed on a scalar level, at least considering the *a priori* information about the location of zero-blocks. More efficient approaches for particular systems are discussed in Appendix E.2.

3.3 Non-canonical specifications

In practical cases, there is often no state function with canonical state variables available, so that the constraints are only linear combinations of these variables. This is for instance the case, if intensive variables are constrained (T, p). Let us consider a storage tank filled with pure vapour at a fixed temperature, but allowing energy exchange with the environment. One obvious approach would be to obtain the H energy by L transformation $\mathfrak{L}_S$ of U, namely $A(T, V, \underset{\sim}{n})$. The canonical derivatives of A contain all thermodynamical obtainable properties, see Appendix C.3.

This approach is convenient for an isolated calculation, but as there is no conservation equation for the intensive variable T, the structure of the coefficient matrix would be destroyed in real applications, namely the integration into a process model. Without the balance equation, the canonical equation system is reduced in size, and the temperature needs to be determined externally, for instance by direct specification. However, the state vector $\underset{\sim}{x}$ of the canonical system no longer contains T, and many thermodynamic properties, such as entropy, heat capacity, and expansivity (see Appendix C.3), would require add-on calculations.

It is therefore most generic to formulate the canonical equation system in solely extensive coordinates, as for instance Equations (3.8) and (3.17). The modelling tool *Modeller* (Westerweele *et al.*, 1999) is based on balance equations as well, and as in this case, extensive state variables help to structure the equation system.

The scope of the methods described in the following subsections are actually not limited to the single equilibrium nodes introduced in the last section, but are equally applicable for entire process models including material streams between different FMs, as described in Section 3.4.

3.3.1 Direct substitution of Lagrange multipliers

A special case occurs, if the non-canonical constraints are direct specifications of conjugated variables, namely the gradient of the state function with respect to its canonical state variables. Such constraints can be interpreted as specifications of parts of $\underset{\sim}{\lambda}$. In order to specify the temperature instead of fulfilling the internal energy balance, the first column of the second block column in Equation (3.17) is removed. The missing term λ_U in the first row of the left hand side is accounted for on the right hand side according to $\lambda_U = -g_U = -1/T$ with $T = T_{\mathsf{spec}}$. Simultaneously, the first row of the second block row is removed, since conservation of U is no longer desired:

$$\begin{pmatrix} \underset{\approx}{H} & \hat{\underset{\approx}{A}}^{\mathrm{T}} \\ \hat{\underset{\approx}{A}} & \end{pmatrix} \begin{pmatrix} \Delta\underset{\sim}{x} \\ \hat{\underset{\sim}{\lambda}} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} - \underset{\approx}{A}^{\mathrm{T}} \underset{\sim}{\begin{pmatrix} -1/T_{\mathsf{spec}} \\ 0 \end{pmatrix}} \\ \hat{\underset{\sim}{\delta}} \end{pmatrix} \quad \text{with} \quad \begin{array}{l} \hat{\underset{\approx}{A}} = \begin{pmatrix} \underset{\sim}{0} & \underset{\approx}{I} \end{pmatrix} \underset{\approx}{A} \\ \hat{\underset{\sim}{\lambda}} = \begin{pmatrix} \underset{\sim}{0} & \underset{\approx}{I} \end{pmatrix} \underset{\sim}{\lambda} \\ \hat{\underset{\sim}{\delta}} = \begin{pmatrix} \underset{\sim}{0} & \underset{\approx}{I} \end{pmatrix} \underset{\sim}{\delta} \end{array} . \tag{3.20}$$

The advantage of this method is the reduction of system size, but again at the expense of structure. $\hat{\underset{\approx}{A}}$ is not invertible even for non-reacting systems, and methods described in Appendix E.2 must be applied. The restriction to specifications of conjugated variables only requires combination with other methods, if arbitrary constraint equations should be applicable. For this reason, the direct substitution of L multipliers cannot be applied practically in a flexible process modelling tool.

3.3.2 Constitutive equation system

The concept of this approach is to substitute selected balance equations of canonical state variables by arbitrary equations depending on $\underset{\sim}{x}$, $\underset{\sim}{\lambda}$, and the process parameters $\underset{\sim}{u}$:

$$\underset{\sim}{h}(\underset{\sim}{x}, \underset{\sim}{\lambda}, \underset{\sim}{u}) = \underset{\sim}{0} . \tag{3.21}$$

The equations of this system are called *constitutive equations*, as they constitute the behaviour of a particular FM or the entire process model. Examples are not only direct specifications, such as $T - T_{\mathsf{spec}} = 0$, but as well heat transfer laws, pressure-flow equations, and characterisation of kinetic reactions.

Each balance equation of the canonical equation system (3.9), which is selected to be replaced by a constitutive equation, represents one DOF. The use of any constitutive equation requires one such DOF, so that the number of state variables is balanced with the number of active equations. From now on, such selected balance equations are referred to as *released*.

Integrating the constitutive equations in a linearised form into the canonical equation system is not an attractive approach, as the block structure would be negatively affected.

Therefore, each released balance equation remains as is in the equation system, but its right hand side δ is replaced by a new variable α.

The objective is to obtain a value for $\underset{\sim}{\alpha}$, such that the solution of the modified canonical equation system (3.9) fulfils also the constitutive equation system (3.21). Substituting the energy balance by a temperature specification, the right hand side of Equation (3.17) is supplemented by $\underset{\sim}{\alpha} = \alpha\, \underset{\sim}{e}_1$ to

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} \\ \underset{\approx}{A} & \end{pmatrix} \begin{pmatrix} \Delta \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} \\ \hat{\underset{\sim}{\delta}} + \Delta \underset{\sim}{\alpha} \end{pmatrix} \tag{3.22}$$

with $\hat{\underset{\sim}{\delta}} = \sum_{i \neq 1} \delta_i\, \underset{\sim}{e}_i$ and $\Delta \underset{\sim}{\alpha} = \underset{\sim}{\alpha}^{(k+1)} - \underset{\sim}{\alpha}^{(k)}$. As $\underset{\approx}{B}$ is calculated at $\underset{\sim}{x}^{(k)}$, the homogeneity Equation (2.15) allows to substitute $\Delta\underset{\sim}{x}$ by $\underset{\sim}{x}^{(k+1)}$. Hence, it follows that

$$\begin{pmatrix} \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix}^{(k+1)} = \underset{\approx}{B}^{-1} \begin{pmatrix} -\underset{\sim}{g} \\ \underset{\sim}{\alpha} \end{pmatrix} \quad \Rightarrow \quad \left. \frac{\partial (\underset{\sim}{x}, \underset{\sim}{\lambda})^{(k+1)}}{\partial \underset{\sim}{\alpha}} \right|_{(\underset{\sim}{x}, \underset{\sim}{\lambda})^{(k)}} = \underset{\approx}{B}^{-1}\, \underset{\approx}{E}_\alpha \,. \tag{3.23}$$

The matrix $\underset{\approx}{E}_\alpha$ represents a set of unity column vectors. Multiplied from the right, it selects the columns of $\underset{\approx}{B}^{-1}$, which correspond to the released balance equations. Considering $\underset{\sim}{x}$ and $\underset{\sim}{\lambda}$ as a function of $\underset{\sim}{\alpha}$, the derivative of $\underset{\sim}{h}$ can be obtained by chain-rule. Typically, only a small subset of canonical variables are used in any constitutive equation. The J matrix $\partial \underset{\sim}{h} / \partial(\underset{\sim}{x}, \underset{\sim}{\lambda})$ therefore contains only a few columns with non-zero elements, and it becomes practical to introduce also a selection matrix $\underset{\approx}{E}_x$, consisting of unity row vectors. Multiplied from the left, it selects rows in $\underset{\approx}{B}^{-1}$ according to the canonical variables that appear in the constitutive equations $(\hat{\underset{\sim}{x}}, \hat{\underset{\sim}{\lambda}})$:

$$\left. \frac{\partial \underset{\sim}{h}(\underset{\sim}{x}, \underset{\sim}{\lambda}, \underset{\sim}{u})}{\partial \underset{\sim}{\alpha}} \right|_{(\underset{\sim}{x}, \underset{\sim}{\lambda})^{(k)}} = \frac{\partial \underset{\sim}{h}}{\partial (\underset{\sim}{x}, \underset{\sim}{\lambda})}\, \underset{\approx}{B}^{-1}\, \underset{\approx}{E}_\alpha = \frac{\partial \underset{\sim}{h}}{\partial (\hat{\underset{\sim}{x}}, \hat{\underset{\sim}{\lambda}})}\, \underset{\approx}{E}_x\, \underset{\approx}{B}^{-1}\, \underset{\approx}{E}_\alpha \,. \tag{3.24}$$

Application of the N-R method suggests an update $\Delta\underset{\sim}{\alpha}$ as

$$\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{\alpha}}\, \Delta\alpha = -\underset{\sim}{h} \quad \Rightarrow \quad \frac{\partial \underset{\sim}{h}}{\partial (\hat{\underset{\sim}{x}}, \hat{\underset{\sim}{\lambda}})}\, \underset{\approx}{E}_x\, \underset{\approx}{B}^{-1}\, \underset{\approx}{E}_\alpha\, \Delta\alpha = -\underset{\sim}{h} \,. \tag{3.25}$$

Equation (3.22) can be decomposed as

$$\begin{pmatrix} \Delta \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} \Delta \underset{\sim}{x}_1 \\ \underset{\sim}{\lambda}_1 \end{pmatrix} + \begin{pmatrix} \Delta \underset{\sim}{x}_2 \\ \Delta \underset{\sim}{\lambda} \end{pmatrix} = \underset{\approx}{B}^{-1} \begin{pmatrix} -\underset{\sim}{g} \\ \hat{\underset{\sim}{\delta}} \end{pmatrix} + \underset{\approx}{B}^{-1} \begin{pmatrix} 0 \\ \Delta \underset{\sim}{\alpha} \end{pmatrix} . \tag{3.26}$$

The partial solution $(\underset{\sim}{x}_1, \underset{\sim}{\lambda}_1)$ is the solution of the canonical constrained optimisation problem (3.17), but assuming the released balance equations to be fulfilled at $\underset{\sim}{x}^{(k)}$. Hence, the L-multipliers $\underset{\sim}{\lambda}_1$ can be interpreted physically as the canonical conjugated variable set at $\underset{\sim}{x}^{(k+\frac{1}{2})} = \underset{\sim}{x}^{(k)} + \Delta\underset{\sim}{x}_1$. The determination of $\Delta\underset{\sim}{\alpha}$ is therefore based on this pair. Subsequently, $\underset{\approx}{B}$ should be updated based on $\underset{\sim}{x}^{(k+\frac{1}{2})}$, but the high computational effort to obtain ${\underset{\approx}{B}^{(k+\frac{1}{2})}}^{-1}$ is not justified, as $\underset{\approx}{B}^{(k+\frac{1}{2})} \approx \underset{\approx}{B}^{(k)}$. The complete solution scheme including a relaxation γ becomes as follows:

```
1   $k := 0$
2   **while** not converged
3       determine $\underset{\approx}{B}$, $\underset{\sim}{g}$ and $\underset{\sim}{\hat{\delta}}$ at $\underset{\sim}{x}^{(k)}$ by state function evaluations
4       solve $\underset{\approx}{B}\binom{\Delta\underset{\sim}{x}_1}{\underset{\sim}{\lambda}_1} = \binom{-\underset{\sim}{g}}{\underset{\sim}{\hat{\delta}}}$
5       $(\underset{\sim}{x}^{(k+\frac{1}{2})}, \underset{\sim}{\lambda}^{(k+\frac{1}{2})}) := (\underset{\sim}{x}^{(k)} + \Delta\underset{\sim}{x}_1, \underset{\sim}{\lambda}_1)$
6       determine $\underset{\sim}{h}^{(k+\frac{1}{2})}$ and $[\partial\underset{\sim}{h}/\partial(\underset{\sim}{x},\underset{\sim}{\lambda})]^{(k+\frac{1}{2})}$
7       solve $[\partial\underset{\sim}{h}/\partial(\underset{\sim}{x},\underset{\sim}{\lambda})\, \underset{\approx}{E}_x\, \underset{\approx}{B}^{-1}\, \underset{\approx}{E}_\alpha]\, \Delta\underset{\sim}{\alpha} = -\underset{\sim}{h}$
8       solve $\underset{\approx}{B}\binom{\Delta\underset{\sim}{x}_2}{\Delta\underset{\sim}{\lambda}_2} = \binom{\underset{\sim}{0}}{\Delta\underset{\sim}{\alpha}}$
9       $\Delta\underset{\sim}{x} := \gamma\,(\Delta\underset{\sim}{x}_1 + \Delta\underset{\sim}{x}_2)$ with $\gamma \in ]0:1]$
10      $\underset{\sim}{x}^{(k+1)} := \underset{\sim}{x}^{(k)} + \Delta\underset{\sim}{x}$
11      $k := k + 1$
12  **end while**
```

This concept of alternating updates is complementary to the concept of nested iterations in an outer and an inner loop. There is no need to converge an inner system in order to perform one step in the outer loop. Instead, this approach is more similar to using a predictor-corrector step when integrating ordinary differential-algebraic systems. The relaxation strategy to obtain γ is further described in Section 3.8.

The simulation tool *Yasim*, as described in Chapter 5, implements this algorithm, and performance characteristics are discussed in Chapter 6.

Though this approach incorporates the use of a structured $\underset{\approx}{B}$ and additionally requires only the solution of a rather small equation system of constitutive equations, the main disadvantage is the demand for the explicit evaluation of $\underset{\approx}{B}^{-1}$, which causes numerical problems and performance loss for larger systems, compared to solution strategies based on decomposition and back-substitution only (Golub and Loan, 1996). The next section therefore introduces a method that avoids the use of $\underset{\approx}{B}^{-1}$.

3.3.3 Augmented equation system

The algorithm described in the previous section can be modified to avoid an explicit evaluation of $\underset{\approx}{B}^{-1}$. Lines 1–6 are left unchanged, while Equation (3.25) can be combined with the canonical equation system to calculate $(\Delta\underset{\sim}{x}_2, \Delta\underset{\sim}{\lambda}_2)$ as follows:

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} & \\ \underset{\approx}{A} & & -\underset{\approx}{E}_\alpha \\ \partial\underset{\sim}{h}/\partial\underset{\sim}{x} & \partial\underset{\sim}{h}/\partial\underset{\sim}{\lambda} & \partial\underset{\sim}{h}/\partial\underset{\sim}{\alpha} \end{pmatrix} \begin{pmatrix} \Delta\underset{\sim}{x}_2 \\ \Delta\underset{\sim}{\lambda}_2 \\ \Delta\underset{\sim}{\alpha} \end{pmatrix} = \begin{pmatrix} \underset{\sim}{0} \\ \underset{\sim}{0} \\ -\underset{\sim}{h} \end{pmatrix}. \tag{3.27}$$

In this case, $\underset{\approx}{E}_\alpha$ is defined more narrowly in order to select specific balance equations from the entire canonical system. The algorithm continues at line 9, and only $\Delta\underset{\sim}{x}_2$

is actually used further on. A disadvantage of this method is that two large equation systems need to be solved in each iteration. Still, the total numerical effort to solve the canonical system and the augmented system can be expected to be lower than the explicit evaluation of $\underset{\approx}{B}^{-1}$. A row and column pivoted LU-decomposition of the augmented coefficient matrix is conducted in analogy to Section 3.2.

Furthermore, it is now natural to consider a direct dependency of $\underset{\sim}{h}$ on $\underset{\sim}{\alpha}$, i.e. constitutive equations can contain source-terms for canonical balance equations directly: $\underset{\sim}{h} = \underset{\sim}{h}(\underset{\sim}{x}, \underset{\sim}{\lambda}, \underset{\sim}{\alpha})$. With regards to advanced process engineering disciplines described in Chapter 4, it is suitable to express all process model parameters $\underset{\sim}{u}$ in constitutive equations rather than using $\underset{\sim}{x}_{\text{initial}}$ as in Equation (3.1). For stand-alone building blocks, all balance equations are released, hence $\underset{\approx}{E}_\alpha = \underset{\approx}{I}$.

Example

Consider a temperature controlled storage tank of an unknown quantity gaseous ammonia, but with specified volume and pressure. Let the current set $(\underset{\sim}{x}, \underset{\sim}{\lambda})$ be a solution of the canonical system based on entropy $S(U, V, n)$:

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{I} \\ \underset{\approx}{I} & \end{pmatrix} \begin{pmatrix} \Delta \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} \\ \underset{\sim}{0} \end{pmatrix} . \tag{3.28}$$

In this case, all balance equations are released, such that $\underset{\sim}{\hat{\delta}} = 0$. For any choice of the state $\underset{\sim}{x}$, the update $\Delta \underset{\sim}{x}$ is a zero-vector. $\underset{\sim}{x}$ can therefore be chosen arbitrary, while $\underset{\sim}{\lambda} = -\underset{\sim}{g}$. Given specifications of temperature T_{spec}, pressure p_{spec}, and volume V_{spec} as process parameters, the constitutive equation system is

$$\underset{\sim}{h} = \left(\lambda_1 + \frac{1}{T_{\text{spec}}},\ \lambda_2 - p_{\text{spec}}\,\lambda_1,\ \alpha_2 - V_{\text{spec}} \right) \quad \text{with} \quad \alpha_2 = V^{(k+\frac{1}{2})} . \tag{3.29}$$

The physical interpretation of $\lambda_1 = -g_1$ and $\lambda_2 = -g_2$ is developed in Appendix C.3. The last line in Table C.1 describes the gradient of the used entropy function as $g_1 = T^{-1}$ and $g_2 = p/T$. Subsequently

$$\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{x}} = \underset{\approx}{0}, \quad \frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{\lambda}} = \begin{pmatrix} 1 & 0 & 0 \\ -p_{\text{spec}} & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{\alpha}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad \text{and} \quad \underset{\approx}{E}_\alpha = \underset{\approx}{I} . \tag{3.30}$$

Evaluation of Equation (3.27) with $\underset{\approx}{A} = \underset{\approx}{I}$ gives $\left[\left(\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{\lambda}}, \frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{\alpha}} \right) \begin{pmatrix} \underset{\approx}{H} \\ -\underset{\approx}{I} \end{pmatrix} \right] \Delta \underset{\sim}{x}_2 = \underset{\sim}{h}$, hence

$$\begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{12} - p_{\text{spec}} H_{11} & H_{22} - p_{\text{spec}} H_{12} & H_{23} - p_{\text{spec}} H_{13} \\ 0 & -1 & 0 \end{pmatrix} \Delta \underset{\sim}{x}_2 = \begin{pmatrix} \lambda_1 + \frac{1}{T_{\text{spec}}} \\ \lambda_2 - p_{\text{spec}}\,\lambda_1 \\ \alpha_2 - V_{\text{spec}} \end{pmatrix} . \tag{3.31}$$

It can be seen that the volume correction $\Delta x_{2,2}$ is independent of the thermodynamic model: $\Delta V = V_{\text{spec}} - V^{(k+\frac{1}{2})}$, while the temperature and the pressure specifications are coupled and therfore model dependent. For an ideal gas with constant heat capacity c_p, and a convenient reference state $\Delta_f H^0 = c_p\,T^0$, the H matrix is

$$\underset{\approx}{H} = \frac{\partial^2 S}{\partial(U,V,n)^2} = \begin{pmatrix} -\frac{1}{T^2 c_V n} & 0 & \frac{1}{T n} \\ 0 & -\frac{p}{T V} & \frac{R}{V} \\ \frac{1}{T n} & \frac{R}{V} & -\frac{c_V+R}{n} \end{pmatrix}. \tag{3.32}$$

The updates Δn and ΔU are calculated as follows:

$$\Delta n = \left(\frac{V_{\text{spec}}}{V} + \frac{p_{\text{spec}}\,T}{p\,T_{\text{spec}}} - 2\right) n \qquad \Delta U = c_V\,T\left[\left(1 - \frac{T}{T_{\text{spec}}}\right) n + \Delta n\right]. \tag{3.33}$$

Table 3.1: *Calculation of an ideal gas storage tank by evaluation of the augmented equation system.*

Starting point	Initialisation	Specification	After 1st step	After 2nd step
$T_0 = 298.15$ K	$U_0 = n_0\,c_V\,T_0$	$T_{\text{spec}} = 400$ K	$T^{(1)} = 302.8$ K	$T^{(2)} = 319.0$ K
$V_0 = 0.1$ m^3	$V_0 = V_0$	$V_{\text{spec}} = 1$ m^3	$V^{(1)} = 1$ m^3	$V^{(2)} = 1$ m^3
$p_0 = 1$ bar	$n_0 = \frac{p_0\,V_0}{R\,T_0}$	$p_{\text{spec}} = 10$ bar	$p^{(1)} = 1.67$ bar	$p^{(2)} = 8.0$ bar
	$= 4.03$ mol	$n^{(\infty)} = 300.7$ mol	$n^{(1)} = 66.4$ mol	$n^{(2)} = 300.7$ mol

Table 3.1 shows the result of a numerical experiment. Obviously, the solution can easily be obtained analytically, but the example shows the capabilities of this generic method.

As expected, the correct volume is calculated in one step. Subsequently, the update Δn reduces to $\Delta n = p_{\text{spec}}\,V_{\text{spec}}/(R\,T_{\text{spec}}) - n$, such that $n_2 = n_\infty$. Due to the ideal gas law, temperature and pressure consequently assume the same relative error. According to the definition given by Nocedal and Wright (1999), the convergence rate is quadratic, that is, with $\psi^{(k)}$ as the numerical value of pressure or temperature after step k,

$$\ln\left\|\psi^{(k+1)}/\psi_{\text{spec}} - 1\right\| = 2\,\ln\left\|\psi^{(k)}/\psi_{\text{spec}} - 1\right\| + \text{const.} \quad \text{with } \psi \in \{T, p\}. \tag{3.34}$$

3.4 Process modelling

The previous section concentrated on the mathematical description of atomic building blocks in canonical process modelling. This section focuses on the interconnection of these blocks by balance equations of canonical variables and constitutive equations.

3.4.1 Mathematical framework for process models

Referring to Section 2.3, and in particular Figure 2.4, the introduced concepts can now be substantiated by a mathematical framework.

Terms and definitions 3.1

Atomic flowsheet module An assembly of at least one phase in restricted physical and chemical equilibrium, represented by a suitable building block described in Section 3.2. An atomic FM has exactly one set of constraint equations, consisting of balance equations supplemented by additional constraints. The coefficient matrix $\underset{\approx}{B}$ of any FM is square and is generally invertible.

Input port A frame for one set of constrained equations in a FM. An Input port defines one constraint vector for each flow of a canonical quantity from an upstream FM. This vector defines the contribution to the constraint equations.

Output port A complete set of canonical variables of one physical phase within a FM. Not all phases within a flowsheet module represent output ports, although at least one phase in each atomic flowsheet module does.

Coupling A set of constraint vectors defined by the downstream FM input port according to the canonical variables of the upstream FM output port agglomerated into a coupling matrix. Coupling matrices are in general of rectangular shape and sparse.

Constitutive equation One equation of $\underset{\sim}{h} = \underset{\sim}{0}$ as $h_i(\underset{\sim}{x}, \underset{\sim}{\lambda}, \underset{\sim}{\alpha}) = 0$. In particular, massless transfer of heat or work between two flowsheet modules are represented by constitutive equations, not couplings.

The coefficient matrix $\underset{\approx}{B}^{[k+1]}$ of a composite flowsheet module couples the blocks $\underset{\approx}{B}_i^{[k]}$ of child flowsheet modules with coupling matrices $\underset{\approx}{C}_{ij}^{[k+1]}$. With $\underset{\approx}{B}_i^{[k]}$ arranged on the block-diagonal, $\underset{\approx}{C}_{ij}^{[k+1]}$ is positioned in block-column i and block-row j, indicating a coupling between module M_i and M_j.

3.4.2 Process model topology

The left side of Figure 3.1 shows the flowsheet representation of a simplified urea synthesis process, the so-called Snamprogetti process (UNIDO and IFDC, 1988). As illustrated in the right side of the figure, the adjacency matrix of the process topology graph directly reflects the block-structure of the coefficient matrix. It also becomes clear that material sinks (stream 8 and 12) are not explicit flowsheet modules, but only representations of otherwise non-coupled output ports.

The sequence of FMs in the coefficient matrix is arbitrary. When sorted by listing upstream FMs before downstream FMs, process models without recycles yield a lower triangular block matrix. It becomes evident how an efficient solver can exploit the topological information. By block elimination, the process model can be solved in linear time regarding the number of FMs.

For each recycle introduced into the process, one coupling block is necessary to be positioned on the other side of the diagonal, as stream 5 in the example of Figure 3.1. These recycle streams are conforming with the tear-streams in sequential-modular approaches (see Section 2.4.1), and as they require iterations in that ap-

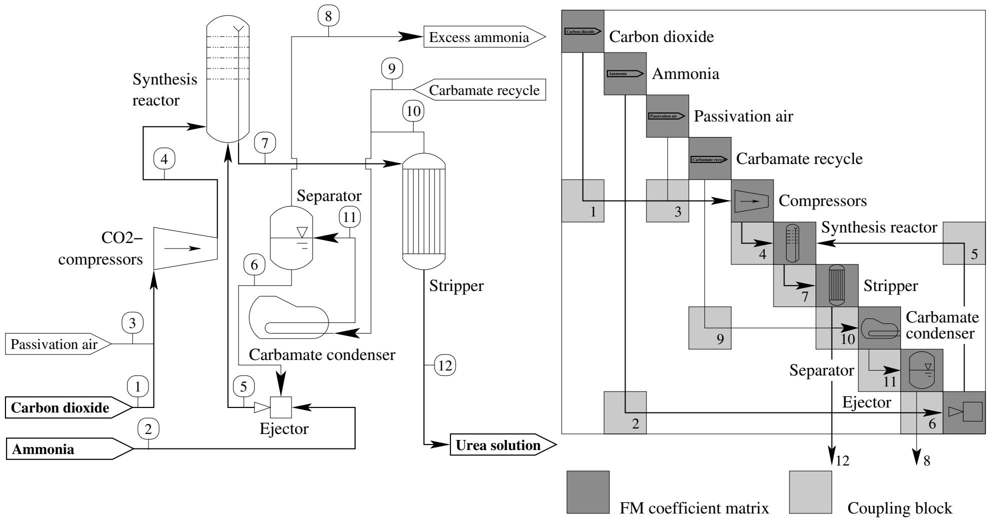


Figure 3.1: *Snamprogetti urea synthesis process and structure of process model coefficient matrix.*

proach, they also require a non-trivial matrix decomposition in the canonical solution strategy. As shown in Section 3.9.2, the increase in model complexity due to occurrence of recycles is inevitable.

Hierarchical process model structure

A large number of process modelling tools implement the concept of hierarchical model structures, such as *Modeller* (Westerweele *et al.*, 1999), M K (Bogusch *et al.*, 2001), gPROMS (Pantelides and Barton, 1992), *Modelica* (Mattsson *et al.*, 1998), and MODEL.LA (Stephanopoulos *et al.*, 1990).

Consider the reactor model in Figure 3.1 to be a composite FM representing 50 vertically arranged discrete volumes. Furthermore, the compressors are arranged in three stages with inter-cooling. The complete reactor model and the complete compressor train are still represented by only one main diagonal block each, and the coupling blocks still remain in the same position. However, it is possible to open the compressor train diagonal block and find a similar structure on lower level. In this context, the compressor train represents an independent process model in itself.

This approach of encapsulation allows one to exchange FMs of same functionality, but at different levels of detail. Using first principle models for process equipment or entire process sections, a process model can quickly be developed on a high level. In order to enhance model predictivity, more detailed FMs can later be substituted in.

3.4.3 Material couplings

A coupling represents a material flow from an output port of an upstream FM M$_1$ to an input port of a downstream FM M$_2$. Let $\underset{\sim}{\dot{x}}_1$ be the state vector representing the outlet stream of M$_1$, and $\underset{\sim}{\dot{x}}_{2,i}$ the state vectors representing the outlet streams of M$_2$. The canonical balance equations to conserve the state variables are

$$\underset{\approx}{A}_1\,\underset{\sim}{\dot{x}}_1 = \sum_i \underset{\approx}{A}_{2,i}\,\underset{\sim}{\dot{x}}_{2,i}\,. \tag{3.35}$$

The constraint matrices are determined by the downstream FM with regard to the actual species set of $\underset{\sim}{\dot{x}}_1$. The building blocks introduced in Section 3.2 only contain the right hand side of Equation 3.35, and the left hand side adds a further contribution. This contribution substitutes the source term $\underset{\sim}{\dot{x}}_{\mathrm{initial}}$ in the particular definitions of $\underset{\sim}{\delta}$.

Without a coupling in between, the canonical equation systems of two FMs are completely independent, and can be arranged as square blocks in the overall coefficient matrix. The left hand side of Equation (3.35) generates an off-diagonal element. The canonical equation system is

$$\begin{pmatrix} \ddots & & & & \\ & \underset{\approx}{B}_1 & & & \\ & & \ddots & & \\ & \underset{\approx}{C}_{12} & & \underset{\approx}{B}_2 & \\ & & & & \ddots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ \underset{\sim}{\Delta}_1 \\ \vdots \\ \underset{\sim}{\Delta}_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \underset{\sim}{l}_1 \\ \vdots \\ \underset{\sim}{l}_2 \\ \vdots \end{pmatrix}. \tag{3.36}$$

The coupling block $\underset{\approx}{C}_{12}$ is sparse and well-structured, since there is no direct link from either L -multipliers or downstream equilibrium equations.

Example

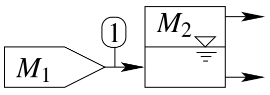


Figure 3.2: *Example process model with a single coupling.*

Figure 3.2 shows a small process model. The reservoir M$_1$ is coupled to a two-phase flash M$_2$. Assuming equal sets of chemical species in both FMs, the coupling matrix $\underset{\approx}{C}_{12}$ contains only one none-zero block:

$$\underset{\approx}{C}_{12} = \begin{pmatrix} \underset{\approx}{0} & \underset{\approx}{0} \\ \underset{\approx}{0} & \underset{\approx}{0} \\ -\underset{\approx}{I} & \underset{\approx}{0} \end{pmatrix}. \tag{3.37}$$

The complete canonical equation system is ($\underset{\sim}{\delta}_1 = -\underset{\sim}{\dot{x}}_1$ and $\underset{\sim}{\delta}_2 = \underset{\sim}{\dot{x}}_1 - \underset{\sim}{\dot{x}}_{2,(\mathrm{l})} - \underset{\sim}{\dot{x}}_{2,(\mathrm{v})}$)

$$\left(\begin{array}{cc|ccc} \underset{\approx}{H}_1 & \underset{\approx}{I} & & & \\ \underset{\approx}{I} & & & & \\ \hline & & \underset{\approx}{H}_{2,(\mathrm{l})} & & \underset{\approx}{I} \\ & & & \underset{\approx}{H}_{2,(\mathrm{v})} & \underset{\approx}{I} \\ -\underset{\approx}{I} & & \underset{\approx}{I} & \underset{\approx}{I} & \end{array}\right) \left(\begin{array}{c} \Delta\underset{\sim}{\dot{x}}_1 \\ \underset{\sim}{\lambda}_1 \\ \hline \Delta\underset{\sim}{\dot{x}}_{2,(\mathrm{l})} \\ \Delta\underset{\sim}{\dot{x}}_{2,(\mathrm{v})} \\ \underset{\sim}{\lambda}_2 \end{array}\right) = \left(\begin{array}{c} -\underset{\sim}{g}_1 \\ \hat{\underset{\sim}{\delta}}_1 + \underset{\sim}{\alpha}_1 \\ \hline -\underset{\sim}{g}_{2,(\mathrm{l})} \\ -\underset{\sim}{g}_{2,(\mathrm{v})} \\ \hat{\underset{\sim}{\delta}}_2 + \underset{\sim}{\alpha}_2 \end{array}\right). \tag{3.38}$$

Coupling equations are balance equations of canonical variables and can be released as described in Section 3.3.2. The contributions $\hat{\underset{\sim}{\delta}}_i$ therefore only contain the right hand side of non-released canonical balance equations.

As applied in Equation (3.38), the initial state vector $\dot{\underset{\sim}{x}}_{\mathsf{initial}}$ can be omitted even for M_1. The reservoir flow is then given by $\underset{\sim}{\alpha}_1$ that is determined by constitutive equations. As a consequence of releasing all balance equations in M_1, $\hat{\underset{\sim}{\delta}} = \underset{\sim}{0}$.

3.5 Atomic flowsheet modules

An interesting idea is to design one generic atomic FM that can serve as a basis for all possible combinations of physical and chemical equilibrium. This FM would always perform a full phase stability test and allow for chemical reactions as well. Output ports of FMs are defined at runtime, and a clever distribution feature defines how to distribute phases to these output ports. Rules define which chemical species they include and how constraints are dependent on the current set of phases.

Apart from the complexity of the task to implement such a general FM, there are incompatible requirements for different FMs. Trusting the thermodynamic model and equilibrium conditions in a phase separation can be desirable in one FM, but have negative side effects in a first principle phase separation if the stability of phases is better known by the user than the thermodynamic model. Therefore, a small set of atomic FMs is suggested in the following subsections. This set can easily be extended, for instance towards multiphase equilibrium calculations, but is still sufficient for most practical applications in steady-state process modelling.

3.5.1 One-phase module

The most primitive FM is that with one physical phase and no reactions enabled. The canonical equation system

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{I} \\ \underset{\approx}{I} & \end{pmatrix} \begin{pmatrix} \Delta\dot{\underset{\sim}{x}} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} \\ \hat{\underset{\sim}{\delta}} + \underset{\sim}{\alpha} \end{pmatrix} \quad \text{with} \quad \hat{\underset{\sim}{\delta}} = -\dot{\underset{\sim}{x}} \tag{3.39}$$

is linear in $\dot{\underset{\sim}{x}}$, which means that $\Delta\dot{\underset{\sim}{x}} = \underset{\sim}{\alpha} - \dot{\underset{\sim}{x}}^{(\mathrm{k})}$ yields the exact canonical update $\underset{\sim}{x}^{(\mathrm{k+1})}$ in each iteration. Furthermore, the decomposition $\underset{\approx}{B} = (\underset{\approx}{P}_{\mathsf{row}}\, \underset{\approx}{L})\,(\underset{\approx}{P}_{\mathsf{col}}\, \underset{\approx}{U})$ is trivial with $\underset{\approx}{P}_{\mathsf{row}}\, \underset{\approx}{L} = \underset{\approx}{B}$ and $\underset{\approx}{P}_{\mathsf{col}}\, \underset{\approx}{U} = \underset{\approx}{I}$. In other words, the inverse matrix $\underset{\approx}{B}^{-1}$ can be obtained without any numerical effort:

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{I} \\ \underset{\approx}{I} & \end{pmatrix}^{-1} = \begin{pmatrix} & \underset{\approx}{I} \\ \underset{\approx}{I} & -\underset{\approx}{H} \end{pmatrix} . \tag{3.40}$$

The $\dim(\dot{\underset{\sim}{x}})$ canonical balance equations can be released and serve as DOFs for constitutive equations.

This module can be applied to: (i) collect multiple streams under mass-balance and two further constraints and obtain a uniform set of canonical conjugated variables, if it is certain that only one phase exists, (ii) implement a material source, which exactly requires all $\dim(\underset{\sim}{\dot{x}})$ DOFs to be specified by constitutive equations, and (iii) represent any one-phase calculation, such as pumps and simplified models of compressors and valves. A source module to represent a two-phase flow or a flow at chemical equilibrium is obtained by combining a one-phase source module with a subsequent flash or reactor module into a composite FM. The canonical variables of that particular module can be used in constitutive equations to specify the DOFs of the source module.

At least one input stream is expected if the one-phase module is not used as a source module. In this case, the state vector of every input stream is added to the residual vector, thus $\hat{\underset{\sim}{\delta}} = -\underset{\sim}{\dot{x}} + \sum_i \underset{\sim}{\dot{x}}_{\mathrm{in},i}$. From now on, the sum of all input streams is combined to the total input stream $\underset{\sim}{\dot{x}}_{\mathrm{in}} = \sum_i \underset{\sim}{\dot{x}}_{\mathrm{in},i}$.

3.5.2 Two-phase equilibrium flash

The case of a stream splitting into two physical phases with equal chemical species sets without reaction is worth being considered as a distinct FM, because it represents a very common operation in practical cases, and as derived in Section 3.3.2, the equation system

$$\begin{pmatrix} \underset{\approx}{H}_{(1)} & & \underset{\approx}{I} \\ & \underset{\approx}{H}_{(2)} & \underset{\approx}{I} \\ \underset{\approx}{I} & \underset{\approx}{I} & \end{pmatrix} \begin{pmatrix} \Delta\underset{\sim}{\dot{x}}_{(1)} \\ \Delta\underset{\sim}{\dot{x}}_{(2)} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g}_{(1)} \\ -\underset{\sim}{g}_{(2)} \\ \hat{\underset{\sim}{\delta}} + \underset{\sim}{\alpha} \end{pmatrix} \tag{3.41}$$

provides valuable structural information. Under conservation of material flow, there are two potential DOFs left for constitutive equations. Among most common specifications used for this are those of temperature, pressure, heat duty, vapour fraction and target concentration of species in one of the phases.

Stand-alone, this module can be used to represent a flash tank or a simple model of partial evaporators and condensers. Applications in a composite context are trays of non-reactive columns and heat exchangers with phase transition.

3.5.3 Reactor modules

To find an intuitive and consistent modelling approach of chemical reactions is a challenge in process modelling, in particular in the combination of equilibrium, stoichiometric and kinetic reactions. Conventional software often requires distinct modules for each type, thus combined reactions are not easily mapped into a process model.

As a motivation, consider the conditions in a urea synthesis reactor. To begin with, the following equilibrium reaction is considered:

$$\mathrm{CO_2} + 2\,\mathrm{NH_3} \leftrightarrows \mathrm{NH_2COONH_4}\ \text{(ammonium carbamate)}\,. \tag{3.42}$$

The actual synthesis reaction is approaching equilibrium inhibited by kinetic effects:

$$\mathrm{NH_2COONH_4} \rightarrow \mathrm{NH_2CONH_2}\ \text{(urea)} + \mathrm{H_2O}\,. \tag{3.43}$$

Furthermore, a stoichiometric relation describes the formation of the undesired by-product biuret:

$$2\,NH_2CONH_2 \;\rightarrow\; NH_2CONHCONH_2\text{ (biuret)} + NH_3\,. \tag{3.44}$$

In a conventional approach, the equilibrium reactions would probably be described by fast kinetics, and the biuret formation would be performed in a subsequent stoichiometric reactor. The consequence is increased numerical effort due to the additional kinetic reactions. Furthermore it is difficult to incorporate the effect of biuret-formation on the main reactions.

Description of chemical reactions

There are two basic approaches to describe chemical reactions, both of them described by Michelsen and Mollerup (2004). A common approach is to define each possible reaction through a vector $\underset{\sim}{\nu}_i$ of stoichiometric coefficients. Starting from an initial molar vector $\underset{\sim}{n}_{\mathsf{initial}}$, each reaction represents a dimension of the space of possible states $\underset{\sim}{n}$:

$$\underset{\sim}{n} = \underset{\sim}{n}_{\mathsf{initial}} + \sum_i \psi_i \underset{\sim}{\nu}_i \quad \text{with} \quad \psi_i \in \mathbb{R} \quad \text{and} \quad n_i \geq 0\,. \tag{3.45}$$

This approach seems to be a good choice for large sets of species with few reactions, and it reflects the traditional way to describe chemical reactions as reaction equations, such as (3.42), (3.43), and (3.44). However, the user, who defines the reactions, is required to provide consistent stoichiometric coefficients, which conform to the balance equations of chemical elements. A subsequent validation is required to ensure a consistent mathematical model. Only stoichiometric vectors $\underset{\sim}{\nu}_i$ in the right null space of the formula matrix $\underset{\approx}{R}$ are allowed: $\underset{\approx}{R}\,\underset{\sim}{\nu}_i = \underset{\sim}{0}$. The formula matrix is the chemical part of the constraint matrix $\underset{\approx}{A}$ in reactor FMs:

$$\underset{\approx}{A} = \begin{pmatrix} 1 & & \\ & 1 & \\ & & \underset{\approx}{R} \end{pmatrix}. \tag{3.46}$$

The direct creation of the formula matrix is a much more intuitive approach. The element balance equations represents the immutable basis for the description of any reacting system. Hence the starting point is to describe a reactive system at complete chemical equilibrium.

To obtain $\underset{\approx}{R}$, the element balance equations based on the chemical formulae of the involved species are established and row-reduced to a linearly independent set. For a simplified urea-synthesis system (CO_2, NH_3, H_2O, NH_2COONH_4, NH_2CONH_2), the balance equations for C, N, O and H yield

$$\underset{\approx}{R}_{\mathsf{raw}} = \begin{pmatrix} 1 & & & 1 & 1 \\ & 1 & & 2 & 2 \\ 2 & & 1 & 2 & 1 \\ & 3 & 2 & 6 & 4 \end{pmatrix}. \quad \begin{array}{l} \leftarrow \mathrm{C} \\ \leftarrow \mathrm{N} \\ \leftarrow \mathrm{O} \\ \leftarrow \mathrm{H} \end{array} \tag{3.47}$$

In this case, the element balance equations yield a rank-deficient matrix $\underset{\approx}{R}_{\mathsf{raw}}$. Using this matrix directly in Equation (3.46) and subsequently in Equation (3.22) yields a singular equation system.

The row-reduced matrix $\underset{\approx}{R}$ indicates that only inert groups CO_2, NH_3 and H_2O are recombining, not the four elements in general:

$$\underset{\approx}{R} = \begin{pmatrix} 1 & & & 1 & 1 \\ & 1 & & 2 & 2 \\ & & 1 & & -1 \end{pmatrix} . \quad \begin{array}{l} \leftarrow CO_2 \\ \leftarrow NH_3 \\ \leftarrow H_2O \end{array} \tag{3.48}$$

Section 3.2.2 describes the general calculation of chemical equilibria. Depending on the number of phases, Equation (3.17) or Equation (3.19) is used. The following paragraphs describe the approach to obtain suitable formula matrices $\underset{\approx}{R}_{(i)}$ considering inert and restricted species, and phases with different sets of species.

Multi-phase reactor

The simplest multi-phase reactor considers equal sets of species in all phases. The balance equations can be formulated with one common formula matrix $\underset{\approx}{R}$ for all phases:

$$\underset{\approx}{R}_{\mathsf{in}}\, \underset{\sim}{\dot{n}}_{\mathsf{in}} - \underset{\approx}{R} \sum_i \underset{\sim}{\dot{n}}_i = \underset{\sim}{0} \, . \tag{3.49}$$

In practice, being able to assign different sets of chemical species to individual phases can drastically improve the robustness of the solution method and avoid phase stability problems. If the balance equations force a certain species into a particular phase, the existence of this phase is assured. Furthermore, most likely, there might not be a thermodynamic model for all species in all phases. Ions and species with high molecular weight will not occur in the vapour phase, and no considerable amounts of light gases might be dissolved in the liquid phase. Eliminating species from particular phases also decreases the size of the canonical equation system.

As an example, consider the set of balance equations for the urea synthesis as in the previous section, but including N_2 to represent the passivation air (see Figure 3.1). During the vapour liquid equilibrium calculations, N_2 is only considered in the vapour phase, while ammonium carbamate (subscripted as ψ_{carb}) and urea are restricted to the liquid phase. In order to apply Equation (3.19) for this system, the matrices $\underset{\approx}{R}_{(\mathrm{l})}$ and $\underset{\approx}{R}_{(\mathrm{v})}$ are defined through the balance equations of elements. For a feed consisting of CO_2, NH_3, H_2O, and N_2, the element balances are

$$\begin{pmatrix} \dot{n}_{\mathrm{C}} \\ \dot{n}_{\mathrm{N}} \\ \dot{n}_{\mathrm{H}} \\ \dot{n}_{\mathrm{O}} \end{pmatrix}_{\mathsf{in}} = \begin{pmatrix} 1 & & & 1 & 1 \\ & 1 & & 2 & 2 \\ & 3 & 2 & 6 & 4 \\ 2 & & 1 & 2 & 1 \end{pmatrix} \begin{pmatrix} \dot{n}_{\mathrm{CO_2}} \\ \dot{n}_{\mathrm{NH_3}} \\ \dot{n}_{\mathrm{H_2O}} \\ \dot{n}_{\mathrm{carb}} \\ \dot{n}_{\mathrm{urea}} \end{pmatrix}_{(\mathrm{l})} + \begin{pmatrix} 1 & & & \\ & 1 & & 2 \\ & 3 & 2 & \\ 2 & & 1 & \end{pmatrix} \cdot \begin{pmatrix} \dot{n}_{\mathrm{CO_2}} \\ \dot{n}_{\mathrm{NH_3}} \\ \dot{n}_{\mathrm{H_2O}} \\ \dot{n}_{\mathrm{N_2}} \end{pmatrix}_{(\mathrm{v})} . \tag{3.50}$$

Note that these constraints not only force N_2 into the vapour phase, but also disallow any chemical reaction including this species. As a consequence of the element balance equations, N_2 is an inert species in this system.

Inert and stoichiometrically restricted chemical species

Reacting systems often contain inert chemical species. In some cases, as for nitrogen in the previous example, this is a consequence of element balance equations.

In other cases, the inertness of a species is a consequence of thermodynamic prohibition, and it needs to be specified by an explicit modification of the formula matrix. This is performed by adding a balance equation to conserve the quantity of the inert species to the reaction matrix.

Considering only element balance constraints, a gas phase containing CO_2, NH_3, H_2O, N_2, and O_2 allows for the oxidation of ammonia:

$$2\,NH_3 + 3\,O_2 \ \leftrightarrows\ 2\,N_2 + 6\,H_2O\,. \tag{3.51}$$

To exclude this reaction, an additional constraint equation to conserve N_2 is added to the formula matrix, represented by the last row of $\underset{\approx}{R}$.

$$\begin{pmatrix} \dot{n}_C \\ \dot{n}_N \\ \dot{n}_H \\ \dot{n}_O \\ \dot{n}_{N_2} \end{pmatrix}_{\text{in}} = \begin{pmatrix} 1 & & & & \\ & 1 & & 2 & \\ & 3 & 2 & & \\ 2 & & 1 & & 2 \\ & & & 1 & \end{pmatrix} \cdot \begin{pmatrix} \dot{n}_{CO_2} \\ \dot{n}_{NH_3} \\ \dot{n}_{H_2O} \\ \dot{n}_{N_2} \\ \dot{n}_{O_2} \end{pmatrix} . \tag{3.52}$$

The same modification is done to the formula matrix $\underset{\approx}{R}$, if a species is reactive, but not supposed to achieve chemical equilibrium. In this case, the species balance equation is added to the formula matrix, but subsequently released (see Section 3.3.2). The DOF can be used for any kind of constitutive equation, specifying for instance the extent of reaction as a direct specification or as an empirical correlation, such as a description of the reaction kinetics.

To complete the urea synthesis example, the byproduct biuret is included into the reaction system (see Equation (3.44)). A species balance equation is added and released, such that a constitutive equation to describe the reaction kinetics can be applied. Equation (3.50) is extended to

$$\begin{pmatrix} \dot{n}_C \\ \dot{n}_N \\ \dot{n}_H \\ \dot{n}_O \\ \dot{n}_{N_2} \\ \dot{n}_{\text{biuret}} \end{pmatrix}_{\text{in}} = \begin{pmatrix} 1 & & & 1 & 1 & 2 \\ & 1 & & 2 & 2 & 3 \\ & 3 & 2 & 6 & 4 & 5 \\ 2 & & 1 & 2 & 1 & 2 \\ & & & & & \\ & & & & & 1 \end{pmatrix} \begin{pmatrix} \dot{n}_{CO_2} \\ \dot{n}_{NH_3} \\ \dot{n}_{H_2O} \\ \dot{n}_{\text{carb}} \\ \dot{n}_{\text{urea}} \\ \dot{n}_{\text{biuret}} \end{pmatrix}_{(\text{l})} + \begin{pmatrix} 1 & & & & \\ & 1 & & 2 & \\ & 3 & 2 & & \\ 2 & & 1 & & 2 \\ & & & 1 & \\ & & & & \end{pmatrix} \cdot \begin{pmatrix} \dot{n}_{CO_2} \\ \dot{n}_{NH_3} \\ \dot{n}_{H_2O} \\ \dot{n}_{N_2} \\ \dot{n}_{O_2} \end{pmatrix}_{(\text{v})} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \alpha \end{pmatrix} . \tag{3.53}$$

The constitutive equation to determine the extent of biuret formation can be defined by a temperature and concentration dependent reaction rate:

$$\dot{n}_{\text{biuret},(\text{l})} - \dot{n}_{\text{biuret,in}} = \psi\left(T, \frac{\dot{\underset{\sim}{n}}_{(\text{l})}}{\dot{V}_{(\text{l})}}\right) . \tag{3.54}$$

This example illustrates the canonical approach to define a consistent reactive system including species in chemical and phase equilibrium, inert species, and kinetically restricted reactions. The required input to define such a system is minimal and in particular not redundant to the material conservation constraints.

Species categories in a reactive system

Considering the element balance equations and additional constraints to the reactive system, all species involved can easily be categorised into different groups as indicated in Figure 3.3. The arrows indicate the possible transitions of the species from one group into another. Transitions along solid arrows are triggered by the modelling engineer, while transitions along dashed arrows are a consequence of this. Figure 3.3

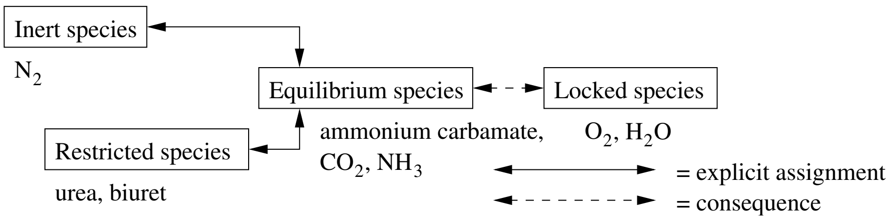


Figure 3.3: *Groups of chemical species in a reactive system.*

includes the species of the urea synthesis example in the previous section. In this configuration, the urea synthesis reaction (3.43) is restricted by a kinetic expression, as it is suitable to describe the chemistry of the carbamate condenser (see Figure 3.1).

The formula matrix $\underset{\approx}{R}$ determines the affiliation of each species to a specific group. This is done by analysis of the null space $\underset{\approx}{N}$ of $\underset{\approx}{R}$. $\underset{\approx}{N}$ is the stoichiometric matrix of the system, as the rows of $\underset{\approx}{N}$ are a set of linear independent stoichiometric vectors $\underset{\sim}{\nu}_i$ of all enabled reactions.

The following definitions ensure a consistent description of any reacting system. No input information redundant to the element balance equations is required.

Terms and definitions 3.2

Equilibrium species are species i with $\underset{\approx}{N}\,\underset{\sim}{e}_i \neq \underset{\sim}{0}$, thus species i is included into at least one reaction. Species of this group can be explicitly changed to *inert* or *restricted* species. Doing this will possibly trigger other species of this group to be *stoichiometrically locked.*

Inert species are defined by a particular species balance equation in $\underset{\approx}{A}$. If $\underset{\approx}{\hat{A}}$ is a system containing an equilibrium species i, the species balance will not be linearly dependent on the original rows in $\hat{\underset{\approx}{A}}$, as the null space is definitely reduced by one dimension.

Restricted species are generated in the same way as the *inert species*, just that the species balance equation is released in favour of a constitutive equation as described in section 3.3.2. This constitutive equation describes the relationship between the formation of a key species and operation conditions in terms of canonical and conjugated state variables.

Specifications of kinetic reactions also belong to this class, as the reaction rate is not more than a function of operating conditions.

Locked species are species i with $\underset{\approx}{N}\,\underset{\sim}{e}_i = \underset{\sim}{0}$, which are not key species in definitions of *inert* or *restricted* species. These species cannot actively be reassigned to another group. The affiliation to this group is a consequence of other balance equations.

Figure 3.3 contains the assignment for the urea synthesis example. To define the reactions as introduced in the beginning of this section, N_2 is defined as inert, biuret and urea as restricted. With this, ammonium carbamate is still in equilibrium with NH_3 and CO_2, but O_2 and H_2O are locked. Here, O_2 happens to be inert, while the amount of H_2O follows the synthesis reaction defined with urea as key component, influenced by the formation of biuret.

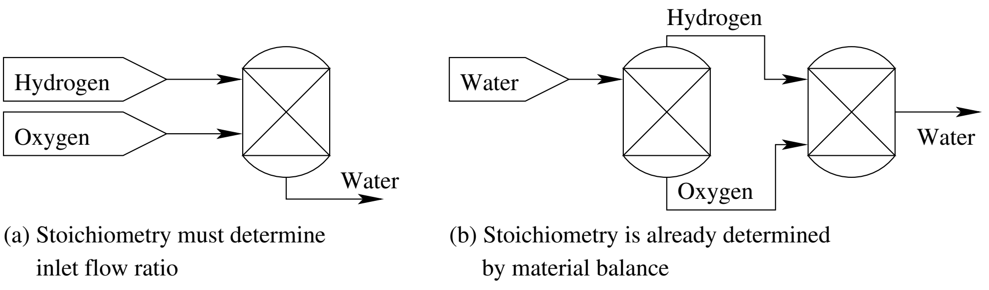
Complete conversion reactor

In some cases, one would like to disregard some of the reactants in the product stream. These species are assumed to disintegrate to full extent. Though thermodynamically not motivated, this approach is practical for avoiding large species sets in the downstream sections. A typical example is the combustion of hydrocarbon fuel. In ordinary cases, only CO_2, H_2O, excess O_2, and maybe CO (and CH_4 for very precise calculations or reducing conditions) are considered as product species. The concentrations of heavier hydrocarbons in the tail-gas are negligible. Any number of non-product input species can be included, if the constraint vector $\underset{\sim}{a}_{\mathsf{in}}$, which maps this species to the set of balance equations, does not increase the rank of $\underset{\approx}{R}$. Consider H_2 and O_2 reacting to H_2O, leaving excess O_2 but no H_2. The element balances yield the formula matrices for the incoming stream $\underset{\approx}{R}_{\mathsf{in}}$ and the reactor outlet $\underset{\approx}{R}$:

$$\underset{\approx}{R}_{\mathsf{in}} = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \quad \text{and} \quad \underset{\approx}{R} = \begin{pmatrix} 2 & 0 \\ 1 & 2 \end{pmatrix} . \tag{3.55}$$

If the excess oxygen is of a negative amount, the molar flow vector with positive and negative entries is most likely not covered by the mathematical domain of the thermodynamic model. In this case, the system is generally not solvable.

However, considering only H_2O in the product stream causes structural problems. In this case, $\underset{\approx}{A}$ cannot be row-reduced without disregarding information in $\underset{\approx}{A}_{\mathsf{in}}$, hence the remaining $\underset{\approx}{A}$ is not of full rank and the FM is not solvable. If an exact stoichiometric match is desired, this represents an additional constraint, which has to be associated with a DOF elsewhere in the process model. To achieve this, inconsistent rows of the balance equation set can be removed from $\underset{\approx}{A}$ and provided as an ordinary constitutive equation. However, use of this equation might still be redundant to other material balance equations.



(a) Stoichiometry must determine inlet flow ratio

(b) Stoichiometry is already determined by material balance

Figure 3.4: *Process with and without redundant stoichiometric constitutive equation.*

Figure 3.4 shows the two different cases. In (a), the reactor module can only ensure the preservation of either hydrogen or oxygen. A constitutive equation representing the other element balance must be used to determine the flow ratio between the inlet streams. Case (b), however, provides hydrogen and oxygen at the correct stoichiometric ratio. One of the element balances determines the outlet flow of water, while the other is linearly dependent on the material balance equations of the first reactor.

3.5.4 Chemical species separator

A first principle separator is a general FM for separating the incoming material stream into two individual outlet streams, only constrained by the total material balance equations.

The modelling approach is based on a restricted two-phase equilibrium calculation. By adding one more constraint for each species that is common to both phases, the phase split can be fully controlled independently of the thermodynamic model. Subsequently, there are four groups of constraints: (i) balance equations for canonical variables common to both phases, (ii) + (iii) balance equations for canonical variables occurring only in the first or second phase respectively, and (iv) constraints on splitting behaviour for canonical variables common to both phases. The constraints of group (iv) serve as DOFs for constitutive equations.

Instead of distinguishing the four groups as such, and handling the combinatorial number of combinations, the categorisation can be simplified: (i) all canonical variables of one phase, and (ii) all canonical variables of the other phase. A constraint matrix $\underset{\approx}{A}$ completes the formulation of balance equations:

$$\begin{pmatrix} \underset{\sim}{\dot{x}}_1 \\ \underset{\sim}{\dot{x}}_2 \end{pmatrix}_{\mathsf{in}} = \begin{pmatrix} \underset{\approx}{I} & \underset{\approx}{A} \\ & \underset{\approx}{I} \end{pmatrix} \cdot \begin{pmatrix} \underset{\sim}{\dot{x}}_1 \\ \underset{\sim}{\dot{x}}_2 \end{pmatrix} + \begin{pmatrix} \underset{\sim}{0} \\ \underset{\sim}{\alpha} \end{pmatrix} \quad \text{with} \quad a_{ij} = \begin{cases} 1 & \text{variable } i \text{ in phase 1 is} \\ & \text{same as } j \text{ in phase 2;} \\ 0 & \text{variables are not identical.} \end{cases} \tag{3.56}$$

Note that a molar flow of an incoming species occurs twice on the left hand side if that species is considered in both outlet streams. For each non-zero a_{ij}, the equation in the first row represents a real balance equation. The corresponding equation in the fourth row is therefore released by α_j to represent a DOF. The belonging constitutive equation specifies the split factor of the state variable j between both output ports. Such state variables include the first two elements of $\underset{\sim}{x}$, for instance S and V. These two DOFs are typically used to specify a correlation between the temperatures and pressures in the outlet streams.

A local optimisation node is defined to obtain the intensive properties through the canonical equation system. The objective function

$$\Lambda(\underset{\sim}{\dot{x}}_1, \underset{\sim}{\dot{x}}_2) = P_1(\underset{\sim}{\dot{x}}_1) + P_2(\underset{\sim}{\dot{x}}_2) - \begin{pmatrix} \underset{\sim}{\lambda}_1 \\ \underset{\sim}{\lambda}_2 \end{pmatrix} \left[\begin{pmatrix} \underset{\sim}{\dot{x}}_1 \\ \underset{\sim}{\dot{x}}_2 \end{pmatrix}_{\mathsf{in}} - \begin{pmatrix} \underset{\approx}{I} & \underset{\approx}{A} \\ & \underset{\approx}{I} \end{pmatrix} \cdot \begin{pmatrix} \underset{\sim}{\dot{x}}_1 \\ \underset{\sim}{\dot{x}}_2 \end{pmatrix} + \begin{pmatrix} \underset{\sim}{0} \\ \underset{\sim}{\alpha} \end{pmatrix} \right] \tag{3.57}$$

yields the following canonical system:

$$\begin{pmatrix} \underset{\approx}{H}_1 & & \underset{\approx}{I} & \\ & \underset{\approx}{H}_2 & \underset{\approx}{A}^{\mathrm{T}} & \underset{\approx}{I} \\ \underset{\approx}{I} & \underset{\approx}{A} & & \\ & \underset{\approx}{I} & & \end{pmatrix} \begin{pmatrix} \Delta\underset{\sim}{\dot{x}}_1 \\ \Delta\underset{\sim}{\dot{x}}_2 \\ \underset{\sim}{\lambda}_1 \\ \underset{\sim}{\lambda}_2 \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g}_1 \\ -\underset{\sim}{g}_2 \\ \underset{\sim}{\delta}_1 \\ \underset{\sim}{\delta}_2 + \underset{\sim}{\alpha} \end{pmatrix} \tag{3.58}$$

There are sufficient identity matrix blocks to solve this system by back-substitution:

$$\underset{\sim}{\dot{x}}_2 = \underset{\sim}{\dot{x}}_{2,\mathsf{in}} + \underset{\sim}{\alpha}\,, \quad \underset{\sim}{\dot{x}}_1 = \underset{\sim}{\dot{x}}_{1,\mathsf{in}} - \underset{\approx}{A}\,\underset{\sim}{\dot{x}}_2 \quad \underset{\sim}{\lambda}_1 = -\underset{\sim}{g}_1\,, \text{ and } \underset{\sim}{\lambda}_2 = \underset{\approx}{A}^{\mathrm{T}}\,\underset{\sim}{g}_1 - \underset{\sim}{g}_2\,. \tag{3.59}$$

3.5.5 Flow splitter

A flow splitter is a special case of the first principle separator. The outlet flows contain identical sets of chemical species and are described by the same thermodynamic model. Their intensive states are equal, and their extensive states only differ by a scaling factor. As with the two-phase non-reacting equilibrium (see Section 3.5.2), the L -multipliers are shared between both phases. However, the flow splitter seeks the trivial solution, such that $H_{(1)} + H_{(2)}$ is singular. To solve the system, one additional constraint, which provides the DOF to hold one more constitutive equation, is necessary. This DOF is used to determine the split ratio of the outlet flows.

The L function given in Equation (3.2) for a two-phase flash is modified to

$$\Lambda(\underset{\sim}{\dot{x}}_{(1)}, \underset{\sim}{\dot{x}}_{(2)}, \underset{\sim}{\lambda}, \lambda_{\mathsf{split}}) = P_{(1)} + P_{(2)} - \underset{\sim}{\lambda}\,(\underset{\sim}{\dot{x}}_{\mathsf{in}} - \underset{\sim}{\dot{x}}_{(1)} - \underset{\sim}{\dot{x}}_{(2)}) - \lambda_{\mathsf{split}}\,\underset{\sim}{a}\,(\underset{\sim}{\dot{x}}_{(1)} - \underset{\sim}{\dot{x}}_{(2)})\,. \tag{3.60}$$

The constraint vector $\underset{\sim}{a}$ can be an arbitrary non-zero vector, but in order to avoid rank loss not orthogonal to any feasible state $\underset{\sim}{\dot{x}}$. Application of the N -R method yields the following equation system:

$$\left(\begin{array}{c:cc:c} \underset{\approx}{H}_{(1)} & & -\underset{\sim}{a} & \underset{\approx}{I} \\ \hdashline & \underset{\approx}{H}_{(2)} & \underset{\sim}{a} & \underset{\approx}{I} \\ -\underset{\sim}{a} & \underset{\sim}{a} & & \underset{\sim}{0} \\ \hdashline \underset{\approx}{I} & \underset{\approx}{I} & \underset{\sim}{0} & \end{array}\right) \left(\begin{array}{c} \Delta\underset{\sim}{\dot{x}}_{(1)} \\ \hdashline \Delta\underset{\sim}{\dot{x}}_{(2)} \\ \lambda_{\text{split}} \\ \hdashline \underset{\sim}{\lambda} \end{array}\right) = \left(\begin{array}{c} -\underset{\sim}{g}_{(1)} \\ \hdashline -\underset{\sim}{g}_{(2)} \\ \delta_{\text{split}} + \alpha_{\text{split}} \\ \hdashline \underset{\sim}{\delta} + \underset{\sim}{\alpha} \end{array}\right) . \tag{3.61}$$

At the solution point, λ_{split} is zero due to the homogeneity of state functions. Still, with suitable starting values, the flow splitter can be triggered to predict a restricted actual phase equilibrium and solve for a non-trivial solution. In this case, $\lambda_{\text{split}} \neq 0$, and $\underset{\sim}{a}$ defines the direction, in which the equilibrium constraints are violated in order to fulfil the third constraint. This case is of little practical value and can easily be avoided by choosing starting values suitable to favour the trivial solution. Possible applications like membranes and partial equilibria are better implemented using the first principle separator described in the previous section.

3.5.6 Saturation node

The calculation of a state vector at the exact phase boundary, namely the boiling point or the saturation point, is often desirable. Experimental data as a basis to adjust and validate thermodynamic models is often available at such saturated conditions. A process model of a heat exchanger needs to identify the saturation point in order to consider changes in the heat transfer characteristics. An ordinary flash calculation with a specified vapour fraction close to 0 or 1 is practically feasible, but not appealing for numerical reasons caused by extremely different scales of the state vectors.

The saturation node contains a main phase and a trial phase. The main phase is constrained by a complete set of canonical constraints, of which one must be made available as a DOF to find the saturation point. Figure 3.5 shows an intersection plane of the multidimensional state space with the state functions of the trial and the main phase. Due to the homogeneity of thermodynamic state functions (see Section 2.5.1), the tangent plane to the state function is in any point $\underset{\sim}{\dot{x}}_0$ defined as the scalar product $\underset{\sim}{\dot{x}}\,\underset{\sim}{g}_0$. As necessary requirement for equilibrium, the tangent planes must coincide. The distance between the tangent planes is defined as

$$\Delta P = P(\underset{\sim}{\dot{x}}_{\text{trial}}) - \underset{\sim}{\dot{x}}_{\text{trial}}\,\underset{\sim}{g}_{\text{main}} = (\underset{\sim}{g}_{\text{trial}} - \underset{\sim}{g}_{\text{main}})\,\underset{\sim}{\dot{x}}_{\text{trial}} \, . \tag{3.62}$$

The minimisation of ΔP at constant $\underset{\sim}{g}_{\text{main}}$ must be subject to at least one constraint in extensive variables to determine the trial phase size: $\underset{\sim}{a}\,\underset{\sim}{\dot{x}} = b$. The formulation of the L -function is

$$\Lambda = P(\underset{\sim}{\dot{x}}_{\text{trial}}) - \underset{\sim}{\dot{x}}_{\text{trial}}\,\underset{\sim}{g}_{\text{main}} - \lambda_{\text{trial}}(b - \underset{\sim}{a}\,\underset{\sim}{\dot{x}}_{\text{trial}}) \tag{3.63}$$

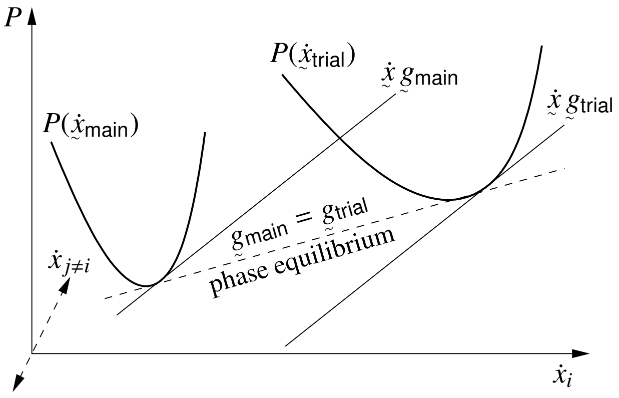

P
$P(\dot{\underset{\sim}{x}}_{\text{trial}})$
$\dot{\underset{\sim}{x}}\,\underset{\sim}{g}_{\text{main}}$
$\dot{\underset{\sim}{x}}\,\underset{\sim}{g}_{\text{trial}}$
$P(\dot{\underset{\sim}{x}}_{\text{main}})$
$\underset{\sim}{g}_{\text{main}} = \underset{\sim}{g}_{\text{trial}}$
phase equilibrium
$\dot{x}_{j\neq i}$
$\dot{x}_i$

Figure 3.5: *Tangent planes of the state functions of two phases in an intersecting plane ($\dot{x}_{j\neq i}$ = const.).*

with

$$\frac{\partial \Lambda}{\partial \dot{\underset{\sim}{x}}_{\text{trial}}} = \underset{\sim}{g}_{\text{trial}} - \underset{\sim}{g}_{\text{main}} - \lambda_{\text{trial}}\,\underset{\sim}{a}\ . \tag{3.64}$$

Thus at the solution point, the difference of the gradient vectors points into the direction of $\underset{\sim}{a}$, and λ_{trial} is a measure for the distance. A constitutive equation to specify $\lambda_{\text{trial}} = 0$ is required to obtain the equilibrium condition $\underset{\sim}{g}_{\text{trial}} = \underset{\sim}{g}_{\text{main}}$. The DOF to host this equation is provided by the main node.

The canonical equation system to obtain a N update is

$$\left(\begin{array}{cc:cc} \underset{\approx}{H}_{\text{main}} & \underset{\approx}{I} & & \\ \underset{\approx}{I} & & & \\ \hdashline & \underset{\approx}{I} & \underset{\approx}{H}_{\text{trial}} & \underset{\sim}{a} \\ -\underset{\sim}{a} & & \underset{\sim}{a} & \end{array}\right) \cdot \left(\begin{array}{c} \Delta\dot{\underset{\sim}{x}}_{\text{main}} \\ \lambda_{\text{main}} \\ \hdashline \Delta\dot{\underset{\sim}{x}}_{\text{trial}} \\ \lambda_{\text{trial}} \end{array}\right) = \left(\begin{array}{c} -\underset{\sim}{g}_{\text{main}} \\ \underset{\sim}{\delta}_{\text{main}} + \underset{\sim}{\alpha} \\ \hdashline -\underset{\sim}{g}_{\text{trial}} \\ \delta_{\text{trial}} \end{array}\right) . \tag{3.65}$$

In this case, the constraint vector $\underset{\sim}{a}$ ensures a similar size of the main and trial phases. In order to avoid a singularity caused by the homogeneity property of the state function in the trial system, $\underset{\sim}{a}$ must not be orthogonal to any feasible state vector.

3.6 Specialised composite flowsheet modules

To minimise low-level maintenance work, a main objective of the modelling concept is to keep the number of atomic flowsheet modules as small as possible. The set introduced in the previous section serves as the basis for a more extensive set of composite flowsheet modules to represent heat exchangers, membranes, turbines, columns and other more complex process equipment.

3.6.1 Limited heat and mass transfer

Process equipment is commonly operated on the edge of its capacity limits. It is therefore rarely possible to obtain an accurate model by assuming complete phase equilibrium. Furthermore, modern process designs apply membranes more frequently as an alternative to thermal separation. The exchange of heat and material is described by gradients of the intensive variables, such as temperature, concentration, and chemical potential.

The approach uses a first principle separator as shown in Figure 3.6. The valves *V1* and *V2* represent trivial modules, which conserve the state and provide the composite flowsheet module with extensive and intensive properties of the two distinct input streams. The separator provides all the properties of the outgoing streams and a sufficient number of DOFs to redistribute each extensive state variable independently. The heat and mass transfer can be described as a set of constitutive equations including the extensive and intensive properties of the incoming and outgoing streams.

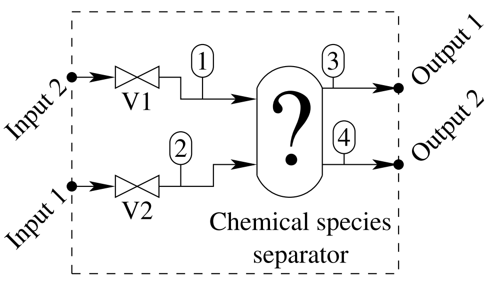


Figure 3.6: *Layout of a general diffusion module as a composite FM.*

The canonical equation system represents the FM as a very general composite implementation:

$$\left(\begin{array}{cc|cc|cccc}
\underset{\approx}{H}_1 & \underset{\approx}{I} & & & & & & \\
\underset{\approx}{I} & & & & & & & \\
\hline
 & & \underset{\approx}{H}_2 & \underset{\approx}{I} & & & & \\
 & & \underset{\approx}{I} & & & & & \\
\hline
 & & & & \underset{\approx}{H}_3 & & \underset{\approx}{I} & \\
 & & & & & \underset{\approx}{H}_4 & \underset{\approx}{A}^{\mathrm{T}} & \underset{\approx}{I} \\
-\underset{\approx}{I} & & -\underset{\approx}{A} & & \underset{\approx}{I} & \underset{\approx}{A} & & \\
 & & -\underset{\approx}{I} & & & \underset{\approx}{I} & &
\end{array}\right) \cdot \left(\begin{array}{c}
\Delta\dot{\underset{\sim}{x}}_1 \\
\underset{\sim}{\lambda}_1 \\
\hline
\Delta\dot{\underset{\sim}{x}}_2 \\
\underset{\sim}{\lambda}_2 \\
\hline
\Delta\dot{\underset{\sim}{x}}_3 \\
\Delta\dot{\underset{\sim}{x}}_4 \\
\underset{\sim}{\lambda}_3 \\
\underset{\sim}{\lambda}_4
\end{array}\right) = \left(\begin{array}{c}
-\underset{\sim}{g}_1 \\
\underset{\sim}{\delta}_1 \\
\hline
-\underset{\sim}{g}_2 \\
\underset{\sim}{\delta}_2 \\
\hline
-\underset{\sim}{g}_3 \\
-\underset{\sim}{g}_4 \\
\hat{\underset{\sim}{\delta}}_3 + \underset{\sim}{\alpha}_3 \\
\hat{\underset{\sim}{\delta}}_4 + \underset{\sim}{\alpha}_4
\end{array}\right) . \qquad (3.66)$$

This equation system is composed of the atomic FMs described in Section 3.5.1 (one-phase module) and Section 3.5.4 (chemical species separator). The off-diagonal blocks represent the material couplings as described in Section 3.4.3. The indices of state vectors are consistent with the stream numbers in Figure 3.6.

An identity matrix can be found for every pivot element, such that the canonical system is a set of explicit equations to determine the state. It provides thermodynamic properties and their derivatives in order to formulate and solve the constitutive equations. These freely configurable constitutive equations determine the entire heat and mass transfer.

3.6.2 Rotating process equipment

Isentropic efficiency modules

The performance of rotating process equipment, such as turbines and compressors, is often characterised by reference to a reversible change in state, like the isentropic efficiency. The ideal module is constrained by an entropy conservation equation and yields the reversible work W_{rev}. The actual work duty or delivery is obtained by multiplication with (turbine) or division by (compressor) a specified efficiency η. The surplus energy is added as process heat at constant pressure. Figure 3.7 shows

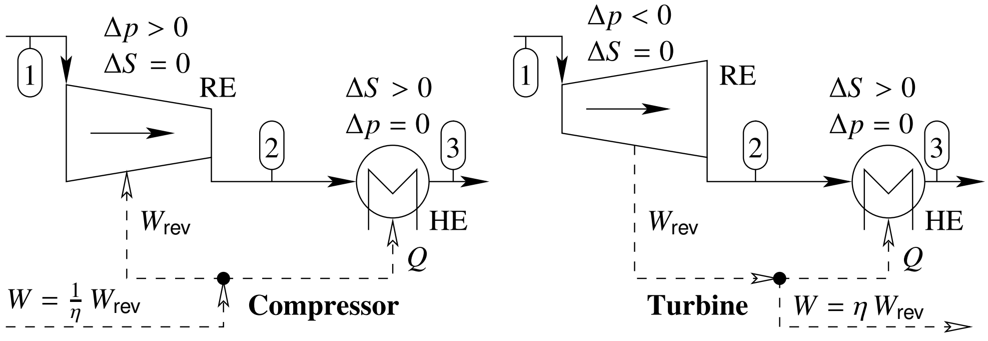


Figure 3.7: *Process models of rotation equipment with isentropic efficiency.*

the approach to incorporate isentropic efficiency into a composite FM. The split into a reversible and an irreversible part is directly reflected in the process topology. As indicated in Figure 3.7, constitutive equations describe the relationship between the reversible and the irreversible processes. With $W = \eta W_{\mathrm{rev}}$, $\Delta p_{\mathrm{HE}} = 0$, and $\Delta S_{\mathrm{RE}} = 0$, the remaining of four DOFs can be used to specify outlet pressure, work load, delivery, or outlet temperature. The canonical system for both FMs is again just a framework to provide thermodynamic properties for the three states, namely the inlet, the reversible state, and the outlet state, each of them calculated by a one-phase module as described in Section 3.5.1. The indices of the state vectors are consistent with the stream numbers in Figure 3.7. The equation system is

$$\left(\begin{array}{cc:cc:cc} \underset{\approx}{H}_1 & \underset{\approx}{I} & & & & \\ \underset{\approx}{I} & & & & & \\ \hdashline & & \underset{\approx}{H}_2 & \underset{\approx}{I} & & \\ -\underset{\approx}{I} & & \underset{\approx}{I} & & & \\ \hdashline & & & & \underset{\approx}{H}_3 & \underset{\approx}{I} \\ & & -\underset{\approx}{I} & & \underset{\approx}{I} & \end{array}\right) \cdot \left(\begin{array}{c} \Delta\underset{\sim}{\dot{x}}_1 \\ \underset{\sim}{\lambda}_1 \\ \hdashline \Delta\underset{\sim}{\dot{x}}_2 \\ \underset{\sim}{\lambda}_2 \\ \hdashline \Delta\underset{\sim}{\dot{x}}_3 \\ \underset{\sim}{\lambda}_3 \end{array}\right) = \left(\begin{array}{c} -\underset{\sim}{g}_1 \\ \underset{\sim}{\delta}_1 \\ \hdashline -\underset{\sim}{g}_2 \\ \hat{\underset{\sim}{\delta}}_2 + \underset{\sim}{\alpha}_2 \\ \hdashline -\underset{\sim}{g}_3 \\ \hat{\underset{\sim}{\delta}}_3 + \underset{\sim}{\alpha}_3 \end{array}\right) . \tag{3.67}$$

Identity blocks can be found for each pivot block in the coefficient matrix of the following equation system:

A precarious issue is the definition of η. Theoretically, the efficiency is defined as above, but practically this makes η depend on the thermodynamic model used to calculate on the equipment. Performance data of compressors is often reported in terms of an efficiency under the assumption of an ideal gas. The process model will therefore give deviating results, if the the simulation is carried out based on a more sophisticated thermodynamic model, such as a cubic equation of state. Even more significant discrepancies might occur, if the saturation line is crossed within a turbine.

Polytropic efficiency

The efficiency of compressors is often reported in terms of a polytropic efficiency (Campbell, 1984):

$$\eta_{\mathrm{poly}} = \frac{\kappa - 1}{\kappa} \frac{\ln p_{\mathrm{out}}/p_{\mathrm{in}}}{\ln T_{\mathrm{out}}/T_{\mathrm{in}}} \quad \text{with} \quad \kappa = \frac{c_p}{c_V}\,. \tag{3.68}$$

Clearly, this formulation is based on the assumption of ideal gas behaviour and constant heat capacity. Therefore, $\eta_{\mathrm{poly}} = 1$ is not equivalent to the reversible process in general. Furthermore, the adiabatic exponent κ is not the actual property of the gas within the compressor, but the value for κ published together with the efficiency data, related to the medium the compressor is designed for. In spite of its inconsistent definition, the polytropic efficiency is widely used to characterise rotating equipment. This justifies the integration of this property as a constitutive equation, but clearly as an empirical measure, in particular not coupled with respect to the adiabatic exponent calculated by the underlying thermodynamic model.

3.6.3 Valves

A valve in this context is defined as a general isenthalpic process equipment. Without exchange of heat and work with the environment, irreversible effects generally cause a pressure drop. Purely descriptive valve models directly specify the outlet pressure or a constant pressure drop throughout the valve. The following paragraphs describe predictive approaches for incompressible and compressible fluids. In both cases, only full turbulent flows are considered, and effects due to variable flow pattern at low R numbers are not discussed. However, the constitutive equations can be refined to describe the dependency of effective cross-section with respect to the R number.

Incompressible fluids

If the fluid density does not change significantly, the assumption of an ideal diffuser yields a single constitutive equation, which can be used in combination with enthalpy conservation within a one-phase non-reacting module (cf. Section 3.5.1):

$$p_{\text{in}} - p_{\text{out}} = \frac{1}{2}\varrho v^2 . \tag{3.69}$$

The velocity v at the most narrow cross-section F is given by the linear relation $F v = \dot{V}$. The cross-section might be modelled as a function of the valve position z: $F = F_{\text{open}}\,\psi(z)$ with $z, \psi \in [0, 1]$.

Compressible fluids

As a basis for a valve model with compressible fluids, the canonical system is identical to those of compressors and turbines, shown in Equation (3.67). The decomposition is trivial, and calculated properties serve as input for constitutive equations.

With increasing pressure drop, the compressibility of the gas has an increasing impact on the decrease of density in the the most narrow cross-section (Smith *et al.*, 2001). Furthermore, the fluid's change of kinetic energy causes a temporary decrease of thermodynamic enthalpy, lowering the fluid temperature in the nozzle. More precise pressure flow relations are obtained by introducing a node to represent the nozzle. Assuming isentropic flow up to this point, the four DOFs are specified by those equations marked by $\star$ in Figure 3.8. As in the simplified incompressible model, velocity

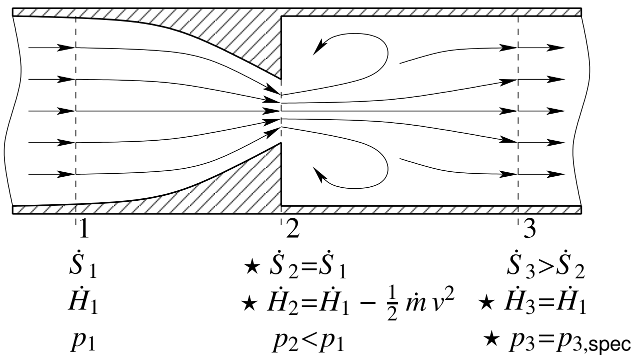

1
2
3
$\dot{S}_1$
$\star\ \dot{S}_2 = \dot{S}_1$
$\dot{S}_3 > \dot{S}_2$
$\dot{H}_1$
$\star\ \dot{H}_2 = \dot{H}_1 - \frac{1}{2}\dot{m}\,v^2$
$\star\ \dot{H}_3 = \dot{H}_1$
p_1
$p_2 < p_1$
$\star\ p_3 = p_{3,\text{spec}}$

Figure 3.8: *Detailed model of a valve containing a compressible medium.*

is a function of volume flow and cross-section. However, a pressure-flow relation is not yet established, and two cases have to be distinguished. For moderate pressure drop, v is below sonic velocity v_{sonic}, in which case $p_2 = p_3$ is an active constraint. A compression shock takes place, if the assumption of equal pressures roughly yields a velocity $v > v_{\text{sonic}}$. In such a case, p_2 and p_3 are uncorrelated with $p_2 > p_3$. The constraint $p_2 = p_3$ is replaced by $v = v_{\text{sonic}}$ to describe the sonic flow.

Speed of sound is a thermodynamic property and more precisely a function of some second-order derivatives of the canonical state function (Perry and Green, 1997):

$$v_{\text{sonic}}^2 = \left.\frac{\partial p}{\partial \varrho}\right|_S = \left.\frac{\partial p}{\partial \dot{V}}\right|_{\dot{S}} \cdot \left(\left.\frac{\partial \varrho}{\partial \dot{V}}\right|_{\dot{S}}\right)^{-1} = -\frac{\dot{V}}{\varrho}\left.\frac{\partial p}{\partial \dot{V}}\right|_S . \tag{3.70}$$

The partial derivative can be substituted by the total molar flow $\dot{N} = \sum_i \dot{n}_i$, adiabatic exponent κ, thermal expansitivity ε_T, and compressibility ε_p (see Appendix C.3):

$$v_{\mathsf{sonic}}^2 = -\frac{\dot{V}}{\varrho}\left.\frac{\partial p}{\partial \dot{V}}\right|_{\dot{S}} = \frac{\dot{V}}{\varrho}\left.\frac{\partial^2 \dot{U}}{\partial \dot{V}^2}\right|_{\dot{S}} = \frac{\kappa}{\varrho\,\varepsilon_p}\left(\frac{c_V}{c_p} + \frac{\varepsilon_T^2\,\dot{V}\,T}{\varepsilon_p\,\dot{N}\,c_p}\right) = \frac{\kappa}{\varrho\,\varepsilon_p}\,. \tag{3.71}$$

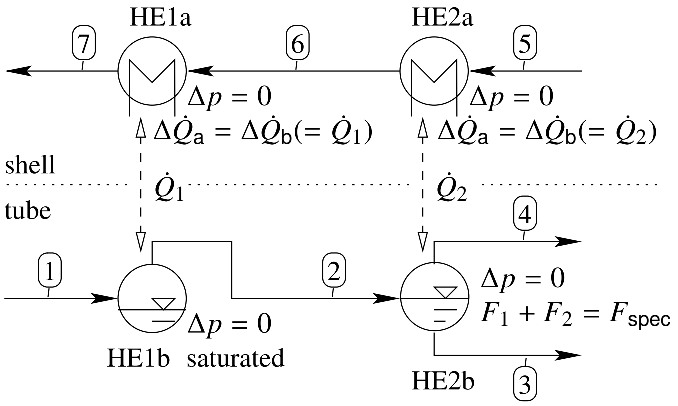
Here, the last calculation step utilises the relation between the heat capacities as

$$c_V = c_p - \frac{\varepsilon_T^2\,\dot{V}\,T}{\dot{N}\,\varepsilon_p}\,. \tag{3.72}$$

Thus, the speed of sound can easily be calculated, but it can not be used directly in the constitutive equations without the loss of the quadratic convergence properties or alternatively the necessity of third derivatives of thermodynamic state functions.

3.6.4 Sub-cooled and super-heated fluids in heat exchangers

If the stable phases change within a heat exchanger, a common approach is to establish a distributed model and conduct stability tests in each volume element. This is probably a necessary choice, if the heat transfer is strongly coupled with the local stream properties. However, a lumped heat exchanger model can be used in some cases and still support distinct regions for different phase sets. Figure 3.9 shows a


HE1a
HE2a
7
6
5
$\Delta p = 0$
$\Delta p = 0$
$\Delta\dot{Q}_a = \Delta\dot{Q}_b (= \dot{Q}_1)$
$\Delta\dot{Q}_a = \Delta\dot{Q}_b (= \dot{Q}_2)$
shell
tube
$\dot{Q}_1$
$\dot{Q}_2$
4
1
2
$\Delta p = 0$
$F_1 + F_2 = F_{\mathsf{spec}}$
$\Delta p = 0$
HE1b saturated
HE2b
3

Process properties

$\Delta T_1 = f(T_1, T_2, T_6, T_7)$

$\Delta T_2 = f(T_2, T_{3/4}, T_5, T_6)$

$k_1 = f(\dot{\underset{\sim}{x}}_1, \dot{\underset{\sim}{x}}_2, \dot{\underset{\sim}{x}}_6, \dot{\underset{\sim}{x}}_7)$

$k_2 = f(\dot{\underset{\sim}{x}}_2, \dot{\underset{\sim}{x}}_3, \dot{\underset{\sim}{x}}_4, \dot{\underset{\sim}{x}}_5, \dot{\underset{\sim}{x}}_6)$

$F_1 = \dot{Q}_1/(k_1\,\Delta T_1)$

$F_2 = \dot{Q}_2/(k_2\,\Delta T_2)$

Figure 3.9: *Heat exchanger to condense from super-heated vapour.*

composite FM to describe the process of partially condensing a super-heated vapour in a tube-shell counter-current heat exchanger. The process unit is divided into two sections *HE1* and *HE2*, which describe the super-heated and the two-phase region. The distribution of physical surface area to these regions is a result of the computation. *HE1b* is a saturation node (see Section 3.5.6) that, constrained by a pressure specification, does not offer further DOFs. *HE1a* and *HE2a* also are fully specified by pressure equations and one equation each to conserve heat flow in the systems

(*HE1a*, *HE1b*) and (*HE2a*, *HE2b*). The only DOF remains in *HE2b*, used to specify the total surface $F_{\text{spec}} = F_1 + F_2$.

The canonical equation system is trivial on the shell-side (first two blocks), while the tube-side (last two blocks) contains equilibrium calculations:

$$
\left(\begin{array}{cc|cc|cccc|ccc}
\underset{\approx}{H}_6 & \underset{\approx}{I} & & & & & & & & & \\
\underset{\approx}{I} & & & & & & & & & & \\
\hline
 & & \underset{\approx}{H}_7 & \underset{\approx}{I} & & & & & & & \\
-\underset{\approx}{I} & & \underset{\approx}{I} & & & & & & & & \\
\hline
 & & & & \underset{\approx}{H}_{2,\text{main}} & \underset{\approx}{I} & & & & & \\
 & & & & \underset{\approx}{I} & & & & & & \\
 & & & & & \underset{\approx}{I} & \underset{\approx}{H}_{2,\text{trial}} & \underset{\sim}{a} & & & \\
 & & & & -\underset{\sim}{a} & & \underset{\sim}{a} & & & & \\
\hline
 & & & & & & & & \underset{\approx}{H}_3 & & \underset{\approx}{I} \\
 & & & & & & & & & \underset{\approx}{H}_4 & \underset{\approx}{I} \\
 & & & & -\underset{\approx}{I} & & & & \underset{\approx}{I} & \underset{\approx}{I} &
\end{array}\right)
\cdot
\left(\begin{array}{c}
\Delta\dot{\underset{\sim}{x}}_6 \\
\underset{\sim}{\lambda}_6 \\
\hline
\Delta\dot{\underset{\sim}{x}}_7 \\
\underset{\sim}{\lambda}_7 \\
\hline
\Delta\dot{\underset{\sim}{x}}_{2,\text{main}} \\
\underset{\sim}{\lambda}_{2,\text{main}} \\
\Delta\dot{\underset{\sim}{x}}_{2,\text{trial}} \\
\lambda_{2,\text{trial}} \\
\hline
\Delta\dot{\underset{\sim}{x}}_3 \\
\Delta\dot{\underset{\sim}{x}}_4 \\
\underset{\sim}{\lambda}_{3,4}
\end{array}\right)
=
\left(\begin{array}{c}
-\underset{\sim}{g}_6 \\
\hat{\underset{\sim}{\delta}}_6 + \underset{\sim}{\alpha}_6 \\
\hline
-\underset{\sim}{g}_7 \\
\hat{\underset{\sim}{\delta}}_7 + \underset{\sim}{\alpha}_7 \\
\hline
-\underset{\sim}{g}_{2,\text{main}} \\
\hat{\underset{\sim}{\delta}}_{2,\text{main}} + \underset{\sim}{\alpha}_{2,\text{main}} \\
-\underset{\sim}{g}_{2,\text{trial}} \\
\alpha_{2,\text{trial}} \\
\hline
-\underset{\sim}{g}_3 \\
-\underset{\sim}{g}_4 \\
\hat{\underset{\sim}{\delta}}_{3,4} + \underset{\sim}{\alpha}_{3,4}
\end{array}\right) .
\tag{3.73}
$$

Note that only two streams indicated in Figure 3.9 are represented as couplings in the coefficient matrix, namely stream 2 and 6. Stream 1 and 5 are input streams, contributing to the balance equations of the first (1) and the third (5) block. Streams 3, 4, and 7 are material sinks in this context. They are represented by the first column of the second block (7), and the first two columns of the fourth block (3, 4). The shell and tube sides do not exchange material and appear therefore completely decoupled in the canonical equation system. Interaction only takes place through constitutive equations.

3.7 Initialisation

3.7.1 Approach for sequential-modular solvers

The initialisation of an arbitrary equation system to assure convergence to the correct solution – if there is any – is an unsolved problem. In general, there is a compromise between using a set of robust estimation equations, which solve the system rather inaccurately, and using the set of original equations of non-linear nature with limited mathematical domain and convergence radius. Furthermore, starting values of only a subset of variables have to be obtained, as the remaining ones can subsequently be calculated by the original system. This effect can be exploited in sequential-modular approaches, such that only tear streams and variables specified through implicit constitutive equations need to be initialised. A possible strategy to initialise an equation based process model is therefore to perform a sequential pre-execution, that is to exploit the process topology and to find a suitable calculation sequence for initialisation (Zitney and Stadtherr, 1988). In general, a suitable set of robust model equations and estimation equations are solved in a proper sequence.

3.7.2 Equation of state thermodynamic models

The most common equations of state, as for instance the SRK equation by Soave (1972), often predict a rather inaccurate liquid density at given pressure. Due to the low compressibility, starting volumes far from the predicted density are likely within a non-physical region or entirely outside the domain of the model. Therefore, a robust and precise density correlation used to calculate a liquid volume, which later is to be predicted by an equation of state, is likely to fail despite (actually caused by) its high accuracy. The calculation of a starting value for the volume as an input to equation of state models must therefore be performed by a model-specific function outside the actual initialisation process.

Therefore, nodes calculated by equation of state models are to be initialised by $\underset{\sim}{x}_{\text{initial}} = (T, p, \underset{\sim}{n})$, even though volume, not pressure, is the canonical state variable. Once, $\underset{\sim}{x}_{\text{initial}} \in \mathcal{X}$, the model is able to calculate the complete set of thermodynamic properties (cf. Appendix C.3), and the ordinary solving process can be launched.

The development and testing of initialisation methods is not included in the main scope of this work, but a general approach suitable for the canonical solving approach is described in the next section.

3.7.3 Approach for the canonical flowsheet solver

This approach is based on the ideas of Zitney and Stadtherr (1988) and is referred to as the *evolutionary approach*. A large number of robust equations is collected from all calculating instances. For instance, a flash module might provide equations to estimate separation factors, and linear constitutive equations and balance equations can be utilised directly. A previously obtained solution or a linear approximation, taking into account the changes in model parameterisation $\underset{\sim}{u}$, represents a valuable set of initialisation equations. As a fallback, global default values for all $(T, p, \dot{\underset{\sim}{n}})$ are available, for example as (298.15 K, 1 bar, $\underset{\sim}{1}$ mol).

The objective of using the most reliable relationships for initialisation can be represented by a cost matrix, which maps equations to variables in a bipartite graph (see Appendix F.3). The solutions of assignment problems result in the optimal set of equations used to determine the required set of variables.

However, a simple example shows the devastating effects of an unfavourable combination of otherwise robust equations. Consider a mixture of three species with the molar flow vector $\dot{\underset{\sim}{n}} = (\dot{n}_1, \dot{n}_2, \dot{n}_3)$. Let $\dot{n}_1 = 0.7\,(\underset{\sim}{1} \cdot \dot{\underset{\sim}{n}})$ and $(\underset{\sim}{1} \cdot \dot{\underset{\sim}{n}}) = 1$ mol/s be specification equations, which obviously are suitable for initialisation. A robust and therefore seemingly harmless estimation equation $\dot{n}_1 = \dot{n}_2$ yields the infeasible solution $\dot{n}_1 = \dot{n}_2 = 0.7$ mol/s, and $\dot{n}_3 = -0.4$ mol/s. Therefore, inequality constraints

($\dot{n}_i > 0$) have to be incorporated into the initialiser, whose task is to find the optimal and most robust and reliable set of equations with a solution within given constraints. A suggestion for an initialisation algorithm is given in Appendix E.4.

3.8 Relaxation scheme

An iterative solving method, as suggested in Section 3.3.2, calculates an update $\Delta\underset{\sim}{x}$ of state variables in each step. Since this update does not yield the solution directly for a general non-linear equation system, the updated state $\underset{\sim}{x}^{(k+1)} = \underset{\sim}{x}^{(k)} + \Delta\underset{\sim}{x}$ is not necessarily within the domain $\mathcal{X}$ of the process model. In particular, the equations of thermodynamic models might require a positive temperature, pressure or volume, and a mole-vector $\underset{\sim}{n}$, for which each element is of the same sign.

Wilhelm and Swaney (1994) recommend the relaxation $\gamma \in (0, 1]$ of each iteration step to ensure $\underset{\sim}{x}^{(k+1)} = \underset{\sim}{x}^{(k)} + \gamma\,\Delta\underset{\sim}{x} \in \mathcal{X}$ by means of linear programming. For process models on a canonical basis, however, all state variables of the process model are state variables of thermodynamic models, and these thermodynamic models can be equipped with the functionality to restrict γ for a given search direction $\Delta\underset{\sim}{x}$.

This section describes a systematic scheme to collect restrictions of γ within a relaxation object R. This object is a representation of the feasible domain of γ. During the solving process, every thermodynamic model is given opportunity to restrict R based on the current state $\underset{\sim}{x}^{(k)}$ and the suggested direction $\Delta\underset{\sim}{x}$.

A sorted sequence of values γ_i is sufficient to describe the domain of feasible relaxation factors R, provided that $\gamma_0 \equiv 0$ is a permitted step length, since $\underset{\sim}{x}^{(k)} \in \mathcal{X}$. Then,

$$\mathrm{R} = (\gamma_0, \gamma_1) \cup \cdots \cup (\gamma_i, \gamma_{i+1}) \cup \cdots \cup (\gamma_{N-1}, \gamma_N) \quad \text{with } i \text{ even and } N \text{ odd.} \tag{3.74}$$

The solver defines a safety factor $f_\gamma > 1$ that defines the minimum distance between any selected relaxation factor $\gamma \in \mathrm{R}$ and the domain boundary values γ_i, such that

$$\gamma \in \bigcup_{i \text{ even}} [f_\gamma\,\gamma_i, \gamma_{i+1}/f_\gamma]\,. \tag{3.75}$$

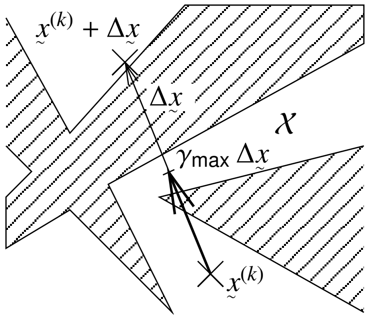
The solver then initiates the relaxation object as $\mathrm{R} = (0, f_\gamma)$, such that, if no further restrictions are contributed, the maximal $\gamma \in [0, 1]$ is selected. With $\gamma = 1$, this permits a full N step with quadratic convergence (Nocedal and Wright, 1999).

A good choice for the safety factor $f_\gamma \approx 1.1$, such that a certain distance to the domain boundaries is maintained, but the solution scheme converges still reasonable fast. If the solution really is close to the boundary, the convergence is linear, and the convergence factor (Nocedal and Wright, 1999) is:

$$\frac{\|\underset{\sim}{x}^{(k+1)} - \underset{\sim}{x}^{(\infty)}\|}{\|\underset{\sim}{x}^{(k)} - \underset{\sim}{x}^{(\infty)}\|} = \frac{f_\gamma - 1}{f_\gamma}\,. \tag{3.76}$$

Given the relaxation object initiated by the solver, each instance of a thermodynamic model can contribute contributes further relaxation objects, which can include positive infinity. Appendix E.3 describes an algorithm to calculate the representation of the union of two relaxation objects. The union of all relaxation objects is used to determine the maximal relaxation factor γ_{max} from the domain that is described in Equation (3.75). With $\gamma_1 > 0$, there is always a feasible step size $\gamma_{\mathsf{max}} > 0$.

Figure 3.10 shows a non-convex domain in white, while the hatched areas represent the infeasible regions. An arrow from the current state $\underset{\sim}{x}^{(k)}$ indicates the direction suggested by the N -R step. Though the full step $\Delta\underset{\sim}{x}$ is possible, the distance to the domain boundaries is too small, and the safer step length γ_{max} is selected.



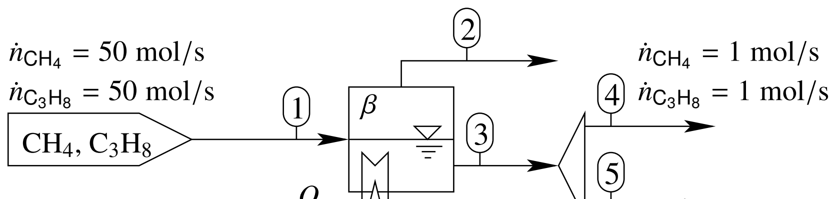
$\underset{\sim}{x}^{(k)} + \Delta\underset{\sim}{x}$
$\Delta\underset{\sim}{x}$
$\mathcal{X}$
$\gamma_{\mathsf{max}}\,\Delta\underset{\sim}{x}$
$\underset{\sim}{x}^{(k)}$

Figure 3.10: *Example iteration step in a non-convex domain.*

It is important that all restrictions to the step length are determined through the relaxation object. Once a relaxation factor γ_{max} is determined, this factor should not be changed. In particular, not all $\gamma \in (0, \gamma_{\mathsf{max}})$ yield state vectors within the mathematical domain of the process model. A further manual relaxation below the calculated γ_{max} has a negative effect on the robustness of the method.

This relaxation method only considers the linearised effect of $\underset{\sim}{x}$ towards calculated properties $\underset{\sim}{y}$ and internally calculated variables within the thermodynamic model including state function transformations. This method may fail if a highly non-linear domain constraint is active and f_γ is chosen too close to unity.

Example



$\dot{n}_{\mathsf{CH_4}}$ = 50 mol/s
$\dot{n}_{\mathsf{C_3H_8}}$ = 50 mol/s
CH_4, C_3H_8
1
Q
β
2
3
4
5
$\dot{n}_{\mathsf{CH_4}}$ = 1 mol/s
$\dot{n}_{\mathsf{C_3H_8}}$ = 1 mol/s

Figure 3.11: *Process model with a restricted domain.*

Consider the process model shown in Figure 3.11. The total species flow specification of methane in stream 4 requires a negative species flow in stream 5, if not enough methane is available. The state is not within the domain, if at the same time there is still a positive amount of butane. A state calculation is only feasible if the volume flow and the molar flows have a common sign, either positive or negative. Figure 3.12 shows a trajectory projected into the molar flow vector of stream 5. Points

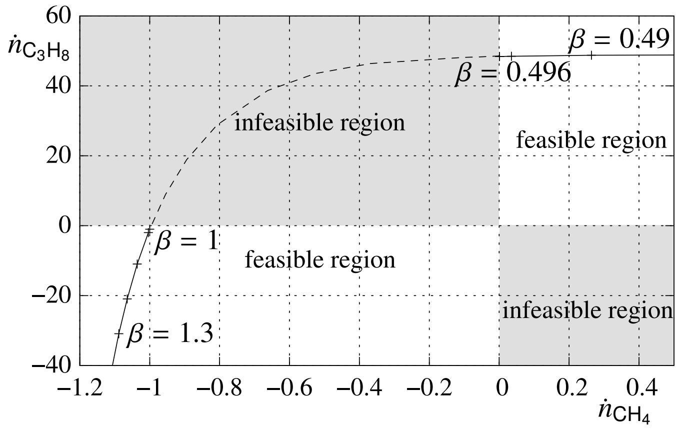


Figure 3.12: *Molar flow trajectory dependent on the vapour fraction β – feasible and infeasible regions.*

close to the infeasible region can successfully be calculated. Approaching a molar vapour fraction of $\beta \approx 0.496$ from below, $\dot{n}_{CH_4}$ approaches zero, while $\dot{n}_{C_3H_8}$ is still positive. For $\beta > 1$, both flows become negative, and calculations can be conducted even though the solution is not physical. The stipulated line within the infeasible region is calculated after removing the splitter, solving its material balance in a post-calculation.

If the model is calculated for an infeasible vapour fraction, γ_{max} is restricted to nearly zero, thus the iteration stalls on the border of the domain. The variables of active domain constraints can be identified by error diagnosis.

Figure 3.12 also shows a limitation of this relaxation method. Both feasible branches enter the infeasible region with an angle such that a huge iteration step would be required in order to jump from one into the other feasible region. Practically, even this rather small example will not converge, if the starting values and result are not in the same feasible region. Independent of the actual solving method, it is therefore important to provide reasonable starting values. However, if the solution can not be obtained, the relaxation factor approaches zero. Identification of the restricted variable (in this case n_{CH_4} or $n_{C_3H_8}$) can help the user to understand and remove the problem.

3.9 Error reporting

Berger and Perris (1979) give a set of objectives for the design of their process simulator, of which the first one deserves far more attention than is usual today – more than 25 years later:

> *FLOWPACK II must be computationally both efficient and reliable. It must either solve the problem posed (which may or may not the problem which the user intended to pose!) or must fail 'gracefully'* ***for clearly identified reasons****.*

The feasible extent of error analysis depends on the solving strategy, but few programs invest any effort in providing user-friendly error-messages. The user is forced to conduct a cumbersome procedure of changing starting values and design specifications in the hope to achieve a converged solution. The actual origin of problems can be:

Terms and definitions 3.3

Linear dependent model equations: A heat exchanger is specified by temperatures of all streams, or a chemical species is captured within a circulation.

Non-existence of a solution in the mathematical domain of the model: Specified pressure drop greater than upstream pressure, or first principle specifications, for instance in species splitter, force values of different sign into the mole vector.

Non-feasible initial values: Typically, the pressure is above or below the stability pressure of the fluid, or the composition is outside the chemical stability region of the phase.

The following sections describe how these problems can be identified, and how this mathematical identification can be translated into a constructive advise for the user of the program.

3.9.1 Potential points of failure in the solving algorithm

A key feature of the modelling approach investigated in this work is the extensive use of structural information. As not only one large equation system is solved, the algorithm can fail in more distinct ways than just to report a general singularity. The following paragraphs describe distinct problems in the solution process, which can be identified individually.

Occurrence of a singular matrix in an atomic flowsheet module

Atomic FMs are guaranteed to be non-singular, if they are based on first principles, like the one-phase calculation node or a component splitter. FMs involving thermodynamic calculations, like equilibrium reactions and phase equilibrium, might become singular. A typical example is the occurrence of a trivial solution in phase equilibrium. These singularities can usually be identified by the particular module, which in turn can give detailed failure information to the user, or even fix the problem.

Occurrence of a singular matrix in a composite flowsheet module

If all atomic flowsheet module coefficient matrices are decomposed successfully, the only reason for a singularity in a composite flowsheet module is a circulation. A deeper description of the problem and how to solve it is given in Section 3.9.2.

Occurrence of a rank-loss in the constitutive equation set

If the J matrix of the constitutive equation set $\underset{\sim}{h}(\underset{\sim}{x}, \underset{\sim}{\lambda}) = 0$ is rank-deficient, some of these equations are linearly dependent. To help the user to overcome the problem, the left null space reveals the sets of linearly dependent equations. Consider three constitutive equations on a set of canonical variables $(\underset{\sim}{x}, \underset{\sim}{\lambda}) = (\ldots, p_1, T_1, p_2, \ldots)$, and the following equations:

$$h_1 = p_1 - p_1^{\mathsf{spec}}, \quad h_2 = p_2 - p_2^{\mathsf{spec}}, \quad h_3 = T_1 - T_1^{\mathsf{spec}}, \text{ and} \quad h_4 = p_2 - p_1 \,. \tag{3.77}$$

The J matrix is

$$\frac{\partial \underset{\sim}{h}}{\partial(\underset{\sim}{x}, \underset{\sim}{\lambda})} = \left(\begin{array}{c:c:c} 0 \ldots 0 & 1\ 0\ 0 & 0 \ldots 0 \\ 0 \ldots 0 & 0\ 0\ 1 & 0 \ldots 0 \\ 0 \ldots 0 & 0\ 1\ 0 & 0 \ldots 0 \\ 0 \ldots 0 & -1\ 0\ 1 & 0 \ldots 0 \end{array}\right) . \tag{3.78}$$

The vector $(1, -1, 0, 1)$ represents the left null space of $\partial\underset{\sim}{h}/\partial(\underset{\sim}{x}, \underset{\sim}{\lambda})$, which means that $h_1 - h_2 + h_4 = 0$. All involved elements of $\underset{\sim}{h}$ represent a set of linear dependent equations. This information can be propagated to the user.

The most common constitutive equations are linear in canonical and conjugated variables. Even if non-linear equations are used, there are no significant conditioning problems regarding the constitutive J matrix. On this level, linear dependent equations are rarely a consequence of a numerical problem, but to a high degree of probability point to an erroneous process model formulation.

Occurrence of a singularity in the combined system

The update equation (3.25) represents an equation system to be solved in order to determine the correction of source-terms within the canonical equation system. The method described in the previous paragraph caught linearly dependent constitutive equations. At this point, the full rank of $\partial\underset{\sim}{h}/\partial(\underset{\sim}{x}, \underset{\sim}{\lambda})$ is verified.

A rank loss of $\partial\underset{\sim}{h}/\partial(\hat{\underset{\sim}{x}}, \hat{\underset{\sim}{\lambda}})\, \underset{\approx}{E}_x\, \underset{\approx}{B}^{-1}\, \underset{\approx}{E}_\alpha$ indicates a linear dependency of a combination of constitutive equations and the canonical equation system. Examples are the attempt to calculate a T, p-flash for a stream of a pure species or an azeotropic mixture, or the specification of both the input and outlet temperatures of a heat exchanger. As for the pure constitutive system, the left null space can be used to identify the constitutive equations involved in linear dependent constraints. However, the impact of the canonical equation system can cause problems at this stage. Actual linearly dependent equations cannot be distinguished easily from singularities caused by other reasons, as for instance general divergence, described in the next paragraph.

General divergence

Unfortunately, even with a full rank equation system, a physical solution might not be found, either because there is none, or because the starting values are not within the convergence radius of the solution method. If the constraints force a solution outside the mathematical domain of the thermodynamic models, the algorithm will not diverge, but as is described in Section 3.8, may stall on the domain boundary. Identification of the troublesome variables, which are involved in active domain boundary constraints, offers valuable information to the user in order to solve the problem. This identification can be performed manually by the user, for instance by observing values for temperature or molar flows close to zero. The functionality of the relaxation object (see Section 3.8) can be extended to record the most restrictive calculation module (e.g. a thermodynamic model).

3.9.2 Problem of circulations

A circulation is defined by a strict conservation of the flow of a chemical species or another canonical quantity within a recycle. In a sequential-modular approach, recycles themselves induce the need for partitioning and tearing, while equation-oriented solvers have the benefit of handling recycles directly. However, if quantities circulate, as material in a closed refrigerator system, the coefficient matrix becomes singular due to linearly dependent balance equations, and no solution can be obtained. Furthermore, no DOFs are available in the process model to actually define the flow and chemical composition. If the flow is specified through a pressure-flow equation, the absolute pressure level is not determined.

Currently available process modelling tools simply ignore this effect or even hide it from the user. Equation-based tools give either a general message to indicate a singularity, or they find an arbitrary solution within the solution space of the equation system. Sequential modular tools as e.g. AspenPlus® (Evans *et al.*, 1979) generally enhance an estimation or even initial values to specifications and ignore the effect. If sources and sinks give no contribution to the circulating quantities, and if the system is numerically stable then the solver finds a solution based on these initial values. Otherwise, the solver diverges and terminates with an error extraneous to the actual problem.

This section shows, how the canonical approach allows one to identify singularities in the coefficient matrix on a composite FM level as circulations, in particular distinct from solving problems within a child FM, for example due to phase stability problems.

Singularity in an hierarchically defined coefficient matrix

Consider the coefficient matrix of a simplified process model of a refrigerator cycle, which consists of a heated valve and a cooled compressor:

$$\underset{\approx}{B} = \left(\begin{array}{cc:cc} \underset{\approx}{H}_1 & \underset{\approx}{I} & & \\ \underset{\approx}{I} & & -\underset{\approx}{I} & \\ \hdashline & & \underset{\approx}{H}_2 & \underset{\approx}{I} \\ -\underset{\approx}{I} & & \underset{\approx}{I} & \end{array}\right) . \quad \begin{array}{l} \leftarrow \text{Compressor} \\ \\ \leftarrow \text{Valve} \end{array} \tag{3.79}$$

Though all diagonal (child FM) blocks are invertible, the overall matrix is not, because the second and fourth row contain linear dependent balance equations. Furthermore, there is no specification of the circulating state at any point.

To safely identify a circulation in the process model, it must be ensured that a singularity during the block decomposition of the process model coefficient matrix is always caused by such a circulation. The following two theorems are essential to map the singularity of a pivot block to the singularity of the entire process model.

Theorem 3.1 *The coefficient matrix of a composite FM with invertible child coefficient matrices yields a singular pivot block, iff the total matrix is singular.*

Proof: The possibility of a singular child coefficient matrix is already excluded. Hence, further singularities must origin from a circulation, and any circulation yields a singularity. Therefore, it is possible to leave out any arbitrary coupling in an invertible coefficient matrix without loss of rank. Even more important is that supplementing new couplings to a singular coefficient matrix cannot restore the rank.

Let $\underset{\approx}{D}^{-1}$ be the block-diagonal matrix of all pre-inverted child flowsheet module coefficient matrices. Consider the G elimination of $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})$ in row i. The next diagonal block $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})_{i,i}$ is only influenced by recycle streams among the child modules of lower index. All couplings involving units with higher index are therefore disregarded. At the given stage of the elimination, the block-row i only consists of the diagonal block. From here, it is clear that this block is singular, iff the total coefficient matrix is singular. □

This is not obvious for arbitrary block matrices. For instance the coefficient matrix of a constrained optimisation problem (cf. equation (3.17)) is not singular, but at the same time not block-invertible.

Theorem 3.2 *A pivoting block $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})_{i,i}$ is block-invertible, if it is invertible.*

Proof: The starting point is the identity matrix, which is obviously invertible. Since adding couplings never restores the rank, a complete matrix of full rank implies a matrix of full rank with any subset of couplings. From here, the influence of each output port involved in a recycle is added in steps. The influence of each output port only affects one column block in $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})_{i,i}$. Assuming that the previously

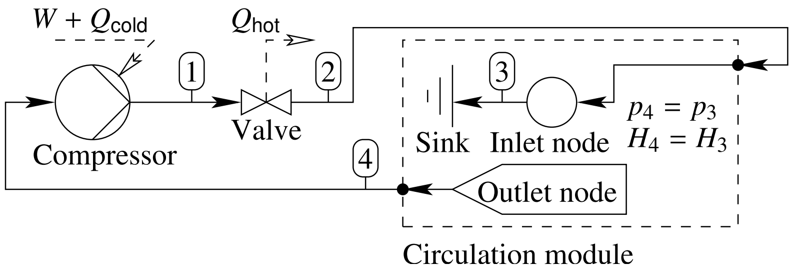


Figure 3.13: *The circulation module in a refrigerator cycle.*

existing diagonal blocks in $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})_{i,i}$ are invertible, they can be used to eliminate any off-diagonal block in the row j, when a contribution of another recycle is added to column j. Therefore, the diagonal block with index j is invertible, iff $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})_{i,i}$ is invertible. This block can be used as a pivot block in the block-inversion. Induction shows that, beginning with the identity matrix, block-invertibility is preserved as long as $(\underset{\approx}{D}^{-1}\,\underset{\approx}{B})_{i,i}$ is invertible. □

As a consequence of Theorem 3.1 and 3.2, circulations can be diagnosed in the canonical flowsheet solver as a singularity in the coefficient matrix, which is not caused by a singularity within a child FM coefficient matrix. The null space on the block level can be computed to determine the actual set of involved stream variables. As an important fact, it is inevitable to involve the user to resolve the problem. In spite of a simple recycle stream, a circulation provides DOFs in itself. The user needs to assign one constitutive equation to each of these DOFs in order to completely specify the model. A specialised FM is introduced in the next section to handle this problem.

Definitely, every recycle introduced into a process model potentially yields a circulation. Technically, a recycle produces a coupling block in the coefficient matrix, which forces modification of a pivot block during decomposition. This pivot block becomes singular due to the linear dependency of balance equations, iff the recycle is a circulation.

Circulation module

A circulation module is a pseudo FM to break up the linear dependent balance equations and to provide the necessary DOFs. It must be inserted somewhere within the circulation. The substitute flowsheet is shown in Figure 3.13. Based on the one-phase FM as described in Section 3.5.1 and the concept of material couplings

(Section 3.4.3), the coefficient matrix of the entire process is given as

$$\underset{\approx}{\hat{B}} = \left(\begin{array}{cc|cc|cc|cc} \underset{\approx}{H}_1 & \underset{\approx}{I} & & & & & & \\ \underset{\approx}{I} & & & & & & -\underset{\approx}{I} & \\ \hline & & \underset{\approx}{H}_2 & \underset{\approx}{I} & & & & \\ -\underset{\approx}{I} & & \underset{\approx}{I} & & & & & \\ \hline & & & & \underset{\approx}{H}_3 & \underset{\approx}{I} & & \\ & & -\underset{\approx}{I} & & \underset{\approx}{I} & & & \\ & & & & & & \underset{\approx}{H}_4 & \underset{\approx}{I} \\ & & & & & & \underset{\approx}{I} & \end{array}\right) . \begin{array}{l} \leftarrow \text{Compressor} \\ \\ \leftarrow \text{Valve} \\ \\ \leftarrow \text{Circulation module inlet node} \\ \\ \leftarrow \text{Circulation module outlet node} \\ \\ \end{array} \qquad (3.80)$$

The circulation block as the last diagonal block is trivial to invert. With only uncoupled sub-matrices, its insertion does not increase the complexity to decompose the coefficient matrix. Compared to an ordinary recycle stream, this formulation actually reduces the complexity, as the recycle is opened up. However, $\dim \underset{\sim}{x}$ DOFs have to be filled by constitutive equations. With $\underset{\sim}{x}_\text{in}$ and $\underset{\sim}{x}_\text{out}$ the incoming and outgoing stream of a circulation module, the set of equations $\underset{\sim}{x}_\text{in} = \underset{\sim}{x}_\text{out}$ is available, but applying the full set would just shift the linear dependency problem to the constitutive system. For each linearly dependent balance equation of x_i, another constraint must be activated, e.g. a direct specification of x_i at that point, or any suitable external constitutive equation. The circulation module must validate $\underset{\sim}{x}_\text{in} = \underset{\sim}{x}_\text{out}$ at the solution point. No changes of state are permitted within this module.

3.9.3 Consistent stoichiometry in chemical reactors

In this work, all definitions of chemical reactions are based on the element balance equations (see Section 3.5.3). As this way to define reacting systems does not allow for the violation of conservation of chemical elements, this already eliminates a major source of error. Furthermore, the state-based approach requires no redundant and potentially inconsistent information about the enthalpy of reaction. Depending on the applied constitutive equations, the partial enthalpies of reacting species determine either the product temperature or the heat duty of the reaction process.

However, with reactions proceeding to their full extent, as described in Section 3.5.3, stoichiometric constraints might get linearly dependent (see Figure 3.4). This case can be detected as a singularity of the combined system (see p. 61). The identification of the involved constitutive equations helps to describe the actual problem.

Chapter 4

Advanced process systems engineering disciplines

4.1 Introduction

The previous chapter concentrates on a basic discipline of process systems engineering (PSE), namely steady-state process simulation. Other PSE disciplines are largely built on the functionality of steady-state simulations, of which Figure 4.1 gives an overview. Case studies are basically a sequence of simulations, therefore posing

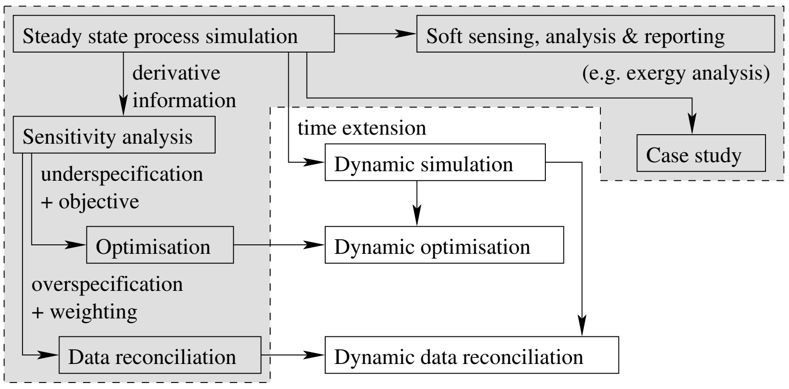


Figure 4.1: *Process system engineering disciplines. The gray background represents the scope of this work.*

no further challenges with respect to the canonical modelling approach. The same could be stated about the subject of soft sensing, analysis and reporting. However, a canonical approach yields an elegant way to define and discuss exergy, as is shown in Section 4.6.

The subject of dynamic process modelling generates a number of challenges that are not specific to the approach of canonical modelling, namely integration methods, discrete handling of events such as topology changes, initial value problem and identification of badly posed problems. A brief introduction to dynamic process modelling on a canonical basis is discussed in Appendix D.

An important extension from ordinary process simulation is the supply of reliable derivative information with various sets of dependent and independent variables. The next section describes how this information can be extracted from a solution of the algorithm described in Section 3.3.2. The subsequent sections show how to utilise this technique in process optimisation and data reconciliation. It must be noted that these fields pose many challenges by themselves. The purpose of the following sections is to investigate the potential of canonical modelling and prove its general suitability.

4.2 Process model derivatives

4.2.1 Computational methods

One common way to obtain derivative information from the result of an algorithm is to perturb the independent variables systematically and use the numerical approach of finite differences: $\mathrm{d}\psi/\mathrm{d}\underset{\sim}{x} \approx [(\psi(x_i + \Delta x_i) - \psi(x_i - \Delta x_i))/2\Delta x_i]\,\underset{\sim}{e}_i$. Not only has the algorithm to be executed $2 \cdot \dim \underset{\sim}{x}$ times to obtain the first derivative, but it also remains a problem to choose Δx_i small enough to eliminate smoothening effects, but large enough to overcome numerical problems due to the precision of the algorithm's results. These two requirements are often irreconcilable in practice.

There are different approaches to obtain analytical derivatives of functions and algorithms. Automatic differentiation compiles existing code of a specific programming language into extended functions in the same language, which produce the required derivative information (Mischler *et al.*, 1995). This method is applicable, if the function and the set of selected dependent and independent variables are defined at compile time. A prime example is the generation of first- and second-order state function derivatives with respect to their state vector. However, in a dynamically configurable process modelling tool, much necessary information is added at runtime, such as

- Process topology and selection of FMs.
- Thermodynamic models and sets of chemical species.
- Sets of constitutive equations, partly first generated by user runtime, subsequently parsed into computer memory.
- Selection of dependent and independent variables in the context of optimisation or reconciliation.

Based on the functional programming paradigm (Hudak, 1989), a general function can be represented by a symbolic algebra graph. Computer algebra systems (CASs) like *Maple*$^{\bigcirc}$ (Čížek *et al.*, 1993) and *Mathematica*$^{\bigcirc}$ (Fateman, 1992) and frameworks for symbolic computations like *GiNaC* (Bauer *et al.*, 2002) utilise this technology.

Elementary functions and operators and literal numbers are represented as nodes in a directed graph, in which the edges point from function and operator nodes to the respective arguments. Literal numbers represent the leaf nodes with zero outgoing cardinality. As shown in Figure 4.2, this symbolic representation not only allows for the calculation of analytical derivatives, but also for code optimisation, for evaluation of expressions with different data-types, and for automatic generation of code in different programming languages. Technical details about the symbolic algebra

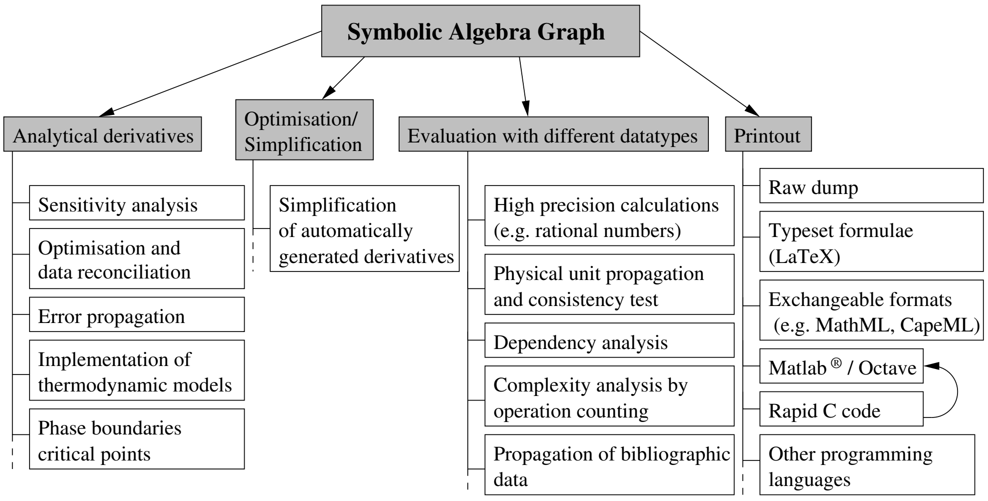

Symbolic Algebra Graph
Analytical derivatives
Optimisation/ Simplification
Evaluation with different datatypes
Printout
Sensitivity analysis
Optimisation and data reconciliation
Error propagation
Implementation of thermodynamic models
Phase boundaries critical points
Simplification of automatically generated derivatives
High precision calculations (e.g. rational numbers)
Physical unit propagation and consistency test
Dependency analysis
Complexity analysis by operation counting
Propagation of bibliographic data
Raw dump
Typeset formulae (LaTeX)
Exchangeable formats (e.g. MathML, CapeML)
Matlab® / Octave
Rapid C code
Other programming languages

Figure 4.2: *Available functionality through symbolic algebra.*

datatype developed and applied in this work are given in Appendix A.1.

However, the pure form of symbolic algebra only works with explicit functions. Iterative algorithms introduce circles in the representative graph that require special treatment. In particular, such a loop would in general not provide full robustness regarding convergence. The use of symbolic algebra is therefore limited to functional constructs and treat algorithms externally.

4.2.2 Symbolic derivatives from the canonical solver

This section follows the idea that if a second-order method terminates successfully with a solution, the first-order derivative information is also provided through the coefficient matrix used in the last iteration step.

Consider a non-linear program without inequality constraints, as described in Section 3.2, the optimisation of a state function $P(\underset{\sim}{x})$ subject to constraints $\underset{\sim}{\delta}(\underset{\sim}{x}) = \underset{\sim}{0}$. The update equation is based on a T series of the first derivative of the L

Function Λ being

$$\Lambda(\underset{\sim}{x},\underset{\sim}{\lambda},\underset{\sim}{\psi}) = P(\underset{\sim}{x},\underset{\sim}{\psi}) - \underset{\sim}{\lambda}\,\underset{\sim}{\delta}(\underset{\sim}{x},\underset{\sim}{\psi})\,. \tag{4.1}$$

In this context, $\underset{\sim}{\psi}$ is a vector of parameters, either of the process model ($\underset{\sim}{u}$) or the underlying thermodynamic models ($\underset{\sim}{c}$). With $\underset{\sim}{\zeta} = (\underset{\sim}{x},\underset{\sim}{\lambda})$ and $\underset{\sim}{l}(\underset{\sim}{\zeta}) = \partial\Lambda/\partial\underset{\sim}{\zeta}$, the truncated T series for $\underset{\sim}{\zeta}^{(k)}$ in the neighbourhood of the solution $\underset{\sim}{\zeta}^{(\infty)}$ is

$$\underset{\sim}{l}(\underset{\sim}{\zeta}^{(\infty)},\underset{\sim}{\psi}) \approx \underset{\sim}{l}(\underset{\sim}{\zeta}^{(k)},\underset{\sim}{\psi}) + \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}}\right|_{\underset{\sim}{\zeta}^{(k)}} (\underset{\sim}{\zeta}^{(\infty)} - \underset{\sim}{\zeta}^{(k)}) \overset{!}{=} \underset{\sim}{0}\,. \tag{4.2}$$

The sensitivities $\partial\underset{\sim}{\zeta}/\partial\underset{\sim}{\psi}$ are to be obtained. The derivative of Equation (4.2) with respect to $\underset{\sim}{\psi}$ is

$$\begin{aligned}
\underset{\approx}{0} = {} & \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\psi}}\right|_{\underset{\sim}{\zeta}^{(k)}} + \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}}\right|_{\underset{\sim}{\psi}} \frac{\mathrm{d}\underset{\sim}{\zeta}^{(k)}}{\mathrm{d}\underset{\sim}{\psi}} + \left(\frac{\partial^2\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}\,\partial\underset{\sim}{\psi}} + \left.\frac{\partial^2\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}^2}\right|_{\underset{\sim}{\psi}} \frac{\partial\underset{\sim}{\zeta}^{(k)}}{\partial\underset{\sim}{\psi}}\right)(\underset{\sim}{\zeta}^{(\infty)} - \underset{\sim}{\zeta}^{(k)}) \\
& + \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}}\right|_{\underset{\sim}{\psi}} \left(\frac{\mathrm{d}\underset{\sim}{\zeta}^{(\infty)}}{\mathrm{d}\underset{\sim}{\psi}} - \frac{\mathrm{d}\underset{\sim}{\zeta}^{(k)}}{\mathrm{d}\underset{\sim}{\psi}}\right) \\
= {} & \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\psi}}\right|_{\underset{\sim}{\zeta}^{(k)}} + \left(\frac{\partial^2\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}\,\partial\underset{\sim}{\psi}} + \left.\frac{\partial^2\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}^2}\right|_{\underset{\sim}{\psi}} \frac{\partial\underset{\sim}{\zeta}^{(k)}}{\partial\underset{\sim}{\psi}}\right)(\underset{\sim}{\zeta}^{(\infty)} - \underset{\sim}{\zeta}^{(k)}) + \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}}\right|_{\underset{\sim}{\psi}} \frac{\mathrm{d}\underset{\sim}{\zeta}_\infty}{\mathrm{d}\underset{\sim}{\psi}}\,.
\end{aligned} \tag{4.3}$$

For a converged iteration at $k \to \infty$, $\underset{\sim}{\zeta}^{(\infty)} = \underset{\sim}{\zeta}^{(k)}$, while the other terms do not approach zero. Omitting the notation to indicate the iteration step at convergence, the limit is

$$\underset{\approx}{0} = \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\psi}}\right|_{\underset{\sim}{\zeta}} + \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}}\right|_{\underset{\sim}{\psi}} \frac{\mathrm{d}\underset{\sim}{\zeta}}{\mathrm{d}\underset{\sim}{\psi}}\,,\ \text{and finally}\quad \frac{\mathrm{d}\underset{\sim}{\zeta}}{\mathrm{d}\underset{\sim}{\psi}} = -\left(\left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\zeta}}\right|_{\underset{\sim}{\psi}}\right)^{-1} \cdot \left.\frac{\partial\underset{\sim}{l}}{\partial\underset{\sim}{\psi}}\right|_{\underset{\sim}{\zeta}}\,. \tag{4.4}$$

Hence the derivative of $\underset{\sim}{\zeta}$ with respect to $\underset{\sim}{\psi}$ can easily be found, if the solution vector $\underset{\sim}{\zeta}^{(k)}$ and the last calculated coefficient matrix $\underset{\approx}{B} = (\partial\underset{\sim}{l}/\partial\underset{\sim}{\zeta})$ is already LU-decomposed. The derivative of the right hand side $\partial\underset{\sim}{l}/\partial\underset{\sim}{\psi}$ can be obtained by means of symbolic algebra as described in the previous section.

Application to one-phase chemical equilibrium calculations

An example should clarify the direct use of Equation (4.4). The objective is to obtain the derivative of the chemical potentials at chemical equilibrium with respect to the parameters of the thermodynamic model. At constant temperature and pressure, the objective function can be defined based on G -energy G as

$$\Lambda(\underset{\sim}{n},\underset{\sim}{\lambda}) = G(\underset{\sim}{n}) - \underset{\sim}{\lambda}\,\underset{\approx}{A}(\underset{\sim}{n}_{\mathsf{initial}} - \underset{\sim}{n})\,. \tag{4.5}$$

Subsequently,

$$\underset{\sim}{\zeta} = \begin{pmatrix}\underset{\sim}{n}\\ \underset{\sim}{\lambda}\end{pmatrix}, \quad \underset{\sim}{l} = \begin{pmatrix}\underset{\sim}{\mu} + \underset{\approx}{A}^T\,\underset{\sim}{\lambda}\\ \underset{\approx}{A}\,(\underset{\sim}{n} - \underset{\sim}{n}_{\text{initial}})\end{pmatrix}, \quad \text{and} \quad \frac{\partial l}{\partial \underset{\sim}{\zeta}} = \begin{pmatrix}\underset{\approx}{H} & \underset{\approx}{A}^T\\ \underset{\approx}{A} & \underset{\approx}{0}\end{pmatrix}. \tag{4.6}$$

At the solution, the derivatives can be expressed as follows (see Equation (4.4)):

$$\frac{\mathrm{d}}{\mathrm{d}\underset{\sim}{\psi}}\begin{pmatrix}\underset{\sim}{n}\\ \underset{\sim}{\lambda}\end{pmatrix} = -\begin{pmatrix}\underset{\approx}{H} & \underset{\approx}{A}^T\\ \underset{\approx}{A} & \underset{\approx}{0}\end{pmatrix}^{-1} \cdot \frac{\partial}{\partial\underset{\sim}{\psi}}\left.\begin{pmatrix}\underset{\sim}{\mu} + \underset{\approx}{A}^T\,\underset{\sim}{\lambda}\\ \underset{\approx}{A}\,(\underset{\sim}{n} - \underset{\sim}{n}_{\text{initial}})\end{pmatrix}\right|_{\underset{\sim}{n},\underset{\sim}{\lambda}} = -\begin{pmatrix}\underset{\approx}{H} & \underset{\approx}{A}^T\\ \underset{\approx}{A} & \underset{\approx}{0}\end{pmatrix}^{-1} \cdot \begin{pmatrix}\left.\dfrac{\partial\underset{\sim}{\mu}}{\partial\underset{\sim}{\psi}}\right|_{\underset{\sim}{n}}\\ \underset{\approx}{0}\end{pmatrix}. \tag{4.7}$$

Furthermore, the derivatives of the chemical potential $\underset{\sim}{\mu}$ can be obtained from $\underset{\sim}{\lambda}$:

$$\underset{\sim}{\mu} = -\underset{\approx}{A}^T\,\underset{\sim}{\lambda} \quad \Rightarrow \quad \frac{d\underset{\sim}{\mu}}{d\underset{\sim}{\psi}} = -\underset{\approx}{A}^T\,\frac{d\underset{\sim}{\lambda}}{d\underset{\sim}{\psi}}. \tag{4.8}$$

Derivative information can be obtained by this technique in order to implement parameter optimisation.

Example

An ideal gas mixture of NO_2 and N_2O_4 is considered. Under the assumption of constant heat capacity, the chemical potential is given as (the universal gas constant is defined as $R = 8.3144$ J/mol K, furthermore $N = \sum_i n_i$)

$$\mu_i = \Delta_\mathrm{f} h_i^{\text{ref}} + T\left[c_{p,i}\left(1 - \frac{T}{T^{\text{ref}}}\right) - s_i^{\text{ref}} + R\,\ln\frac{p}{p^{\text{ref}}} + R\,\ln\frac{n_i}{N}\right]. \tag{4.9}$$

Table 4.1 shows the thermodynamic properties and the equilibrium quantities n_{eq} at $T = 600$ K and $p = 1$ bar. Using the equilibrium condition $\mu_{NO_2} = 2\,\mu_{N_2O_4}$, the equilibrium composition is calculated analytically from Equation (4.9). To investigate

Table 4.1: *Ideal gas model parameters and equilibrium state of the one-phase system* $NO_2 - N_2O_4$ *at* $T = 600$ K *and* $p = 1$ bar.

	s_0 [J/(mol K)]	c_p [J/(mol K)]	$\Delta_f h^0$ [kJ/mol]	n_{eq} [mol]
NO_2	204	45.8	33.1	0.31
N_2O_4	304	104	9.08	0.69

the sensitivity of the chemical equilibrium with respect to the standard state entropy s_i^{ref}, Equation (4.7) can be substantiated as follows:

$$\frac{\mathrm{d}}{\mathrm{d}\underset{\sim}{s}^{\text{ref}}}\begin{pmatrix}n_{\mathrm{NO_2}}\\ n_{\mathrm{N_2O_4}}\\ -2\,\mu_{\mathrm{NO_2}}\end{pmatrix} = -\begin{pmatrix}RT\left(n_{\mathrm{NO_2}}^{-1} - N^{-1}\right) & -RT/N & 2\\ -RT/N & RT\left(n_{\mathrm{N_2O_4}}^{-1} - N^{-1}\right) & 1\\ 2 & 1 & 0\end{pmatrix}^{-1} \cdot \begin{pmatrix}-T & 0\\ 0 & -T\\ 0 & 0\end{pmatrix}.$$

(4.10)

The numerical outcome is

$$\frac{\mathrm{d}}{\mathrm{d}\underset{\sim}{s}^{\mathsf{ref}}}\begin{pmatrix} n_{\mathrm{NO_2}} \\ n_{\mathrm{N_2O_4}} \end{pmatrix} = \begin{pmatrix} 0.015 & -0.030 \\ -0.030 & 0.060 \end{pmatrix} \mathrm{mol^2K/J}\,. \tag{4.11}$$

This calculated sensitivity is valid for the material constraint $2\,n_{\mathrm{NO_2}} + n_{\mathrm{N_2O_4}} = \mathrm{const.}$ The chain-rule yields the correct sensitivity regarding the molar fraction of $\mathrm{NO_2}$:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\underset{\sim}{s}^{\mathsf{ref}}}\left(\frac{n_{\mathrm{NO_2}}}{n_{\mathrm{NO_2}} + n_{\mathrm{N_2O_4}}}\right) &= \frac{1}{(n_{\mathrm{NO_2}} + n_{\mathrm{N_2O_4}})^2}\begin{pmatrix} n_{\mathrm{N_2O_4}} \\ -n_{\mathrm{NO_2}} \end{pmatrix}^{\mathrm{T}} \cdot \frac{\mathrm{d}}{\mathrm{d}\underset{\sim}{s}^{\mathsf{ref}}}\begin{pmatrix} n_{\mathrm{NO_2}} \\ n_{\mathrm{N_2O_4}} \end{pmatrix} \\ &= \begin{pmatrix} 0.019567 \\ -0.039133 \end{pmatrix} \mathrm{mol\,K/J}\,. \end{aligned} \tag{4.12}$$

As expected, increasing the standard entropy for $\mathrm{NO_2}$ stabilises this species and yields an increased equilibrium concentration. Furthermore, increasing $s^{\mathsf{ref}}_{\mathrm{NO_2}}$ has exactly the same effect as decreasing $s^{\mathsf{ref}}_{\mathrm{N_2O_4}}$ by twice the amount. This is a consequence of the equilibrium condition $\mu_{NO_2} = 2\,\mu_{N_2O_4}$.

4.2.3 Derivatives with respect to process parameters

For process simulation, the information about the derivative of state variables with respect to parameters in constitutive equations is of great value and accessible directly from the result of the 2nd order solver. A modelling task often involves variation of the process parameters to observe impact on the calculated state. In process optimisation (Section 4.4), and data reconciliation (Section 4.5), these derivatives are mandatory.

The derivative information obtained in Equation (4.4) was general and not restricted to parameters of the thermodynamic model. As described in Section 3.3, the canonical equation system is solved in combination with a set of constitutive equations:

$$\underset{\sim}{h}(\underset{\sim}{\zeta}(\underset{\sim}{\alpha}), \underset{\sim}{u}) = \underset{\sim}{0}\,. \tag{4.13}$$

Vector $\underset{\sim}{\alpha}$ is a contribution to the right hand side $\underset{\sim}{l}$ of the inner equation system. Furthermore, $\underset{\sim}{\alpha}$ is the only contribution in $\underset{\sim}{l}$, which is directly dependent on the process parameters $\underset{\sim}{u}$. Considering the sparse contribution of $\underset{\sim}{\alpha}$ to $\underset{\sim}{l}$ by the selection matrix $\underset{\approx}{E}_\alpha$ as introduced in Section 3.3.2, Equation (4.4) can be interpreted as

$$\frac{\mathrm{d}\underset{\sim}{\zeta}}{\mathrm{d}\underset{\sim}{\alpha}} = -\left(\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right)^{-1} \underset{\approx}{E}_\alpha \quad \text{or as a total differential} \quad \mathrm{d}\underset{\sim}{\zeta} = -\left(\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right)^{-1} \underset{\approx}{E}_\alpha\, \mathrm{d}\underset{\sim}{\alpha}\,. \tag{4.14}$$

As mentioned above, only $\underset{\sim}{\alpha}$ is dependent on $\underset{\sim}{u}$, such that

$$\frac{\mathrm{d}\underset{\sim}{\zeta}}{\mathrm{d}\underset{\sim}{u}} = -\left(\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right)^{-1} \underset{\approx}{E}_\alpha \frac{\mathrm{d}\underset{\sim}{\alpha}}{\mathrm{d}\underset{\sim}{u}}\,. \tag{4.15}$$

The change of $\underset{\sim}{\alpha}$ with respect to $\underset{\sim}{u}$ can be obtained from the total differential of $\underset{\sim}{h}$ as defined in Equation (4.13). With one selection matrix $\underset{\approx}{E}_x$ defined to map $\bar{\underset{\sim}{\zeta}} = \underset{\approx}{E}_x^{\mathrm{T}} \underset{\sim}{\zeta}$, the total differential is:

$$\mathrm{d}\underset{\sim}{h} = \left.\frac{\partial \underset{\sim}{h}}{\partial \bar{\underset{\sim}{\zeta}}}\right|_{\underset{\sim}{\alpha},\underset{\sim}{u}} \left.\frac{\partial \bar{\underset{\sim}{\zeta}}}{\partial \underset{\sim}{\alpha}}\right|_{\underset{\sim}{u}} \mathrm{d}\underset{\sim}{\alpha} + \left.\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{u}}\right|_{\bar{\underset{\sim}{\zeta}}} \mathrm{d}\underset{\sim}{u} \overset{!}{=} 0 \quad \Rightarrow \quad \left.\frac{\partial \underset{\sim}{h}}{\partial \bar{\underset{\sim}{\zeta}}}\right|_{\underset{\sim}{\alpha},\underset{\sim}{u}} \left.\frac{\partial \bar{\underset{\sim}{\zeta}}}{\partial \underset{\sim}{\alpha}}\right|_{\underset{\sim}{u}} \frac{\mathrm{d}\underset{\sim}{\alpha}}{\mathrm{d}\underset{\sim}{u}} = -\left.\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{u}}\right|_{\bar{\underset{\sim}{\zeta}}} . \tag{4.16}$$

Here, $\left.\partial \bar{\underset{\sim}{\zeta}}/\partial \underset{\sim}{\alpha}\right|_{\underset{\sim}{u}}$ can be substituted by the differential quotient from Equation (4.14), such that

$$\underbrace{\left[\left.\frac{\partial \underset{\sim}{h}}{\partial \bar{\underset{\sim}{\zeta}}}\right|_{\underset{\sim}{\alpha},\underset{\sim}{u}} \underset{\approx}{E}_x \left(\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right)^{-1} \underset{\approx}{E}_\alpha\right]}_{\underset{\approx}{J}} \frac{\mathrm{d}\underset{\sim}{\alpha}}{\mathrm{d}\underset{\sim}{u}} = \left.\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{u}}\right|_{\underset{\sim}{\zeta}} . \tag{4.17}$$

Matrix $\underset{\approx}{J}$ already was computed within the solution method itself (see Section 3.3.2). As mentioned in Section 3.9.1, a singular matrix $\underset{\approx}{J}$ indicates a linear dependency of constitutive equations in combination with the canonical system. With invertible $\underset{\approx}{J}$, the sensitivity of $\underset{\sim}{\alpha}$ with respect to the process parameters $\underset{\sim}{u}$ is given as

$$\frac{\mathrm{d}\underset{\sim}{\alpha}}{\mathrm{d}\underset{\sim}{u}} = \underset{\approx}{J}^{-1} \left.\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{u}}\right|_{\bar{\underset{\sim}{\zeta}}} , \tag{4.18}$$

and can be substituted into Equation (4.15), hence

$$\frac{\mathrm{d}\underset{\sim}{\zeta}}{\mathrm{d}\underset{\sim}{u}} = -\left(\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right)^{-1} \underset{\approx}{E}_\alpha \underset{\approx}{J}^{-1} \left.\frac{\partial \underset{\sim}{h}}{\partial \underset{\sim}{u}}\right|_{\underset{\sim}{\zeta}} . \tag{4.19}$$

$\left.\partial \underset{\sim}{h}/\partial \underset{\sim}{u}\right|_{\underset{\sim}{\zeta}}$ is easy to obtain by means of symbolic algebra, while the other matrices involved are already available for a solved process model. The equation obtained therefore provides valuable information with very little additional effort. Naturally, the derivatives of every set of derived quantities $\underset{\sim}{y}(\underset{\sim}{\zeta})$ can be obtained applying the chain-rule:

$$\frac{\mathrm{d}\underset{\sim}{y}}{\mathrm{d}\underset{\sim}{u}} = \left.\frac{\partial \underset{\sim}{y}}{\partial \underset{\sim}{\zeta}}\right|_{u} \frac{\mathrm{d}\underset{\sim}{\zeta}}{\mathrm{d}\underset{\sim}{u}} + \left.\frac{\partial \underset{\sim}{y}}{\partial \underset{\sim}{u}}\right|_{\zeta} . \tag{4.20}$$

As an important fact, Equation (4.19) can be evaluated as a post-calculation. Compared to an ordinary process simulation, no computational overhead is required during the iterations to solve the model.

4.2.4 Derivatives with respect to thermodynamic parameters

In order to gain the derivative of state variables with respect to thermodynamic parameters $\underset{\sim}{c}$, Equation (4.4) requires the derivative of the right hand side at constant

$\underset{\sim}{\zeta}$. But in a relevant simulation, $\underset{\sim}{\zeta}$ will consist of states transformed by state function transformations (see Section 2.5.2). The performed transformations are applied numerically and therefore only valid in the calculated point for a converged solution. With $\underset{\sim}{\hat{\zeta}}$ as the native[1] state of a system, $\partial\underset{\sim}{l}/\partial\underset{\sim}{c}\big|_{\underset{\sim}{\hat{\zeta}}}$ is available instead. The objective of this section is to find an explicit expression for $\partial\underset{\sim}{l}/\partial\underset{\sim}{c}\big|_{\underset{\sim}{\zeta}}$ based on $\partial\underset{\sim}{l}/\partial\underset{\sim}{c}\big|_{\underset{\sim}{\hat{\zeta}}}$.

Considering $\underset{\sim}{l}$ as a function of $\underset{\sim}{\hat{\zeta}}$ and $\underset{\sim}{c}$, whereas $\underset{\sim}{\hat{\zeta}}$ is a function of $\underset{\sim}{\zeta}$, for $\underset{\sim}{l} = \underset{\sim}{l}(\underset{\sim}{\hat{\zeta}}(\underset{\sim}{\zeta}, \underset{\sim}{c}), \underset{\sim}{c})$ the total differential is

$$\mathrm{d}\underset{\sim}{l} = \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} \left(\left.\frac{\partial \underset{\sim}{\hat{\zeta}}}{\partial \underset{\sim}{\zeta}}\right|_{\underset{\sim}{c}} \mathrm{d}\underset{\sim}{\zeta} + \left.\frac{\partial \underset{\sim}{\hat{\zeta}}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}} \mathrm{d}\underset{\sim}{c} \right) + \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}} \mathrm{d}\underset{\sim}{c}\,. \tag{4.21}$$

As $\underset{\sim}{\zeta}$ is constant, $d\underset{\sim}{\zeta} = 0$ and therefore

$$\left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\zeta}} = \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} \left.\frac{\partial \underset{\sim}{\hat{\zeta}}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\zeta}} + \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}}\,. \tag{4.22}$$

To obtain the missing expression for the derivative $\partial\underset{\sim}{\hat{\zeta}}/\partial\underset{\sim}{c}\big|_{\underset{\sim}{\zeta}}$, the total differential of $\underset{\sim}{\zeta}(\underset{\sim}{\hat{\zeta}}, \underset{\sim}{c})$ can be utilised:

$$\mathrm{d}\underset{\sim}{\zeta} = \left.\frac{\partial \underset{\sim}{\zeta}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}} \mathrm{d}\underset{\sim}{c} + \left.\frac{\partial \underset{\sim}{\zeta}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} \mathrm{d}\underset{\sim}{\hat{\zeta}} = 0 \quad \Rightarrow \quad \left.\frac{\partial \underset{\sim}{\hat{\zeta}}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\zeta}} = -\left(\left.\frac{\partial \underset{\sim}{\zeta}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} \right)^{-1} \cdot \left.\frac{\partial \underset{\sim}{\zeta}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}} = -\underset{\approx}{J} \cdot \left.\frac{\partial \underset{\sim}{\zeta}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}}\,. \tag{4.23}$$

Furthermore, the total differential $\underset{\sim}{l}(\underset{\sim}{\hat{\zeta}})$ at constant $\underset{\sim}{c}$ can be used to obtain $\partial\underset{\sim}{l}/\partial\underset{\sim}{\hat{\zeta}}\big|_{\underset{\sim}{c}}$:

$$\mathrm{d}\underset{\sim}{l} = \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} \mathrm{d}\underset{\sim}{\hat{\zeta}} = \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} \underset{\approx}{J}\mathrm{d}\underset{\sim}{\zeta} \quad \Rightarrow \quad \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\hat{\zeta}}}\right|_{\underset{\sim}{c}} = \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right|_{\underset{\sim}{c}} \underset{\approx}{J}^{-1}\,. \tag{4.24}$$

These results can be substituted into Equation (4.22). Hence

$$\left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\zeta}} = \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}} - \left.\frac{\partial \underset{\sim}{l}}{\partial \underset{\sim}{\zeta}}\right|_{\underset{\sim}{c}} \cdot \left.\frac{\partial \underset{\sim}{\zeta}}{\partial \underset{\sim}{c}}\right|_{\underset{\sim}{\hat{\zeta}}}\,. \tag{4.25}$$

All the terms on the right hand side of this equation can be easily obtained. However, the implementation of thermodynamic models in their native state function must have the functionality to provide derivatives with respect to thermodynamic parameters.

[1]regarding the underlying thermodynamic models

4.3 Sensitivity Analysis

4.3.1 Motivation

The most fundamental utilisation of the derivatives obtained in the previous section is in sensitivity analyses, i.e. to interpret the direct physical meaning of the derivatives. A sensitivity analysis can therefore be a substitute or supplement for a case-study, giving valuable information to set up a meaningful optimisation.

In general, to understand the derivatives of process properties with respect to process parameters it is necessary to understand the process model and to check the rationale of selected process constraints. In contrast to a descriptive process model, a predictive model must deliver relevant sensitivity information correctly. This is a necessary requirement to conduct further disciplines as for instance process optimisation. The effort to obtain a realistic set of process constraints is often underestimated. Unsuitable constraints yield non-optimal or even infeasible operating conditions.

4.3.2 Sensitivity analysis of an air compression process

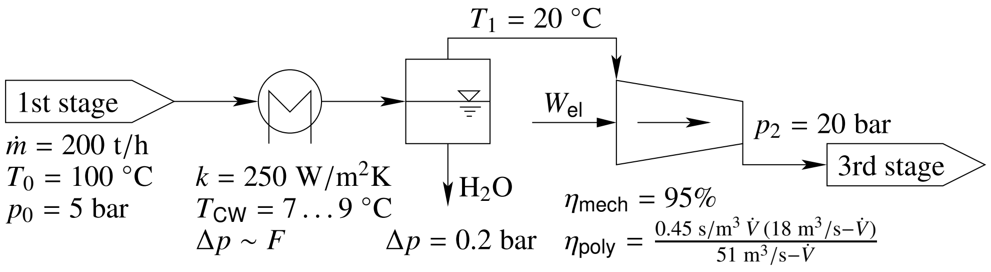


Figure 4.3: *A process air compression stage with inter-cooling.*

As an example, the second stage of a process air compression train is considered as shown in Figure 4.3. The pre-compressed air was originally saturated with water at 15 °C and 1 atm, and it is first cooled by cooling water. Condensate is removed in a separator, before the actual compression takes place. A larger surface area F in the heat exchanger will provide a lower T_1, but also increase pressure drop proportionally with a rate of 23 mbar/m^2. These effects suggest the existence of an optimal surface regarding a minimal compressor work duty. The S -R -W equation of state (Schwartzentruber *et al.*, 1990; Schwartzentruber and Renon, 1989) as described in Appendix B is used to calculate the properties of air and water in this chapter.

The core of the process simulation tool *Yasim* (cf. Chapter 5) is used to generate numerical results to the examples in this chapter. The process model topology is therefore defined as described in Section 3.4. All specifications shown in Figure 4.3 are formulated as constitutive equations. The algorithm in Section 3.3.2 is used to obtain the simulation results shown in Table 4.2. Applied on the solution, Equation (4.20) yields the derivatives with respect to the process parameter T_1. In

Table 4.2: *Sensitivity study results of an intermediate compressor stage.*

T_1 [°C]	F [m^2]	p_1 [bar]	W_{el} [MW]	T_2 [°C]	$\dot{m}_{\mathrm{H_2O}}$ [t/h]
20	578.9	4.67	10.6	197.6	1.74
$\partial\psi/\partial T_1$	-24.4	0.0056	**0.032**	1.44	-0.046
10	1028.4	4.56	10.4	184.8	2.09
$\partial\psi/\partial T_1$	-100.6	0.023	**-0.0042**	0.87	-0.025

the example, the derivative of compressor duty with respect to T_1 actually shifts sign, indicating an optimum within the range 10 °C $< T_1 <$ 20 °C.

Naturally, the derivatives and consequently the optimum will be highly dependent on those process parameters that remain constant, as for example the surface-specific pressure drop. Considering a constant pressure drop of 0.4 bar instead, the derivative $\partial W_{\mathrm{el}}/\partial T = 38.2$ kW/K at $T = 8$ °C is completely different from the reference case shown in Figure 4.3. There is no longer an indication for the existence of an optimal finite heat exchanger surface. Other constraints and formulation of objective functions would become relevant, not least the investment costs for the heat exchanger.

4.4 Process optimisation

4.4.1 Comparison to data reconciliation

Even though the scope of this work is limited to steady-state process models, process optimisation of these models still covers a wide range of applications, with a smooth transition towards the discipline of data reconciliation. In both cases, an objective function of state variables is optimised with respect to an independent set of process parameters. However, in data-reconciliation, the objective is to match redundant measurements in an optimal way, i.e. to describe inconsistent state information by one consistent state as well as possible. Process optimisation aims for an unknown state that optimises a given objective function. Section 4.2.3 describes a suitable way to obtain the J matrix $\underset{\approx}{J} = \mathrm{d}\underset{\sim}{y}/\mathrm{d}\underset{\sim}{u}$.

4.4.2 Selection of independent variables

Given a process model with a suitable set of process constraints, the actual set of independent variables within an optimisation is of secondary importance, as long as the desired DOFs are addressed, i.e. $\underset{\approx}{J}$ is not singular. Considering the example of Figure 4.3, an optimisation can be performed on the temperature T_1, on the heat removed in the cooler, or on the heat exchanger surface F with identical results. Obviously, it is a good choice to select independent variables for which most conceivability is given in terms of physically feasible domain and expected optimal value. Clearly, a prior sensitivity analysis can provide much of this inside knowledge.

4.4.3 Substitution of independent variables

Inequality constraints potentially add significantly to the complexity of an optimisation problem. The non-linear program is supplemented by a discrete set of conditions, converting it into a mixed integer non-linear program. With a suitable set of independent variables, many – in some cases all – inequality constraints can be hidden by substitution by bounded functions. The chain rule is applied to map $\partial\Lambda/\partial \underset{\sim}{u}$ to the sensitivity of the objective function with respect to the modified process parameter vector $\underset{\sim}{\hat{u}}$:

$$\frac{\partial\Lambda}{\partial\underset{\sim}{\hat{u}}} = \frac{\partial\Lambda}{\partial\underset{\sim}{u}}\,\frac{\partial\underset{\sim}{u}}{\partial\underset{\sim}{\hat{u}}}\,. \tag{4.26}$$

Consider a process parameter u bounded by $u_{\mathsf{min}} \le u \le u_{\mathsf{max}}$. A suitable substitution can be

$$u = (u_{\mathsf{max}} - u_{\mathsf{min}})\frac{1-\cos\pi\,\hat{u}}{2} + u_{\mathsf{min}} \quad \text{with} \quad \frac{\partial u}{\partial\hat{u}} = \frac{\pi}{2}(u_{\mathsf{max}} - u_{\mathsf{min}})\,\sin\pi\,\hat{u}\,. \tag{4.27}$$

Even if the solver overshoots into another period of the harmonic function, $\hat{u}$ still maps to a feasible value of u. However, $\partial u/\partial\hat{u} = 0$ at $\hat{u} \in \mathbb{Z}$, which is a singularity that has to be handled by the solver. As a positive side effect, such a substitution scales the independent variable into a defined and comparable dimensionless range, which is especially important for application of first-order optimisation methods.

Substitution of independent variables can also be used to reduce the problem dimension. Consider a separation column with individually heated/cooled stages. With u_i as the heat duty of stage i, $\dim\underset{\sim}{u}$ is unnecessarily high, as a certain continuity will be expected for proximate stages. Especially for preliminary optimisations, a profile function can reduce the dimension significantly, for instance in a linear form as follows:

$$u_i = \hat{u}_1 + i\,\hat{u}_2 \quad \text{with} \quad \frac{\partial u_i}{\partial(\hat{u}_1,\hat{u}_2)} = (1,\ i)\,. \tag{4.28}$$

4.4.4 Optimisation of compressor intake temperature

For the process introduced in Section 4.3.2, the temperature derivatives listed in Table 4.2 indicate the existence of an optimal compressor inlet temperature T_1. The existence of a minimum compressor duty is asserted at an intake temperature T_1 between 10 °C and 20 °C.

This example uses T_1 as the only one independent variable. The derivative $\partial W_{\mathsf{el}}/\partial T_1$ can be used to apply a first-order method to minimise the compressor duty. The secant method (Nocedal and Wright, 1999) yields the following update formula:

$$T_1^{(k+1)} = T_1^{(k)} - (\partial W_{\mathsf{el}}/\partial T_1)^{(k)}\,\frac{T_1^{(k-1)} - T_1^k}{(\partial W_{\mathsf{el}}/\partial T_1)^{(k-1)} - (\partial W_{\mathsf{el}}/\partial T_1)^{(k)}}\,. \tag{4.29}$$

With $T_1^{(0)}$ = 10 °C and $T_1^{(1)}$ = 20 °C, the subsequent iterations result in

$$T_1^{(2)} = 11.16\ °\text{C}, \quad T_1^{(3)} = 10.39\ °\text{C}, \quad T_1^{(4)} = 10.29\ °\text{C}, \quad T_1^{(5)} = 10.31\ °\text{C}, \ldots \tag{4.30}$$

The optimal temperature is $T_{1,\mathsf{opt}}$ = 10.3 °C, and W_{el} = 10.42 MW, F = 998 m^2 and T_2 = 185.1 °C.

This calculation is a practical example for the efficient use of analytical derivatives in process optimisation. The derivatives $\mathrm{d}y/\mathrm{d}u$ are calculated using Equation (4.20).

However, this tiny example already indicates challenges, which are beyond what can be solved by providing the technical framework. The realistic replication of actual process constraints, as mentioned in the previous section, has a significant impact on the location and even existence of an optimal point.

No less important is the formulation of the objective function, in particular, if penalty contributions of different metrics are to be combined. With this, it becomes clear how important it is to perform sensitivity studies prior to process optimisation. In many cases, the optimal process parameter is bounded by technical feasibility. For instance the compressor inlet should not be cooler than 5 °C in order to avoid icing on the compressor blades.

The treatment of such inequality constraints is essential to process optimisation, but decoupled from the canonical modelling approach. Edgar and Himmelblau (2001) give a comprehensive overview over the broad field of process optimisation.

4.5 Data reconciliation

4.5.1 Weighted least-squares method

As mentioned in Paragraph 4.4.1, data reconciliation is really a specialised case of process optimisation. The objective function measures the difference between calculated properties $\underset{\sim}{y}$ and measured properties $\underset{\sim}{y}_{\mathsf{meas}}$. The simplest applicable approach is to define a sum of weighted least-squares without constraints outside the process model itself:

$$\min_{\underset{\sim}{u}} \frac{1}{2}(\underset{\sim}{y}(\underset{\sim}{u}) - \underset{\sim}{y}_{\mathsf{meas}})\, \underset{\approx}{W}\, (\underset{\sim}{y}(\underset{\sim}{u}) - \underset{\sim}{y}_{\mathsf{meas}}) \quad \text{with } \underset{\approx}{W} \text{ as diagonal weight matrix.} \tag{4.31}$$

Exactly one process parameter u_i is selected as an independent variable for each DOF. As explained in Paragraph 4.4.3, inequality constraints of the process parameters can potentially be eliminated by substitution to ease the optimisation scheme. An overview of different objective functions with respect to gross error detection in particular is given by Özyurt and Pike (2004). The least-squares method yields a linear optimisation problem, but defect measurements contribute strongly. More advanced formulations are based on so-called *redescending influence functions*. These objective functions assign low weight to gross error measurements.

To solve the least-squares problem, Equation (4.31) is derived with respect to $\underset{\sim}{u}$. With $\underset{\approx}{J} = \mathrm{d}\underset{\sim}{y}/\mathrm{d}\underset{\sim}{u}$ from Equation (4.20), the zero-gradient condition becomes

$$\underset{\approx}{J}^{\mathrm{T}}\,\underset{\approx}{W}(\underset{\sim}{y}(\underset{\sim}{u}) - \underset{\sim}{y}_{\text{target}}) = \underset{\sim}{0}\,. \tag{4.32}$$

Linearisation of $\underset{\sim}{y}(\underset{\sim}{u})$ in $\underset{\sim}{u}^{(k)}$ yields the well known equation for the weighted linear least-squares problem, which is an overdetermined equation system:

$$\underset{\approx}{J}^{\mathrm{T}}\,\underset{\approx}{W}\,\underset{\approx}{J}(\underset{\sim}{u}^{(k+1)} - \underset{\sim}{u}^{(k)}) = \underset{\approx}{J}^{\mathrm{T}}\,\underset{\approx}{W}\,(\underset{\sim}{y}_{\text{target}} - \underset{\sim}{y}(\underset{\sim}{u}^{(k)}))\,. \tag{4.33}$$

Figure 4.4 shows the main strategy to implement data reconciliation based on the

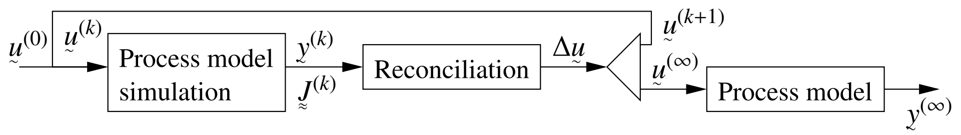

$\underset{\sim}{u}^{(0)}$
$\underset{\sim}{u}^{(k)}$
Process model simulation
$\underset{\sim}{y}^{(k)}$
$\underset{\approx}{J}^{(k)}$
Reconciliation
$\Delta\underset{\sim}{u}$
$\underset{\sim}{u}^{(k+1)}$
$\underset{\sim}{u}^{(\infty)}$
Process model
$\underset{\sim}{y}^{(\infty)}$

Figure 4.4: *Flow-diagram of a data reconciliation process.*

canonical flowsheet simulation. The process model delivers $\underset{\sim}{y}(\underset{\sim}{u})$ and $\underset{\approx}{J}(\underset{\sim}{u})$ as a representation of a linearised process model. The reconciliation block evaluates the regression Equation (4.33) to update the independent process parameters $\underset{\sim}{u}$. The converged set of independent parameters $\underset{\sim}{u}^{(\infty)}$ is applied to the process model to obtain the complete set of reconciled data $\underset{\sim}{y}^{(\infty)}$. It is advisable to converge the process simulation step before any reconciliation step. Not only is the derivative obtained by Equation (4.33) valid only at a converged simulation result, but even the intensive variables themselves only receive their physical interpretation at the solution point. For instance, the L multiplier, which in the solution point is interpreted as pressure, can assume large negative values during the iteration procedure.

4.5.2 Data reconciliation of a compressor stage model

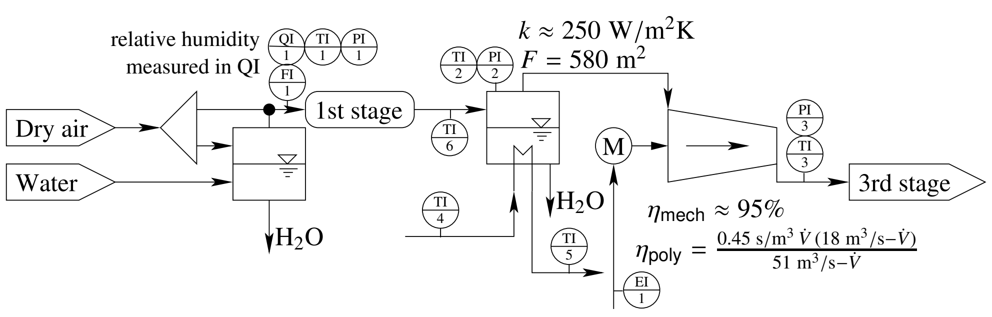

relative humidity measured in QI
QI 1
TI 1
PI 1
FI 1
Dry air
Water
H_2O
1st stage
TI 6
TI 2
PI 2
$k \approx 250\ \mathrm{W/m^2K}$
$F = 580\ \mathrm{m^2}$
TI 4
H_2O
TI 5
M
EI 1
PI 3
TI 3
3rd stage
$\eta_{\text{mech}} \approx 95\%$
$\eta_{\text{poly}} = \frac{0.45\ \mathrm{s/m^3}\ \dot{V}\,(18\ \mathrm{m^3/s} - \dot{V})}{51\ \mathrm{m^3/s} - \dot{V}}$

Figure 4.5: *Process model of the compressor stage for data reconciliation.*

Figure 4.5 shows the process flowsheet of the compressor train, slightly modified to suit the data reconciliation case. The relative humidity of air determines the split factor to supply dry and saturated air into the first compressor stage. Ambient air conditions (QI1, PI1 and TI1) and cooling water inlet temperature (TI4) are typically measured outside the actual process. The nominal compressor efficiency (both mechanical and isentropic), and heat transfer in the heat exchanger are used as indirect measurements, i.e. the empirically calculated efficiency and heat transfer are used as if they were measurements. Weight factors can be employed to use the indirect measurements actively in order to reconcile the only flow measurement. Alternatively it is an option to just monitor heat transfer and compressor efficiency in order to observe operational problems (e.g. fouling and corrosion).

Originating from the base-case, a set of distorted potential measurements is created, adding a statistical error, a systematic error, and, for some quantities, a drift of the data to replicate real measurements as input for a data reconciliation run. However, in order to concentrate on the general principles, no gross-errors have been generated within the measurements in this example. The volume flow measurement

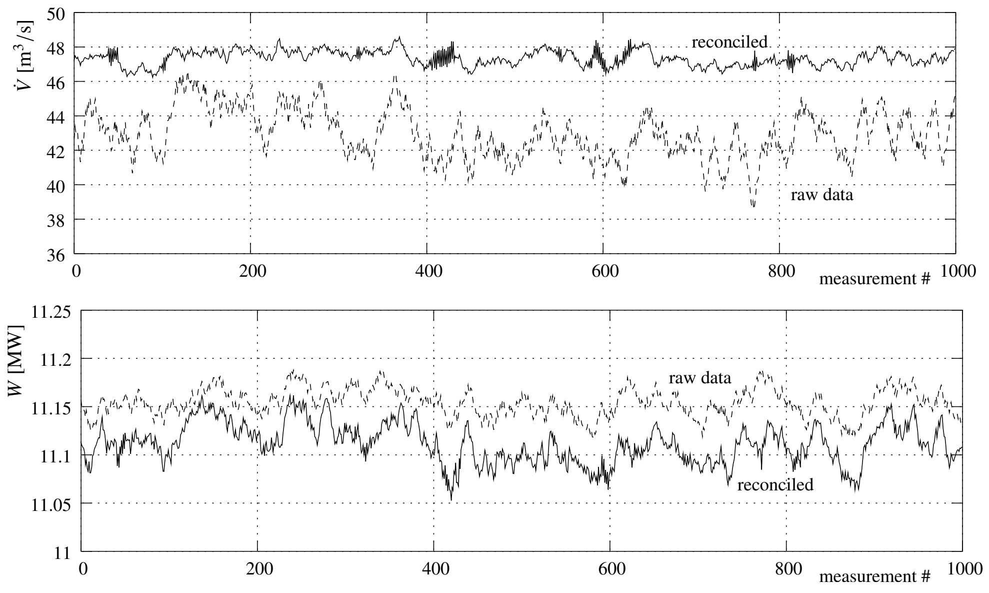


Figure 4.6: *Reconciled data series of volume flow and compressor effect.*

is redundant to the compressor energy duty and affecting temperature and pressure measurements. From Figure 4.6, a systematic error can clearly be identified. The volume flow is measured too low, and/or the compressor duty is measured too high. Trusting both measurements simultaneously, the reconciled values stay in between. Figure 4.7 shows typical data, which is not directly measurable, but calculated process properties as a result of the data reconciliation. Such data is of special interest for a process operator. With measured cooling water temperatures and heat exchanger surface, the heat transfer coefficient can be determined. In spite of dominant statistical errors, a slight trend towards lower conductivity can be observed, which might

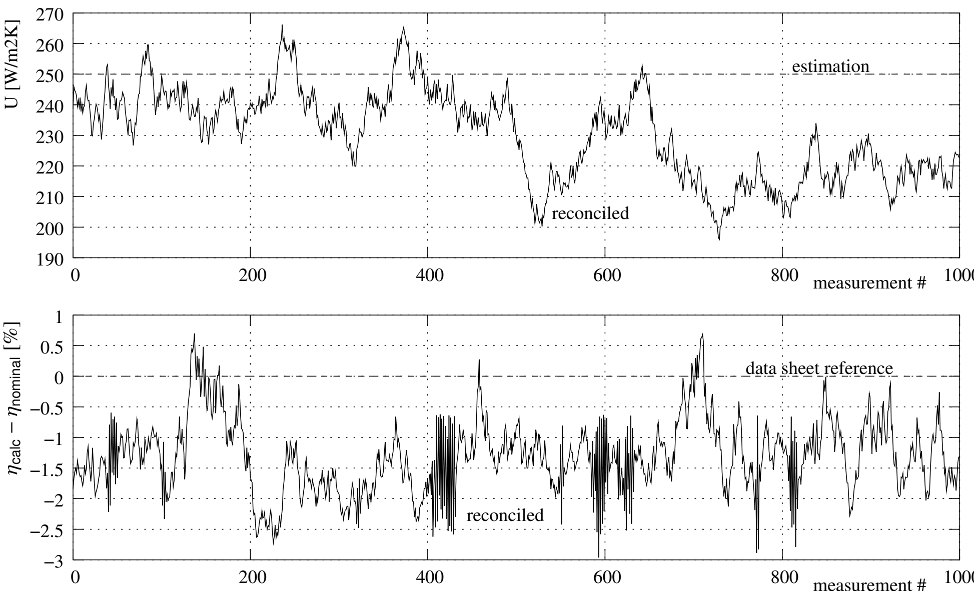


Figure 4.7: *Reconciled data series of heat transfer coefficient and compressor efficiency.*

indicate a fouling problem.

The compressor efficiency in this example stays approximately 1% below the nominal values, but does not show a deteriorating behaviour. The deviation can be caused by systematic measurement errors or a loss of performance on a larger time scale.

4.6 Exergy analysis

As is obvious from the problem formulations of the previous sections, process modelling is a key factor for improving chemical processes, during both the design phase and operation. Data reconciliation enhances accuracy through the appropriate interpretation of available measurements, allowing one to tune more precisely towards a target state of operation, while process optimisation actually determines an optimal target state. In this regard, exergy analysis can help to first identify inefficient process parts and then to estimate a potential improvement, based on the second law of thermodynamics.

4.6.1 Concepts of second law analysis

The literature often defines exergy solely considering temperature gradients (Callen, 1985; Tester and Modell, 1997), while not considering chemical reactions or pressure changes. This special case yields the C -efficiency $\eta = 1 - T_0/T$, while Wall

(1986) uses a more general definition, namely:

> *Exergy is the totally convertible part of the energy, i.e. that part which may be converted into any other energy form.*

However, since chemical potentials will be considered here, the definition of an ambient chemical potential for each chemical species is required. For a consistent description, one recipient species for each chemical element is sufficient, as it is shown below. Furthermore, the concept of exergy is often mixed up with that of available energy. This work uses therefore following definitions:

Terms and definitions 4.1

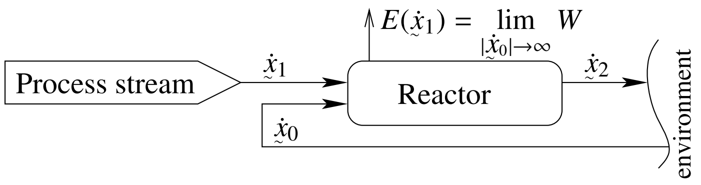
Exergy The totally convertible part of energy in a *stream* represented by a state $\dot{\underset{\sim}{x}}$. Exergy is based on enthalpy as the conserved property regarding the first law of thermodynamics for adiabatic stream-based systems.

Available energy The totally convertible part of energy in an *accumulated state* represented by $\underset{\sim}{x}$. Available energy is based on internal energy as the conserved property regarding the first law of thermodynamics for closed systems.

In particular, it is meaningless to define exergy on an accumulated state like the content of a buffer tank, or to define available energy on a stream.

As in this work, the main focus is put on steady-state processes, exergy is the measure for second law analysis in this section.

4.6.2 Definition of exergy



$E(\dot{\underset{\sim}{x}}_1) = \lim_{|\dot{\underset{\sim}{x}}_0|\to\infty} W$
Process stream
$\dot{\underset{\sim}{x}}_1$
Reactor
$\dot{\underset{\sim}{x}}_2$
$\dot{\underset{\sim}{x}}_0$
environment

Figure 4.8: *Process flowsheet to define the exergy of a process stream. The limit* $|\dot{\underset{\sim}{x}}_0| \to \infty$ *indicates a process of infinite size or infinite time.*

Given a stream of an arbitrary state $\dot{\underset{\sim}{x}}_1$, the exergy E is defined as the maximum amount of work that can be extracted by conforming the intensive properties towards the intensive state of a defined infinite reservoir (from now this will be referred to as *environment*). To archive the environmental state, species are reactants of chemical reactions. The product species are most stable in the environment, and constrained by the balance equations of chemical elements, there is exactly one such *recipient species* for each present chemical element.

Figure 4.8 shows a possible setup to obtain a suitable mathematical definition of exergy. The environmental intensive state is reached by infinitely diluting the process stream with a stream of environmental state. Naturally infinite streams can not be evaluated by means of straightforward process simulation, but the following derivation describes a way to obtain exergy as a flow property.

All chemical species are converted by chemical reaction into recipient species. In spite of heat, work is allowed to be exchanged, such that $\dot S_2 \geq \dot S_0 + \dot S_1$, furthermore $W = \dot H_0 + \dot H_1 - \dot H_2$.

Substituting the E -integrated representation of H into the latter equation yields (note the mass balance $\underset{\sim}{\dot n}_2 = \underset{\sim}{\dot n}_0 + \underset{\approx}{A}\,\underset{\sim}{\dot n}_1$):

$$\begin{aligned}
W &= T_1\,\dot S_1 + T_0\,\dot S_0 - T_2\,\dot S_2 + \underset{\sim}{\mu}_1\,\underset{\sim}{\dot n}_1 + \underset{\sim}{\mu}_0\,\underset{\sim}{\dot n}_0 - \underset{\sim}{\mu}_2\,\underset{\sim}{\dot n}_2 \\
&= T_1\,\dot S_1 + T_0\,\dot S_0 - T_2\,(\dot S_0 + \dot S_1) + \underset{\sim}{\mu}_1\,\underset{\sim}{\dot n}_1 + \underset{\sim}{\mu}_0\,\underset{\sim}{\dot n}_0 - \underset{\sim}{\mu}_2\,(\underset{\sim}{\dot n}_0 + \underset{\approx}{A}\,\underset{\sim}{\dot n}_1) \\
&= (T_1 - T_0)\,\dot S_1 + (\underset{\sim}{\mu}_1 - \underset{\approx}{A}^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot n}_1 \\
&\qquad - (T_2 - T_0)(\dot S_0 + \dot S_1) - (\underset{\sim}{\mu}_2 - \underset{\sim}{\mu}_0)(\underset{\sim}{\dot n}_0 + \underset{\approx}{A}\,\underset{\sim}{\dot n}_1)\,.
\end{aligned} \tag{4.34}$$

Here, the inequality for entropy is substituted by an equality, as any higher $\dot S_2$ due to irreversibility would clearly reduce W by the positive product $T_2\,\Delta\dot S_{\mathsf{irreversible}}$. To calculate the limit at $|\underset{\sim}{\dot x}_0| \to \infty$, T_2 and $\underset{\sim}{\mu}_2$ are approximated by linearisations around $\underset{\sim}{\dot x}_0$ at constant $\dot S$ and p:

$$T_2 \approx T_0 + \left.\frac{\partial T}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} (\underset{\sim}{\dot n}_2 - \underset{\sim}{\dot n}_0) = \left.\frac{\partial T}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} \underset{\approx}{A}\,\underset{\sim}{\dot n}_1 \quad \text{and} \quad \underset{\sim}{\mu}_2 \approx \left.\frac{\partial \underset{\sim}{\mu}}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} \underset{\approx}{A}\,\underset{\sim}{\dot n}_1\,. \tag{4.35}$$

Using the symmetry of $\partial\underset{\sim}{\mu}/\partial\underset{\sim}{\dot n}$, the work can be written as

$$W = (T_1 - T_0)\,\dot S_1 + (\underset{\sim}{\mu}_1 - \underset{\approx}{A}^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot n}_1 - \underset{\approx}{A}\,\underset{\sim}{\dot n}_1 \left[\left.\frac{\partial T}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} (\dot S_0 + \dot S_1) + \left.\frac{\partial \underset{\sim}{\mu}}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} (\underset{\sim}{\dot n}_0 + \underset{\approx}{A}\,\underset{\sim}{\dot n}_1) \right]. \tag{4.36}$$

With the homogeneity property of enthalpy $\partial^2 H/\partial(S,\underset{\sim}{n})^2 \cdot (S,\underset{\sim}{n}) = \underset{\sim}{0}$, this equation simplifies to

$$W = (T_1 - T_0)\,\dot S_1 + (\underset{\sim}{\mu}_1 - \underset{\approx}{A}^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot n}_1 - \underset{\approx}{A}\,\underset{\sim}{\dot n}_1 \left[\left.\frac{\partial T}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} \dot S_1 + \left.\frac{\partial \underset{\sim}{\mu}}{\partial \underset{\sim}{\dot n}}\right|_{\dot S,p} \underset{\approx}{A}\,\underset{\sim}{\dot n}_1 \right]. \tag{4.37}$$

The partial derivatives are reciprocally proportional to $|\underset{\sim}{\dot x}_0|$. The expression of exergy of a general stream $\underset{\sim}{\dot x}$ is therefore

$$E(\underset{\sim}{\dot x}) = \lim_{|\underset{\sim}{\dot x}_0| \to \infty} W = (T - T_0)\,\dot S + (\underset{\sim}{\mu} - \underset{\approx}{A}^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot n} = H - T_0\,\dot S - \underset{\sim}{\mu}_0\,\underset{\approx}{A}\,\underset{\sim}{\dot n} \tag{4.38}$$

As an example, consider a stream of a pure species or azeotropic mixture within two-phase equilibrium conditions. Similar to a reservoir, adding heat will not influence its intensive state, in particular T and $\underset{\sim}{\mu}$. The change in exergy is therefore simply given by $\Delta E = (T - T_0)\,\Delta S$ with $\Delta H = T\,\Delta S$, which yields the C efficiency

$$\eta_{\mathsf{Carnot}} = \frac{\Delta E}{\Delta H} = 1 - \frac{T_0}{T}\,. \tag{4.39}$$

Considering the E -integrated representation of enthalpy, the definition of exergy according to Equation (4.38) can be interpreted as the tangent plane distance of enthalpy. Due to the convexity of $\dot{H}$, this distance function is positive for all $\underset{\sim}{x}_0 = (\dot{S}, p_0, \underset{\sim}{\dot{n}})$, if $\underset{\sim}{x}_0$ is at chemical equilibrium, i.e. no spontaneous chemical reactions are possible in the environment.

However, pressures below p_0 yield a negative exergy contribution, describing the work necessary to compress the stream to environmental pressure. The pressure dependency of exergy is

$$\left.\frac{\partial E}{\partial p}\right|_{\dot{S},\dot{n}} = \left.\frac{\partial \dot{H}}{\partial p}\right|_{\dot{S},\dot{n}} = \dot{V} > 0 \quad \text{for} \quad \underset{\sim}{\dot{n}} > 0\,. \tag{4.40}$$

There is an important difference between exergy and available energy, as positive work can be extracted from an accumulated state at vacuum. Available energy is therefore always non-negative.

With a fixed T_0 and $\underset{\sim}{\mu}_0$, the exergy defined as in Equation (4.38) is purely a function of canonical variables, and can therefore be a contribution to constitutive equations in the simulation context. With E being a process property y, Equation (4.20) can be applied, and exergy analysis can be combined smoothly with the tasks of sensitivity analysis and process optimisation.

4.6.3 Selection of an ambient state

The ambient state is in general different from the reference state of the underlying thermodynamic model. The latter one depends on the availability of data, hence most models are based on $T_{\text{ref}} = 298.15$ K and $p_{\text{ref}} = 1$ bar. Merely the chemical potential is easily converted to different recipient species by a linear enthalpy shift.

The ambient state could be selected freely depending on the environment of the considered process, but this selection poses a practical problem in many cases. In general, cooling water has a different temperature than ambient air. Selecting the air-temperature as T_0, the exergy of cooling water is found to be positive, which causes lower efficiency values for process parts dealing with cooling water. Selecting cooling water temperature as T_0, process parts interacting with ambient air are disadvantaged. Even if only differences in exergy are evaluated, the non-linearity of exergy still yields a dependency of T_0 for irreversible processes. For this reason, a suitable individual ambient temperature has to be selected for each process part in order to compare results of an exergy analysis. A possible way to couple process parts of different ambient conditions into one exergy analysis is discussed in the next section.

However, the chemical potential $\underset{\sim}{\mu}_0$ does not contribute to the exergy balances, as long as the atom balance is fulfilled. Consider an isothermal reactor at T_0 with an input stream $\underset{\sim}{\dot{x}}_1$ and a product stream $\underset{\sim}{\dot{x}}_1$, where the reaction is constrained by

$\underset{\sim}{\dot{n}}_2 = \underset{\approx}{A}_{1,2}\,\underset{\sim}{\dot{n}}_1$. The change of exergy is

$$\begin{aligned}\Delta E &= (\underset{\sim}{\mu}_2 - \underset{\approx}{A}_2^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot{n}}_2 - (\underset{\sim}{\mu}_1 - \underset{\approx}{A}_1^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot{n}}_1 = (\underset{\approx}{A}_{1,2}^{\mathrm{T}}\,\underset{\sim}{\mu}_2 - \underset{\approx}{A}_1^{\mathrm{T}}\,\underset{\sim}{\mu}_0 - \underset{\sim}{\mu}_1 + \underset{\approx}{A}_1^{\mathrm{T}}\,\underset{\sim}{\mu}_0)\,\underset{\sim}{\dot{n}}_1\\ &= (\underset{\approx}{A}_{1,2}^{\mathrm{T}}\,\underset{\sim}{\mu}_2 - \underset{\sim}{\mu}_1)\,\underset{\sim}{\dot{n}}_1\,. \qquad (4.41)\end{aligned}$$

The selection of ambient pressure p_0 affects E only indirectly by the pressure-dependency of the ambient chemical potential $\underset{\sim}{\mu}_0$. Exergy differences are therefore as well independent of ambient pressure.

4.6.4 Processes of multiple ambient states

Most plants have access to more than one reservoir, typically water and air with different temperatures and chemical potentials of the recipient species. Generally, one could suggest to exploit the driving-forces in an infinitely sized engine, and thereby assign the zero efficiency to all finite processes. This strict criterion is obviously not suitable for real processes.

However, without entering the deep subject of finite-time thermodynamics, it can be observed that in order to combine two reservoirs within one process, at least one of them has to be acquired, e.g. by a material stream. The process must be separated into sub-processes with a definite ambient reservoir associated to each of them. This approach requires a minimum of process insight, namely which streams are exchanged between sub-processes within different environments. Within these sub-processes, the ambient conditions are used to define exergy. Consequently, the calculated value of exergy steps up or down on the interfaces between them.

A step downwards means that exergy, which could have been utilised in the source environment, is wasted into the other system, where it is less valuable – similarly to exporting goods to a country with lower prices for this article. Clearly, such a transition must be considered as a loss. A step upwards, however, indicates a potential for utilisation of a finite amount of exergy from one reservoir within another. The gain in exergy is clearly an input to the downstream process.

This approach does not require a process to utilise a potential difference in available ambient states, but once a process acquires exergy from one environment within another, it is considered as input to the process – hence a loss if not exploited.

4.6.5 Relative exergy efficiency

It is in general a bad idea to define key performance indicators as quotients of energy figures, as the zero-level is arbitrarily chosen. For an oil-pipeline, an efficiency of nearly 100% is calculated, if the heat of formation of chemical species is considered. A more suitable approach then considers only the pressure drop, as the pipe does not (and is not supposed to) utilise the oil's heat of combustion. The efficiency based on energy or exergy is therefore zero, which is typical for any kind of horizontal transport. Sorin *et al.* (2000) therefore introduces *transiting* exergy as the unaffected part, *consumed* exergy as the input exergy to be converted, and *produced* exergy as output. Considering complex processes, it is a challenge to assign these fractions correctly, and necessary information might not be available. Sorin *et al.* (1998b), Sorin *et al.* (1998a) and Siepmann *et al.* (2001) invested effort to provide a consistent basis for comparability. Hinderink *et al.* (1996) also suggest a split of exergy into different contributions, to which they refer to as *mixing*, *chemical* and *physical* exergy. However, considering the canonical approach, it is more natural to consider contributions associated to canonical state variables, hence a thermal, mechanical and a chemical part based on changes in T, p and $\underset{\sim}{\mu}$:

$$E = \underbrace{(\underset{\sim}{\mu}(T_0, p_0) - \underset{\approx}{A}^{\mathrm{T}}\underset{\sim}{\mu}_0))\dot{\underset{\sim}{n}}}_{E_{\mathsf{ch}}(\dot{\underset{\sim}{n}})} + \underbrace{(\underset{\sim}{\mu}(T_0, p) - \underset{\sim}{\mu}(T_0, p_0))\dot{\underset{\sim}{n}}}_{E_{\mathsf{mc}}(p, \dot{\underset{\sim}{n}})} + \underbrace{(T - T_0)\dot{S} + (\underset{\sim}{\mu} - \underset{\sim}{\mu}(T_0, p))\dot{\underset{\sim}{n}}}_{E_{\mathsf{th}}(T, p, \dot{\underset{\sim}{n}})} \, . \tag{4.42}$$

Other decompositions are possible, e.g. $E = E_{\mathsf{th}}(T) + E_{\mathsf{mc}}(T, p) + E_{\mathsf{ch}}(T, p, \dot{\underset{\sim}{n}})$, but less practical, if the ambient chemical potential must be evaluated at process conditions, and a thermodynamic model must be available to perform such a calculation.

As an example to clarify the benefit of a decomposition as in Equation (4.42), a hydrogen burner to generate high pressure steam from condensate is considered.

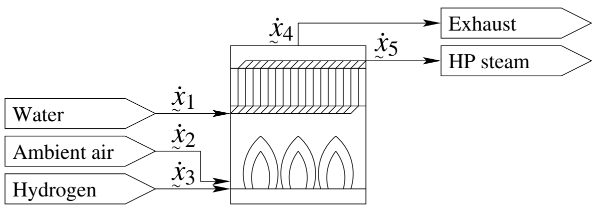

$\dot{\underset{\sim}{x}}_4$
Exhaust
$\dot{\underset{\sim}{x}}_5$
HP steam
Water
$\dot{\underset{\sim}{x}}_1$
Ambient air
$\dot{\underset{\sim}{x}}_2$
Hydrogen
$\dot{\underset{\sim}{x}}_3$

Figure 4.9: *Hydrogen burner to generate high pressure steam.*

Table 4.3: *Stream table of the hydrogen combustion process. In the scope of this table, x_i denotes the mole fraction of species i.*

	Condensate	Ambient air	Hydrogen	Exhaust	Steam
T [°C]	90.0	20.0	20.0	123.5	393.2
p [bar]	45.0	1.013	200.0	1.013	45.0
m [kg/h]	295.1	346.0	7.26	353.2	295.1
x_{N_2} [%]		77.7		67.6	
x_{O_2} [%]		20.7		5.0	
x_{H_2} [%]			100		
x_{H_2O} [%]	100	1.6		27.4	100
E [kW]	8.8	0.0	250.3	7.5	105.6

The process is shown in Figure 4.9, supplemented by the stream table 4.3 from the process simulation. The total exergy figures are based on ambient air and evaluate to an absolute exergy loss of 146 kW with exhaust gas considered as a byproduct to be utilised later. A plain quotient of outgoing divided by incoming exergy suggests an efficiency of $\eta_{\mathsf{max}} = 44\%$, assuming all exergy being converted. Considering the process as a black box and only viewing the exergy figures, the assumption might be that no exergy is converted at all. The amount of 113.1 kW would be assigned to transiting exergy, and 259.1 kW accounted for as loss. This interpretation results into $\eta_{\mathsf{min}} = 0\%$. The true efficiency η is therefore constrained by $\eta_{\mathsf{min}} < \eta < \eta_{\mathsf{max}}$. However, the process gas and the steam systems are two decoupled material systems. Due to the second law of thermodynamics, the exergy increase in one material system can only be explained by internal exergy conversion, so it is possible to find a better lower limit: $\eta_{\mathsf{min}} = (E_5 - E_1)/(E_3 + E_2 - E_4) = 40\%$. Decomposing the exergy values,

Table 4.4: *Decomposed exergy E* [kW] *of the hydrogen combustion process.*

System	Type	Input	Output	ΔE	Comment
	thermal	0.0	1.76	1.76	heat in exhaust gas
process	mechanical	13.2	0.0	-13.2	pressure drop of hydrogen fuel
	chemical	237.1	5.7	-231.4	heat of combustion
	thermal	2.7	99.4	96.7	evaporation of water
steam	mechanical	0.36	0.36	0.0	constant steam pressure
	chemical	5.8	5.8	0.0	no chemical reactions

as defined in Equation (4.42), yields values as reported in Table 4.4. Input and output figures are balanced for each material system. In this example, the exergy of process input is delivered through the hydrogen feed, while the exhaust gas represents the output. Water and HP steam represent respectively the input and the output for the steam system.

Without any knowledge about the process, it is clear that differences in net values are consumed (negative) and produced (positive) fractions. Assuming all other exergy to be transiting, η_{min} can be recalculated as $\eta_{\mathsf{min}} = \frac{1.76+96.7}{13.2+231.4} = 40.25\%$. Even if more exergy was converted in practice, this conversion would not be necessary to provide the functionality of the process, such that $\eta = 40.25\%$ is a representative figure. The decomposed exergy figures also quantitatively indicate reasons for irreversible effects, e.g. the loss of 13.2 kW (5.4%) mechanical exergy due to non-utilised expansion of high pressure hydrogen gas.

This approach considers the exergy of the exhaust gas as a product. In fact, downstream processes can utilise the temperature and composition difference to ambient air, and it is not a property of the considered hydrogen burner process whether this is done or not. However, one might include the stack into the process. The stack has zero efficiency, as no work is extracted, while the ambient state is approached. The efficiency in this case is $\eta_{\mathsf{min}} = \frac{96.7}{13.2+237.1} = 38.63\%$.

However, an exergy analysis of this kind requires some amount of logical and computational overhead compared to the basic process simulation. In order to apply Equation (4.42), the chemical potentials in each considered stream have to be evaluated not only for (T, p), but as point calculations also for (T_0, p_0) and (T_0, p). The available thermodynamic models might not be predictive at ambient conditions. Furthermore, even though the total exergy is a derived property of canonical state variables, this is not the case for its contributions. This detailed exergy analysis is therefore not easily applicable to process optimisation.

As a solution to the problem, a specialised FM can be implemented to evaluate the state not only at process conditions, but as well at (T_0, p_0) and (T_0, p). Such a FM can evaluate the exergy figures required for the detailed analysis described in this section.

Chapter 5

Yasim

5.1 Introduction

In parallel to the development of methods and technologies as a basis for canonical modelling, an actual process simulator tool called *Yasim* has been designed and implemented in this work. The name *Yasim* is an abbreviation for *Yara simulator*, as its first industrial applications and therefore a significant driving force for development of a graphical user interface were simulation assignments of the international fertiliser producer *Yara International ASA*. In particular urea synthesis processes require a strong flexibility regarding thermodynamic modelling and handling of numerous significant recycle streams and external constitutive equations. Despite high licence costs for commercial software, the required flexibility for this kind of modelling was not available. It is in particular problematic to find a flexible equation oriented process simulator, which supports tailor-made thermodynamic models in a consistent maintainable framework.

Yasim is therefore developed as a canonical process simulator also driven by industrial needs instead of pure academic aspects. The aim of design is therefore in particular a suitable mix of flexibility and simplicity. The main concept can be described as follows:

Solve simple problems in a simple way, and make it possible to solve advanced tasks.

Furthermore, three different levels of process knowledge are identified as shown in Figure 5.1. The computer requires a mathematical representation of a process model. This primary process information includes not more than a set of variables ψ_i and equations, as well as suitable initial values and numerical specifications of process parameters. Internally, a heat transfer equation has the form $\psi_1 - \psi_2\,\psi_3(\psi_4 - \psi_5) = 0$ with $\psi_5 = 298.15$. However, this representation is of little value for the human engineer, and reverse engineering towards a more understandable form is difficult. A process modelling tool must therefore preserve e.g. the physical interpretation of

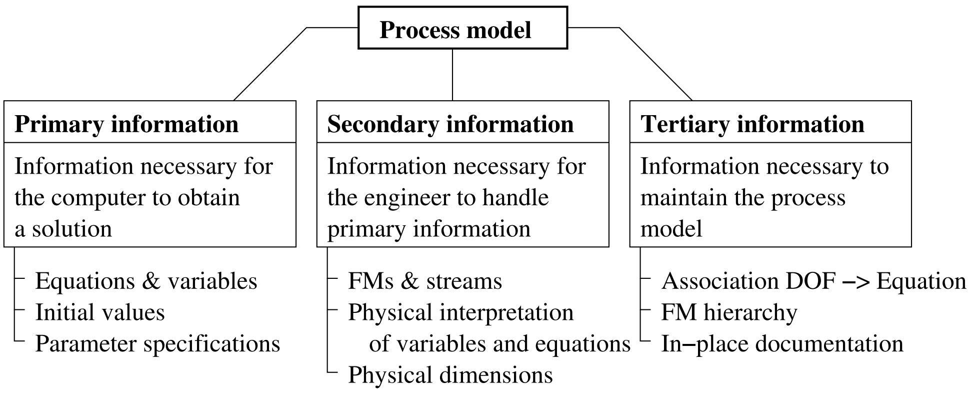

Process model
Primary information
Information necessary for the computer to obtain a solution
Equations & variables
Initial values
Parameter specifications
Secondary information
Information necessary for the engineer to handle primary information
FMs & streams
Physical interpretation of variables and equations
Physical dimensions
Tertiary information
Information necessary to maintain the process model
Association DOF –> Equation
FM hierarchy
In–place documentation

Figure 5.1: *Different levels of process knowledge.*

variables and equations as secondary information. In this case, we have a heat transfer equation formulated as $Q - (kF)(T - T_0) = 0$ with $T_0 = 25$ °C. The tertiary information is important to pick up and re-understand a process model, even with many weeks between the creation of the model and the continuation of the work. It is furthermore of high value for new engineers, who get involved into the development and maintenance of an existing process model. In today's practice, this is typically put into reports besides the process model and easily yields inconsistencies between documentation and the actual process model. It is therefore desirable to enforce as much self-documentation as possible.

The analysis and maintenance of degrees of freedom (DOFs) is a central issue in process modelling. Most tools offer two big containers, one for equations and one for variables – simulation is possible if both containers are equally full. The canonical modelling basis in *Yasim* however allows for *one-to-one mappings between DOFs and equations*. This is very useful, in particular to comprehend the intentions and thoughts of the process model's author.

Yasim consists of two main parts: An inner core that implements the administration of thermodynamic models, process models, model parameterisation and all the calculations including the solution scheme described in Section 3.3.2. This kernel is written in the programming language *C*++ and provides a programmer's interface as a set of libraries. The functionality available on this level covers the complete scope of *Yasim*.

The second main part is a graphical user interface, which has been developed using *Microsoft Visio* as a front-end. Through this interface, the basic functionality has been used efficiently in various projects within the research facilities of *Norsk Hydro ASA* and *Yara International ASA* in Porsgrunn, Norway.

This chapter concentrates on the main design aspects of the calculation core, which are based on the derived concepts of the previous chapters, but supplemented to enhance maintainability and ensure consistency of process models. Section 5.3 gives an overview over design features on the top level, after the basic concept is explained in the following section. In particular, a detailed and complete documentation of the entire software is not in scope of this work.

5.2 General process modelling approach

This section describes a general approach to establish a process model, which is not necessarily limited to the canonical approach. However, each step is naturally associated to certain concepts of this work, such that a brief discussion will clarify the context of the following sections. As shown in Figure 5.2, the first sub-task for

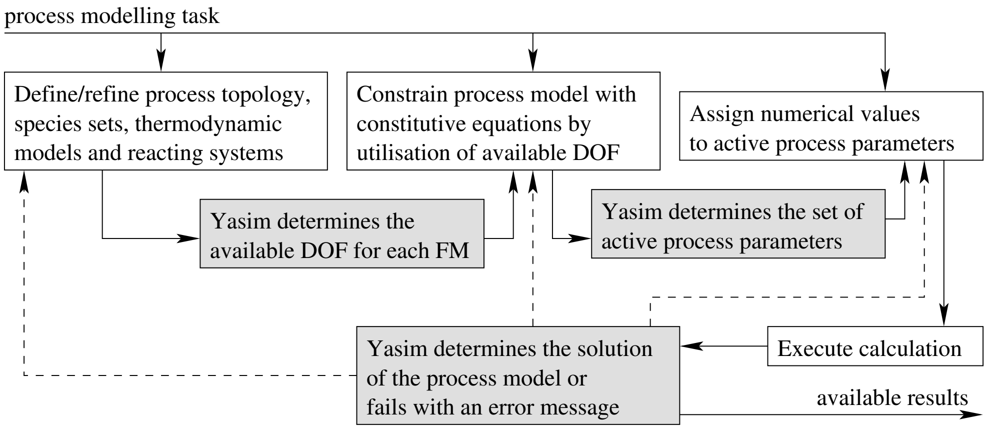

process modelling task
Define/refine process topology, species sets, thermodynamic models and reacting systems
Constrain process model with constitutive equations by utilisation of available DOF
Assign numerical values to active process parameters
Yasim determines the available DOF for each FM
Yasim determines the set of active process parameters
Yasim determines the solution of the process model or fails with an error message
Execute calculation
available results

Figure 5.2: *Interaction between user and Yasim to solve a process modelling task.*

the user is to define the process topology, instantiating flowsheet modules (FM), establishing couplings, defining sets of chemical species to be considered and reacting systems. This step determines the canonical equation system completely, while no information is yet provided to start defining the constitutive equation system. However, *Yasim* identifies the available DOFs for each FM as described in Section 3.3.2. The next sub-task to establish the process model is to define process parameters, properties and constitutive equations where necessary, and constrain the process model by one constitutive equation for each DOF. This step defines the constitutive equation system and *Yasim* determines the set of active process parameters. Finally, these process parameters are given numerical values and the calculation is executed. Each of the described three modelling steps can be refined, if *Yasim* does not find a solution or the model should be further modified or extended.

5.3 Software design

Figure 5.3 shows a typical representation of a process model in *Yasim*. The concept reflects the structure shown in Figure 2.4, enabling a hierarchical module structure.

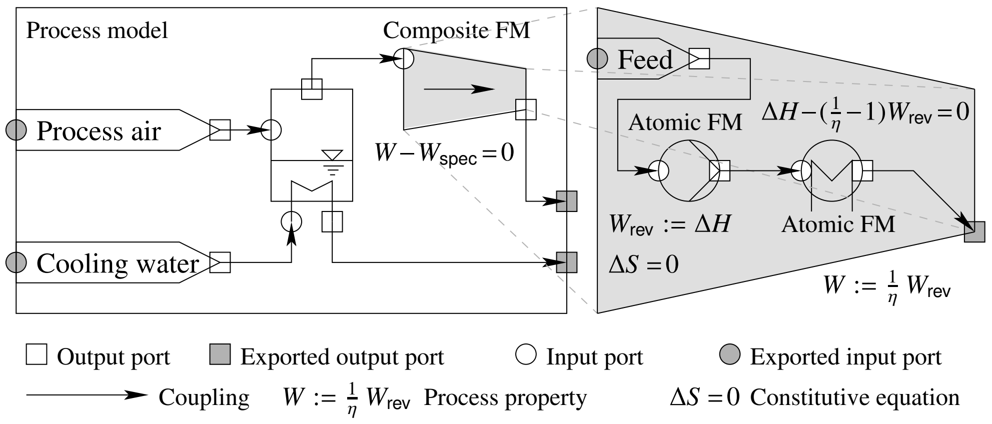

Process model
Composite FM
Process air
Cooling water
$W - W_{\text{spec}} = 0$
Feed
Atomic FM
$\Delta H - (\frac{1}{\eta} - 1) W_{\text{rev}} = 0$
$W_{\text{rev}} := \Delta H$
$\Delta S = 0$
Atomic FM
$W := \frac{1}{\eta} W_{\text{rev}}$
Output port
Exported output port
Input port
Exported input port
Coupling
$W := \frac{1}{\eta} W_{\text{rev}}$ Process property
$\Delta S = 0$ Constitutive equation

Figure 5.3: *Example of a typical hierarchical process model structure.*

Within the process model, the compressor appears as an ordinary FM with one input port and one output port. $W - W_{\text{spec}} = 0$ is a constitutive equation attached to this FM, and the user can adjust the value for the process parameter W_{spec}. As described in Section 3.6.2, the compressor is a composite FM. The right side of Figure 5.3 shows the inner topology with internal couplings, constitutive equations, and process parameters.

A key design requirement is to keep FMs maximally independent of their parent FM. The following sections describe the software design of various groups of functionality to a FM, which are designed to smoothly fit into this concept. The functionality is grouped into equations and DOFs, continuous and discrete process parameters and properties, thermodynamic models, chemical reactions, input and output ports and executive functionality, such as simulation and optimisation. A FM provides these groups of functionality through various handlers as shown in Figure 5.4. Basic design ideas are inspired by the European CAPE-OPEN (computer aided pro-

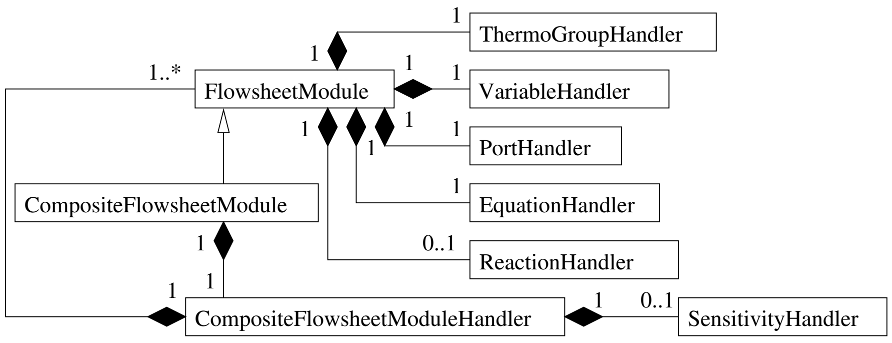


Figure 5.4: *Functionality of flowsheet modules with its interfaces partitioned into handlers.*

cess engineering – open process environment) project (Braunschweig *et al.*, 1999, 2000).

5.3.1 Handler for thermodynamic groups

As shown in Figure 5.5, a thermodynamic group in *Yasim* is defined as a tuple consisting of the following attributes:

Identifier: A textual name of the thermodynamic group. The identifier is unique within its scope, which it is defined for, i.e. the FM it is contained in.

Thermodynamic model: The implementation of a thermodynamic model, capable of performing point calculations on the given set of chemical species.

Set of chemical species: A set of identifiers of chemical species, which is used to gather relevant thermodynamic properties from the database, to test the validity of couplings between two material ports, to collect stoichiometric data for establishing element balance equations within a reactor FM, and as secondary information (cf. Figure 5.1) for the engineer to be able to interpret species-specific data.

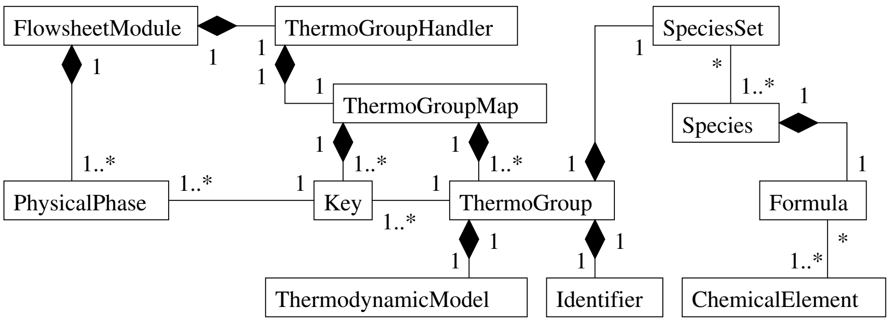


Figure 5.5: *Handling of thermodynamic models in Yasim.*

The thermodynamic model itself provides a state function $P(\underset{\sim}{x})$ with first and second-order derivatives $\underset{\sim}{\nabla}$ and $\underset{\approx}{H}$, furthermore an interface to access the thermodynamic parameters. Symbolic derivatives of P, $\underset{\sim}{\nabla}$ and $\underset{\approx}{H}$ with respect to parameters can be obtained as well. In order to utilise a thermodynamic model in *Yasim*, a series of state transformations is applied as described in Section 2.5.2.

Within the handler, thermodynamic groups are hosted in a map, of which the keys are used by actual physical phase objects to obtain the correct thermodynamic group. Consider a vapour liquid equilibrium of moist air over NaCl-solution. The liquid phase will seek for a key '*liquid*', for which the handler will probably host an NRTL-model (Non Random Two Liquid) considering the chemical species H_2O and NaCl. The vapour phase will find an SRK-model hosted under the key '*vapour*'.

A more detailed example, demonstrating the application in a hierarchical context, is given at the end of this chapter on page 100.

5.3.2 Handler for process properties and parameters

The main focus of this handler is the definition of process properties $\underset{\sim}{y}$ and process parameters $\underset{\sim}{u}$ within the scope of a FM. All process variables consist of an identifier, which is unique in its scope, a numerical value and a physical dimension. The latter one is identified by a set of basic dimensions (currently length, time, mass, temperature and quantity) associated with an exponent. The physical dimension of a heat duty Q is therefore represented by $[Q] = \text{mass}^1 \cdot \text{length}^2 \cdot \text{time}^{-3}$. This approach allows for consistency checks and to obtain physical dimensions of successive expressions, but does not specify the actual unit of measurement, namely MW or kWh/s. The physical dimension solely defines the set of valid units of measurement for a given process variable.

Basic process properties (e.g. $\underset{\sim}{n}$, $\underset{\sim}{\mu}$), which are available through the solutions $\underset{\sim}{x}$ and $\underset{\sim}{\lambda}$ of the canonical equation system, process parameters (e.g. p_{spec}), and derived process properties (e.g. $\underset{\sim}{\varrho}$) are specialisations of process variables. As shown in Figure 5.6, each process variable can be used within the definition of new process properties and constitutive equations. These definitions are based on variable collectors,

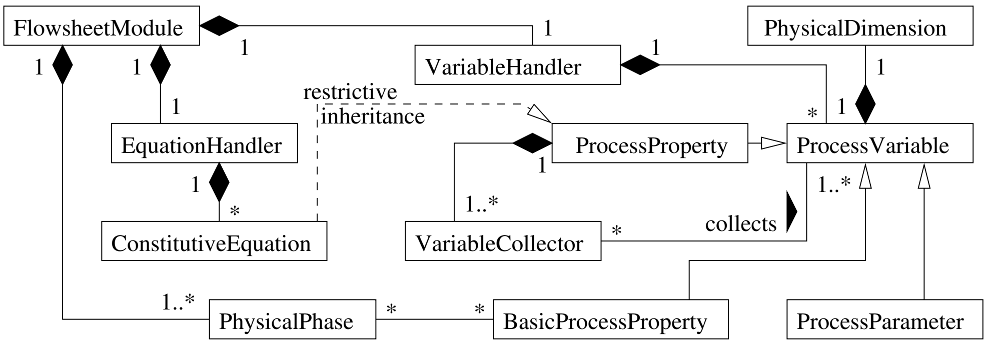


Figure 5.6: *Design of process parameters and properties.*

which link a symbol or a set of symbols within an algebraic expression to process variables within the scope of the defined object. Variable collectors represent an important layer to separate definition and instantiation. Abstract expressions, e.g. for pressure drop from an upstream module, can be defined, before the upstream module is connected or even instantiated. Just before actual calculations are conducted, all expressions link to their symbols and generate a symbolic algebra graph. Symbols can point to process variables that are defined in the same FM, a direct child FM or based on a material flow between two child FMs.

Constitutive equations are a restricted specialisation of process properties. Like for other process variables, variable collectors are used to associate symbols within its definition to other process variables. A constitutive equation also needs to be consistent regarding physical dimensions. The value of a constitutive equation is actually the current residual during the solution process. However, in the context of process model parameterisation, it is not of particular interest. The restriction is therefore that a constitutive equation can not be included as a symbol inside the definition of another process property.

The approach to define properties as explicit expressions of already defined variables avoids additional load to the solver. The canonical approach only allows for one implicit equation for each natural DOF. As proven so far in many applications of *Yasim*, process models do not require additional independent variables as a supplement to the canonical basis. *However, Yasim is not a general equation solver, but clearly limited to physical systems, of which the state is completely described by the thermodynamic state vector $\underset{\sim}{x}$ of physical phases.*

5.3.3 Handler for equations and degrees of freedom

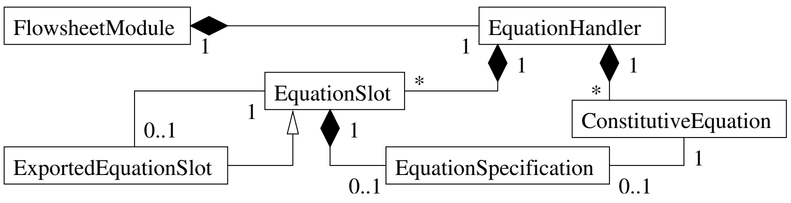


Figure 5.7: *Design of constitutive equations and DOFs.*

The equation handler hosts objects to represent DOFs and constitutive equations. An equation slot is a released balance equation according to the concept described in Section 3.3.2, and represents a DOF. An equation is represented by an expression, which calculates the residual of the equation dependent on imported process variables as described in the previous section. The equation specification object works in the same way as the variable collector, as it represents a link to an equation, which is resolved just before actual calculations. As an important restriction, each defined equation can be used maximally once. An equation slot can be unused, so that the underlying canonical balance equation (e.g. conserving enthalpy) is used. Even if the equation specification links to an equation, the equation slot can still be exported. A constitutive equation defined in the parent FM can then be associated with this DOF. For exported equation slots, the locally linked equation will only be used, if the FM itself is the process model. Otherwise, the exported slot determines the actual equation used. Within nested FMs, equation slots can always be exported up to the global process model level.

However, like process variables, constitutive equations cannot be exported. If export of constitutive equations was enabled, the contained variable collectors would not necessarily have access to their target process variables within the parent FM. The data encapsulation, which prohibits this access, is an important paradigm to preserve maintainability of process models.

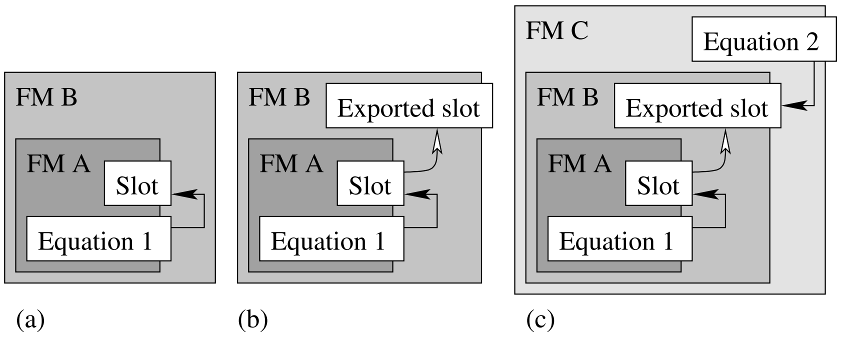


Figure 5.8: *The principle of equation slots and constitutive equations.*

Figure 5.8 shows a typical configuration example in the context of a composite flowsheet module:

(a) FM *A* provides an equation slot and a constitutive equation (*Equation 1*) assigned to the equation slot. Instantiated into FM *B*, *Equation 1* is therefore an active constraint to the process model. In parallel to *Equation 1*, there might be other constitutive equations defined, which however, if not associated to other equation slots, are inactive.

(b) Composite FM *B* is configured to be instantiated as a child FM. For this purpose, the equation slot is exported in order to be visible in the parent FM context. The represented DOF can subsequently be utilised from there. FM *B* still can be executed as a process model. In this case, *Equation 1* is still active.

(c) In the context of FM *C*, *Equation 1* is no longer active. A new equation (*Equation 2*) is defined, and contributions of process variables from various FMs next to FM *B* might be the motivation to define this equation at the outer level. If *Equation 2* would only be contributed by process variables of FM *B*, the process model would be most maintainable with this equation being defined in FM *B*. Finally, *Equation 2* is activated by assigning it to the exported slot.

5.3.4 Handler for input and output ports

Figure 5.7 shows the handling of objects related to material flow. Every FM hosts a port handler that defines input and output ports. As a composite FM contains child FMs, the composite FM handler holds coupling objects that represent a material stream from exactly one output port to one input port of another FM. An output port can only be linked to one coupling, but might as well remain unconnected if the material stream leaves the system boundaries. An input port must be connected at

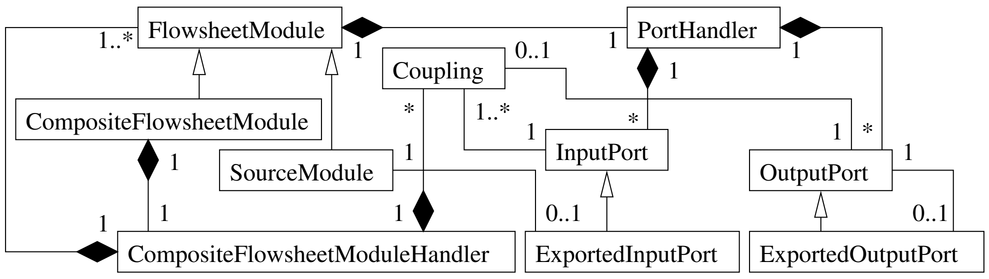


Figure 5.9: *Design of material ports and couplings.*

least once, but might retrieve many couplings, in which case all incoming material flows are considered. The FM implementation determines the boundary conditions of mixing, most commonly $\dot{\underset{\sim}{n}}_{\text{total}} = \sum_i \dot{\underset{\sim}{n}}_i$, $H_{\text{total}} = \sum_i H_i$ and $p_{\text{total}} = \min_i p_i$.

A coupling between an output port and an input port is valid, if the chemical species provided by the output port are accepted by the input port. In particular, not all species accepted by the input port have to be provided through a single coupling.

As shown in Figure 5.3, an output port can either be exported or coupled to the input port of another FM. An exported output port is hosted by a composite FM and represents an output port of a child FM. An exported input port however does not represent the input port of a child FM, as this would make it impossible to calculate a child FM as a stand-alone module. As it can be seen for the compressor in Figure 5.3, a source module, which in local context represents a material reservoir, can be exported as an input port in a global context. An outer process model will then disregard the local source module, and link the material flow directed to the exported input port to the input port downstream of the source module in the local context. This mechanism is further clarified by an example at the end of this chapter on page 100.

5.3.5 Handler for reactions

Reactions are only supported by a subset of FMs, therefore not all FMs host a reaction handler. The current implementation only allows for at most one reaction handler per FM, but composite FMs could host many, related to different child FMs. As shown in Figure 5.10, the reaction handler maintains a number of different species sets. Initially, the inert and key species set is empty, hence the constraint matrix for each physical phase and input port is generated as element balances respectively based on the species defined in phase and accepted species sets. According to the approach described in Section 3.5.3, additional balance equations are introduced for key and inert species. While the species balance for inert species is meant to be an active constraint, the species balance of a key species serves as a DOF for any kind of constitutive equation. Figure 5.11 shows the reactivity of a system containing N_2, O_2, NO_2, and N_2O_4. Initially, all species are reactive, and two independent reactions

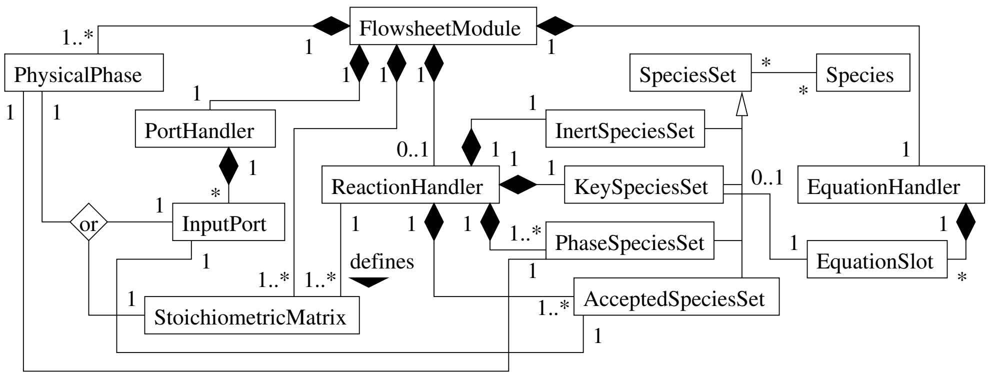


Figure 5.10: *Design of chemical reaction handling.*

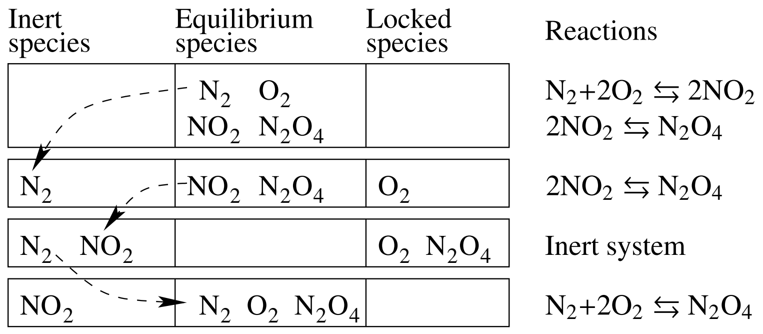

Inert species
Equilibrium species
Locked species
Reactions
N_2 O_2 NO_2 N_2O_4
$N_2+2O_2 \leftrightarrows 2NO_2$
$2NO_2 \leftrightarrows N_2O_4$
N_2
NO_2 N_2O_4
O_2
$2NO_2 \leftrightarrows N_2O_4$
N_2 NO_2
O_2 N_2O_4
Inert system
NO_2
N_2 O_2 N_2O_4
$N_2+2O_2 \leftrightarrows N_2O_4$

Figure 5.11: *Different stoichiometric constraints on the nitrogen – oxygen reactive system.*

are possible. Defining N_2 as an inert species, there is no possible reaction involving O_2. When NO_2 is declared as inert, the entire system is locked and no chemical reaction is enabled. In the last step, N_2 is again permitted to participate in chemical equilibrium. With this, O_2 and N_2O_4 also become reactive again.

An important design limitation is the one-to-many relation between a reaction handler and physical phases. The union of chemical species in all phases determine the constraint matrix. The desirable association between definitions of reactions and thermodynamic groups is therefore not applicable.

5.3.6 Handler for composite flowsheet modules and optimisation

The composite flowsheet module handler is mainly responsible to host the flowsheet topology, i.e. child FMs and couplings. Because every well-defined composite FM is a functional process model in itself, a solver object and a sensitivity handler are created on demand. Figure 5.12 shows the structure diagram of this context. The solver

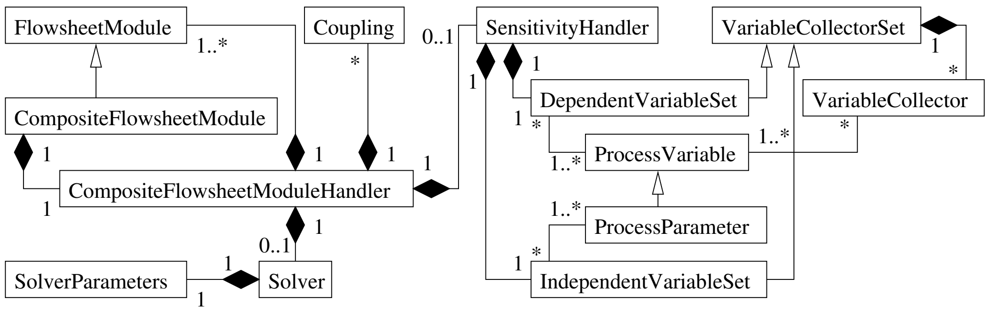


Figure 5.12: *Design of process topology and sensitivity handler.*

object is generated prior to a simulation run. The separated solver parameters contain options to tune convergence criteria for the constitutive and canonical equation system, a tolerance limit for near-zero pivot elements to detect linear dependencies, and the relaxation parameter f_γ according to Section 3.8. If defined, the sensitivity handler hosts a set of independent and dependent variables. Equation (4.20) is then used to obtain the desired J matrix, after the simulation is completed. The variable sets are implemented through variable collectors as introduced in Section 5.3.2. While every process variable can be declared as a dependent variable, only process parameters are candidates for independent variables. The trivial case of defining a process parameter as a dependent variable is not considered. However, the user can force this by defining a process property as $y = u$ if desired.

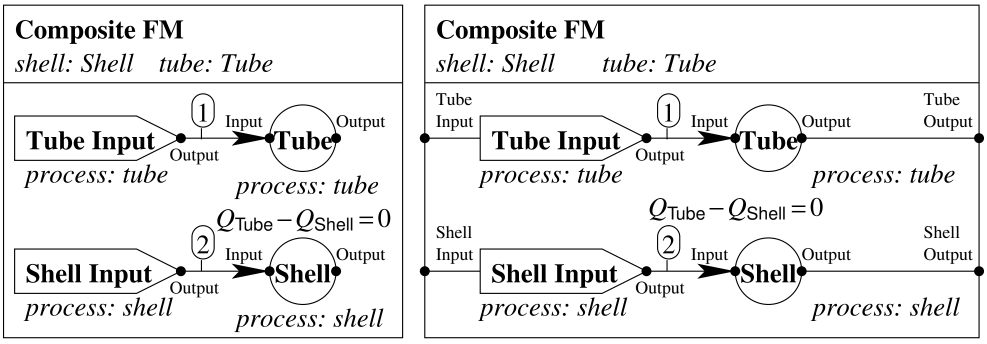
Considering the example shown in Figure 5.3, W_{spec} and η are typical process parameters, therfore candidates for independent variables. The compressor outlet pressure and temperature are examples of process properties, which can represent dependent parameters in this context.

As soon as the optimisation handler is activated, there must be at least one dependent and one independent variable declared in order to obtain a J matrix of non-zero size. The sensitivity handler provides functionality to set, get and update the independent variable vector, get the dependent variable vector and the J matrix.

5.4 A configuration example

The practical example given in this section clarifies the direct application and functionality. As one example can not be exhaustive, the intention is to substantiate the contents of the previous section. The objective of the example is to define a simple process model for a heat exchanger. This composite flowsheet module should be usable from a parent context and provide standard constitutive equations for heat transfer. Naturally, different fluids are considered on the tube and shell side.

The first step to set up this process model is to define thermodynamic groups. Two identical pure water models are sufficient as place-holders for different groups when applied as a composite FM, named *Shell* and *Tube*. The keys within the map are not identical to the identifiers, but denoted by lowercase *shell* and *tube*. The next step



(a) (b)

Figure 5.13: *Definition of process topology for a simple heat exchanger.*

is to define the process topology as shown in Figure 5.13(a). A one-phase module is defined for both, shell and tube side, each fed by a source module. These child modules inherit the thermodynamic groups from their parent context. The mechanism for the FM *Shell Input* to define and maintain its thermodynamic group is as follows: The map of the FM contains a key called *process*. The thermodynamic group addressed is a copy of *Shell* in the parent context, named after the key hosting it.

In Figure 5.13(b), the non-connected output ports are exported to the composite level and named as *Tube Output* and *Shell Output*. The source modules are as well exported and named *Tube Input* and *Shell Input*. Both actions have no impact on the process model as such, but define the interface, when later used as a child FM.

The equation slots provided by the source modules are used to specify the flows for a local test-case. For pure water flows, there are three DOFs each, which might be specified as temperature, pressure and mass flow. The equations associated to these DOFs and the entire source modules will however only be used for local calculations, not if instantiated as a child FM.

The one-phase modules provide two DOFs each. The material balances stay constrained in the canonical system, but the balance equations of the first two state variables, for instance S and V, are released. For a simple model, both pressure drops are defined as zero, leaving two DOFs to define the heat transfer. The representing equation slots are both exported to the composite level. Both modules make their heat duty Q visible on the composite level as a process property. Here, the equation $Q_{\mathsf{Tube}} - Q_{\mathsf{Shell}} = 0$ is defined and plugged into one of the exported equation slots. Various heat transfer equations can now be defined on the composite level, including incoming and outgoing process properties, such as e.g. temperatures and flows. On

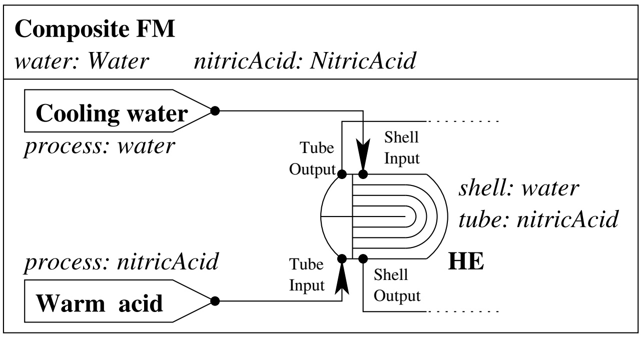


Figure 5.14: *Usage of the heat exchanger as a child FM.*

instantiation as a child FM, these predefined equations can be offered to the user, selecting one of them to utilise the remaining DOF. Figure 5.14 shows the usage of the new heat exchanger model in a parent context. The instantiation is called *HE* and appears basically indistinguishable from atomic FMs. The exported input and output ports are visible, while the inner process topology is hidden. Furthermore, place-holders for two thermodynamic groups are defined, now filled with one group for water (shell) and the other for nitric acid (tube). When instantiated, the thermodynamic group *water* inserted into the *shell* placeholder will trigger the thermodynamic groups called *shell* in each child FM to be replaced by a copy of *water* recursively.

The heat exchanger offers its defined heat transfer equations and one DOF. However, the parent FM might define and utilise additional equations or actually apply two heat transfer equations simultaneously – one physically motivated and one based on first principles – e.g. to determine the required temperature difference. This technique requires the utilisation of DOFs external to the the heat exchanger.

5.5 Software architecture of Yasim

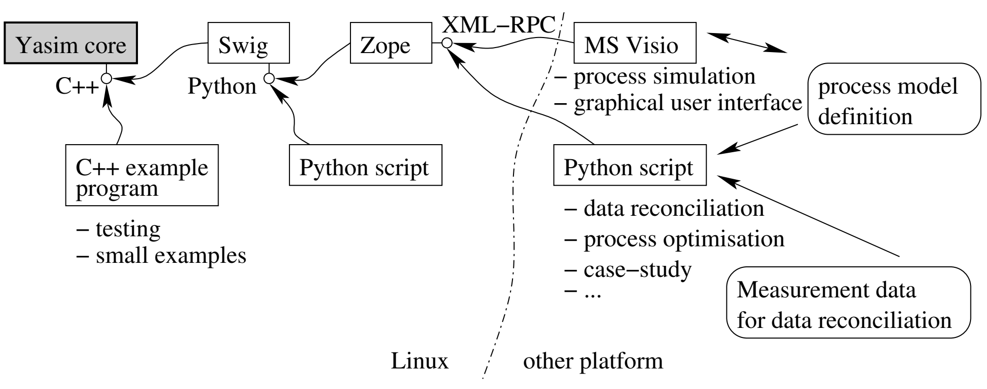


Figure 5.15: *Yasim software architecture and programmers' access points.*

As shown in Figure 5.15, the core of *Yasim* is implemented as a framework in the programming language *C++* on *Linux*-platforms. Most examples in this work have been programmed through direct access by *C++* main-programs. The main functionality is made available through *Swig* (Swig, 2005) as a *python*-interface. The programming language *python* (Python, 2005) is much more suitable to administrate process models than *C++*. Through the web application server *Zope* (Zope, 2005), *Yasim* can be accessed remotely via *XML-RPC* (XML-RPC, 2005) on arbitrary platforms. A graphical user-interface is developed using *Microsoft Visio* (MS Visio, 2003).

With this variety of access-points, *Yasim* can be utilised with high efficiency in industrial relevant projects. Typically, the process model is established using the graphical interface. A tailor-made *python*-script picks up that process model to execute advanced tasks as described in Chapter 4. This approach combines the maintainability of larger process models through the graphical user interface, while the full flexibility of a programming language can be applied on the same process model.

Chapter 6

Performance characteristics

6.1 Introduction

This work does not provide the solution to a specific process modelling problem, but investigates the canonical approach as a formulation of process models in general. The actual implementation of *Yasim*, as described in the previous chapter, clearly is a practical outcome and serves as a basis for future work, both to extend the scope of *Yasim* and in combination to apply the existing functionality in industrial projects, as it is done in several cases already.

This chapter focuses on the performance of the solution methods and other numerical techniques used throughout this work in general. After a discussion of the solver properties, such as convergence rate and region, the numerical effort is investigated for different types of process models. The consistent use of symbolic derivatives poses questions about the quality of differently obtained derivatives, which is the subject of Section 6.4. Scaling problems can occur when using the current implementation of *Yasim* for large process models. This problem is addressed in the last section of this chapter.

6.2 Solver characteristics

Unlike conventional equation solvers, the algorithms described in Section 3.3 solve two equation systems simultaneously. The canonical equation system is well structured and is only dependent on thermodynamic and stoichiometric data, while the constitutive equation system has no defined structure, and contains all geometric information and process parameters.

6.2.1 Convergence rate

The challenge of implementing the solver is to find an iteration scheme that efficiently solves the combination of canonical and constitutive equation system. The availability of second-order derivatives makes it possible to obtain quadratic convergence, if the equation systems are updated correctly. The algorithms in Section 3.3 only interpret the L multipliers of the canonical system as mathematical derivatives of the objective function with respect to the constraints, when the constraints are fulfilled, which is a necessary requirement for quadratic convergence. A typical

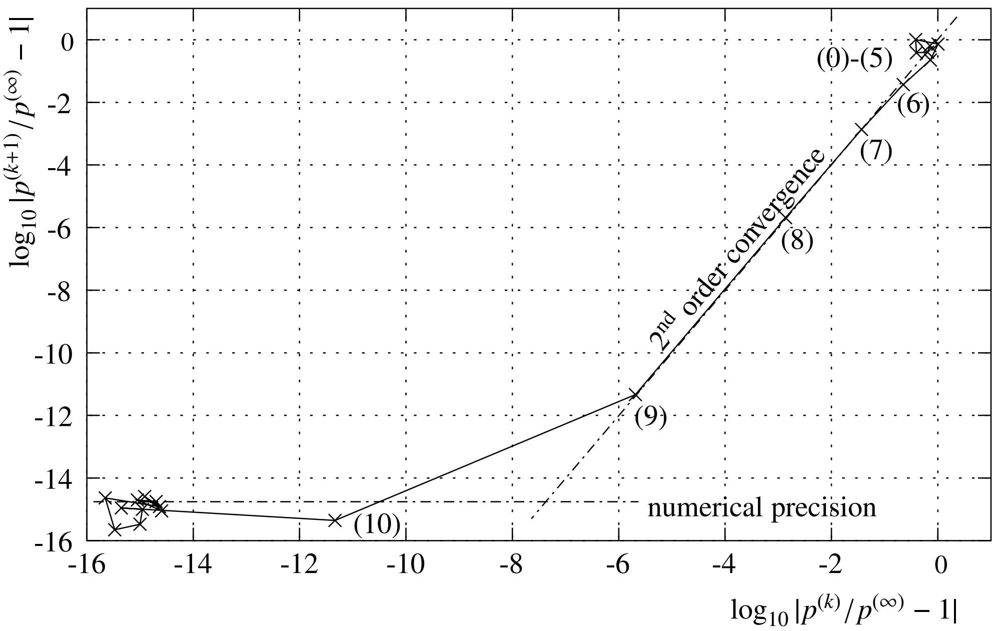


Figure 6.1: *Convergence characteristics for the separator pressure p of the compressor example shown in Figure 4.3.*

progression of a variable involved in non-linear equations is shown in Figure 6.1. Far from the final solution, the first iterations do not reduce the residual of the merit function. The step length is even reduced in order to remain within the domain of thermodynamic models (cf. Section 3.8). Shortly after full steps are taken, convergence is of second-order, such that the residual falls rapidly beneath the limit of numerical precision. The numerical precision depends on the solution method, and the process model, which influences the condition of coefficient matrices. Avoiding the calculation of the inverse canonical coefficient matrix, as described in Section 3.3.3, could further reduce this level, as less critical subtractions of numerical values are performed (Golub and Loan, 1996).

6.2.2 Convergence regions

The current implementation facilitates a trivial generation of starting values, only reading T, p and $\underset{\sim}{n}$ for each thermodynamic phase from an *XML*-file. Because the user will have some ideas about the approximate figures here, practical problems of finding suitable starting values are not significant. The problem of finding more intelligent initialisation routines is therefore not emphasised in this work. However, it is very effective to restart a modified process model using recent results as starting values.

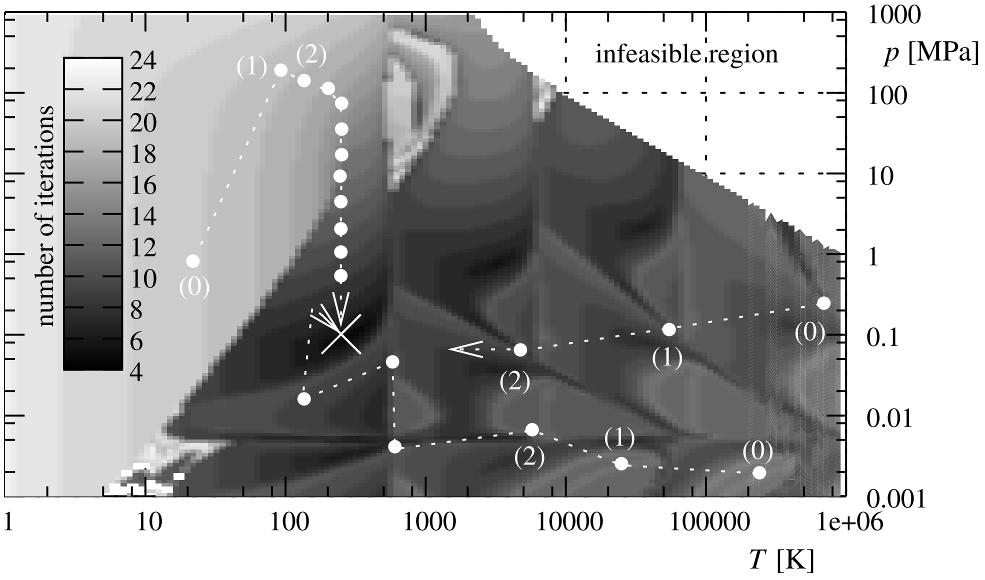


Figure 6.2: *Convergence of a single-phase node with varying starting values.*

Figure 6.2 shows the number of necessary iterations to achieve convergence in a single-phase source module with atmospheric air containing N_2, O_2, Ar, H_2O and CO_2. The starting values are given by equimolar amounts of each species and varying temperature and pressure. The process model converges for a wide range of T and p around the solution marked by a white cross. Far-off starting values do not allow for proper thermodynamic calculations and cause immediate problems. There is a sharp separation line, at the left of which significantly more iterations are required. This is caused by the cubic equation of state model, which predicts only a liquid root at lower temperatures. The enthalpy jumps when iterating towards the desired solution into the vapour region, and this highly non-linear feature causes the effect. Another region of slow convergence occurs at high pressures between 500 K and 1000 K around the critical point of the mixture. The repetitive structures at high temperatures and moderate pressures repeat within approximately one decade and are caused by the relaxation scheme (cf. Section 3.8). With the requirement $T > 0$, each temperature reducing step is restricted to change temperature no more than 90% of its current value. The two dotted polygons starting on the high temperature side show iteration paths constrained by this scheme.

Figure 6.3 shows a similar plot of convergence regions. This time, the starting values for each point calculation are fixed to atmospheric conditions, but the target specifications are altered. Most significant is the extended region of non-successful

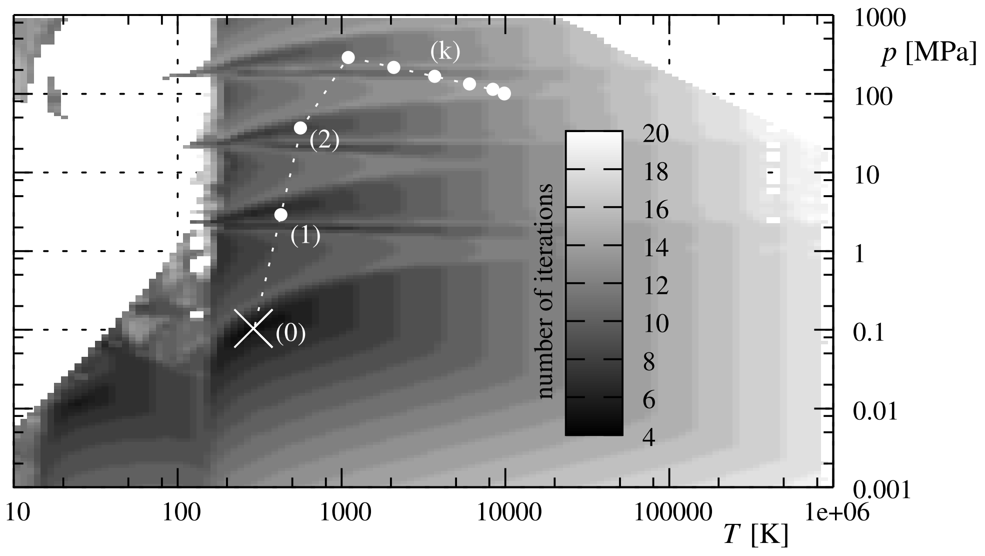


Figure 6.3: *Convergence of a single-phase node with varying specifications.*

calculations nearly covering the complete range, in which the solution is forced within the liquid root of the equation of state. While reducing the volume during iteration, the relaxation method does not prevent the state vector from entering the unphysical domain with $\partial p/\partial V > 0$. Once inside this region, the solver suggests an increase of volume to reach a higher pressure, such that the state oscillates between the unphysical and the vapour region. Only extreme compressed conditions allow the solver to overleap the unphysical region directly into the liquid root and solve the system. The first steps of the indicated example calculation with $T_{\mathsf{spec}} = 10000$ K and $p = 100$ MPa are limited by the relaxation scheme, as the volume can not be reduced by more than 90% of its value in each iteration.

Within regions of ordinary process conditions, it can be concluded that convergence towards a liquid solution can not be obtained if the starting values suggest a vapour phase. But Figure 6.2 indicates no problems to predict vapour phase results with starting values suggesting a liquid phase. The first update then yields a clear vapour state, if the target state is not too close to critical conditions.

As described in Section 3.8, constitutive equations can generate similar effects. In both cases, the current implementation of the solver relies on starting values within the same feasible region.

6.3 Computational effort

The computational effort for one iteration is the sum of different contributions. The current implementation according to Section 3.3 requires the

1. Calculation of state function gradient and H for each thermodynamic

phase within the process model.

2. Calculation of inverse coefficient matrices for each single atomic FM and arranging them in composite FM coefficient matrices.

3. Inversion of the composite FM coefficient matrix and solving the canonical system.

4. Evaluation of the J of constitutive equations and calculating updates of the right hand side.

For process models with some recycle streams, the main bottleneck is identified to be the inversion of the main composite FM coefficient matrix. In this section, three different process model structures are considered: (i) A linear process with no recycles, (ii) a counter-current column, and (iii) a particular strongly recycled structure. The species set chosen for this example is propane, n-butane and n-hexane. Each flash tank is specified to atmospheric pressure and a 50% vapour fraction. Figure 6.4

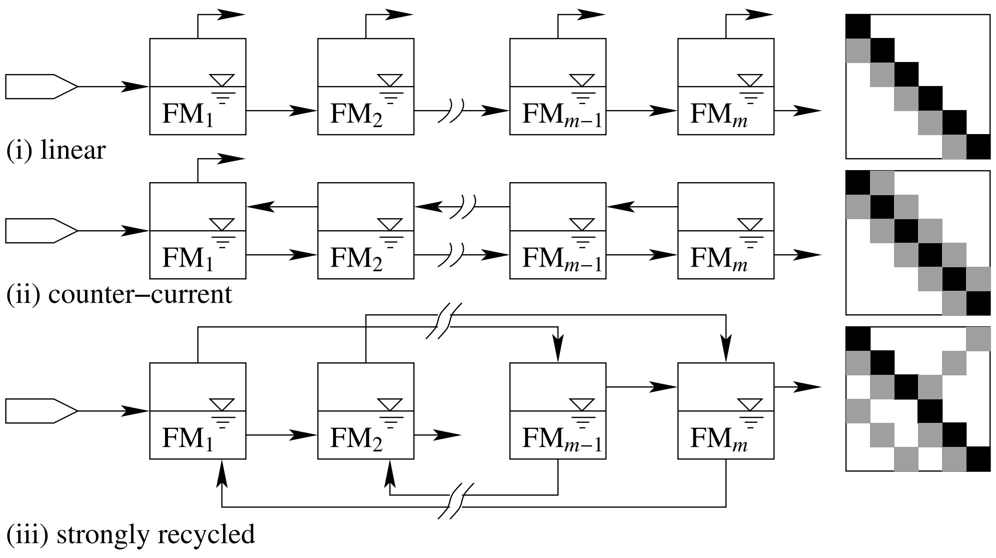


Figure 6.4: *Different topological structures to analyse computational effort.*

shows the process topologies and resulting coefficient matrices of these three structures. Structure (i) contains no recycles. As can be seen in Figure 6.5, the sparse block matrix structure is exploited to reduce the complexity from order 3 for general matrix inversions to 1.85. The constitutive equation system for 100 flash modules is of size 205 and contributes to about 10% of the total calculation time. However, solving the canonical system of size 1500 could be performed in linear time, if the inverse matrix is avoided as in the approach discussed in Section 3.3.3.

Process models (ii) and (iii) generate similar performance characteristics. Most of the sparse matrix block structure is lost during explicit inversion, such that the

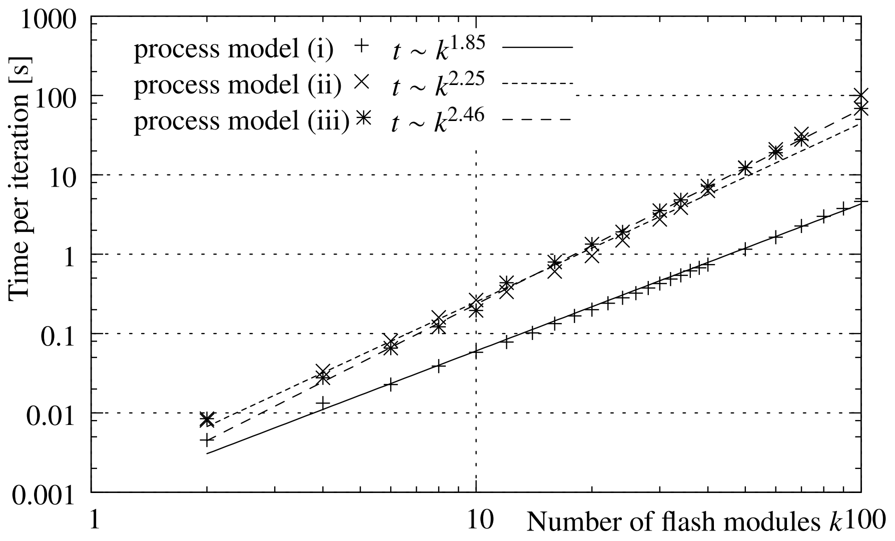


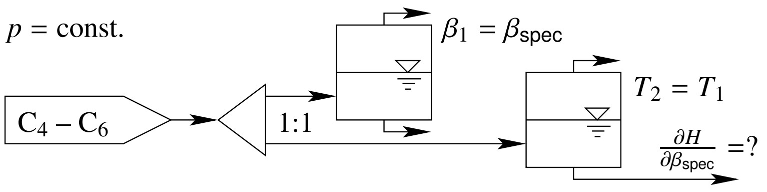
Figure 6.5: *Performance of the test implementation for different process types on a* 2.0 GHz *Intel© XEON™ CPU.*

computational effort is between quadratic and cubic with respect to the number of FMs. Still, process (ii) could be solved in linear time by avoiding explicit matrix inversion, and even though process (iii) yields a rather unstructured coefficient matrix, the sparsity still limits the necessary effort to solve the equation system in quadratic time, if the explicit inversion is avoided.

The current implementation is surely a prototype mainly to show capabilities of canonical modelling, but also to detect potentials such as those to increase efficiency and robustness in subsequent development. At to this phase of development, a comparison of performance with similar process modelling tools is not representative for the potential of this approach.

6.4 Comparison of derivation methods

Within this work, a small symbolic algebra package is implemented as described in Appendix A.1. The two main benefits are: the possibility of runtime parsing of algebraic expressions, but even more important: the availability of derivatives obtained by symbolic computations (as from now called symbolic derivatives). In this section, symbolic derivatives are compared to analytical and numerical derivatives in the context of the canonical flowsheet solver. A comparison of derivatives of analytical functions is trivial, as the symbolic derivatives are identical to the analytical ones. This section therefore concentrates on derivatives necessary to evaluate equations derived in Section 4.2.2, in particular Equation (4.20) used for process optimisation and data reconciliation.



p = const.
$C_4 - C_6$
1:1
$\beta_1 = \beta_{\rm spec}$
$T_2 = T_1$
$\frac{\partial H}{\partial \beta_{\rm spec}} = ?$

Figure 6.6: *Process model to analyse differently obtained derivatives.*

The purpose of the process model shown in Figure 6.6 is to represent a pair of one process parameter and one property with an analytical relationship. The example is chosen such that both constitutive equations and the thermodynamic model contribute to this relationship.

The total effect of the specified vapour fraction $\beta_{\rm spec}$ to the liquid enthalpy from the second flash can be obtained by means of symbolic calculations through Equation (4.20). The same way, the derivatives $\partial T/\partial\beta_{\rm spec}$ and $\partial \underset{\sim}{n}/\partial\beta_{\rm spec}$ can be calculated. The second derivatives of the thermodynamic state function are implemented as explicit analytical expressions. As shown in Appendix C.3, these include heat capacity c_p and partial entropy $\underset{\sim}{\bar{S}}$. Interpreting enthalpy as $H = H(T, p, \underset{\sim}{n}(\beta))$ the total differential at constant pressure is

$$\mathrm{d}H = \left.\frac{\partial H}{\partial T}\right|_{p,\beta} \mathrm{d}T + \left.\frac{\partial H}{\partial \underset{\sim}{n}}\right|_{T,p} \frac{\mathrm{d}\underset{\sim}{n}}{\mathrm{d}\beta}\mathrm{d}\beta \quad\Rightarrow\quad \frac{\mathrm{d}H}{\mathrm{d}\beta} - \left[c_p \frac{\mathrm{d}T}{\mathrm{d}\beta} + (\underset{\sim}{\mu} + T\,\underset{\sim}{\bar{S}})\,\frac{\mathrm{d}\underset{\sim}{n}}{\mathrm{d}\beta}\right] = 0\,. \quad (6.1)$$

This equation must hold, if the symbolic derivatives are correct, i.e. consistent with the analytically available information. Figure 6.7 shows a plot of the residual of Equation (6.1) over the accuracy of a representative process property, in this case the enthalpy of the liquid stream from the second flash. A linear relationship can clearly be identified, and in this case, the derivatives are well over one order of magnitude more precise than the property itself. With high vapour fractions ($\beta_{\rm spec} \to 1$), the condition of the process model deteriorates, such that enthalpy can not be obtained with full precision. Even with a precision limit of 10^{-5}, the observed derivative is still far more accurate, yielding a residual of only 10^{-10}.

The traditional alternative to symbolic derivatives is the numerical approach, typically central differences. $\beta_{\rm spec}$ is perturbed by $\pm\Delta\beta$ to obtain

$$\frac{\mathrm{d}\underset{\sim}{y}}{\mathrm{d}\beta} = \frac{\underset{\sim}{y}(\beta_{\rm spec} + \Delta\beta) - \underset{\sim}{y}(\beta_{\rm spec} - \Delta\beta)}{2\Delta\beta} + O(\Delta\beta^2) \quad \text{with} \quad \underset{\sim}{y} = (H,\, T,\, \underset{\sim}{n})\,. \quad (6.2)$$

Figure 6.8 shows the quality of numerical and symbolic derivatives for various precisions obtained in the process simulation step. Obviously, the symbolic derivative is independent of $\Delta\beta$, but at high precision calculations, fluctuations around a constant value due to the numerical precision limit of that process model become visible.

The precision of numerically obtained derivatives is limited by the precision of symbolical derivatives at low precision of the simulation, but otherwise not correlated. The deviation increases quadratically for large $\Delta\beta$, as is expected according

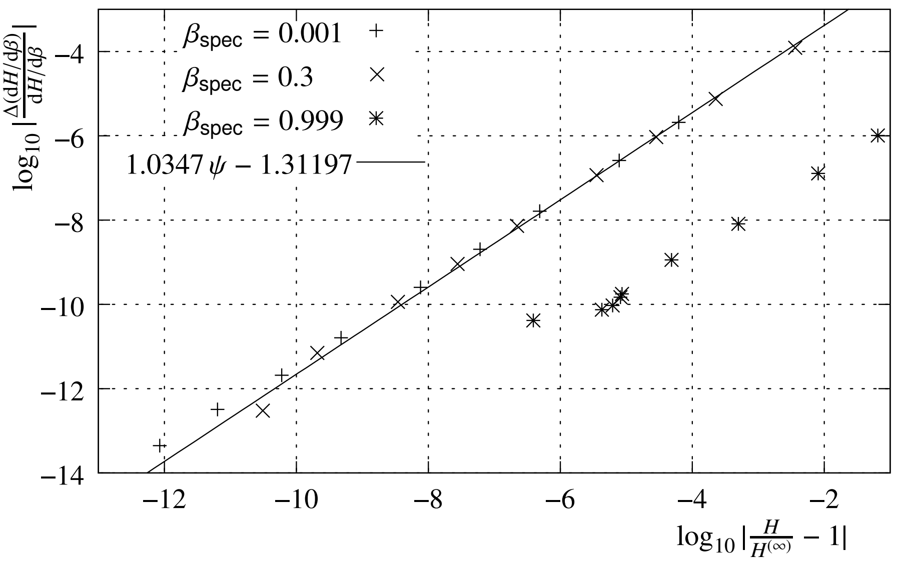


Figure 6.7: *Precision of symbolic derivatives compared to process properties.*

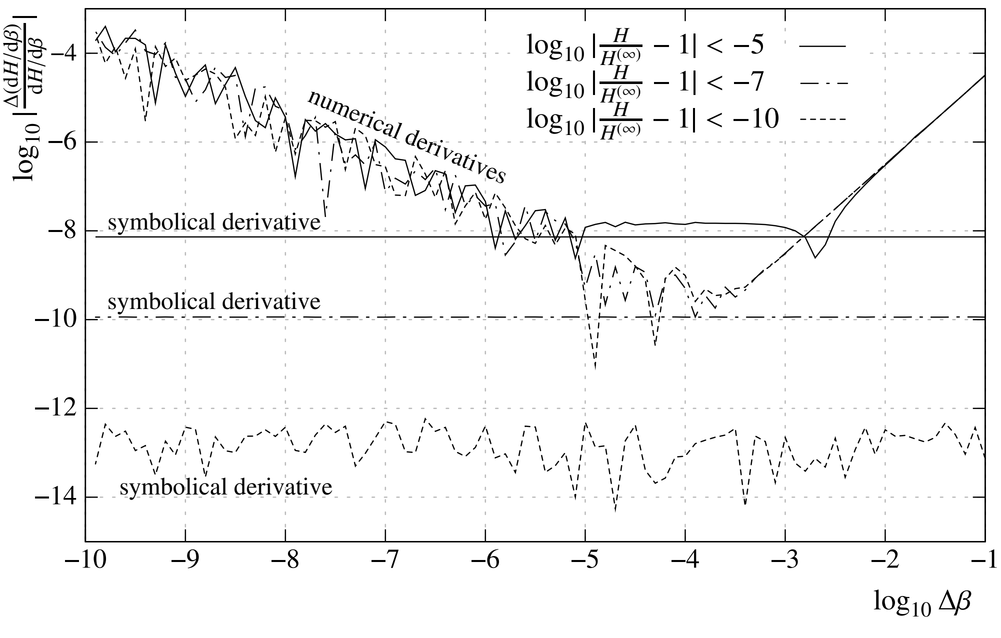


Figure 6.8: *Comparison between symbolical and numerical derivatives.*

to Equation (6.2). For small $\Delta\beta$, the finite difference $y(\beta_{\mathsf{spec}} + \Delta\beta) - y(\beta_{\mathsf{spec}} - \Delta\beta)$ develops a constant non-zero contribution, such that $\lim_{\Delta\beta\to 0} \mathrm{d}y/\mathrm{d}\beta = O(\Delta\beta^{-1})$. This limitation is equally visible in Figure 6.8.

6.5 Properties of the coefficient matrix

The main limitations to the obtainable numerical precision are the conditions of the coefficient matrices both of the canonical and the constitutive equation systems. With measures of various physical quantities such as pressure, energy, volume, etc. forced into one matrix, conditioning problems can become a limitation to the obtainable numerical precision. A gas separation process model of three pressure stages, contain-

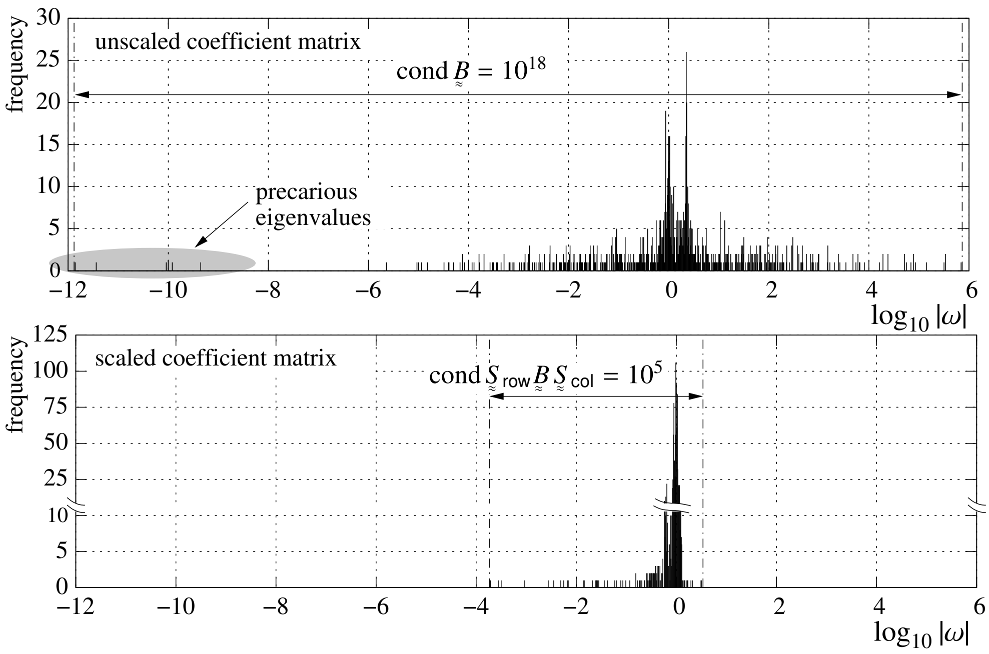


Figure 6.9: *Eigenvalue distribution of scaled and original coefficient matrices.*

ing 22 FMs, three recycles and vapour-liquid-liquid equilibria is established. Considering 11 chemical species (C_{1-6}, i-$C_{4,5}$, H_2O, N_2, CO_2), the canonical system is of size 997×997. The top diagram in Figure 6.9 shows the distribution of eigenvalues of the original matrix, identifying in particular 5 precarious eigenvalues of a magnitude below 10^{-8}. All corresponding eigenvectors are linear combinations of state variables representing enthalpy, substantiating the hypothesis of this being caused by bad scaling. As enthalpy is represented in measures of Joule (J), rather big numbers ($\approx 10^6$) are produced compared to the conjugated measure in K^{-1} ($\approx 10^{-3}$). Thus, an eigenvalue of 10^{-9} is natural in this context. Extreme phase size ratios in separation modules enhance this situation. To generate the lower diagram in Figure 6.9, two scaling matrices $\underset{\approx}{S}_{\mathsf{col}}$ and $\underset{\approx}{S}_{\mathsf{row}}$ are obtained by repeated normalisation of columns and rows. This way, the condition[1] can be reduced significantly down to 10^5, no longer being a serious limit of numerical precision. This result is even more significant for the constitutive equation system, which in this case is of size 210×210. The condition number is here reduced from 10^8 down to $10^{1.6}$.

However, the current implementation of *Yasim* as described in Chapter 5 does not use active scaling. The internal units of measurement are adapted in order to obtain a similar order of magnitude for the conjugated variables pressure and volume. With this, the numerical precision rarely becomes a problem in practical applications. As an example, Figure 6.7 indicates the precision limit for the specified split factor $\beta_{\mathsf{spec}} = 1 - 10^{-3}$ to be in the order of 10^{-5}. This represents a typical limitation, as values of β_{spec} closer to one yield increasingly unstable iteration paths and inaccurate results.

[1]For simplicity reasons, the norm of a matrix is defined here as the maximum ratio of the eigenvalues' absolute values

Chapter 7

Discussion and Conclusions

A new process modelling tool emerged as the practical result of this work. This tool called *Yasim* has already been used in several projects within the Corporate Research Centre of Norsk Hydro ASA in Porsgrunn. Simulations and case studies are performed on models of different plant sections related to urea production. Within a data reconciliation project, *Yasim* generates linearised representations of the gas separation process on one of Norsk Hydro's offshore platforms.

The equation oriented approach made it a valuable tool for medium sized process models, strongly coupled by several material streams and constitutive equations.

Built on the fundament of thermodynamic state functions in transformed coordinates, the highly non-linear equations of thermodynamic models are encapsulated from the main (canonical) equation system. The complexity of these models therefore hardly effects the performance in terms of calculation time and robustness. The H matrices of the thermodynamic models are utilised in multiple ways:

- As a basis to evaluate thermodynamic properties, such as heat capacity, compressibility, thermal expansion coefficient, J -T coefficient, and speed of sound (cf. Table C.2 and Table C.3).
- In order to transform canonical state vectors between different state functions, as described in Appendix C.2.2.
- As H matrices in local optimisation nodes, thus in the actual flowsheet solver (cf. Section 3.2).

The structure of the canonical equation system directly represents the process topology. Flowsheet modules (FMs) represent diagonal blocks, while couplings relate to the off-diagonal block, which shares row and column with down- and upstream FM. This transparent structure supports the implementation of hierarchical FMs and error diagnosis. Not only is the sparsity in general known a priori, but also entire blocks of zeros and identity matrices are recognised, so that the solver can take advantage of it. In a recursive manner, FMs contribute actively to solve the system, currently by inverting, but as suggested in Section 3.3.3, potentially by decomposing their own subset of equations.

Constitutive equations form a separated equation system of much smaller size. The use of symbolic algebra is essential to allow run-time defined user-equations and to obtain symbolic derivatives, reliable sparsity information of the resulting J matrix, and validation of physical dimensions to avoid consistency errors.

Clearly, as a result of continuous research, the current implementation does not represent the current state of knowledge presented in this work. A pure symbolic representation of thermodynamic models would have a significantly positive impact on the maintainability of these models without noticeable loss of performance. Furthermore, the use of the $W(H, V/T, \underset{\sim}{n})$ state function involves many more state transformations than $U(S, V, \underset{\sim}{n})$. Thanks to the constitutive equation system, individual coordinate systems for each FM turned out to be unnecessary and hard to maintain. Restricting all FMs to the system $U(S, V, \underset{\sim}{n})$ contributes only marginal to the size of the constitutive equation system, but avoids M transformations.

Since M transformations are currently used in *Yasim*, further reference state information contributes to the H matrix through the chemical potential $\underset{\sim}{\mu}$ (cf. Table C.1) and therefore may amplify conditioning problems.

A general limitation is given by state functions, which assume volume as a canonical variable, conjugated to pressure in the gradient (e.g. the H function). For incompressible fluids, the pressure is not defined though the state function, as a volume change at constant temperature and pressure is not possible. Consequently, the H matrix contains infinite values, and no calculations can be performed. As a consequence, all thermodynamic models must describe a nonzero compressibility in order to be used for canonical modelling. Appendix B.3 describes a suitable model contribution to ensure a nonzero compressibility.

The implementation of the augmented solver version that avoids explicit inverse matrices will improve performance significantly with respect to both calculation time and numerical stability in ill-conditioned systems. This applies especially to larger process models, exceeding 50 FMs, in particular those containing separation columns.

Automatic initialisation of process models is not yet implemented, but starting values are provided in XML-Files. As cumbersome as manual tweaking of these files sometimes can be, the direct access for the user to fill in the starting values is essential and can prevent from high work effort in terms of trial and error tweaking of the process model towards a converging equation system.

The same applies to the phase stability analysis. The current version of *Yasim* works on a constant topology with each material stream representing one physical phase. Implementation of phase stability tests and consequently considering multiple phase streams is desirable, especially for heat exchangers with phase transitions. The possibility to restrict possible phase sets must be preserved in order to utilise first principle FMs.

This work and the resulting tool *Yasim* is committed to steady-state process models. Still today, a wide range of relevant problems in process design, data reconciliation, optimisation and plant performance projects are performed using steady-state models. However, Appendix D demonstrates the general feasibility of dynamic process modelling on a canonical basis. The topological structure and the constitutive equations can be handled without any changes in the methodology.

Appendix A

Software utilities

A.1 Lazy evaluation datatype

An object oriented symbolic algebra datatype is developed in C++. An extensible set of algorithm classes provides functionality for a large variety of applications. Applied in particular to thermodynamic functions, the framework can be used for complex modelling tasks. E.g. a derivative algorithm is applied for tasks like automatic model implementation, parameter optimisation, data reconciliation, and phase boundary tracking.

A.1.1 General Software Design

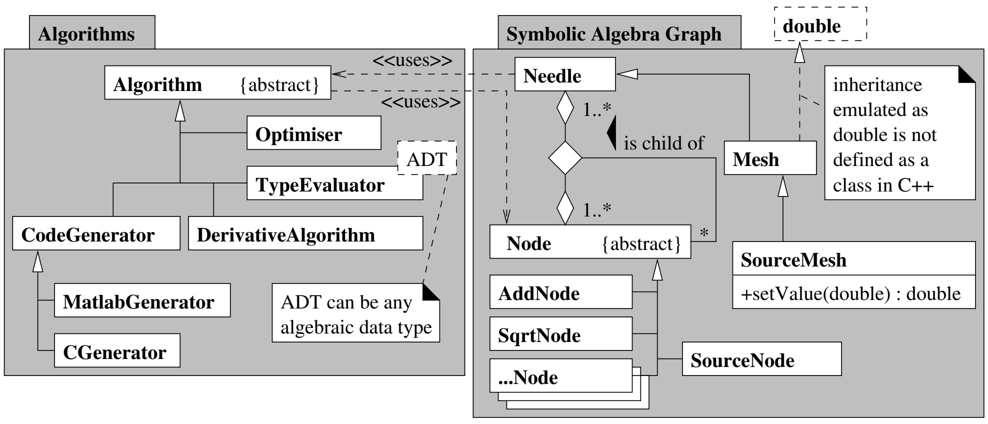


Figure A.1: *UML class diagram of algorithm objects and symbolic algebra graph design.*

The UML class diagram (OMG, 2001) of algorithms and representation of the symbolic algebra graph is shown in Figure A.1. The vertices of the symbolic algebra graph are objects of class *Node*. All operators and standard functions are represented by such a vertex. Via reference counting (Stroustrup, 1997), *Node* instances can be shared among other vertices or user objects called *Needle*. A *Needle* can contain any subset of existing *Node*s, and provides a method to apply algorithm objects. A *Mesh* is a specialisation that carries exactly one *Node*. On *Mesh*, all common arithmetic operators and functions are defined, so it can be treated like the builtin C++ *double* datatype. One further specialisation is a holder of a single source node *SourceMesh*, to which a new value can be assigned without altering the graph.

The following code of simple assignment expressions generates the graph shown in Figure A.2:

```
SourceMesh a = 1.0, b = 3.0;
Mesh c = a + b;
Mesh d = sqrt(c) - (c + a * b);
```

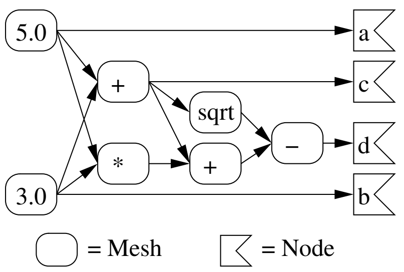


Figure A.2: *Graph representation of a simple expression.*

Though graph optimising algorithms can be implemented, assignments are represented by shared nodes to preserve the benefit of manual coding, namely to avoid redundant evaluations. *a* and *b* can also be declared of type *Mesh*, if the values are not to be changed later in the program. In this case however, printing *d* will display −5 initially only. The subsequent line of code

```
a.setValue(6.0); // call setValue() on SourceMesh instance a
```

yields the output of −24. It becomes clear, how little effort is required to translate existing function implementations into functions generating a symbolic algebra graph. Template-based numerical packages can often be utilised directly.

The operands of a symbolic node, represented by child nodes, are connected at construction for the whole lifespan of the node. Thus, the symbolic algebra graph is assured to be non-cyclic, which is a necessary requirement for algorithms to terminate in a finite number of steps.

A.1.2 Algorithms

The main concept of algorithms on symbolic algebra graphs is based on the idea to separate functionality from the actual function implementation. The UML sequence diagram (OMG, 2001) in Figure A.3 explains the application of an algorithm object. It's illustrated, how the function object only creates the symbolic algebra graph and is subsequently not involved, when algorithm objects are executed.

It's shown in Figure A.1 that all algorithm classes implement an interface called *Algorithm*, which defines methods for the following stages of application:

Initialisation: The algorithm object is created and given the necessary information to function. E.g. a derivative algorithm requires a set of independent variables.

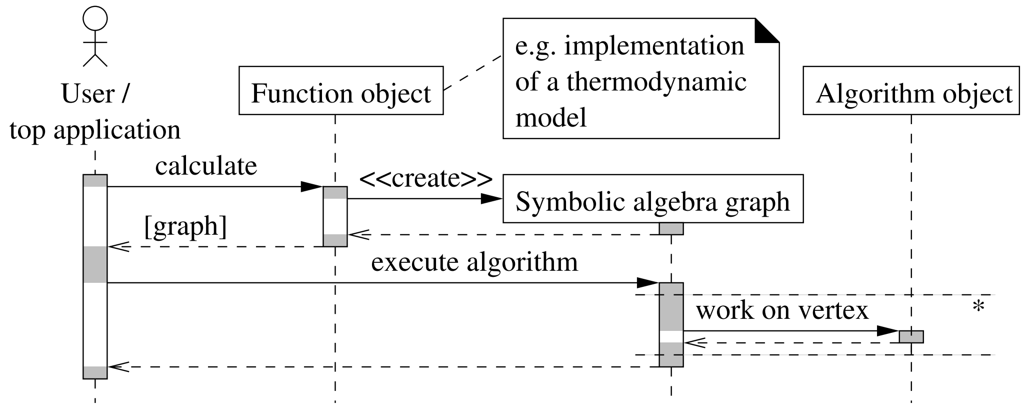


Figure A.3: *The role of a function object in the context of lazy evaluation.*

Execution: The needle object executes the algorithm object. Each node in the needle is called recursively by the algorithm object. Intermediate data is created and temporarily saved in each node. The data belonging to the direct child nodes of the needle represents the result of the algorithm. This can for instance be other nodes in case of a derivative algorithm, or a string in case of an equation-filter. It is saved inside the algorithm object.

Deallocation: The intermediate data of each traversed node is released.

Result query: The specific result is requested from the algorithm object.

The node objects offer functionality for algorithms to traverse to child nodes and determine the type and value of each addressed node. Algorithms – as top layer code – can supplement the graph, but not modify or delete existing nodes.

Generation of Simplified Derivatives

Figure A.4 shows a simple example code and a belonging sequence diagram, whereas the belonging symbolic algebra graphs are visualised in Figure A.5. Line 1–3 define the variables a, b, and $c = a \cdot b$. A derivative algorithm object is declared and a is given as the independent variable. The boxed line in the code executes the algorithm itself. As shown in the sequence diagram, node N_3 is first processed. The algorithm then descends recursively to N_1 and N_2, generating their derivatives, before constructing its own, represented by N_8.

The result of the algorithm is of type *Needle*, which in general can hold many nodes. The *Needle* class supports an STL[1]-style interface (Austern, 1998; Schildt, 1999), such that the first and in this case only element is obtained by the *front()* method.

[1]Standard Template Library

```
 1 Mesh a = 3.0;
 2 Mesh b = 5.0;
 3 Mesh c = a * b;

 4 DerivativeAlgorithm D;
 5 D.setIndepNeedle(a);
 6 c.execute(D);
 7 Mesh d =
     D.getResult().front();

 8 Optimiser O;
 9 d.execute(O);
10 Mesh e =
     O.getResult().front();
```

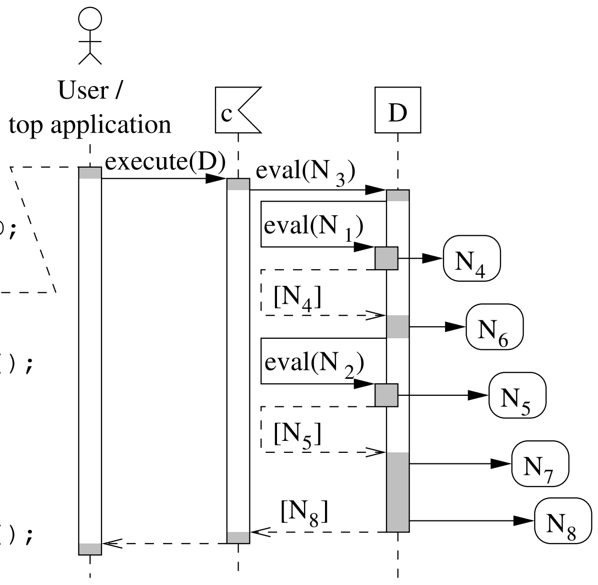


Figure A.4: *Application of a derivative algorithm to a simple example.*

The use of distinct nodes for exact 0 and 1 instead of using an ordinary source node of regarding value makes it possible to efficiently simplify the result by another algorithm. The algorithm class *Optimiser* is declared and used much like the previous one. By exploiting $1 \cdot x = x$, $0 \cdot x = 0$, and $0 + x = x$, it is found that N_2 itself represents $\partial c/\partial a$. Considering this piece of code as a sub-function, which returns e, the user object d will run out of scope and release $N_4 - N_8$.

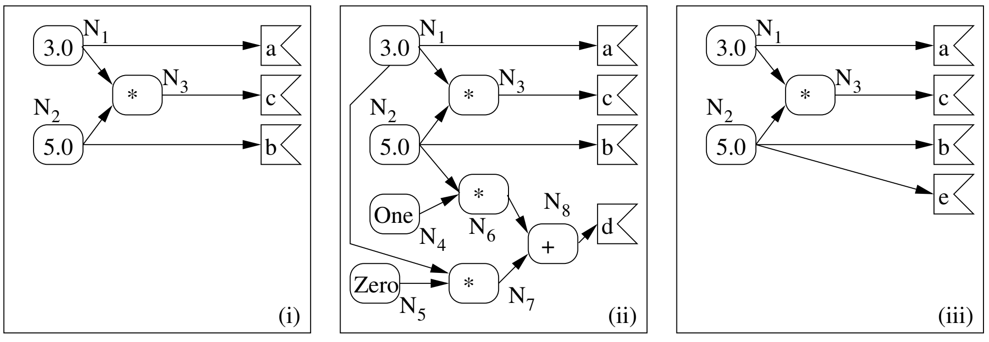


Figure A.5: *Symbolic algebra graphs for the example in Figure A.4: (i) line 3 completed; (ii) line 7 completed; (iii) The optimiser algorithm has generated e in line 10, and with d running out of scope,* $N_4 - N_8$ *are released.*

A.2 Linear algebra package

A.2.1 Requirements

The canonical equation system contains a complete state vector for each physical phase and L multipliers for each distinct intensive state. A process model with typically 50 FMs of two extensive and one intensive state, considering 8 chemical species, yields an equation system of size 1200×1200. In order to benefit from the canonical modelling approach as described in the main part of this work, the linear algebra package must provide the following key functionality:

- Each scalar element of a linear algebra object can be either an ordinary floating point variable or an instance of the symbolic datatype as described in Section A.1.

- There are block structures, which are compatible to the elementary linear algebra objects. The algebra preserves the block structure in its operators. As a consequence, this block structure can be applied recursively, i.e. a block structure contains child linear algebra objects, which again can be block structures.

- A limited number of special linear algebra objects are identified as such. Operators on these objects are implemented to efficiently exploit the additional information. In addition to full matrices and vectors, it is desirable to recognise the following special entities: zero matrix $\underset{\approx}{0}$, zero vector $\underset{\sim}{0}$, identity matrix $\underset{\approx}{I}$, and diagonal matrix $\underset{\approx}{D}$.

- Due to a significant amount of trivial operators, such as $\underset{\approx}{0} + \underset{\approx}{A}$ or $\underset{\approx}{I} \cdot \underset{\approx}{A}$, specific data access objects are handled by reference to avoid extensive copying efforts. These accessors therefore implement reference counting technology (Stroustrup, 1997).

With this, the linear algebra package is in itself a significant part of the solver, with functionality exceeding the scope of freely or commercially available software on this field. On the other hand, the elementary matrices, i.e. those not represented as a block structure, are of moderate size, typically 10×10. Thus, there is no need for highly developed algorithms made to handle huge matrices efficiently.

A.2.2 General software design

Figure A.6 shows the main classes of the library according to the current design. With a floating ownership, accessors are at first accumulated in the user objects, which are *Matrix* and *Vector*. These user objects implement algebraic operators and provide further functionality to access and manipulate the underlying accessors. Instances of block structures always accumulate further accessors. Block matrices of symmetric block structure are treated distinct from general block matrices. They represent the

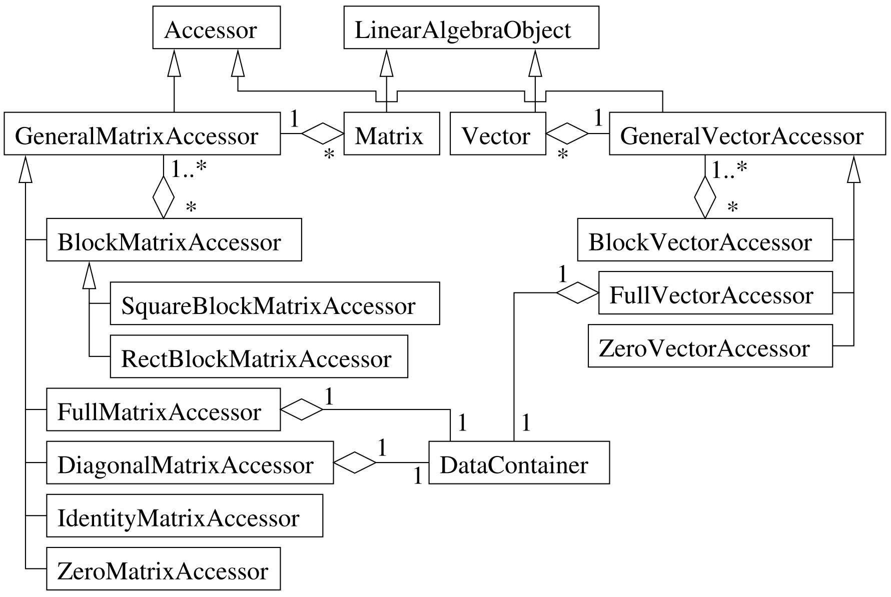


Figure A.6: *Static UML structure diagram of user objects, accessors and data containers.*

diagonal blocks in the canonical coefficient matrix and therefore assume a special role. The storage of actual floating point numbers is only required for a subset of linear algebra object types. Each instance of of these accessors accumulate one data container.

The current implementation supports a wide range of additional accessors, among others for symmetric matrices, matrices of constant value and dyadic matrices. However, in praxis, the accessor types shown in Figure A.6 are most relevant to accomplish an efficient implementation to solve the canonical equation system.

Figure A.7 shows a simplified sequence diagram of the inversion operation

$$\underset{\approx}{B}_{\mathsf{inv}} = \underset{\approx}{B}^{-1} \quad \text{with} \quad \underset{\approx}{B} = \begin{pmatrix} \underset{\approx}{I} & \underset{\approx}{C} \\ \underset{\approx}{0} & \underset{\approx}{A} \end{pmatrix}, \tag{A.1}$$

whereas $\underset{\approx}{A}$ is square but not necessarily of same size as $\underset{\approx}{I}$. There is a table for each operation, which holds a reference to operator objects. Tables of unary operators contain one operator object for each type of linear algebra object, this operation can be applied to. For binary operators, this table contains an operator object for each possible combination, of which many are trivial as $\underset{\approx}{A} + \underset{\approx}{0}$, while others are identical through the commutative law.

The operator object create a new accessor object, which is either processed on a higher block level operator, or as a result wrapped into the user classes *Matrix* and *Vector*.

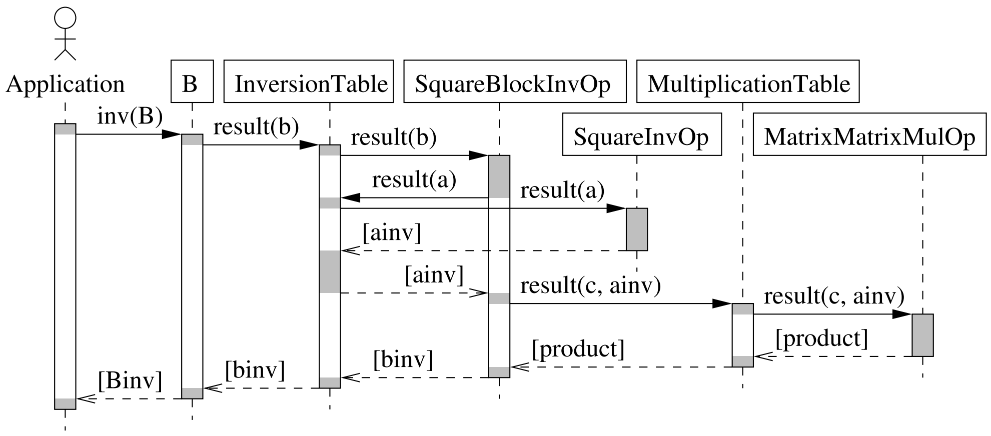


Figure A.7: *UML sequence diagram of a execution of a typical operator.*

A.2.3 Transposed and negated linear algebra objects

The computational effort of transposing or negating a matrix is quadratic in size and therefore not negligible. But obviously, $\underset{\approx}{A}+(-\underset{\approx}{B}) = \underset{\approx}{A}-\underset{\approx}{B}$, hence subsequent operators can often efficiently integrate these kind of modification steps by a simplified lazy evaluation technique. With this, it is

$$\begin{aligned}[(-\underset{\approx}{A}^{\mathrm{T}})(-\underset{\approx}{B}^{\mathrm{T}})]^{-1} + \underset{\approx}{C}^{\mathrm{T}} &= [\underset{\approx}{A}^{\mathrm{T}}\underset{\approx}{B}^{\mathrm{T}}]^{-1} + \underset{\approx}{C}^{\mathrm{T}} = [(\underset{\approx}{B}\underset{\approx}{A})^{\mathrm{T}}]^{-1} + \underset{\approx}{C}^{\mathrm{T}} = [(\underset{\approx}{B}\underset{\approx}{A})^{-1}]^{\mathrm{T}} + \underset{\approx}{C}^{\mathrm{T}} \\ &= [(\underset{\approx}{B}\underset{\approx}{A})^{-1} + \underset{\approx}{C}]^{\mathrm{T}} \qquad \text{(A.2)}\end{aligned}$$

such that only the multiplication, inversion and addition involve floating point operations, while the transposing and negating could be avoided.

However, this technique's drawback is the need for many new binary operators for matrices. The current implementation supports

$$\underset{\approx}{A} \pm \underset{\approx}{B}, \quad \underset{\approx}{A} \pm \underset{\approx}{B}^{\mathrm{T}}, \quad \underset{\approx}{A} \cdot \underset{\approx}{B}, \quad \underset{\approx}{A} \cdot \underset{\approx}{B}^{\mathrm{T}}, \quad \underset{\approx}{A}^{\mathrm{T}} \cdot \underset{\approx}{B}, \quad \underset{\approx}{A} \overset{E}{\cdot} \underset{\approx}{B} \quad \text{and} \quad \underset{\approx}{A} \overset{E}{\cdot} \underset{\approx}{B}^{\mathrm{T}}. \qquad \text{(A.3)}$$

With 9 different binary operators and 23 different accessors implemented, the number of potentially possible operators is approaching 5000. Even though only a fraction of these is defined and some more are identical or trivial, this approach generates a maintenance problem.

A.3 Remarks

Handling a symbolic data-type within a linear algebra package represents a working solution for the purpose of this work. However, performance problems can occur, if systems with large number of chemical species are instantiated. Furthermore, the rudimental realisation of lazy evaluation techniques on linear algebra level, restricted to transposition and negation, generates a huge number of required different binary operators, which represents a maintenance problem for current implementation.

It is desirable to alter the approach towards a fully symbolic linear algebra package with generalised operators. A standard linear algebra package can be utilised to represent the low level entities in order to ensure a validated and efficient processing. Operators can be generalised in many ways. For instance, multiplication of two diagonal matrices is identical to element-wise multiplication of the diagonal vectors. Other operators merely differ by different indexing of two-dimensional data-containers.

With such a symbolic linear algebra package, still fulfilling the requirements given in Section A.2.1, the canonical system could be symbolically decomposed and simplified. Significant amounts of maintenance in each iteration step can be avoided this way. Bauer *et al.* (2002) have developed a framework for symbolic computation, probably suitable to be extended towards block structure handling.

Appendix B

Thermodynamic models

B.1 Implemented contributions of thermodynamic models

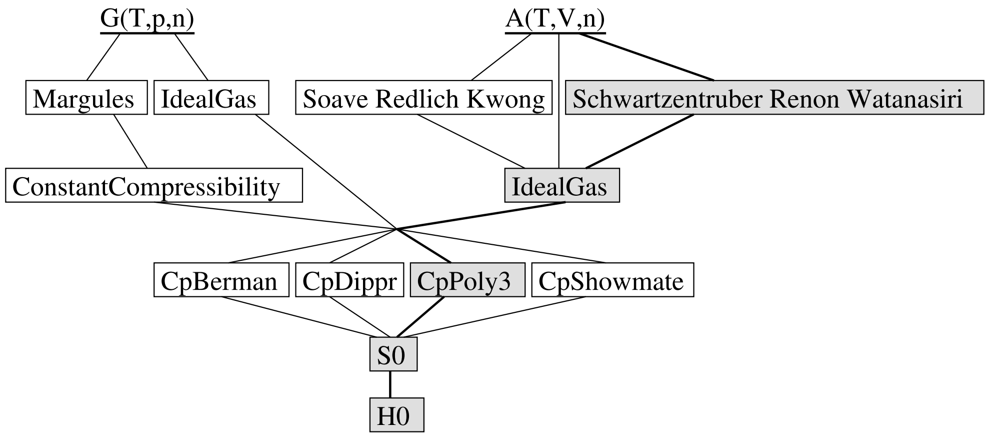


Figure B.1: *Simplified structure of available thermodynamic models in* Yasim.

Figure B.1 shows a graph of thermodynamic model contributions that are available in *Yasim* today. Grey shaded boxes indicate the contributions, which were used in the examples of this work. These contributions will be described more detailed in the following section. While H   models necessarily need to describe a nonzero compressibility in order to define pressure, this is no general requirement for G   models. In many applications, the compressibility of liquid phases is not of particular interest, but the canonical modelling approach relies on in in order to correlate volume and pressure as conjugated variables. The simplest way to describe a nonzero compressibility is implemented into the model contribution *ConstantCompressibility* and further described in Appendix B.3[1].

[1]The following sections contain model equations with many mathematical symbols. To maintain readability, some symbols receive a local interpretation in the scope of this appendix.

B.2 Schwartzentruber-Renon-Watanasiri equation of state

The thermodynamic model described in this section is used for all *Yasim* calculations within this work, in particular Chapter 4 and 6.

B.2.1 Pure species contributions

The reference state chemical potential μ_i^{ref} at reference conditions ($T^{\mathsf{ref}}, p^{\mathsf{ref}}$) is given by

$$\mu_i^{\mathsf{ref}} = \Delta_{\mathrm{f}} h_i^{\mathsf{ref}} - T\, s_i^{\mathsf{ref}} . \tag{B.1}$$

Here, $\Delta_{\mathrm{f}} h_i^{\mathsf{ref}}$ is the molar reference state heat of formation, and s_i^{ref} the molar reference state entropy. The next contribution is related to the ideal gas heat capacity, given as a third degree polynomial:

$$c_{p,i}(T) = c_{p,i}^{[0]} + (T - T^{\mathsf{ref}})c_{p,i}^{[1]} + (T^2 - T^{\mathsf{ref}^2})c_{p,i}^{[2]} + (T^3 - T^{\mathsf{ref}^3})c_{p,i}^{[3]} . \tag{B.2}$$

This heat capacity contributes as follows to the pure species chemical potential:

$$\mu_i(T) = \mu_i^{\mathsf{ref}} + \int_{T^{\mathsf{ref}}}^{T} c_{p,i}(\hat{T})\,\mathrm{d}\hat{T} - T \int_{T^{\mathsf{ref}}}^{T} \frac{c_{p,i}(\hat{T})}{\hat{T}}\,\mathrm{d}\hat{T} . \tag{B.3}$$

B.2.2 Helmholtz ideal gas state function

The ideal gas model contribution incorporates the effect of ideal mixing, and the pressure dependency given by the ideal gas law $p\,V = N\,R\,T$ with $N = \sum_i n_i$ and R the universal gas constant:

$$\mu_i^{\mathsf{ig}}(T, p, \underset{\sim}{n}) = \mu_i(T) + R\,T\,\ln \frac{n_i\,R\,T}{p_0\,V} . \tag{B.4}$$

The H energy is a state function given as $A = G - p\,V$:

$$A(T, V, \underset{\sim}{n})^{\mathsf{ig}} = \sum_i \mu_i^{\mathsf{ig}}(T, p, \underset{\sim}{n})\,n_i - N\,R\,T . \tag{B.5}$$

B.2.3 Schwartzentruber-Renon-Watanasiri residual contribution

The S -R -W equation of state (Schwartzentruber and Renon, 1989; Schwartzentruber *et al.*, 1990) is an extended version of the well-known S -R -K (SRK) equation of state (Soave, 1972). The P´ contribution C allows for a more precise description of liquid volumes (Péneloux *et al.*, 1982). Asymmetric interaction coefficients $\underset{\approx}{L}$ and polar parameters p_i are introduced to enhance the model for polar species and at supercritical temperatures. The equation of state is formulated as

$$p = \frac{NRT}{V+\bar{C}-\bar{B}} - \frac{\bar{A}}{(V+\bar{C})(V+\bar{C}+\bar{B})} \quad \text{with} \quad N = \sum_i n_i\,. \tag{B.6}$$

Furthermore:

$$\bar{A} = n_i n_j \sqrt{a_i a_j}\left(1 - k_{a,ij} - \frac{1}{N}\, l_{ij}(n_i - n_j)\right) \tag{B.7}$$

$$\bar{B} = \frac{1}{2N}\, n_i n_j (b_i + b_j)(1 - k_{b,ij}) \qquad \bar{C} = c_i n_i \tag{B.8}$$

$$a_i = \Omega_a\, \alpha_i \frac{R^2\, T_{c,i}^2}{p_{c,i}} \quad \text{with} \quad \Omega_a = \frac{1}{9}(2^{1/3}-1)^{-1} \approx 0.427480\ldots \tag{B.9}$$

$$b_i = \Omega_b\, \frac{R\, T_{c,i}}{p_{c,i}} \quad \text{with} \quad \Omega_b = \frac{1}{3}(2^{1/3}-1) \approx 0.086640\ldots \tag{B.10}$$

$$c_i = c_{0,i} + c_{1,i}\, T_{r,i} + c_{2.i}\, T_{r,i}^2 \quad \text{with} \quad T_{r,i} = T/T_{c,i} \tag{B.11}$$

$$\alpha_i = \begin{cases} \left[1 + m_i(1 - T_{r,i}^{1/2}) - (1 - T_{r,i})(p_{1,i} + p_{2,i}\, T_{r,i} + p_{3,i}\, T_{r,i}^2)\right]^2 & \text{for } T_{r,i} \le 1 \\ \left[\exp\left(\gamma_i (1 - T_{r,i}^{d_i})\right)\right]^2 \text{ with } \begin{array}{l} \gamma_i = 1 - \frac{1}{d_i} \text{ and} \\ d_i = 1 + \frac{m_i}{2} - (p_{1,i} + p_{2,i} + p_{3,i}) \end{array} & \text{for } T_{r,i} > 1 \end{cases} \tag{B.12}$$

$$m_i = 0.48508 + 1.55171\, \omega_i - 0.15613\, \omega_i^2 \tag{B.13}$$

$$k_{a,ij} \overset{\text{def.}}{=} k_{a,ji} = k_{a,ij}^{[0]} + k_{a,ij}^{[1]}\, T + k_{a,ij}^{[2]}/T \tag{B.14}$$

$$k_{b,ij} \overset{\text{def.}}{=} k_{b,ji} = k_{b,ij}^{[0]} + k_{b,ij}^{[1]}\, T + k_{b,ij}^{[2]}/T \tag{B.15}$$

$$l_{ij} \overset{\text{def.}}{=} -l_{ji} = l_{ij}^{[0]} + l_{ij}^{[1]}\, T + l_{ij}^{[2]}/T. \tag{B.16}$$

The parameters are critical temperatures $T_{c,i}$, critical pressures $p_{c,i}$, acentric factors ω_i, polar parameters $p_{1,i}$, $p_{2,i}$, and $p_{3,i}$, interaction coefficient matrices $k_{a,ij}^{[0])}$, $k_{a,ij}^{[1]}$, $k_{a,ij}^{[2]}$, $k_{b,ij}^{[0]}$, $k_{b,ij}^{[1]}$, $k_{b,ij}^{[2]}$, $l_{ij}^{[0]}$, $l_{ij}^{[1]}$, and $l_{ij}^{[2]}$, and liquid volume parameters $c_{0,i}$, $c_{1,i}$, and $c_{2,i}$.

The residual H state function is then obtained by integration of residual pressure over volume as

$$A^{\text{res}} = \int_V^\infty p - \frac{NRT}{V}\, dV = NRT\, \ln\frac{V}{V+\bar{C}-\bar{B}} + \frac{\bar{A}}{\bar{B}}\, \ln\frac{V+\bar{C}}{V+\bar{C}+\bar{B}}\,. \tag{B.17}$$

The complete H energy state function is given as

$$A(T, V, \underset{\sim}{n}) = A^{\text{ig}} + A^{\text{res}}\,. \tag{B.18}$$

B.3 Constant compressibility model contribution

This model contribution describes a phase with constant compressibility $\varepsilon_{p,i}$, thermal expansion coefficient $\varepsilon_{T,i}$, and a given reference molar volume $\bar{v}_i^{\mathrm{ref}}$.

Compressibility and the thermal expansion coefficient are defined as follows:

$$\varepsilon_T = \frac{1}{V}\frac{\partial V}{\partial T} \quad \text{and} \quad \varepsilon_p = -\frac{1}{V}\frac{\partial V}{\partial p}\,. \tag{B.19}$$

These definitions can be formulated also on the partial volume $\bar{v}_i$, such that $\varepsilon_{T,i}$ and $\varepsilon_{p,i}$ are interpreted as pure species properties. The calculated compressibility of a mixture is then a consequence of these properties and possible mixing effects:

$$\varepsilon_{T,i} = \frac{1}{\bar{v}_i}\frac{\partial \bar{v}_i}{\partial T} \quad \text{and} \quad \varepsilon_{p,i} = -\frac{1}{\bar{v}_i}\frac{\partial \bar{v}_i}{\partial p}\,. \tag{B.20}$$

Integration and combination gives

$$\bar{v}_i(T,p) = \bar{v}_i^{\mathrm{ref}}(T^{\mathrm{ref}},p^{\mathrm{ref}})\,\exp\left[\varepsilon_{T,i}(T-T^{\mathrm{ref}}) - \varepsilon_{p,i}(p-p^{\mathrm{ref}})\right]\,. \tag{B.21}$$

Furthermore

$$\Delta\mu_i = \int_{p^{\mathrm{ref}}}^{p} \bar{v}_i\,dp = \frac{\bar{v}_i^{\mathrm{ref}}(T^{\mathrm{ref}},p^{\mathrm{ref}})}{\varepsilon_{p,i}}\,\exp\left[\varepsilon_{T,i}(T-T^{\mathrm{ref}})\right]\left(1-\exp\left[-\varepsilon_{p,i}(p-p^{\mathrm{ref}})\right]\right). \tag{B.22}$$

With a realistic parameterisation for condensed phases, moderate pressures and temperatures do not yield a significant contribution to calculated thermodynamic properties, in particular regarding phase equilibrium calculations. However, the contribution ensures a consistent correlation between pressure and volume.

Appendix C

State functions and transformations

C.1 Properties of homogeneous state functions

Theorem C.1 *A state function $P(\utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})$ of extensive canonical variables $\utilde{x}_{\mathcal{E}}$ and intensive canonical variables $\utilde{x}_{\bar{\mathcal{E}}}$ can be represented in the* E *-integrated form, iff it is first-order homogeneous:*

$$P(\utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}}) = \frac{\partial P}{\partial \utilde{x}_{\mathcal{E}}} \utilde{x}_{\mathcal{E}} \quad \Leftrightarrow \quad P(\utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}}) = \frac{1}{\psi} P(\psi \utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}}) \quad \textit{for } \psi \neq 0\,. \tag{C.1}$$

Proof (⇐): Let $\hat{\utilde{x}}_{\mathcal{E}} = \frac{1}{\psi}\, \utilde{x}_{\mathcal{E}}$ and derive with respect to ψ:

$$\frac{\partial}{\partial \psi} P(\utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}}) = \frac{\partial}{\partial \psi}\left(\frac{1}{\psi} P(\psi\, \hat{\utilde{x}}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})\right) \Rightarrow 0 = \frac{1}{\psi} \frac{\partial P}{\partial(\psi\, \hat{\utilde{x}}_{\mathcal{E}})} \hat{\utilde{x}}_{\mathcal{E}} - \frac{1}{\psi^2} P(\psi\, \hat{\utilde{x}}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})\,. \tag{C.2}$$

Multiplication with ψ^2 and back-substitution of $\psi\, \hat{\utilde{x}}_{\mathcal{E}} = \utilde{x}_{\mathcal{E}}$ yields the E -integrated form. □

(⇒): Integration of the differentiated form in (⇒) to ψ yields

$$\frac{1}{\psi} P(\psi\, \hat{\utilde{x}}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}}) = \hat{P}(\hat{\utilde{x}}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})\,. \tag{C.3}$$

In particular, $\hat{P}$ is not dependent on ψ, but a yet unknown function of all canonical variables $(\hat{\utilde{x}}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})$. The equation must still hold for all $\psi \neq 0$ including $\psi = 1$, concluding $\hat{P} = P$. □

Furthermore, derivation of the E -integrated form with respect to $\utilde{x}_{\mathcal{E}}$ yields

$$\frac{\partial P(\utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})}{\partial \utilde{x}_{\mathcal{E}}} = \frac{\partial^2 P}{\partial \utilde{x}_{\mathcal{E}} \partial \utilde{x}_{\mathcal{E}}} \utilde{x}_{\mathcal{E}} + \frac{\partial P(\utilde{x}_{\mathcal{E}}, \utilde{x}_{\bar{\mathcal{E}}})}{\partial \utilde{x}_{\mathcal{E}}} \quad \Rightarrow \quad \frac{\partial^2 P}{\partial \utilde{x}_{\mathcal{E}} \partial \utilde{x}_{\mathcal{E}}} \utilde{x}_{\mathcal{E}} = \utilde{0} \tag{C.4}$$

as a necessary, but not sufficient property of first-order homogeneous state functions.

C.2 State function transformations

C.2.1 Preservation of homogeneity

The L and M transformations are used to obtain a state function representation from another. The homogeneity of thermodynamic state functions is an important feature in the concept of canonical modelling. The preservation of homogeneity throughout transformations is therefore proven here.

Theorem C.2 *Any* L *transformed state function $\hat{P} = \mathfrak{L}_j[P]$ of a given a homogeneous first-order state function P is also first-order homogeneous.*

Proof: For the case $x_j \in \mathcal{E}$, the subtracted term in Equation (2.16) is identical to the contribution to be removed from the E -integrated form. $\hat{x}_j \in \bar{\mathcal{E}}$ does not give a new contribution. If $x_j \in \bar{\mathcal{E}}$, consider the implicit formulation of Equation (2.16). With $\partial P/\partial x_j = \hat{x}_j$ and $x_j = -\partial\hat{P}/\partial\hat{x}_j$, it is

$$\hat{P}(\underset{\sim}{\hat{x}}) = P(\underset{\sim}{x}) + \frac{\partial\hat{P}(\underset{\sim}{\hat{x}})}{\partial\hat{x}_j}\,\hat{x}_j\,. \tag{C.5}$$

The added term is identical to the term to be added to the E -integrated form, since $\hat{x}_j \in \mathcal{E}$. As proven for theorem C.1, this is sufficient condition for a homogeneous state function. □

Theorem C.3 *Any* M *transformed state function $\hat{P} = \mathfrak{M}_j[P]$ of a given a homogeneous first-order state function P with $x_j \in \mathcal{E}$ and $\partial P/\partial x_j \neq 0$ is likewise first-order homogeneous.*

Proof: Consider the total differential of P at constant $\underset{\sim}{x}_{\bar{\mathcal{E}}}$, separating out the term containing x_j and divide by $\frac{\partial P}{\partial x_j}$ and solve for $\mathrm{d}x_j$ to obtain the total differential of $\mathfrak{M}_j[P]$:

$$\mathrm{d}P = \frac{\partial P}{\partial x_j}\,\mathrm{d}x_j + \sum_{i\in\mathcal{E}\setminus\{j\}} \frac{\partial P}{\partial x_i}\,\mathrm{d}x_i \quad\Leftrightarrow\quad \mathrm{d}x_j = \frac{1}{\frac{\partial P}{\partial x_j}}\,\mathrm{d}P - \sum_{i\in\mathcal{E}\setminus\{j\}} \frac{\frac{\partial P}{\partial x_i}}{\frac{\partial P}{\partial x_j}}\,\mathrm{d}x_i\,. \tag{C.6}$$

In transformed notation regarding the differentials, this is

$$\mathrm{d}\hat{P} = \frac{1}{\frac{\partial P}{\partial x_j}}\,\mathrm{d}\hat{x}_j - \sum_{i\in\mathcal{E}\setminus\{j\}} \frac{\frac{\partial P}{\partial x_i}}{\frac{\partial P}{\partial x_j}}\,\mathrm{d}\hat{x}_i \quad\Leftrightarrow\quad \frac{\partial\hat{P}}{\partial\hat{x}_j} = \frac{1}{\frac{\partial P}{\partial x_j}} \wedge \frac{\partial\hat{P}}{\partial\hat{x}_i} = -\frac{\frac{\partial P}{\partial x_i}}{\frac{\partial P}{\partial x_j}} \quad \text{for } i \neq j\,. \tag{C.7}$$

Based on the partial derivatives obtained, the E -integrated form is

$$\hat{P} = \frac{1}{\frac{\partial P}{\partial x_j}}\,\hat{x}_j - \sum_{i\in\mathcal{E}\setminus\{j\}} \frac{\frac{\partial P}{\partial x_i}}{\frac{\partial P}{\partial x_j}}\,\hat{x}_i \quad\text{or}\quad x_j = \frac{1}{\frac{\partial P}{\partial x_j}}\,P - \sum_{i\in\mathcal{E}\setminus\{j\}} \frac{\frac{\partial P}{\partial x_i}}{\frac{\partial P}{\partial x_j}}\,x_i\,, \tag{C.8}$$

which is equivalent to the presumed E -integrated form of P. □

C.2.2 Jacobian matrices

With different state functions applied simultaneously within one process model, the sensitivity of a transformed state $\underset{\sim}{\hat{x}}$ with respect to the original state $\underset{\sim}{x}$ is required to calculate consistent updates. The J matrix for a chain of transformations can be subdivided by chain-rule into the product of J -matrices for single transformations.

For the L transformation, it is

$$\hat{x}_j = g_j \quad \text{, therefore} \quad \frac{\partial \hat{x}_j}{\partial \underset{\sim}{x}} = \frac{\partial^2 P}{\partial x_j \underset{\sim}{x}} \,. \tag{C.9}$$

Accordingly, the inverse L transformation yields

$$\hat{x}_j = -g_j \quad \text{, therefore} \quad \frac{\partial \hat{x}_j}{\partial \underset{\sim}{x}} = -\frac{\partial^2 P}{\partial x_j \underset{\sim}{x}} \,. \tag{C.10}$$

As in a M transformation, $\hat{x}_j = P(\underset{\sim}{x})$, therefore $\partial \hat{x}_j / \partial \underset{\sim}{x} = \partial P / \partial \underset{\sim}{x}$.

C.3 State functions and thermodynamic properties

The columns x_t, x_m and x_c in Table C.1 show the canonical variable sets of some selected state functions, followed by the physical interpretation of first and second-order derivatives of that state function with respect to the canonical variables. Horizontal lines separate groups of state functions, which can be transformed into

Table C.1: *Thermodynamic state functions P and physical interpretation of the derivatives with respect to their canonical variables.*

P	**x**$_\mathbf{t}$	**x**$_\mathbf{m}$	**x**$_\mathbf{c}$	**P**$_\mathbf{t}$	**P**$_\mathbf{m}$	**P**$_\mathbf{c}$	**P**$_\mathbf{tt}$	**P**$_\mathbf{tm}$	**P**$_\mathbf{tc}$	**P**$_\mathbf{mm}$	**P**$_\mathbf{mc}$
G	T	p	n_i	$-S$	V	μ_i	$-\frac{C_p}{T}$	$V\,\varepsilon_T$	$-\bar{S}_i$	$-V\,\varepsilon_p$	$\bar{V}_i$
A	T	V	n_i	$-S$	$-p$	μ_i	$-\frac{C_V}{T}$	$-\frac{\varepsilon_T}{\varepsilon_p}$	$\frac{\varepsilon_T}{\varepsilon_p}\,\bar{V}_i - \bar{S}_i$	$\frac{1}{V\,\varepsilon_p}$	$-\frac{\bar{V}_i}{V\,\varepsilon_p}$
U	S	V	n_i	T	$-p$	μ_i	$\frac{T}{C_V}$	$-\frac{\varepsilon_T\,T}{\varepsilon_p\,C_V}$	$(\frac{\varepsilon_T}{\varepsilon_p}\bar{V}_i - \bar{S}_i)\,\frac{T}{C_V}$	$\frac{1}{\varepsilon_p\,V} + \frac{\varepsilon_T^2\,T}{\varepsilon_p{}^2\,C_V}$	$\frac{T\,\varepsilon_T(\bar{S}_i\,\varepsilon_p - \varepsilon_T\,\bar{V}_i)}{C_V\,\varepsilon_p{}^2} - \frac{\bar{V}_i}{\varepsilon_p\,V}$
H	S	p	n_i	T	V	μ_i	$\frac{T}{C_p}$	$\frac{V\,T\,\varepsilon_T}{C_p}$	$-\frac{T\,\bar{S}_i}{C_p}$	$\frac{V^2\,\varepsilon_T^2\,T}{C_p} - V\,\varepsilon_p$	$\bar{V}_i - \frac{T\,V\,\bar{S}_i\,\varepsilon_T}{C_p}$
S	H	p	n_i	$\frac{1}{T}$	$-\frac{V}{T}$	$-\frac{\mu_i}{T}$	$-\frac{1}{T^2\,c_p}$	$\frac{V(1-T\,\varepsilon_T)}{T^2\,c_p}$	$\frac{\mu_i + T\,\bar{S}_i}{T^2\,c_p}$	$\dots$	
$\mathbb{W}$	H	$\frac{V}{T}$	n_i	$\frac{1}{T}$	p	$-\frac{\mu_i}{T}$	$\dots$				
S	U	V	n_i	$\frac{1}{T}$	$\frac{p}{T}$	$-\frac{\mu_i}{T}$	$-\frac{1}{T^2 c_V}$	$\frac{\varepsilon_T\,T - \varepsilon_p\,p}{T^2\,\varepsilon_p\,c_V}$	$\dots$		

each other by L transformations. A M transformation is necessary to reach from one group into another (see Section 2.5.2). The H elements of M -transformed surfaces S and $\mathbb{W}$ can be physically interpreted, but the complexity of their analytical expressions in many cases prohibits a practical use. Therefore, second-order information of these state functions is used solely as the sensitivity of the gradient with respect to the canonical variables. The first H elements of $S(U, V, \underset{\sim}{n})$ are shown in Table C.1.

Table C.1 can also be used to extract M -relations, as for example

$$P_{\mathrm{tm}} = \frac{\partial}{\partial x_{\mathrm{m}}}\left(\left.\frac{\partial P}{\partial x_{\mathrm{t}}}\right|_{x_{\mathrm{m}},x_{\mathrm{c}}}\right)\Bigg|_{x_{\mathrm{t}},x_{\mathrm{c}}} = \frac{\partial}{\partial x_{\mathrm{t}}}\left(\left.\frac{\partial P}{\partial x_{\mathrm{m}}}\right|_{x_{\mathrm{t}},x_{\mathrm{c}}}\right)\Bigg|_{x_{\mathrm{m}},x_{\mathrm{c}}} \quad \text{e.g.} \quad \left.\frac{\partial V}{\partial T}\right|_{p,n_i} = -\left.\frac{\partial S}{\partial p}\right|_{T,n_i} . \tag{C.11}$$

Further interpretations are available with help of the E -integrated form of state functions:

$$\begin{aligned} G &= \underset{\sim}{\mu}\,\underset{\sim}{n}\,, & A &= -p\,V + \underset{\sim}{\mu}\,\underset{\sim}{n}\,, \\ U &= T\,S - p\,V + \underset{\sim}{\mu}\,\underset{\sim}{n}\,, \text{ and} & H &= T\,S + \underset{\sim}{\mu}\,\underset{\sim}{n}\,. \end{aligned}$$

As an example, using the E -representation of H, the following non-canonical derivative can be analysed:

$$\left.\frac{\partial H}{\partial p}\right|_{T,n_i} = \frac{\partial}{\partial T}\left(T\,S + \underset{\sim}{\mu}\,\underset{\sim}{n}\right)\Big|_{p,n_i} = S - T\,\left.\frac{\partial(-S)}{\partial T}\right|_{p,n_i} + \left.\frac{\partial \underset{\sim}{\mu}}{\partial T}\right|_{p,n_i}\underset{\sim}{n} = -T\,\left.\frac{\partial(-S)}{\partial T}\right|_{p,n_i} = C_p\,. \tag{C.12}$$

Table C.2 can be used to back-calculate heat capacities C_p and C_V, thermal expansion

Table C.2: *Thermodynamic properties as a function of canonical derivatives.*

P	$\mathbf{x_t\ x_m\ x_c}$	$\mathbf{C_p}$	$\mathbf{C_V}$	$\boldsymbol{\varepsilon_T}$	$\mathbf{\bar{S}_i}$	$\boldsymbol{\varepsilon_p}$	$\mathbf{\bar{V}_i}$
G	$T\ \ p\ \ n_i$	$-G_{\mathrm{tt}}\,T$	$T\left(\frac{G_{\mathrm{tm}}^2}{G_{\mathrm{mm}}} - G_{\mathrm{tt}}\right)$	$\frac{G_{\mathrm{tm}}}{V}$	$-G_{\mathrm{tc}}$	$-\frac{G_{\mathrm{mm}}}{V}$	G_{mc}
A	$T\ \ V\ \ n_i$	$T\left(\frac{A_{\mathrm{tm}}^2}{A_{\mathrm{mm}}} - A_{\mathrm{tt}}\right)$	$-A_{\mathrm{tt}}\,T$	$-\frac{A_{\mathrm{tm}}}{V\,A_{\mathrm{mm}}}$	$\frac{A_{\mathrm{tm}}\,A_{\mathrm{mc}} - A_{\mathrm{mm}}\,A_{\mathrm{tc}}}{A_{mm}}$	$\frac{1}{V\,A_{\mathrm{mm}}}$	$-\frac{A_{\mathrm{mc}}}{A_{\mathrm{mm}}}$

coefficient ε_T, compressibility ε_p, partial entropy $\bar{S}_i$ and partial volume $\bar{V}_i$ from given derivative information. Heat capacity at constant p (C_p) and at constant V (C_V) are linked by $C_V = C_p - T\,V\,\varepsilon_T^2/\varepsilon_p$. Combining these quantities, a set of dependent thermodynamic properties can be obtained. The symbols introduced in Table C.3 are not consistent with the main part of this work.

Table C.3: *Derived thermodynamic properties – M_i is the molar weight of species i.*

Quantity	Formula	Quantity	Formula
Total molar quantity	$N = \sum_i n_i$	Molar fraction	$x_i = n_i / N$
Total mass	$M = \sum_i M_i\, n_i$	Mass fraction	$w_i = M_i\, n_i / M$
Concentration	$c_i = n_i / V$	Average molar mass	$\bar{M} = M / N$
Density	$\varrho = M / V$	Partial enthalpy	$\bar{H}_i = \mu_i + T\, \bar{S}_i$
Molar heat capacities	$c_{p/V} = C_{p/V} / N$	Adiabatic exponent	$\kappa = C_p / C_V$
Joule Thomson coefficient	$JT = V/C_p\,(T\,\varepsilon_T - 1)$	Speed of sound	$v_{\mathsf{sonic}} = \sqrt{\kappa / (\varrho\, \varepsilon_p)}$

Appendix D

Dynamic simulation

The main scope of this work is focused on canonical process modelling, in this appendix exploring the feasibility to perform process modelling tasks beyond steady-state simulation. Dynamic process simulation definitely holds more challenges then those which can be addressed in this appendix. Some of them are consistent initialisation, stiffness, event handling, and a wide range of index problems. This appendix therefore only sketches the basic approach, how dynamic behaviour can be described within the framework of canonical modelling.

In order to explore the feasibility of dynamic simulation based on a canonical process model representation, it is necessary to define different forms of dynamic simulation. The data-reconciliation example in Section 4.5 is generally not considered as dynamic, even though the process state is calculated as a function of time. Hence, if the process model itself has no memory, but only time-dependent process parameters give variations of state in time, the process model is called *quasi-steady-state*. On the other hand, a *dynamic* process model contains some kind of memory, represented by an accumulated (or integrated) state. In the context of canonical process models, there are two distinct kinds of potential dynamic elements: (i) canonical (thermodynamic) dynamics (in $\underset{\sim}{x}$), as for instance a buffer tank or a pipe hold-up, and (ii) non-canonical dynamics (in $\underset{\sim}{y}$), as for instance any control structures and limited valve-opening rates:

$$\underset{\sim}{x}_2(t) = \underset{\sim}{x}_2(t_0) + \int_{t_0}^{t} \sum_i \dot{\underset{\sim}{x}}_{1,i}(\hat{t})\,\mathrm{d}\hat{t} \quad \text{(i)} \qquad \underset{\sim}{y}_2(t) = \underset{\sim}{y}_2(t_0) + \int_{t_0}^{t} \underset{\sim}{y}_1(\hat{t})\,\mathrm{d}\hat{t} \quad \text{(ii)}\,. \quad \text{(D.1)}$$

An example is a buffer tank with the difference in state variables of incoming and outgoing streams $\dot{\underset{\sim}{x}}_{1,i}$, and the accumulated state $\underset{\sim}{x}_2$ inside the buffer tank. However, the integrands can depend on the accumulated variables, for instance if the outgoing flow is dependent on the liquid level in the tank. This dependency can be direct or indirect through canonical or constitutive process constraints.

The second case requires integration not only of state variables (see $\dot{\underset{\sim}{x}}_{1,i}(t)$ in

Equation (D.1), but of calculated properties $y_1(t)$ as well. As an example, the actual valve position is no longer a process parameter u, but the integrated actuator speed.

With this, the integrated properties $y_2(t)$ become non-canonical state variables as well. State variables other than those of thermodynamic nature are an inevitable consequence of the fact that non-canonical dynamic effects, such as control structures are considered.

D.1 Transition from steady-state to dynamic simulation

Even though material flows described by state vectors $\dot{\underset{\sim}{x}}$ are supplemented by time derivatives of accumulated states $\mathrm{d}\underset{\sim}{x}/\mathrm{d}t$, the interface between two FM remains restricted to couplings of streams. Interactions of accumulated states with each other always is described by either material streams between them, or constitutive equations. This restriction is not limiting the flexibility of the simulation tool, but greatly improves maintainability, as the collaborations are not changed from figure 2.4.

In a traditional switch from steady-state to dynamic simulation, all FM are supplemented with dynamic features instantaneously, i.e. hold-up volumes are assigned to every flash tank and even valve, and constitutive equations are exchanged by others more suitable for dynamic simulation automatically. As a result, the dynamic behaviour of the system is immediately very complex, and the origins of high-frequency oscillations can hardly be understood. Extensive use of default geometric parameters yields a process model, which looks much more predictive than it really is.

As the interface between dynamic FMs can be kept compatible with steady-state FMs, the strategy to switch from steady-state to a dynamic simulation from a user's point of view can be designed as a continuous transition:

1. Originally, there is a steady-state process model. The simulation can be interpreted as a single point calculation.
2. Without changes in the process model, an integrator can be started. As there is no accumulated state, and all process parameters are independent of time, the calculated properties are constant in time and still represent the steady-state solution.
3. Time-dependent process parameters are defined. There is still no dynamic behaviour (accumulated states), and the calculation results represent a series of point calculations of the steady-state process model.
4. Individual FMs are replaced with a dynamic equivalent, for instance a relevant buffer tank. From now, the process model shows its own dynamic effects.
5. Accumulated and stream-based variables are used in constitutive equations. The pressure of a liquid outlet from a tank is set into an algebraic relationship to the liquid level and the pressure in the tank.
6. The explicit integration and differentiation of process properties is used, e.g. to implement control structures and limited valve opening rates. This technique requires the maintenance of non-canonical state variables.

D.2 A sketch example

Consider the example process shown in Figure D.1. To emphasise the paradigm of

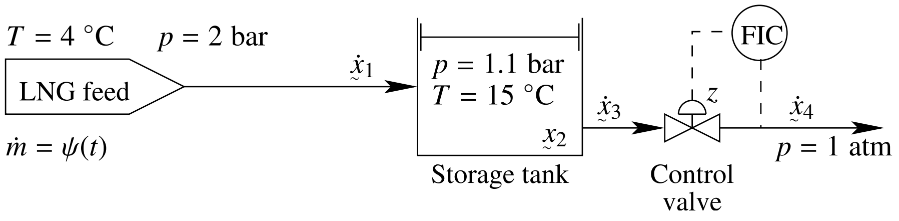


Figure D.1: *Dynamic process with PID control.*

only considering relevant dynamics, only the storage tank holds an accumulated state $\underset{\sim}{x}_2$, of which pressure and temperature are even specified. The feed stream enters with constant temperature and pressure, but time-variant flow. The valve with constant outlet pressure utilises a pressure-flow relation as described in Section 3.6.3.

The canonical system is not entirely different from that of a steady-state process:

$$
\left[\begin{array}{cc|cc:cc|cc}
\underset{\approx}{H}_1 & \underset{\approx}{I} & & & & & & \\
\underset{\approx}{I} & & & & & & & \\
\hline
 & & \underset{\approx}{H}_2 & \underset{\approx}{I} & & & & \\
-\underset{\approx}{I} & & \underset{\approx}{I} & & \underset{\approx}{I} & & & \\
\hdashline
 & & & \underset{\approx}{I} & \underset{\approx}{H}_3 & \underset{\sim}{a} & & \\
 & & & & \underset{\sim}{a} & & & \\
\hline
 & & & & & & \underset{\approx}{H}_4 & \underset{\approx}{I} \\
 & & & & -\underset{\approx}{I} & & \underset{\approx}{I} &
\end{array}\right]
\cdot
\left[\begin{array}{c}
\Delta\dot{\underset{\sim}{x}}_1 \\
\underset{\sim}{\lambda}_1 \\
\hline
\Delta(\mathrm{d}\underset{\sim}{x}_2/\mathrm{d}t) \\
\underset{\sim}{\lambda}_{2/3} \\
\hdashline
\Delta\dot{\underset{\sim}{x}}_3 \\
\underset{\sim}{\lambda}_3 \\
\hline
\Delta\dot{\underset{\sim}{x}}_4 \\
\underset{\sim}{\lambda}_4
\end{array}\right]
=
\left[\begin{array}{c}
-\underset{\sim}{g}_1 \\
\underset{\sim}{\delta}_1 + \underset{\sim}{\alpha}_1 \\
\hline
-\underset{\sim}{g}_2 \\
\underset{\sim}{\delta}_2 + \underset{\sim}{\alpha}_2 \\
\hdashline
-\underset{\sim}{g}_3 \\
\underset{\sim}{\alpha}_3 \\
\hline
-\underset{\sim}{g}_4 \\
\underset{\sim}{\delta}_4 + \underset{\sim}{\alpha}_4
\end{array}\right]
\begin{array}{l}
\leftarrow \text{LNG feed} \\
\\
\leftarrow \text{Storage tank (hold-up)} \\
\\
\leftarrow \text{Storage tank (outlet)} \\
\\
\leftarrow \text{Valve} \\
\\
\end{array}
\tag{D.2}
$$

The indices of the state variable vectors are consistent with the stream numbers in Figure D.1. Both valve and source module are identical, while the tank coefficient matrix reminds one of that of a flow splitter (cf. Section 3.5.5). Actually, as the flow splitter shares a single intensive state for both of its physical phases, the same applies to the storage tank. The outlet stream $\dot{\underset{\sim}{x}}_3$ is split from the derivative of the accumulated state $\underset{\sim}{x}_2$.

The constitutive equation system is modified only by including two new operations, namely integration and differentiation. The integrator is necessary to include non-canonical dynamic effects, such as limited changes in valve position, while the differentiator mainly is applied for process control equipment. In this example, the immediate valve position is determined by the flow control (FIC), implemented as a

PID controller as follows:

$$z = z_0 + k\Big[\underbrace{(\dot{V}_4 - \dot{V}_{4,\mathsf{SP}})}_{\mathtt{V-V_{SP}}} + \frac{1}{t_{\mathsf{Int}}}\underbrace{\int_{t_0}^{t}(\dot{V}_4 - \dot{V}_{4,\mathsf{SP}})\,\mathrm{d}t}_{\mathtt{int(V-V_{SP})}} + t_{\mathsf{Diff}}\underbrace{\frac{\mathrm{d}\dot{V}_4}{\mathrm{d}t}}_{\mathtt{diff(V)}}\Big]. \tag{D.3}$$

The expressions beneath the braces indicate a possible syntax for the parser of constitutive equations. The PID control is simply coded as a constitutive equation:

$$\mathtt{z_0} + \mathtt{k}\,((\mathtt{V} - \mathtt{V_{SP}}) + \mathtt{int}(\mathtt{V} - \mathtt{V_{SP}})/\mathtt{T_{Int}} + \mathtt{T_{Diff}}\,\mathtt{diff}(\mathtt{V})) - \mathtt{z} = \mathtt{0}\,. \tag{D.4}$$

The internal implementation of the `int` and `diff` operations are dependent on the actual integration method.

As an important fact, $\underset{\approx}{H}_2$ and $\underset{\sim}{g}_2$ are naturally calculated from $\underset{\sim}{x}_2$. For an explicit solving strategy, these derivatives are therefore constant during iterations of one time step. In this case, constitutive equations must be based on the state $\underset{\sim}{x}_2$ of the previous time-step in order to obtain correct derivatives to maintain second-order convergence. For an implicit solution method, $\underset{\approx}{H}_2$ and $\underset{\sim}{g}_2$ are to be evaluated on the next time step.

Appendix E

Numerical methods and matrix computations

E.1 Block LU-decomposition

In order to solve equation systems related to canonical process models efficiently, the sparse block-structure of linear algebra objects is exploited to avoid redundant floating point operations. The LU-decomposition is conducted using the following algorithm on a block-matrix $\underset{\approx}{B}$:

```
for $k = 1 : n - 1$
    for $r = k + 1 : n$
        solve for $\underset{\approx}{\hat{B}}_{r,k}$: $\underset{\approx}{\hat{B}}_{r,k}\,\underset{\approx}{B}_{k,k} = \underset{\approx}{B}_{r,k}$
        $\underset{\approx}{B}_{r,k} := \underset{\approx}{\hat{B}}_{r,k}$
    end for
    for $c = k + 1 : n$
        for $r = k + 1 : n$
            $\underset{\approx}{B}_{k,c} := \underset{\approx}{B}_{r,c} - \underset{\approx}{B}_{r,k}\,\underset{\approx}{B}_{k,r}$
        end for
    end for
end for
```

This version without pivoting requires $\underset{\approx}{B}$ to contain invertible diagonal blocks, which is the case for solvable composite flowsheet module coefficient matrices.

However, building blocks of equilibrium flowsheet modules contain singular H -matrices of thermodynamic state-functions and zero-matrices. In order to decompose these matrices on block-level, it is therefore necessary to pivot both rows and columns, even though row-pivoting is sufficient to perform a scalar LU-decomposition on a non-singular matrix (Golub and Loan, 1996). A typical example is given in Section 3.2.1. Still, the number of atomic flowsheet modules is limited and their internal structure fixed, such that an appropriate permutation can be performed prior to the application of the non-pivoting algorithm above.

E.2 Solution strategies for non-blockinvertible systems

E.2.1 One-phase systems of chemical equilibrium

The non-singularity of a matrix does not imply block-invertibility. A common example is any reactor coefficient matrix as given in Section 3.2.2:

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} \\ \underset{\approx}{A} & \end{pmatrix} \begin{pmatrix} \Delta \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} \\ \underset{\sim}{\delta} \end{pmatrix} . \tag{E.1}$$

According to Equation (2.15), $\underset{\approx}{H}$ is singular, while $\underset{\approx}{A}$ is not even square, hence no pivot block can be found. Furthermore, no LU-decomposition exists, since

$$\begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} \\ \underset{\approx}{A} & \end{pmatrix} = \begin{pmatrix} \underset{\approx}{L}_{1,1} & \\ \underset{\approx}{L}_{2,1} & \underset{\approx}{L}_{2,2} \end{pmatrix} \begin{pmatrix} \underset{\approx}{U}_{1,1} & \underset{\approx}{U}_{1,2} \\ & \underset{\approx}{U}_{2,2} \end{pmatrix} \quad \Rightarrow \quad \underset{\approx}{L}_{1,1}\,\underset{\approx}{U}_{1,1} = \underset{\approx}{H} . \tag{E.2}$$

As $\underset{\approx}{H}$ is singular, either of $\underset{\approx}{L}_{1,1}$ and $\underset{\approx}{U}_{1,1}$ must be singular, which can not be a result of a successful decomposition.

Repartitioning

Repartitioning can be an efficient strategy for a one-phase reactor, if the number of reactions is small compared to the number of chemical species involved. Utilising the full row-rank of $\underset{\approx}{A}$, the contained balance equations can be recombined to partition $\underset{\approx}{A} = (\underset{\approx}{I}\ \hat{\underset{\approx}{A}})$. System (E.1) then becomes

$$\begin{pmatrix} \underset{\approx}{H}_{1,1} & \underset{\approx}{H}_{1,2} & \boxed{\underset{\approx}{I}}_{2} \\ \underset{\approx}{H}^{\mathrm{T}}_{1,2} & \boxed{\underset{\approx}{H}_{2,2}}_{3} & \hat{\underset{\approx}{A}}^{\mathrm{T}} \\ \boxed{\underset{\approx}{I}}_{1} & \hat{\underset{\approx}{A}} & \end{pmatrix} \begin{pmatrix} \Delta \underset{\sim}{x}_1 \\ \Delta \underset{\sim}{x}_2 \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g}_1 \\ -\underset{\sim}{g}_2 \\ \underset{\sim}{\delta} \end{pmatrix} \tag{E.3}$$

with

$$\underset{\approx}{H} = \begin{pmatrix} \underset{\approx}{H}_{1,1} & \underset{\approx}{H}_{1,2} \\ \underset{\approx}{H}^{\mathrm{T}}_{1,2} & \underset{\approx}{H}_{2,2} \end{pmatrix} , \Delta \underset{\sim}{x} = \begin{pmatrix} \Delta \underset{\sim}{x}_1 \\ \Delta \underset{\sim}{x}_2 \end{pmatrix} , \underset{\sim}{g} = \begin{pmatrix} \underset{\sim}{g}_1 \\ \underset{\sim}{g}_2 \end{pmatrix} \tag{E.4}$$

and can be solved by using the boxed elements as pivot blocks in the indicated sequence.

System modification

Preserving the structure to the cost of efficiency, Equation (E.1) can be modified by multiplication as follows:

$$\begin{pmatrix} \underset{\approx}{I} & \\ \underset{\approx}{A} & \underset{\approx}{I} \end{pmatrix} \begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} \\ \underset{\approx}{A} & \end{pmatrix} \begin{pmatrix} \Delta \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} \underset{\approx}{I} & \\ \underset{\approx}{A} & \underset{\approx}{I} \end{pmatrix} \begin{pmatrix} -\underset{\sim}{g} \\ \underset{\sim}{\delta} \end{pmatrix} \Leftrightarrow \begin{pmatrix} \underset{\approx}{H} & \underset{\approx}{A}^{\mathrm{T}} \\ \underset{\approx}{A}(\underset{\approx}{H} + \underset{\approx}{I}) & \underset{\approx}{A}\,\underset{\approx}{A}^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \Delta \underset{\sim}{x} \\ \underset{\sim}{\lambda} \end{pmatrix} = \begin{pmatrix} -\underset{\sim}{g} \\ \underset{\sim}{\delta} - \underset{\approx}{A}\,\underset{\sim}{g} \end{pmatrix} . \tag{E.5}$$

Since $\underset{\approx}{A}$ is of full row rank, $\underset{\approx}{A}\,\underset{\approx}{A}^{\mathrm{T}}$ is non-singular. The balance equations can be rearranged to obtain $\underset{\approx}{A}$ orthonormal, thus $\underset{\approx}{A}\,\underset{\approx}{A}^{\mathrm{T}} = \underset{\approx}{I}$. Still, a sub-system of size $\dim \underset{\approx}{H}$ must be solved, which is the drawback of this method.

E.3 Domain restrictions in a relaxation object

Given a relaxation object R_1 with its representation as a sorted sequence of values $\gamma_{1,i}$ (see Section 3.8), the object describes a currently permitted domain for a relaxation factor γ as

$$\gamma \in \mathrm{R}_1 = [\gamma_{1,0}, \gamma_{1,1}] \cup \cdots \cup [\gamma_{1,i}, \gamma_{1,i+1}] \cup \cdots \cup [\gamma_{1,N-1}, \gamma_{1,N}]\,. \tag{E.6}$$

Open intervals can be described by the formal notation $\gamma_{1,N} = \infty$. The following algorithm determines the relaxation object $\mathrm{R} = \mathrm{R}_1 \cap \mathrm{R}_2$:

```
R := R_1
**for** $k = 1 : 2 : N - 1$
    $\hat{\mathrm{R}} := \mathrm{R}$
    $(b, e) = (\gamma_{2,k}, \gamma_{2,k+1})$
    Remove all $\gamma_j$ with $b < \gamma_j < e$ from R
    **if** $b \in \hat{\mathrm{R}}$ **then** insert $b$ into R.
    **if** $e \in \hat{\mathrm{R}}$ **then** insert $e$ into R
**end for**
```

Example:

Let $\mathrm{R}_1 = [0 : 3] \cup [6 : 9]$ and $\mathrm{R}_2 = [0 : 2] \cup [5 : 8]$. The representing coefficients are $\langle \gamma_{1,i} \rangle = (3, 6, 9)$ and $\langle \gamma_{2,i} \rangle = (2, 5, 8)$.

The first pair of R_2 to consider is $(b, e) = (2, 5)$, such that $\gamma_1 = 3$ is to be removed from $\mathrm{R} := \mathrm{R}_1$, which yields $\hat{\mathrm{R}} = [0 : 6] \cup [9 : \infty]$. Since $b = 2 \in \mathrm{R}$, we modify $\hat{\mathrm{R}} = [0 : 2] \cup [6 : 9]$. As $e = 5 \notin \mathrm{R}$, no further modification is taken in this step.

The second pair of R_2 is $(b, e) = (8, \infty)$. $\gamma_3 = 9 \in [b, e]$ must be removed: $\mathrm{R} = [0 : 2] \cup [6 : \infty]$. Now, $b = 8 \in \mathrm{R}$, such that the final result is

$$\mathrm{R} = \mathrm{R}_1 \cap \mathrm{R}_2 = [0 : 2] \cup [6 : 8]\,. \tag{E.7}$$

E.4 A suggestion for an initialisation algorithm

The input to the algorithm is a set of robust (mostly linear) equations, a set of inequality constraints, and the complete set of canonical and canonical conjugated variables. Each equation is associated to a non-negative cost value, which describes its degree of reliability. Furthermore associated to the equation are all involved canonical variables. Each inequality constraint points to an equation, which – if applied – forces the constraint to be fulfilled. For example the constraint $T > 0$ points to an equation $T - 298.15\ \mathrm{K} = 0$. Most variables do not really need to be initialised. Therefore, obligatory variables are specially marked as part of the algorithm input, that is temperature T, pressure p, and molar vector $\underset{\sim}{\dot{n}}$ of each phase in the process model (see Section 3.7.2).

The algorithm framework can be described by the following work-flow:

1. Find an optimal set of robust equations (a square system) to determine all obligatory variables.

2. Detect linear dependent equations and modify equation cost attributes, such that a new solution of step 1 does not include these singularities.

3. Solve equation system.

4. Test for non-fulfilled inequality constraints. If there are any, lower the cost of the equation, this inequality points to. Goto step 1.

5. With no singularities and non-fulfilled inequality-constraints, the solution from step 3 represents a feasible set of initial values.

E.4.1 Obtaining a minimal structural invertible sub-system

The core of the algorithm is to solve assignment problems as described in Section E.4.2, which however requires a matching number of equations and variables. In our case, dummy variables can be added to the system. These variables do neither appear in any equation, nor are they required for initialisation. The cost c_{ij} of a matching between equation i and variable j is set to the cost-value of the equation, if the variable appears within the equation, an infinite value else.

Theorem E.1 *A small modification of the assignment problem algorithm finds not only one, but all optimal matchings, i.e.* $\{M_i \,|\, c(M_i) = \min\limits_{M_j} c(M_j)\}$. *As a fact, exactly all possible perfect matchings in the last iteration k of the algorithm are optimal.*

Proof: It is trivial to see that all these matchings are optimal, since they are found on the same cost-reduced matrix $\underset{\approx}{C}^{(\mathrm{k})}$. To prove that no others are optimal, consider an optimal matching M_2, for which $\gamma^{(\mathrm{k})} = \sum\limits_{e_{ij}\in M_2} c_{ij}^{(\mathrm{k})} \neq 0$, then $\gamma^{(\mathrm{k})} > 0$ because $c_{ij}^{(\mathrm{k})} \geq 0\ \forall_{i,j}$. But the distance $\gamma^{(0)} - \gamma^{(\mathrm{k})}$ is due to the reduction of complete rows and columns only depending on k, but not the matching M. Hence $c(M_2) > \min\limits_{M_i} c(M_i)$, M_2 is not optimal. □

All optimal matchings can therefore found by subsequent disallowing of all matching edges recursively until no perfect matching exists for $\underset{\approx}{C}^{(\mathrm{k})}$.

Theorem E.2 *With all optimal matchings M_i and the most reduced sub-matchings $\bar{M}_i$, such that $\bar{M}_i$ contains all necessary variables x_i, there is no non-optimal matching M_n with $c(\bar{M}_n) < c(\bar{M}_i)$, i.e. all optimal sub-matchings are among sub-sets of the optimal matchings:*

Proof: Let M_1 be an optimal matching with the sub-assignment $\bar{M}_1 \ni x_i$. M_n is a non-optimal matching, but $c(\bar{M}_n) < c(\bar{M}_1)$. Since $\bar{M}_n$ is per definition a structural solvable sub-system, the graph $G = (X \cup F, E)$ can be partitioned by defining X_n and F_n, such that $\bar{M}_n \subseteq (X_n \cup F_n, X_n \times F_n)$, as shown in Figure E.1. It is obvious that

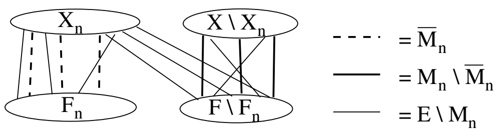


Figure E.1: *Partitioning of the bipartite graph to isolate an optimal sub-assignment.*

there exists no complete optimal matching M_i in G with $M_i \cap X_n \times (F \setminus F_n) \neq \emptyset$, since $|X_n| = |F_n|$ and $c_{ij} > c_{\text{opt}}$ for $\{(i, j)|X_i \notin X_n \wedge F_j \in F_n\}$. Hence, both partial graphs $G_n = (X_n \cup F_n, X_n \times F_n)$ and $\bar{G}_n = G \setminus G_n$ are decoupled. It is

$$G \text{ is optimal} \quad \Leftrightarrow \quad G_n \text{ is optimal} \quad \wedge \quad \bar{G}_n \text{ is optimal}\,. \tag{E.8}$$

It can be assumed w.l.o.g. that $\bar{M}_n$ is optimal, hence selection of an optimal supplementing matching generates an optimal matching $M_2 \supset \bar{M}_n$. □

The important conclusion of this is that there exists a polynomial algorithm to find the optimal set of equations to determine all required variables.

E.4.2 Assignment Problem Algorithm for Square Systems

The algorithm to solve the assignment problem includes an algorithm called "Hungarian method" to obtain a maximum matching and a minimum vertex cover for a bipartite graph:

The Hungarian method

Given a bipartite graph $G = (U \cup W, E)$ and an initial (not necessary maximum) matching M of G, the maximum matching and minimum vertex cover can be obtained as follows:

1. Marking vertices:
   (a) Every $u \in U \setminus M$ is marked by '0'
   (b) If all marks are processed, goto step 3. Else select a non-processed $v \in U \cup W$. Goto step 1c, if $v \in U$, or to step 1d, if $v \in W$.

(c) Processing a $v \in U$: Each unmarked $w \in W$ with $vw \in E \setminus M$ is marked with v. Goto step 1b.

(d) Processing a $v \in W$: If $v \notin M$, goto step 2. Else select matching partner $u \in U$ with $vu \in M$ and mark u with v. Goto step 1b.

2. There is an M-prolonging path P from a vertex $u \in U \setminus M$ to $v \in W \setminus E$ as follows: The first vertex is v, which is marked with u. P starts with $vu \in E$. If the mark of u is "0", P is complete and used to extend M, afterwards clearing all marks and going to step 1a. Else, u is marked with $w \in W$, P is extended by $uw \in E$. w is marked also and P follows the marks.

3. M is a maximum matching. The non-marked points of U and the marked points of W represent a minimum vertex cover.

The weighted matching problem

A cost matrix $\underset{\approx}{C}^{(0)}$ is provided as input to the algorithm. To find the least-cost matching in a complete bipartite graph $G = (U \cup W, E)$ with $E = U \times W$, the following steps are carried out:

1. Obtain a cost-reduced matrix from $\underset{\approx}{C}^{(0)}$ by subtracting the minimum of each column from the regarding column, and the minimum of each row from the regarding row:

$$c_{ij}^{(1/2)} = c_{ij}^{(0)} - \min_i c_{ij}^{(0)}, \quad c_{ij}^{(1)} = c_{ij}^{(1/2)} - \min_j c_{ij}^{(1/2)} . \tag{E.9}$$

   Now, $c_{ij}^{(1)} \geq 0 \ \forall_{ij}$, $\exists_i \ c_{ij}^{(1)} = 0 \ \forall_j$ and $\exists_j \ c_{ij}^{(1)} = 0 \ \forall_i$. Set the iteration counter $k = 1$.

2. Construct a bipartite graph $G(U \cup W, E)$ with $U = \langle u_i \rangle$, $W = \langle w_j \rangle$ and $u_i w_j \in E \Leftrightarrow c_{ij}^{(k)} = 0$. Use the Hungarian method to find a maximum matching and a minimum vertex cover $X \cup Y$ with $X \subseteq U$ and $Y \subseteq W$. The reduction number m is defined as

$$m = \min_{ij} \{ c_{ij}^{(k)} | u_i \in X \wedge w_j \in Y \} . \tag{E.10}$$

   The assignment problem is solved as soon as the obtained matching is complete. The matching edges represent the assignment.

3. Add $m/2$ to all rows i with $u_i \in X$, subtract $m/2$ from all other rows. Add $m/2$ to all columns j with $w_j \in Y$, subtract $m/2$ from all other columns. The resulting matrix is $\underset{\approx}{C}^{(k+1)}$. Increment k and goto step 2.

The algorithm terminates latest in n^2 iterations, but much faster for problems with many edges of same costs.

Appendix F

Notation

F.1 Landau symbols

In this work, asymptotic notation is used in two different contexts, namely to describe the precision of approximate functions and to characterise computational effort solving a particular problem. Latter one can be measured in terms of memory or runtime. If not stated otherwise, computational effort describes the runtime aspect within the scope of this work, more precise: the asymptotic number of floating point operations (flop) necessary to perform a particular task.

Literature offers various ways to define the *Landau symbols* (e.g. von Bronstein *et al.*, 1999; Gilbert and Peierls, 1988). A consistent definition, which can be used for complexity analysis, but as well for error estimations of approximate functions is the following:

Let $\psi_1(\xi)$ and $\psi_2(\xi)$ be two arbitrary positive functions of a variable ξ, and $\xi_0 \in \mathbb{R} \cup \{\pm\infty\}$ an agreed limit, typically $\xi_0 = \infty$ for complexity analysis, $\xi_0 = 0$ for error estimations. We define:

$$\begin{aligned}
\psi_2 \in O(\psi_1) &\quad\Leftrightarrow\quad \lim_{\xi\to\xi_0} \psi_2(\xi)/\psi_1(\xi) = \nu \text{ with } \nu \in \mathbb{R}^{\neq 0} \\
\psi_2 \in o(\psi_1) &\quad\Leftrightarrow\quad \lim_{\xi\to\xi_0} \psi_2(\xi)/\psi_1(\xi) = 0 \\
\psi_2 \in \bar{O}(\psi_1) &\quad\Leftrightarrow\quad \psi_1 \in O(\psi_2) \\
\psi_2 \in \bar{o}(\psi_1) &\quad\Leftrightarrow\quad \psi_1 \in o(\psi_2) \\
\psi_2 \sim \psi_1 &\quad\Leftrightarrow\quad \psi_2 \in O(\psi_1) \wedge \psi_1 \in O(\psi_2)
\end{aligned} \tag{F.1}$$

F.2 Unified modelling language

A complete description of the Unified Modelling Language (UML) can be found in OMG (2001), but only a small subset is used within the main part of this work, namely exclusively static structure diagrams. Due to different versions and dialects, some notation conventions might be deprecated in the future. Figure F.1 contains the

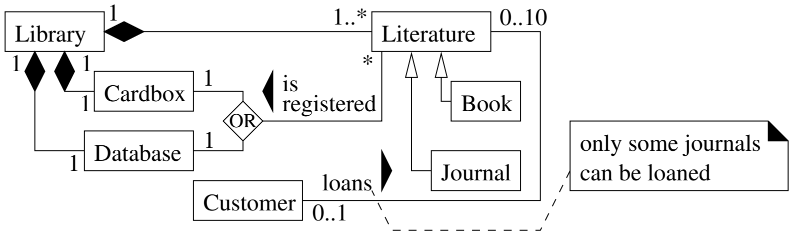


Figure F.1: *Example of a UML static structure diagram*

elements used in this work, which are:

Terms and definitions F.1

Class A box filled by a noun indicates a class of objects, e.g. *Book*. There might be many instances of type *Book*, and the common set of properties defines the class.

Inheritance The non-filled arrowheads on lines connecting *Book* and *Journal* with *Literature* indicate an inheritance relationship. The sub-classes *Book* and *Journal* inherit properties from the base-class *Literature*, which might be an abstract object, i.e. all instances of this class are represented by a sub-class. In most cases, the arrow can be read as an "*is a*"-relationship: "A book is a [piece of] literature".

Association Any solid line, which does not represent an inheritance, expresses an association, which is further specified by cardinality.

Cardinality The cardinality is indicated by numbers or ranges of numbers, including the symbol * for an arbitrary number. *Customer* can loan between zero and ten instances of *Literature*. Vice versa, an instance of *Literature* can be loaned out only to zero or one customer at a time. The *Library* contains at least one instance of *Literature*.

Description For clarification, a solid triangle can give a short description of a particular relationship, usually expressed by a verb. This directed indicator can be read as a sentence, e.g. "A customer loans literature".

OR-Block The block indicating that an instance of *Literature* is either registered in *Cardbox* or *Database* is a simplification to avoid the necessity to display numerous classes. Alternatively, both *Cardbox* and *Database* could inherit from a new class called *Register*, to which they would be associated instead.

Accumulation A black rhombus on the end of an association line indicates an ownership relation. The class next to the rhombus is the owner of the counterpart class. *Library* is the owner of both, *Literature*, *Cardbox* and *Database*.

Annotation If there is relevant information, which can not be expressed by any other notation, a dashed line can connect any symbol mentioned above to a text-box with a dog-ear on the upper right corner.

However, in the appendix, a broader subset of UML is utilised for documentation (cf. Appendix A). On these occasions, it is referred to literature for further documentation of UML.

F.3 Graph theory

Graph theory is a suitable discipline to describe the field of discrete mathematics. This work utilises graph theory to describe process topology as well as mappings between equations and variables for initialisation purposes (cf. Appendix E.4).

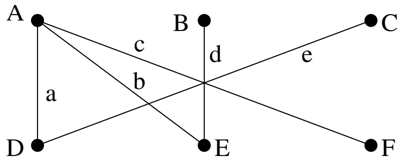


Figure F.2: An example graph

The following terms related to graph theory are used within this work. Figure F.2 shows a simple example of a graph to illustrate these definitions. Trudeau (1993) gives an introduction to graph theory.

Terms and definitions F.2

Graph A set of vertices $\{A, B, \ldots, F\}$ and edges $\{a, b, \ldots, e\}$. One edge connects two vertices. Vertices can be endpoints of zero, one, or many edges.

Path An alternating sequence of vertices and edges, starting and ending with a vertex. A path contains each edge not more than once. In Figure F.2, (F, c, A, a, D, e, C) is an example for a path.

Circle A path, in which the starting vertex is identical to the ending vertex.

Bipartite graph A graph, in which the set of all vertices can be partitioned into two subsets, such that no edge connects two vertices of the same subset. In Figure F.2, the vertices can be partitioned into a bipartite graph as follows: $\{A, B, C\} \cap \{D, E, F\}$.

Matching A subset of edges in a graph, which do not share any vertices, as $\{c, e\}$ in the example.

Maximum matching A matching, such that any further inclusion of an arbitrary edge does not yield a new matching. In the example, $\{b, e\}$ is a maximum matching.

Perfect matching A (maximum) matching, which covers all vertices of a graph, as $\{c, d, e\}$ in the example.

Bibliography

Abbott, K. A., Allan, B. A., Westerberg, A. W., 1997. Global preordering for Newton equations using model hierarchy. AIChE J 43 (12), 3193–3204.

Austern, M. H., 1998. Generic Programming and the STL. Addison Wesley.

Bauer, C., Frink, A., Kreckel, R., 2002. Introduction to the GiNaC framework for symbolic computation within the C++ programming language. J. Symb. Comp. 33, 1–12.

Berger, F., Perris, F. A., 1979. Flowpack II – A new generation of system for steady state process flowsheeting. Comput. Chem. Eng. 3, 309–317.

Biegler, L. T., Grossmann, I. E., Westerberg, A. W., 1997. Systematic Methods of Chemical Process Design. International Series in the Physical and Chemical Engineering Science. Prentice Hall PTR, London, ISBN 0-13-492422-3.

Bogle, I. D. L., Perkins, J. D., 1988. Sparse Newton-like methods in equation oriented flowsheeting. Comput. Chem. Eng. 12 (8), 791–805.

Bogusch, R., Lohmann, B., Marquardt, W., 2001. Computer-aided process modelling with M K . Comput. Chem. Eng. 25, 963–995.

Boston, J. F., Britt, H. I., 1978. A radically different formulation and solution of the single-stage flash problem. Comput. Chem. Eng. 2, 109–122.

Braunschweig, B. L., Pantelides, C. C., Britt, H. I., Sama, S., 1999. Open software architectures for process modeling: Current status and future perspectives. In: Computer-Aided Design for the 21st Century. FOCAPD '99.

Braunschweig, B. L., Pantelides, C. C., Britt, H. I., Sama, S., 2000. Process modelling: The promise of open software architectures. Chemical Engineering Progress 96 (9), 65–76.

Brendsdal, E., 1999. Computation of phase equilibria in fluid mixtures. Ph.D. thesis, NTNU Trondheim, ISBN 82-471-0371-0.

Callen, H. B., 1985. Thermodynamics and an Introduction to Thermostatistics, 2nd Edition. John Wiley & Sons, New York, ISBN 0-471-86256-8.

Campbell, J. M., 1984. Gas Conditioning and Processing. Vol. 2. Campbell Petroleum Series.

Chen, H., Stadtherr, M. A., 1981. A modification of Powell's dogleg method for solving systems of nonlinear equations. Comput. Chem. Eng. 5 (3), 143–150.

Čížek, J., Vinette, F., Weniger, E. J., 1993. On the use of the symbolic language Maple in physics and chemistry - Several examples. Int. J. Mod. Phys. C – Phys. Comput. 4, 257–270.

Cofer, H. N., Stadtherr, M. A., 1996. Reliability of iterative linear equation solvers in chemical process simulation. Comput. Chem. Eng. 20 (9), 1123–1132.

Dluzniewski, J. H., Adler, S. B., 1972. Calculation of complex reaction and/or phase equilibria problems. I. Chem. E. Symp. Ser. 35, 4:21–4:26.

Edgar, T. F., Himmelblau, D. M., 2001. Optimization of Chemical Processes. McGraw-Hill.

Evans, L. B., Boston, J. F., Britt, H. I., Gallier, P. W., Gupta, P. K., Joseph, B., Mahalic, V., Ng, E., Seider, W. D., Yagi, H., 1979. Aspen: An advanced system for process engineering. Comput. Chem. Eng. 3, 319–327.

Fateman, R. J., 1992. A review of Mathematica. J. Symb. Comput. 13, 545–579.

George, A., Ng, E., 1985. An implementation of Gaussian elimination with partial pivoting for sparse systems. SIAM J. Sci. Stat. Comp. 6 (2), 390–409.

Gilbert, J. R., Peierls, T., 1988. Sparse partial pivoting in time proportional to arithmetic operations. SIAM J. Sci. Stat. Comp. 9 (5), 862–874.

Golub, G. H., Loan, C. F. V., 1996. Matrix Computations, 3rd Edition. The Johns Hopkins University Press, Baltimore, ISBN 0-8018-5414-8.

Haug-Warberg, T., 1988. Computation of thermodynamic equilibria. Ph.D. thesis, NTNU Trondheim.

Hernandez, R., Sargent, R. W. H., 1979. A new algorithm for process flowsheeting. Comput. Chem. Eng. 3, 363–371.

Hinderink, A. P., Kerkhof, F. P. J. M., Lie, A. B. K., De Svaan Arons, J., van der Kooi, H. J., 1996. Exergy analysis with a flowsheeting simulator – I. Theory; Calculating exergies opf material streams. Chem. Eng. Sci. 51 (20), 4693–4700.

Hudak, P., 1989. Conception, evolution, and application of functional programming languages. ACM Comput. Surv. 21 (3), 359–411.

Jungnickel, D., 1999. Optimierungsmethoden – Eine Einführung. Springer, ISBN 3-540-66057-7.

Li, X., Shao, Z., Qian, J., 2004. Module-oriented automatic differentiation in chemical process systems optimization. Comput. Chem. Eng. 28, 1551–1561.

Mahoney, D., Santollani, O., 1994. HYSYS – An integrated System for Process Engineering and Control. Hyprotech, Ltd., Calgary, Alberta.

Markowitz, H. M., 1957. The elimination form of the inverse and its application to linear programming. Manage. Sci. 3 (3), 255–269.

Marquardt, W., 1996. Trends in computer-aided process modeling. Comput. Chem. Eng. 20 (6/7), 591–609.

Mattsson, S. E., Elmqvist, H., Otter, M., 1998. Physical system modeling with modelica. Control Eng. Pract. 6, 501–510.

Michelsen, M., Mollerup, J., 2004. Thermodynamic models: Fundamentals & Computational Aspects. Tie-Line Publications.

Michelsen, M. L., 1982. The isothermal flash problem. Part II. Phase-split calculation. Fluid Phase Equilibr. 9, 21–40.

Michelsen, M. L., 1987. Multiphase isenthalpic and isentropic flash algorithms. Fluid Phase Equilibr. 33, 13–27.

Michelsen, M. L., 1994. Calculation of multiphase equilibrium. Comput. Chem. Eng. 18 (7), 545–550.

Michelsen, M. L., 1999. State function based flash specifications. Fluid Phase Equilibr. 160 (158), 617–626.

Mischler, C., Joulia, X., Hassold, E., Galligo, A., Esposito, R., 1995. Automatic differentiation applications to computer aided process engineering. Comput. Chem. Eng. 19, 779–784.

MS Visio, 2003. Microsoft visio 2003. http://www.microsoft.com/office/visio/.

Nocedal, J., Wright, S. J., 1999. Numerical Optimisation. Springer.

Oh, M., Pantelides, C. C., 1996. A modelling and simulation language for combined lumped and distributed parameter systems. Comput. Chem. Eng. 20 (6/7), 611–633.

OMG, 2001. OMG Unified Modeling Language Specification Version 1.4. Object Management Group (OMG).
URL `http://www.omg.org`

OMG, 2003. OMG Unified Modelling Language Specification, Version 1.5. Object Management Group Inc., http://www.omg.org.

Özyurt, D. B., Pike, R. W., 2004. Theory and practice of simultaneous data reconciliation and gross error detection for chemical processes. Comput. Chem. Eng. 28, 381–402.

Pantelides, C. C., Barton, P. I., 1992. Equation-oriented dynamic simulation – Current status and future perspectives – 2. In: Depeyre, D. (Ed.), European Symposium on Computer Aided Process Engineering. Vol. 17 of Computers and Chemical Engineering. European Federation of Chemical Engineers, Pergamon Press, pp. 263–285.

Péneloux, A., Rauzy, E., Fréze, R., 1982. A consistent correction for Redlich-Kwong-Soave volumes. Fluid Phase Equilibr. 8, 7–23.

Perkins, J. D., 1979. Efficient solution of design problems using a sequential-modular flowsheeting programme. Comput. Chem. Eng. 3, 375–381.

Perry, R. H., Green, D. W. (Eds.), 1997. Perry's Chemical Engineers' Handbook, 7th Edition. McGraw-Hill.

Python, 2005. Python programming language. http://www.python.org.

Schildt, H., 1999. STL Programming from the Ground Up. McGraw-Hill.

Schwartzentruber, J., Renon, H., 1989. Extension of UNIFAC to high pressures and temperatures by the use of a cubic equation of state. Ind. Eng. Chem. Res. 28 (7), 1049–1055.

Schwartzentruber, J., Renon, H., Watanasiri, S., 1990. K-values for non-ideal systems: An easier way. Chem. Eng. , 118–124.

Shewchuk, C. F., 1987. Massbal MKII – New process simulation system. Pulp Pap.-Canada 88 (5), 76–82.

Siepmann, V., Haug-Warberg, T., Mathisen, K. W., 2001. Analysis and consistency of process models with application to ammonia production. In: Gani, R., Jørgensen, S. B. (Eds.), European Symposium on Computer Aided Process Engineering. Vol. 11 of Computer Aided Chemical Engineering. Computer Aided Process Engineering Center (CAPEC), Elsevier, pp. 297–302.

Smith, J. M., van Ness, H. C., Abbot, M. M., 2001. Introduction to Chemical Engineering Thermodynamics, 6th Edition. McGraw-Hill.

Soave, G., 1972. Equilibrium constants from a modified Redlich-Kwong equation of state. Chem. Eng. Sci. 27, 1197–1203.

Sorin, M., Bonhivers, J.-C., Paris, J., 1998a. Exergy efficiency and conversion of chemical reactions. Energy Convers. Mgmt. 39 (16–18), 1863–1868.

Sorin, M., Hammache, A., Diallo, O., 2000. Exergy based approach for process synthesis. Energy (Oxford) 25, 105–129.

Sorin, M., Lambert, J., Paris, J., 1998b. Exergy flows analysis in chemical reactors. Chem. Eng. Res. Des. 76 (A3), 389–395.

Stadtherr, M. A., Wood, E. S., 1984. Sparse matrix methods for equation-based chemical process flowsheeting – I. Comput. Chem. Eng. 8 (1), 9–18.

Stephanopoulos, G., Henning, G., Leone, H., 1990. MODEL.LA.A modeling language for process engineering – I. The formal framework. Comput. Chem. Eng. 14 (8), 813–846.

Stroustrup, B., 1997. The C++ Programming Language, 3rd Edition. Addison Wesley.

Swig, 2005. Simplified wrapper and interface generator. http://www.swig.org.

Tester, J. W., Modell, M., 1997. Thermodynamics and Its Applications, 3rd Edition. Prentice Hall International Series in the Physical and Chemical Engineering Sciences. Prentice Hall PTR, ISBN 0-13-915356.

Tolsma, J. E., Barton, P. I., 1998. On computational differentiation. Comput. Chem. Eng. 22 (4/5), 475–490.

Tolsma, J. E., Clabaugh, J. A., Barton, P. I., 2002. Symbolic incorporation of external procedures into process modeling environments. Ind. Eng. Chem. Res. 41, 3867–3876.

Trudeau, R. J., 1993. Introduction to Graph Theory. Dover Publications, Inc.

UNIDO, IFDC (Eds.), 1988. Fertilizer Manual. Kluwer Academic Publishers, Dordrecht, The Netherlands, ISBN 0-7923-5032-4.

von Bronstein, I. N., Semendjajew, K. A., Musiol, G., Mühlig, H., 1999. Taschenbuch der Mathematik, 4th Edition. Verlag Harry Deutsch AG, Frankfurt am Main, Germany.

Wakeham, W. A., Stateva, R. P., 2004. Numerical solution of the isothermal isobaric phase equilibrium problem. Rev. Chem. Eng. 20 (1/2).

Wall, G., 1986. Exergy flows in industrial processes. Tech. Rep. 83-11, Physical Resource Theory Group, Chalmers Univ. of Technology and Univ. of Göteborg.

Westerweele, M. R., Preisig, H. A., Weiss, M., 1999. Concept and design of Modeller, a computer-aided modelling tool. Comput. Chem. Eng. Supp. , S751–S754.

White, W. B., Johnson, S. M., Dantzig, G. B., 1958. Chemical equilibrium in complex mixtures. J. Chem. Phys. 28 (5), 751–755.

Wilhelm, C. E., Swaney, R. E., 1994. Robust solution methods of algebraic process modelling equations. Comput. Chem. Eng. 18 (6), 511–531.

XML-RPC, 2005. Xml-rpc homepage. http://www.xmlrpc.com.

Zitney, S. E., Stadtherr, M. A., 1988. Computational experiments in equation-based chemical process flowsheeting. Comput. Chem. Eng. 12 (12), 1171–1186.

Zitney, S. E., Stadtherr, M. A., 1993. Supercomupting strategies for the design and analysis of complex separation systems. Ind. Eng. Chem. Res. 32 (4), 604–612.

Zlatev, Z., 1980. On some pivotal strategies in Gaussian elimination by sparse technique. SIAM J. Numer. Anal. 17 (1), 18–30.

Zope, 2005. Z object publishing environment. http://www.zope.org.