Device-Oriented Modeling and Simulation in Aircraft Energy Systems Design

Vom Promotionsausschuss der
Technischen Universität Hamburg-Harburg
zur Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

genehmigte Dissertation

von

Michael Sielemann

aus Detmold

2012

1. Gutachter	Prof. Dr.-Ing. Gerhard Schmitz
2. Gutachter	Prof. Dr.-Ing. Martin Otter
Prüfungsvorsitzender	Prof. Dr.-Ing. Günter Ackermann
Weitere Gutachter	Prof. Dr.-Ing. Alfons Kather
	Prof. Dr.-Ing. Frank Thielecke

Tag der Abgabe	22. November 2011
Tag der mündlichen Prüfung	14. Mai 2012

To My Dear Melissa
and our daughter Maria Elisa.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my doctoral thesis supervisor Prof. Dr.-Ing. Gerhard Schmitz, under whose constant supervision the work presented in this thesis was carried out. The scientific freedom, which you granted under concise general conditions, allowed me to pursue these extraordinary opportunities. I thank you for your guidance and support, and for the enthusiasm towards your students.

I am glad to be indebted to my adviser Prof. Dr.-Ing. Martin Otter for many revealing discussions and excellent suggestions. Without your day-to-day assistance and constantly open door, this thesis would not have been possible. Thank you for this support and for taking me to the Modelica Design Meetings when we initially identified robustness issues with thermo-fluid models.

I thank Prof. Dr.-Ing. Günter Ackermann for chairing the examining board and Prof. Dr.-Ing. Alfons Kather as well as Prof. Dr.-Ing. Frank Thielecke for their scientific reviews.

I would also like to extend my appreciation to Dr.-Ing. Johann Bals, the head of System Dynamics and Control. Thank you for giving me exciting opportunities and the freedom to write this thesis.

This thesis would not have been possible without the DENECS project ("Design Environment for New ECS") supported by Airbus Operations GmbH, which provided me with an outstanding industrial context. I wish to thank Dr.-Ing. Tim Giese and Bettina Öhler for their tremendous technical expertise, patience with new developments, and an open and very kind collaboration in general. Thank you for providing me with a rationale and motivation for the sometimes overly theoretic endeavors.

During the last years, I was fortunate to meet many great people. In particular, the following persons had a relevant impact on this thesis: Francesco Casella, Rüdiger Franke, Kathrin Prölß, Hubertus Tummescheit, Hilding Elmqvist, and Dimitri Mavris.

I also want to mention at this point that the material presented in chapter 5 partially resulted from of a collaboration with various persons. Therefore, section 5.3 provides a detailed list of who contributed what.

I would like to thank my colleagues Jan Kladetzke, Thiemo Kier, Roland Ziegler, Dirk Zimmer, and Andreas Pfeiffer at System Dynamics and Control, and my friends Sebastian Breyer, Janis Mühlratzer, Claus Christmann, Eike Pauli, Bastiaan Klik, and Marcelo Gerin.

Moreover, I want to thank my family for their unconditional company and support. I thank my sister Claudia for the good times we have together. I thank my parents in law for accepting me wholeheartedly as part of their family. I thank my parents for being here for me, and for opening up perspectives and opportunities to me. I hope I can become as great of a parent to my children as you have been to me.

Finally, I thank my daughter Maria Elisa for the love and easiness she gives me. And I thank my wife Melissa for her expertise in academic matters, the confidence, understanding, and patience she has with me. Meeting you was the best thing that happened in my life.

Munich, May 2012 *Michael Sielemann*

TABLE OF CONTENTS

SYMBOLS

a	Left-most coordinate of one-dimensional computational mesh, random parameter.
A	Cross section area, area, Jacobian matrix of conservation laws.
b	Right-most coordinate of one-dimensional computational mesh.
$\mathfrak{B}$	Unit ball.
c	Speed of sound, signal speed, specific heat capacity (at constant pressure or volume depending on subscript).
d	Optimal weighting parameter, distance, search direction.
f	Generic map, flux.
F	Generic DAE for time simulation, fixed point problem, force.
g	Numerical flux approximation.
G	Additional equation system for initialization, scaling or leakage matrix.
h	Specific enthalpy, driving-point characteristic.
I	Cell of computational mesh, current.
J	Staggered cell of computational mesh, Jacobian matrix.
k	Order of accuracy of a discretization or interpolation method, resistive coefficient.
K	Right eigenvector, matrix of right eigenvectors.
l	Lower bound.
L	Linear operator.
m	Momentum.
$\dot{m}$	Mass flow rate.
n	Dimension, surface normal.
N	Nonlinear operator.
p	Pressure (static or total depending on subscript), non-primitive interpolation polynomial, generic problem.
P	Primitive interpolation polynomial, probability, power.
$\mathcal{P}$	Set of problems.
q	Generic quantity, quaternion.
$\dot{q}$	Volumetric heat transfer rate.
r	Ratio of cell side slopes, left-shift of computational stencil, radius.
R	Resistance.
s	Source term, right-shift of computational stencil, solution, arc length.

S	Surface of a control boundary, computational stencil.
$\dot{S}$	Rate of entropy change.
$\mathcal{S}$	Set of solvers.
$\mathfrak{S}$	Surface of unit ball.
t	Time.
T	Temperature (static or total depending on subscript).
u	Specific internal energy (static or total depending on subscript), conserved variable, upper bound.
v	Velocity, non-primitive function, transformed conserved variables.
V	Interpolating primitive function, voltage.
$\dot{V}$	Volumetric flow rate.
w	Algebraic unknown.
$\dot{w}$	Volumetric work rate or power.
x	Generic unknown, cartesian coordinate, water content.
y	Generic dependant variable, cartesian coordinate.
z	Cartesian coordinate, state variable.
α	Wave strength, essentially non-oscillatory coefficient, auxiliary variable, current gain.
β	Limited slopes parameter, smooth indicator.
Γ	Generic diffusion coefficient, zero curve.
Δ	Slope or difference of conserved quantity.
ε	Small constant.
ζ	Generic loss factor for dissipation of momentum.
η	Dynamic viscosity.
κ	Ratio of specific heats.
λ	Interpolation factor, eigenvalue, homotopy parameter, thermal conductivity.
Λ	Diagonal matrix of eigenvalues.
ξ	Slope limiter, auxiliary averaging variable.
ρ	Density, homotopy map.
τ	Maximum performance metric factor, mechanical torque.
φ	Generic conserved quantity, flux limiter function, generic flow variable.
χ	Indicator function, thermal conductivity times area per thickness.
ω	Weighting parameter, angular velocity.
Ω	Volume inside a control boundary.

SUB- AND SUPERSCRIPTS

0	Total or stagnation quantity, initial time, start of continuation.
-1	Inverse.
a	At a constant parameter vector a.
aj	Analytic Jacobian.
$\langle a \rangle$	Connector a-type polarity.
b	Bounding.
$\langle b \rangle$	Connector b-type polarity.
$cmp2$	Compressor outlet.
db	Drain–to–bulk.
$demanded$	Demanded value.
dis	Discharge.
ds	Drain–to–source.
e	External.
F	Forward.
$Force$	First-Order Centered monotone flux.
fr	Wall friction.
gb	Gate–to–bulk.
gs	Gate–to–source.
HI	Higher-order.
$HLLE$	Harten-Lax-van-Leer monotone flux.
hyd	Hydraulic component.
l	Left.
L	Leftmost.
LF	Lax-Friedrichs monotone flux.
LO	Low-order.
lr	Linearization using Roe's average values.
max	Maximum.
mb	Minbee limiter.
min	Minimum.
mix	Mixing quantity.
mu	First-order monotone upstream flux.

n	Normal component, n-dimensional space.
N	Newton.
p	At constant pressure.
pT	Pressure and temperature.
r	Right, Total Variation Diminishing region.
R	Rightmost, reverse.
Ri	Richtmyer flux.
Roe	Roe's monotone flux.
Rus	Rusanov monotone flux.
s	Solution.
sb	Source–to–bulk, Superbee limiter.
SW	Steger-Warming monotone flux.
t	Derivative with respect to time.
T	Transpose.
th	Thermal component.
TVD	Total Variation Diminishing.
$TVDc$	Total Variation Diminishing centered flux.
$TVDu$	Total Variation Diminishing upstream flux.
up	Upstream quantity.
ut	Before symbolic reduction.
vl	Van Leer limiter.
w	Algebraic variables.
WE	Water extractor.
x	Derivative with respect to space.
Xi	Independent mass fractions.
z	State variables.
$+$	Limited to positive value, approximation in right limit.
$-$	Limited to negative value, approximation in left limit.
$*$	Root or solution, embedding in device model.

SUMMARY

In the past, aircraft energy systems had only marginal relevance during the early design of commercial aircraft. The design problem was decomposed hierarchically based on a conventional architecture using pneumatic, hydraulic, and electric secondary power. Requirements were allocated and the actual design of individual aircraft energy systems was established in isolation downstream in the design process. Today, this approach is in question, as airframers realize that global optimization of the entire aircraft is the only remaining way to achieve substantial improvements in the total aircraft package.

The ongoing integration of aircraft energy systems architecture design leads toward the reduced use of hydraulic and pneumatic secondary power, and a “More Electric Aircraft”. At the same time, business relationships change from “purchaser and supplier”-cooperations to risk-sharing partnerships. These changes require methods and tools to support designing globally optimized architectures. As a contribution toward this overall goal, the objective of this thesis is to develop and implement a design methodology for the air conditioning system (Environmental Control System), which imposes key requirements on the aircraft energy systems architecture.

The proposed concept is based on a physics-based design methodology. Herein, energy-optimal open-loop control has to be defined. Otherwise, a rigorous assessment using aircraft-level metrics cannot be established. For this purpose, inverse-modeling approaches are introduced. These allow establishing energy-optimal open-loop control concurrently with optimal aircraft energy system architecture.

As a physics-based design methodology requires a substantial modeling effort, flexibility and reusability of mathematical plant models are of pivotal importance. For this purpose, the equation-based object-oriented modeling language Modelica is adopted. In this regard, substantial contributions are made to further improve the robustness of the code generated from such modeling languages. In particular, robustness issues in established non-causal thermo-fluid interfaces (“connectors”) are identified. Based on a rigorous analysis, a robust yet user-friendly interface called “stream connectors” is proposed. Additionally, robustness issues with steady-state initialization are addressed. In order to analyze this problem, a quantitative metric is proposed. Then, a probability-one homotopy method is introduced to equation-based object-oriented modeling languages. Using theorems from topology, the established method guarantees convergence with probability one. This property is demonstrated on a number of case studies.

Exploiting these improvements, a robust and flexible modeling and simulation framework for Environmental Control Systems and general cooling systems is implemented. It covers both conventional low-speed fluid dynamics as well as high-speed gas dynamics. Building on this modeling and simulation framework and the proposed design methodology, a physics-based design environment for Environmental Control Systems and aircraft energy systems in general is implemented. It can be applied to conventional and unconventional system architectures alike.

CHAPTER 1

INTRODUCTION

Progress in engineering and technology is sometimes made in small, and sometimes in large steps. The development of object-oriented modeling languages was such a large step for system-level simulation. This thesis explores the use of such concepts for the design of aircraft systems architecture, a domain in which a similarly revolutionary change is currently taking place. As the activities described in this thesis test and push the limits of the underlying modeling and simulation technology, the subject area is in the intersection of aerospace systems design, fluid dynamics, and applied mathematics.

This chapter provides a high-level overview of the industrial challenges motivating this thesis and the problems it addresses. The state of the art is introduced and overarching research objectives are derived. Detailed literature reviews of the sub-topics are presented in the subsequent chapters.

1.1 Motivation

Despite recent political and economical crises and other enduring threats, the long term outlook for world air travel is positive. According to the Airbus Global Market Forecast [2], the economic recovery, the return of business confidence and corporate investment, the sustained trade in commodities and a demand in worldwide leisure travel, have all resulted in a strong rebound of air traffic.

- Passenger traffic in terms of revenue passenger kilometers is forecast to grow by an average of 4.8% annually over the 2010-2029 period according to Airbus [2] and by 5.3% according to Boeing [19].

- Cargo traffic will grow even faster: Airbus [2] and Boeing [19] both forecast an average growth rate in terms of freight tonne kilometers of 5.9% over 2010-2029.

These growth rates are occurring in a competitive and globalized environment. The two key competitors are under constant pressure to continuously improve the commercial aircraft. The demanded improvements translate to superior capabilities at reduced emissions and cost. According to Jackson [88], the aircraft can be divided into the airframe, propulsion, and other segments. The other segments are referred to by the term aircraft systems (or aircraft energy systems, aircraft equipment systems). While substantial improvements were achieved over the last decades using traditional aircraft design, which puts the focus on airframe and propulsion [172], the aircraft systems become increasingly relevant today [144]. This is due to two reasons. First, their contributions to emissions, weight, and cost are substantial. On long range aircraft for example, the installation and operation of the aircraft systems account for more than 5% of the fuel burn [102]. Additionally, they contribute arround 30% to both aircraft empty weight and direct operating costs [38].

The second key reason for increased relevance of aircraft systems in early design is due to a pending revolution [144]. Since the conception of modern, gas-turbine driven aircrafts, nearly all airplanes, both commercial and military, have used three conventional types of secondary power: hydraulics, pneumatics and electricity. Advances in aircraft systems occurred mostly locally and brought the infusion of new technologies into existing systems, but no modifications to the overall architecture. Following the principles of Systems Engineering [88], requirements were formulated and addressed downstream in the design process. For this purpose, the so-called Air Transport Association chapters [1] (ATA chapters) were used on civil aircraft. Based on them, the architecture of the aircraft systems was broken down and requirements were allocated. Design of the aircraft systems took place downstream in the design process.

Today, improvements are becoming increasingly difficult to achieve by local changes to existing designs, "as both fundamental physical barriers and conflicts between requirements emerge" [144]. For this reason, engineers cannot continue to optimize each component and subsystem individually. Instead, global optimization of the whole aircraft is considered as "the only way to produce meaningful improvements in the total aircraft package" [144].

This global optimization leads toward the reduced use of hydraulic and pneumatic secondary power, and a "More-Electric Aircraft" (MEA). Such concepts were initially proposed in the 1970s [37]. The reasons to promote the increased use of electric power include intrinsic losses in pneumatic Engine Bleed Air Systems (both intrinsic to the turbofan thermodynamic cycle at increased bypass ratios and related to losses in the pre-cooler and pressure regulating valves), increased reliability of electric systems as well as the all but exhausted potential of power electronics technology (in contrast to pneumatics and hydraulics). For examples of such MEA architectures, see Moir and Seabridge [125].

The ongoing integration of aircraft systems design does not only have impact on technical factors, but also directly affects the way business is conducted. In particular, the former purchaser and supplier relationship is increasingly replaced by a risk-sharing partnership on several commercial aircraft development programs. At the same time, the conventional nomenclature in terms of ATA chapters becomes less adequate. The reason is that ATA chapters are based on a conventional aircraft systems architecture. In unconventional architectures, system boundaries, interfaces and so on change substantially. In this thesis, sub-problems in aircraft systems design are thus classified according to the physical domain (thermal, thermo-fluid, electric, hydraulic etc.).

The given need for globally optimized aircraft systems architectures and the implications of more highly integrated architectures motivate the development of the methods and tools in this thesis. An emphasis is put on aircraft system design problems involving thermo-fluid dynamics (i.e., in the thermo-fluid domain). This includes the air conditioning system called Environmental Control System (ECS) and electrified pneumatic systems in general.

1.2 Thesis Overview

The contributions of this thesis span three different levels. First and foremost, design and optimization methods for thermo-fluid aircraft systems are considered. This is the first (highest) level of the problem decomposition. These methods in turn rely on mathematical models of the systems. Their development is addressed on the second level. Finally, such mathematical models have to be formulated in a suitable programming or modeling language. These aspects are considered on the third (lowest) level.

The structure is illustrated in figure 1. It shows the different elements on the three levels. The definition of requirements and architecture is done informally in the present chapter with an introductory review of the state of the art, the identification of knowledge gaps and the formulation of objectives. The implementation, test, and integration is described in the following chapters of this thesis. In particular, the bottom aspect of "Equation-based object-oriented modeling and simulation technology" is addressed in chapters 2 to 10. The middle topic concerning "Thermo-fluid dynamics modeling and simulation of aircraft systems" is studied in chapters 11 and 12. The top subject "Physics-based aircraft system design" is addressed in chapter 13.

Note that these chapters do not include a documentation of all possible details but emphasize the academically relevant topics.

1.3 Design of aircraft systems

In order to formulate requirements and objectives toward aircraft systems architecture design, the state of the art is reviewed.

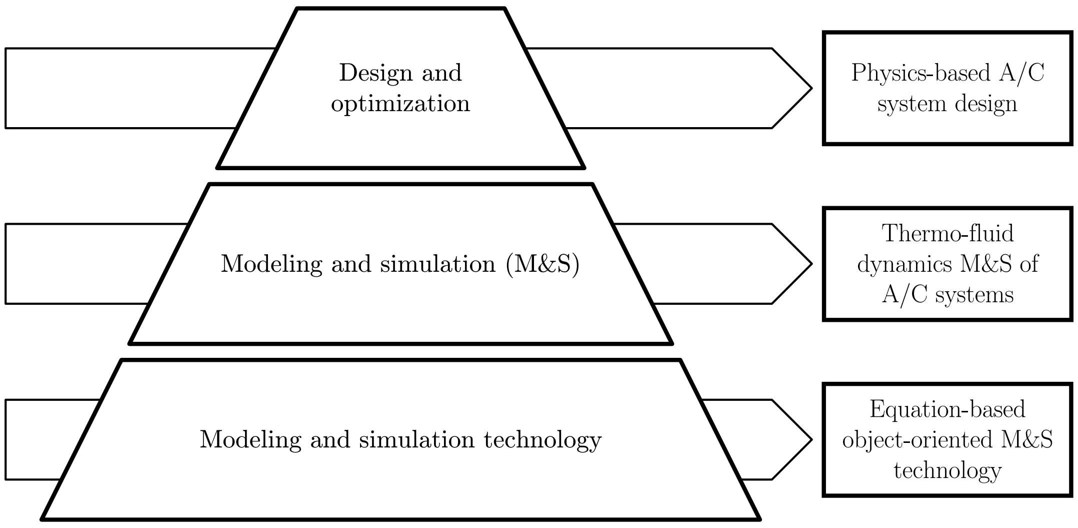

Design and optimization
Modeling and simulation (M&S)
Modeling and simulation technology
Physics-based A/C system design
Thermo-fluid dynamics M&S of A/C systems
Equation-based object-oriented M&S technology

Figure 1: Thesis overview via a methods and tools pyramid

1.3.1 Aircraft systems in conventional aircraft conceptual design

Conventional methods to cover aircraft systems in early aircraft design are based on statistical regression of system weight. Such techniques have been used for several decades in conceptual design and are well suited for handling the details of these aircraft systems downstream in the design process. The purely statistical relations represent certain technological levels up to a specific state of the art. Usage of such methods to estimate the performance of next-generation systems or of aircraft using different architectural or technological features depends, if at all possible, on modifying regression factors which do not have a physical meaning. Such methods have been proposed by several authors including Torenbeek [172], Roskam [150] and Beltramo et al. [16].

Several authors proposed ways to improve the quality of regression-based predictions for different technological levels or physical architectures. In [7], for instance, the RDT&E period was introduced as an explanatory variable into the regression to account for overall technological trends. In the Fast and Advanced Mass Estimation method [107], a comparatively large number of explanatory variables was used to establish regressions. The Subsystems Integrated Design Assessment Technology [136] proposed the regression of sensitivities of key aircraft system physical parameters against aircraft level performance parameters. Koeppen [99] proposed the combination of simplified physical models with regression of physical parameters as predicted by the simplified models against their actual values.

These methods are useful for a comparative assessment of different technologies and partially aircraft system architectures. Kirby [97] for instance demonstrates how to use such methods for technology identification, evaluation, and selection. After the reduction of a technological portfolio to a small number of candidate solutions, more stringent methods are of interest. The difficulty in the assessment of such portfolios using regression-based methods lies in the lack of rigorous methods to adapt simplified models and non-physical regression factors.

1.3.2 Conceptual design of aircraft system architecture

The industrial interest in unconventional aircraft systems architecture such as the MEA has triggered the development of suitable design methodologies and supporting tools, both in industry itself and academia. Even though several relevant contributions have been made along this direction, integrated aircraft system architecture design methods that span multiple physical domains are only at the beginning of development.

Until today, in the conceptual design of aircraft system architecture, the problem is frequently decomposed using systems engineering into single physical domain aircraft system design. So-called trade factors are used to link one aircraft system design problem to the aircraft level. The trade-factors represent sensitivities (i.e., gradients) of aircraft level metrics such as take-off gross weight and fuel consumption against aircraft system physical parameters such as weight, secondary power consumption and so on. While trade-factors allow to take aircraft level metrics into consideration in the design of individual aircraft systems, they neglect interaction. First, interaction takes place between the aircraft and the aircraft system, because a change in aircraft system physical parameters strictly requires an additional design iteration on aircraft level. Frequently, this does not take place. Second, interaction takes place via changes in derived requirements of one system toward the other. This is neglected, because every system is often sized individually.

Eventually, such methods in conceptual aircraft design will be superseded by integrated aircraft systems architecture design methods. Recently, first steps toward such methods and software platforms were proposed. Bals et al. [13, 14] presented an integrated aircraft system architecture simulation framework and applied MEA concepts. The simulation framework used an inverse modeling technique to establish derived requirements on power distribution and generation systems. It is also applied to optimize power consumption [156]. Liscouët-Hanke later suggested a similar simulation platform [106, 105]. The latter approach was limited in applicability in that it required structural assumptions on aircraft system-specific elements to hold (a forward and a reverse loop, see section 13.1 for a relevant example not meeting these assumptions). Mavris et al. [114, 39, 8] proposed a methodology for aircraft system architecture definition based on functional decomposition and applied concepts from optimization.

Several authors contributed toward the goal of integrated aircraft systems architecture design and optimization via building blocks of a future integrated design platform. Schallert [157, 158] for instance suggested a design methodology for electrical power systems, which unifies the aspects of performance, weight, and reliability. Scholz [159] presented a method for conceptual design of flight control and hydraulic systems. Dollmayer and Carl [50, 49] proposed a method to quantify the impact of aircraft system installation and operation on the propulsion system. Kaslusky et al. [94] suggested processes to integrate the design of aircraft system architecture into industrial context.

The ECS was subject of a number of manuscripts. In particular, Vargas and Bejan [186] proposed a notional design methodology for ECS taking into account the assessment on aircraft level. The problem was simplified in several regards however. For instance, the authors considered a single point along the mission only, studied a simplified system (a single heat exchanger, no water separation, no ram air fan or ejector), addressed nominal on-design performance only and optimized only a single component. Tipton et al. [171, 61] use a similar approach on a complex architecture of ECS and thermal systems and similarly low fidelity. Ordonez and Bejan [134] established ECS requirements based on a simple cabin model, which contained a lumped heat generation and heat transfer element. The derived requirements are imposed on a mostly idealized pack model (e.g., reversible compression and expansion, no pressure drops on the heat exchanger). While their work is very interesting from a theoretical perspective, the presented trade-off between pack outlet temperature and pack mass flow rate is of limited practical value as it does not consider all relevant constraints. In industrial practice, the pack mass flow rate for instance is directly driven by the fresh air requirements due to the high cost of fresh air in terms of specific fuel consumption. These requirements are imposed by the certification authorities in [92, paragraph 25.831a] and [59, paragraph 25.831a].

1.3.3 Objectives and contributions

The objective of this thesis with respect to the conceptual design of aircraft systems architecture was to contribute an aircraft system design environment, which is suitable for use in integrated aircraft systems architecture design. This aircraft system design environment used a generic, physics-based approach[1] and was implemented for the ECS. This thesis contributes to the state of the art in the following areas.

[1]Statistical regression was still useful in the environment (e.g., heat transfer correlations), but the plant model was implemented using rigorous balance equations.

- A suitable design methodology for *general* aircraft systems was defined. Herein, the industrial perspective in commercial aircraft was taken into consideration. In contrast to the contributions in literature, the methodology does not neglect or idealize the open-loop control parameters, which is required for a rigorous assessment using aircraft-level metrics. For this purpose, inverse-modeling approaches were introduced and compared. These allow establishing energy-optimal open-loop control concurrently with optimal aircraft energy system architecture.

- In contrast to the mentioned earlier publications, off-design performance was considered. This was a relevant improvement in aircraft systems architecture optimization, as design is all about making compromises and industrial applications require to routinely consider a large number of different scenarios, in which certain performance constraints have to be met.

- Causality of the physical plant model was not fixed to a predefined state. During design, it was possible to establish model causality starting with the top-level aircraft requirements and the functions to fulfill. Casting them into requirements allowed the formulation of derived requirements for all other parts of the system. The conventional causality in turn enabled performance computations to assess a given design.

These contributions were described in chapter 13.

1.4 Modeling and simulation of thermo-fluid dynamics

A physics-based design methodology required physics-based plant models. As the focus of this manuscript was on ECS and thermo-fluid aircraft systems, a particular set of physical phenomena were of interest. Within the present work, the term thermo-fluid dynamics was used to refer to an informal union of fluid dynamics, thermodynamics as well as heat and mass transfer, which was relevant for these applications.

1.4.1 State of the art

System-level simulation [32] in the domain of thermo-fluid dynamics is a wide topic yet relatively mature. For instance, see Ding [47] for a review of system-level simulation for vapor compression cycle applications. For reasons, which are outlined in the following section, the focus was put on a particular modeling and simulation technology termed equation-based, object-oriented modeling languages.

Several authors present applications using such languages in various technical domains. For instance, Casella [29, 30] considers power plant simulation, Pfafferott [141], Tummescheit et al. [180], Richter [147], and Prölß [143] study applications in sub-critical vapor compression cycles, Casas [26, 27] addresses air conditioning using desiccant assisted cycles, and Vasel and Schmitz [187] consider air conditioning using trans-critical cycles.

In all of the given applications, the governing equations are adapted to the specifics of the underlying flow phenomena. According to the best knowledge of the author, the assumptions are identical for all applications involving equation-based, object-oriented modeling languages reported in literature. The corresponding flow, which allows to make these assumptions, is called a *low-speed compressible flow* in this thesis. All previously referenced authors assume that the flow is incompressible with respect to the flow phenomena, as it is low-speed. Density variation is only encountered due to heat transfer and in lumped parameter components such as compressors. Density variation due to flow phenomena is neglected, i.e., the Mach number is typically below 0.3.

In particular, an analysis of model code revealed that the difference between static and total pressure is neglected as the dynamic pressure is considered small and not of interest. For the given applications in power plants or vapor compression cycle refrigeration systems this is reasonable. Only in special devices, which involve large variations in flow cross-section such as adapters between different pipe diameters or nozzles, total pressure is of interest. Total or stagnation enthalpy is often treated similarly; the kinetic term $v^2/2$ is neglected. A typical argument is that the order of magnitude of change in specific enthalpy due to heat transfer is larger than that of such kinetic terms.

If kinetic terms in pressure and specific enthalpy are not neglected for such applications and the common assumption of a steady-state momentum balance is made then coupled nonlinear algebraic equation systems arise for density, which is required to establish flow velocity. These coupled equation systems deteriorate simulation performance.

A relevant part of the applications of interest in this thesis involve a different type of flow, which is called *high-speed compressible flow* herein. This applies to the ECS and electrified pneumatic systems in particular. Kinetic terms and dynamic pressure may not be neglected and have to be included in compressible formulations. Density variation is also encountered with respect to flow phenomena, in particular dynamic conservation of momentum is relevant and also shock waves may be part of the solution. The Mach number may be > 0.3 (including the supersonic regime). The term "gas dynamics" refers to the same type of flow.

The key theoretical area to enable applications involving high-speed compressible flow is the discretization method for the governing equations. The foundations of numerical solution methods in thermo-fluid dynamics are well understood. However, in the framework of equation-based, object-oriented modeling languages, only methods suitable for low-speed compressible flow have been applied. The classic finite volume method has been studied in particular by Tummescheit [181]. Moving boundary methods have been applied by Jensen [90, 89] and Tummescheit [181]. Casella [30] proposed a mean density discretization, which is non-conservative but results in continuous and continuously differentiable thermodynamic properties at phase boundaries of two-phase flow. Prölß and Schmitz [142] discretized the governing equations for frost formation on heat exchanger surfaces.

1.4.2 Objectives and contributions

System-level simulation of thermo-fluid dynamics is rather mature. However, even though discretization methods suitable for high-speed compressible flow are available in literature (e.g., [174, 162]), they have not been applied in the framework of equation-based, object-oriented modeling languages. The objective of this thesis with respect to modeling and simulation was to contribute toward the application of such methods in the given modeling languages to enable rigorous modeling and simulation of aircraft systems involving high-speed compressible flow in the scope of integrated aircraft systems architecture design. The thesis contributes to the state of the art in the following areas.

- Relevant concepts of the theory in numerical solution methods for high-speed compressible flow were reviewed and translated from the algorithmic perspective taken in literature to the acausal concepts of equation-based, object-oriented modeling languages.
- The elements of discretization schemes were decomposed in an object-oriented fashion and implemented in a generic library.
- Object-oriented concepts were exploited for increased flexibility such as parametric polymorphism for exchangeable thermodynamic property models.

These contributions are described in chapter 12.

Additionally, the thesis presents how to address the requirements of the design method for physics-based plant models at the example of the ECS. The aspects mentioned in section 1.3.3 such as variable causality and off-design performance simulation as well as a general requirement for flexibility were addressed. These contributions are dealt with in chapter 11.

1.5 Modeling and simulation technology

Åström et al. [9] reviewed the evolution of continuous-time modeling and simulation. Among others, they mention the graphical block diagram modeling paradigm and the physical modeling or equation-based, object-oriented approach.

1.5.1 Graphical block diagram modeling

In order to use digital computers for system-level modeling and simulation, several researchers adopted the analog simulation principles on digital computers. As Åström et al. [9] wrote, "it seemed easier to change the technology than to change the paradigm".

Graphical block diagram modeling is based on the notion of graphical blocks, which have fixed inputs and outputs. Arithmetic operations, integrators and transfer functions are typical examples. Blocks may be composed hierarchically.

Typical implementation examples are EASY5 [129], originally developed by the Boeing Company around 1976, and Simulink [72], developed by MathWorks around 1991.

Following the graphical block diagram modeling paradigm, models need to be expressed in explicit ordinary differential equations (state form). As Åström et al. [9] report, "a severe consequence is that it is cumbersome to build physics-based model libraries in the block diagram languages". In particular, "a general solution to this problem required a paradigm shift".

1.5.2 Equation-based object-oriented modeling languages

Several limitations of the graphical block diagram modeling paradigm can be lifted by allowing the modeler to state the underlying physical balance equations of mass, energy, and momentum in their natural form, i.e., differential algebraic equations (DAEs)[2]. Several advantages immediately emerge from this approach. As the problem is largely posed in terms of equations, symbolic processing of the resulting DAE is enabled. Like this, symbolic and numerical solution techniques can be combined to allow for efficient simulation on real-world problems. At the same time, this type of problem definition is declarative (in contrast to algorithmic). This implies that a user has only to define *what* the problem is, not *how* to solve it. Finally, the problem definition becomes non-causal, and therefore a single model can be used in place of a set of models with permuted inputs and outputs, which is required in the graphical block diagram modeling paradigm. Additionally, equation-based, object-oriented modeling languages combine this equation-based approach with the concept of object-orientation. This largely increases the flexibility and power of the resulting languages. For instance, parametric polymorphism allows to separate and exchange distinct model elements such as the thermodynamic property model and the device model.

Åström et al. [9] also discuss the history of such equation-based, object-oriented modeling languages. One of the most advanced generic (i.e., non domain-specific) equation-based, object-oriented modeling languages is Modelica [113], even though there are others (VHDL-AMS [85], gPROMS™ [133] etc.). Modelica emerged from a unification effort "bringing together expertise in object-oriented physical modeling" [9]. The open language standard is developed by the non-profit Modelica Association. Relevant predecessors were for instance Smile [98] and Omola [112].

This language is particularly successful both in academia and industry. The Modelica web page [121] currently lists eight commercial Modelica simulation environments and four free Modelica simulation environments, each building on the same standardized language definition.

1.5.3 Objectives and contributions

As the aircraft systems architecture becomes more and more tightly integrated, non domain-specific modeling and simulation technology was applied in this thesis. For the advantages they provide, an equation-based, object-oriented modeling language was used in this context. Like this, the requirements to allow performance calculation and requirement-driven computation were fulfilled elegantly, i.e., with the same models. Furthermore, the object-oriented principles mimicked the way physical systems are built and allowed for increased flexibility and reusability of the models.

[2]Obviously, the natural form of most balance equations is in terms of partial differential algebraic equations. If the partial derivatives with respect to space dimensions are discretized however, then differential algebraic equation systems involving time derivatives only suffice.

System-level simulation using equation-based, object-oriented modeling languages is rather mature. However, during research into physics-based aircraft systems design methods certain reoccurring issues were identified in two areas. The first one involved thermo-fluid dynamics plant models in general and the second one steady-state initialization. With respect to the former, the issues were traced to the thermo-fluid interfaces ("connectors"). Therefore, this thesis contributes to the state of the art in the following areas.

- Established thermo-fluid interface definitions were reviewed and discussed in a consistent manner (chapter 2).
- Requirements toward such interfaces were rigorously defined (chapter 3).
- A consistent line of argument was established to trace the reported difficulties to deficiencies of the thermo-fluid interfaces with respect to the given requirements. For this purpose, a set of tools and test cases was established (chapter 4).
- Improvements and alternatives for thermo-fluid interfaces were proposed (chapter 5).
- Evidence was given to highlight how the improved thermo-fluid interface meets the given requirements (chapter 6).

With respect to the reported difficulties with steady-state initialization, the thesis contributes to the state of the art in the following regard.

- The stated observation of robustness issues was analyzed rigorously. For this purpose, a suitable quantitative metric was introduced and a testing environment was developed (chapter 7).
- A well-known class of alternative solution methods called homotopy methods was introduced in a comprehensible, informal fashion (chapter 8).
- Homotopy methods using generic maps were analyzed. Substantial reasons were given why it is unlikely that they resolve the observed problem (chapter 9).
- A proposal for homotopy methods using problem-specific maps in the context of equation-based, object-oriented modeling languages was formulated. The proposal was based on the theory of probability-one homotopy methods. Using theorems from topology, such methods guarantee convergence with probability one. Additionally, the proposal was illustrated on several case studies. Problems, which were time-consuming to solve before, were solved robustly without manual interaction. Improvements were quantified in terms of a rigorous metric (chapter 10).

In summary, this thesis contributes in several distinct aspects toward the overall goal of integrated, physics-based design of aircraft systems architecture.

CHAPTER 2

ESTABLISHED NON-CAUSAL INTERFACE DEFINITIONS FOR THERMO-FLUID DYNAMICS

The objective in this chapter is to review the results of several years of research about the optimal design of non-causal thermo-fluid interfaces in equation-based, object-oriented modeling languages. All relevant design concepts that have been used for simulation of thermo-fluid dynamics are reviewed together with the corresponding connector design and notional implementations of three exemplary standard components. For the reasons given in section 1.5, emphasis is put on the equation-based, object-oriented modeling language Modelica.

The fundamental issue in the design of such thermo-fluid interfaces is how to treat the quantities that are transported via convection, i.e., the quantities that are transported by the fluid flow *in the flow direction*. Examples of such quantities are specific enthalpy and substance mass fractions. All convected quantities can be handled similarly and thus, for the sake of readability, only specific enthalpy is addressed explicitly. The extension to additional convected quantities is trivial.

Historically, the development of non-causal thermo-fluid interfaces focused on low-speed compressible flow as described in section 1.4.1. Therefore such applications are considered in the chapters 2 to 6 of this thesis. High-speed compressible flow is addressed in chapter 12.

Before introducing established interface definitions, the governing equations and discretization schemes in primitive variables are introduced. The presentation is short; for a more elaborate general treatment refer to Ferziger and Perić [60] or Patankar [139], for a presentation specific to equation-based, object-oriented modeling languages refer to Tummescheit [181].

2.1 Governing equations of thermo-fluid dynamics

In fluid dynamics, the governing equations of mass, energy, and momentum conservation may be formulated under several different assumptions. The Navier-Stokes and Euler equations each involve different assumptions. By convention, the Navier-Stokes equations cover a viscous flow with the dissipative transport phenomena of friction, thermal conductance and mass diffusion. The Euler equations model an inviscid flow, which neglects the effects of said transport phenomena.

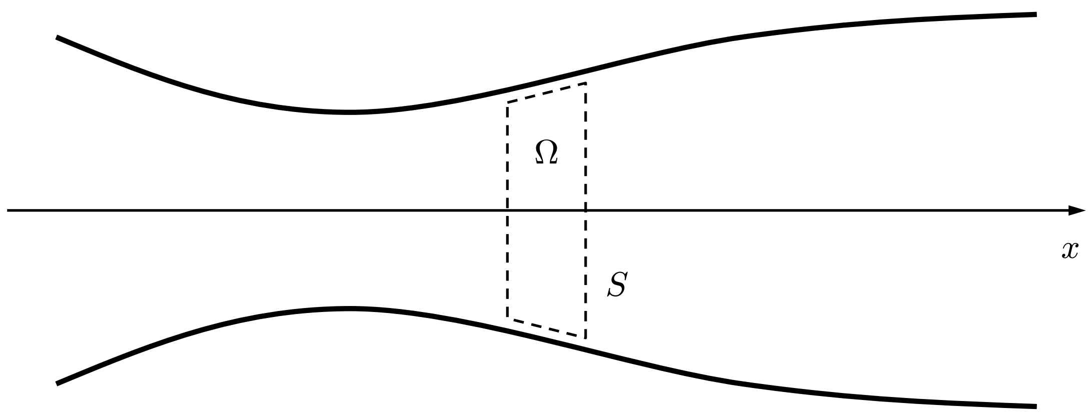


Figure 2: One-dimensional problem domain with coordinate x, volume Ω, and surface S

In one-dimensional system-level simulation, "real" viscous effects in terms of spatial derivatives (shear and normal stresses on an infinitesimal control volume such as the ones that dominate a boundary layer flow) cannot be resolved as no relevant space dimension is available (to resolve, e.g., a boundary layer). Instead, generic friction forces based on empirical correlations in terms of a ζ loss factor, Fanning friction factor or the like are usually introduced. Moreover, the thermal conduction and mass diffusion are neglected in several cases. This is also done in this thesis. Consequently, unsteady quasi one-dimensional Euler equations suitable for system-level simulation are considered. The equations are hyperbolic independently of the Mach number. Next, the governing equations for a one-dimensional domain as depicted in figure 2 are introduced. In differential form, they are

Continuity:

$$\frac{\partial}{\partial t}(\rho A) + \frac{\partial}{\partial x}(\rho v A) = 0 \tag{1}$$

Conservation of momentum:

$$\frac{\partial}{\partial t}(\rho v A) + \frac{\partial}{\partial x}\left(\rho v^2 A\right) = -A\frac{\partial p}{\partial x} - \Delta p_{fr} \cdot A \tag{2}$$

Conservation of energy:

$$\frac{\partial}{\partial t}(\rho u_0 A) + \frac{\partial}{\partial x}\left(\rho v \left(u_0 + {}^{p}\!/_{\rho}\right) A\right) = \dot{q}_e \cdot A + \dot{w}_e \cdot A \tag{3}$$

These equations are derived from first principles, see, e.g., Anderson [6]. Equations (1) to (3) additionally consider a pressure difference due to viscous friction Δp_{fr} and a volumetric heat transfer rate $\dot{q}_e$. Variable $\dot{w}_e$ denotes a volumetric work rate or power. In several cases, it is not included in the equations below. This means that the work term was neglected.

The integral forms of the equations are as follows (here, the subscript $()_x$ denotes a reference to the component in x-direction, Ω is the volume inside a control boundary, and S is its surface, see figure 2).

Continuity:

$$\frac{\partial}{\partial t}\int_\Omega \rho \, d\Omega + \int_S \rho \vec{v} \cdot \vec{n} \, dS = 0 \tag{4}$$

Conservation of momentum:

$$\frac{\partial}{\partial t}\int_\Omega \rho v_x \, d\Omega + \int_S \rho v_x \vec{v} \cdot \vec{n} \, dS = -\int_S (p \cdot \vec{n} \, dS)_x - \Delta p_{fr} \cdot A \tag{5}$$

Conservation of energy

$$\frac{\partial}{\partial t}\int_\Omega \rho u_0 \, d\Omega + \int_S \rho \left(u_0 + {}^{p}\!/_{\rho}\right) \vec{v} \cdot \vec{n} \, dS = \int_\Omega \rho \dot{q}_e \, d\Omega + \int_\Omega \rho \dot{w}_e \, d\Omega \tag{6}$$

Generic Conservation Equation: To discuss the different methods to discretize the governing equations, a generic conservation equation of a quantity φ is introduced. In the conservation equation, the first term stands for storage, the second term for convective flux, the third one for dissipative flux, and the last one for a generic volumetric source.

Differential form:

$$\frac{\partial}{\partial t}(\rho\varphi) + \operatorname{div}(\rho\varphi\vec{v}) = \operatorname{div}(\Gamma \operatorname{grad}\varphi) + q_\phi \tag{7}$$

Integral form:

$$\frac{\partial}{\partial t}\int_\Omega \rho\varphi \, d\Omega + \int_S \rho\varphi\vec{v} \cdot \vec{n} \, dS = \int_S \Gamma \operatorname{grad}\varphi \cdot \vec{n} \, dS + \int_\Omega q_\phi \, d\Omega \tag{8}$$

2.2 *Discretization methods in primitive variables*

2.2.1 The Finite Volume Method

A classic approach in Computational Fluid Dynamics is to discretize the governing equations using a Finite Volume scheme. One advantage is that the conservation laws are automatically fulfilled *exactly* independently of the grid resolution.

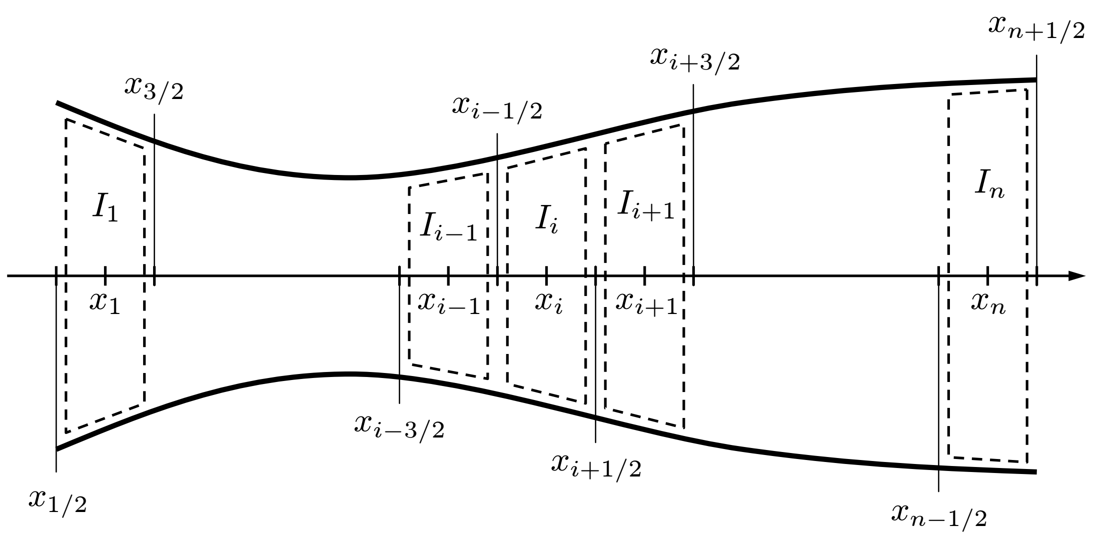


Figure 3: Computational mesh of a one-dimensional problem domain with cells I_i, cell centers x_i, and cell sides $x_{i+1/2}$

In the Finite Volume Method, the problem domain is discretized on a suitable computational mesh, see figure 3. The control volumes are defined based on a grid of cell side coordinates on an interval $[a, b]$

$$a = x_{1/2} < x_{3/2} < \ldots < x_{n-1/2} < x_{n+1/2} = b \tag{9}$$

Based on it, cells, cell centers and cell sizes are defined for $i = 1, 2, \ldots, n$.

$$\begin{aligned} I_i &= \left[x_{i-1/2}, x_{i+1/2}\right] \\ x_i &= \tfrac{1}{2}\left(x_{i-1/2} + x_{i+1/2}\right) \\ \Delta x_i &= x_{i+1/2} - x_{i-1/2} \end{aligned} \tag{10}$$

In this notation, $x_{i+1/2}$ is the coordinate of the right side of a computational cell I_i with cell center x_i. This grid is colocated. Furthermore, the maximum cell size is defined as follows.

$$\Delta x = \max_{1 \leqslant i \leqslant n} \left(\Delta x_i\right) \tag{11}$$

The discretization scheme allows to deduce algebraic equations or differential algebraic equations that properly approximate the governing equations. Note that, in the context of equation-based, object-oriented modeling languages, the goal is to deduce differential algebraic equations and thus the equations have only to be discretized in space, not in time ("semi-discretized").

The set of cell centers, which is used in a discretization scheme to deduce such equations for each cell, is called the stencil. For the most simple schemes, the stencil for cell I_i includes I_i itself and the cells to the left and to the right,

$$S(i) = \{I_{i-1}, I_i, I_{i+1}\} \tag{12}$$

2.2.1.1 Approximation of Surface Integrals

The surface integrals of governing equations such as equation (8) in integral form express the influence of a *flux* f that crosses the control volume boundary. The surface integral is split up into the contributions of each face according to the chosen grid with the number of faces per control volume n.

$$\int_S f\, dS = \sum_n \int_{S_n} f\, dS \tag{13}$$

A simple approximation is the product of the flux at the face center and the face surface area (midpoint rule). Applying this rule to face $i + 1/2$ yields the following.

$$\int_{S_{i+1/2}} f\,dS = \overline{f_{i+1/2}}S_{i+1/2} \approx f_{i+1/2}S_{i+1/2} \tag{14}$$

An approximation of a surface integral involves two approximations. First, the integral is approximated in terms of variable values on the cell face. Then, the required values on the cell faces are approximated in terms of variables at the control volume center, i.e., on the nodes.

2.2.1.2 Approximation of Volume Integrals

The remaining terms in the generic conservation equation involve volume integrals (storage and source term). Consider the simple yet second-order approximation analogous to the above. In place of the volume integral (the product of the average value with the volume), the product of the value at the control volume center with volume of the control volume is used.

$$\int_{\Omega_i} q\,d\Omega = \overline{q_i}\Omega_i \approx q_i\Omega_i \tag{15}$$

2.2.1.3 Interpolation Schemes

Herein, only two basic schemes for discretization in primitive variables are introduced. One of them is the second-order central difference scheme (CDS). Just like all other schemes of order higher than one, it is not universally bounded and can lead to oscillations. The other scheme is the upstream discretization (UDS), which avoids this. However, it is only first-order accurate and introduces artificial diffusion.

The CDS uses linear interpolation between the neighboring nodes. The interpolation factor is $\lambda_{i+1/2} = \frac{x_{i+1/2}-x_i}{x_{i+1}-x_i}$ in the general case and $\lambda_{i+1/2} = 1/2$ for equidistant meshes.
Central difference scheme:

$$\varphi_{i+1/2} = \varphi_{i+1}\lambda_{i+1/2} + \varphi_i\left(1-\lambda_{i+1/2}\right) \tag{16}$$

The CDS can also be used for the evaluation of diffusive fluxes. It assumes a linear profile and is second-order accurate if the face is located midway between the nodes.
Central difference scheme for gradients:

$$\left(\frac{\partial \varphi}{\partial x}\right)_{i+1/2} \approx \frac{\varphi_{i+1}-\varphi_i}{x_{i+1}-x_i} \tag{17}$$

The UDS approximates the interpolated value by the upstream value and thus resembles the nature of convection. As the scheme is bounded it does not lead to oscillations.
Upstream discretization scheme:

$$\varphi_{i+1/2} = \begin{cases} \varphi_i \text{ if } (\vec{v}\cdot\vec{n})_{i+1/2} > 0 \\ \varphi_{i+1} \text{ if } (\vec{v}\cdot\vec{n})_{i+1/2} < 0 \end{cases} \tag{18}$$

2.2.2 Application to the governing equations

Continuity Equation: Consider the continuity equation in integral form, equation (4). It is discretized using the second-order approximation to the volume integral with centroid values and the second-order approximation to the fluxes using the face values.

$$\frac{\partial}{\partial t}\left(\rho_i\Omega_i\right) + (\rho v A)_{i-1/2} - (\rho v A)_{i+1/2} \tag{19}$$

Energy Balance: The energy equation in the integral form is used as introduced in equation (6). The stagnation internal energy is labeled u_0 and the stagnation specific enthalpy h_0. The Finite Volume Method approximations presented above are applied. This yields the following equation.

$$\frac{\partial}{\partial t}\left(\rho_i u_{0,i}\Omega_i\right) + \left(\rho v h_0 A\right)_{i-1/2} - \left(\rho v h_0 A\right)_{i+1/2} = \rho_i \dot{q}_{e,i}\Omega_i + \rho_i \dot{w}_{e,i}\Omega_i \tag{20}$$

The interpolation equations discussed above allow to establish the centroid velocity v_i and face velocity $v_{i+1/2}$ using the central difference scheme.

Momentum Balance:

The momentum balance in integral form is discretized as posed in equation (5).

$$\frac{\partial}{\partial t}\left(\rho_i v_i \Omega_i\right) + \left(\rho v^2 A\right)_{i-1/2} - \left(\rho v^2 A\right)_{i+1/2} = -\left(pA\right)_{i-1/2} + \left(pA\right)_{i+1/2} - \Delta p_{fr} A \tag{21}$$

The pressures are assigned to the cell centers, and consequently interpolation has to be used to establish pressures $p_{i-1/2}$ and $p_{i+1/2}$. The CDS is used according to equation (16). For the moment, an evenly spaced grid is assumed (i.e., $\lambda_{i-1/2} = \lambda_{i+1/2} = 1/2$).

$$p_{i-1/2} = \frac{p_{i-1} + p_i}{2}, \qquad p_{i+1/2} = \frac{p_i + p_{i+1}}{2} \tag{22}$$

If additionally constant cross-section area is assumed then the discretization of the pressure term in equation (5) is as follows.

$$-\left(pA\right)_{i-1/2} + \left(pA\right)_{i+1/2} = -A\left(\frac{p_{i-1} + p_i}{2} - \frac{p_i + p_{i+1}}{2}\right) = -A\left(\frac{p_{i-1} - p_{i+1}}{2}\right) \tag{23}$$

As pressures p_i cancel the grid resolution effectively is halved. This is a well-known problem in Computational Fluid Dynamics known as odd-even decoupling or checker-board effect [139, 60]. Practically, this pathologic property results in oscillations in pressure or divergence in the solution algorithm.

2.2.3 The staggered grid

A classic remedy against odd-even decoupling on discretization in primitive variables is the staggered grid [78]. Some of the quantities of interest are allocated to a grid that is shifted by half the length of a cell.

The mass and energy balances are formulated on the control volumes of the original grid introduced in section 2.2.1. At the center of each such control volume, the thermodynamic state variables, cross-section area, and momentum flux are stored. The momentum balances are applied to staggered control volumes J_i.

$$J_i = [x_i, x_{i+1}] \tag{24}$$

At the center $i + 1/2$ of each of them, a mass flow rate and enthalpy flow rate are stored. As a consequence, the faces $i - 1/2$, $i + 1/2$ of the original grid coincide with the centers of the control volume on the staggered grid and vice versa.

For the momentum balance in integral form (5), $\Omega_{i+1/2}$ consequently spans the volume J_i around the centroid $i + 1/2$ between the faces i and $i + 1$. Using the finite volume approximations to surface and volume integrals presented above, the following equation is obtained.

$$\frac{\partial}{\partial t}\left(\rho_{i+1/2} v_{i+1/2} \Omega_{i+1/2}\right) + \left(\rho v^2 A\right)_i - \left(\rho v^2 A\right)_{i+1} = -\left(pA\right)_i + \left(pA\right)_{i+1} - \Delta p_{fr} A \tag{25}$$

The goal of the staggered grid approach was to have the pressures p_i, p_{i+1} available at the required locations without the need of an interpolation scheme.

2.3 Common aspects

Before introducing established thermo-fluid interfaces, common aspects of non-causal thermo-fluid interfaces are described. They result from the use of the concept of effort or potential and flow variables in equation-based, object-oriented modeling languages. These imply either equality-type equations at connection points or "sum to zero"-type equations. The semantics of thermo-fluid interfaces therefore classify the static[1] thermo-dynamic balance equations into either of these types.

The mass balance is considered a "sum to zero"-type equation. This does not involve any approximation or idealization. Pressure in turn is usually treated as effort variable and thus results in equality-type equations. The result is a momentum balance. Again, storage of momentum is neglected (static balance). Furthermore, assuming constant velocity (no variation of density or cross-section area in an infinitesimal control volume) and considering conservation of mass immediately leads to constant momentum flux. Therefore, only pressure equality remains. Assuming that this one-dimensional treatment is exact (the momentum balance is a vector equation), this momentum balance is also exact if total pressure is conserved. The resulting connection semantics include an idealization, as multiple-way connections usually result in dissipative losses, but this is justified if no explicit junction model is used. All of the established interfaces can be used this way, but previous authors always used static instead of total pressure. In this case, the pressure equality is only correct in case of equal cross-section. This is motivated by applications involving low-speed compressible flow (see section 1.4.1).

The energy balance is formulated as "sum to zero"-type equation for the enthalpy flow rates. As with the momentum balance, this equation is correct if total or stagnation enthalpy is used (idealization of no heat transfer). Again, previous authors focused on low-speed compressible flow and used static specific enthalpy. Therefore, the semantics are again only correct for one-to-one connections with equal cross-sections.

2.4 Stencil based on staggered grids

In the context of equation-based, object-oriented modeling languages, Casella et al. [29, 30] proposed a stencil thermo-fluid interface. As described in section 2.2.1, a stencil is the set of the cells on a discretization mesh required to establish a specific discretization scheme. Casella's interface was adapted to a staggered grid such that the stencil accounts only for convectively transported quantities but not pressure or mass flow rate. For the interface, two complementary connector definitions are required.

```
connector FluidPort_a
  Pressure p;
  flow MassFlowRate m_flow;
  output SpecificEnthalpy h_a;
  input SpecificEnthalpy h_b;
end FluidPort_a;
```

Listing 1: Fluid connector type A using Stencil interface

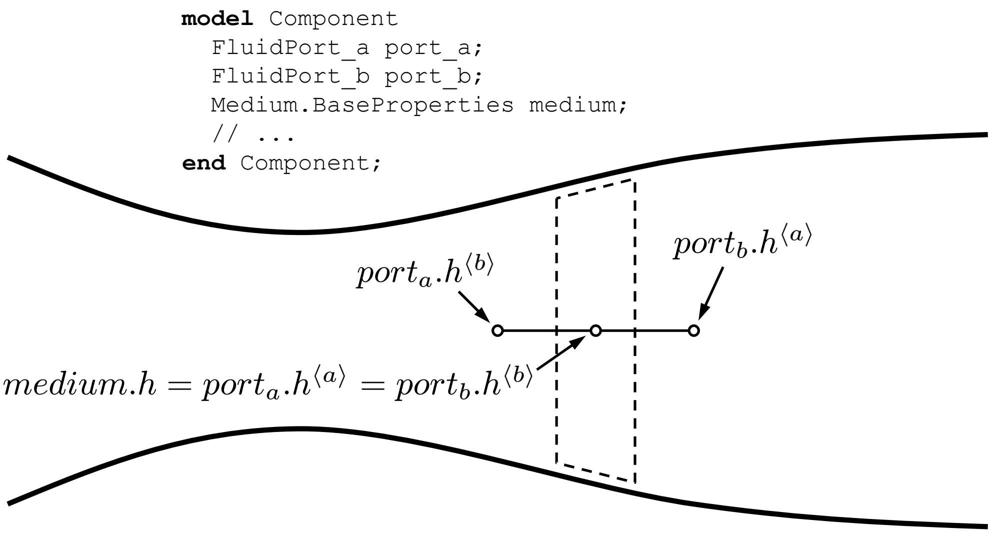
```
connector FluidPort_b
  Pressure p;
  flow MassFlowRate m_flow;
  input SpecificEnthalpy h_a;
  output SpecificEnthalpy h_b;
end FluidPort_b
```

Listing 2: Fluid connector type B using Stencil interface

[1]Connection sets use static thermo-dynamic balance equations as they are supposed to mimic an infinitesimally small control volume without storage.

To avoid numerical problems with the convected quantities, the interface provides state variables or state variables that were adequately transformed for both potential upstream thermodynamic properties. It has been first implemented in the context of the ThermoPower library by the same author.

Figure 4 illustrates the concept of this interface with a three-point stencil. A dynamic control volume model has one `FluidPort_a` instance `port_a` on the left, and one `FluidPort_b` instance `port_b` on the right. To provide the own state variables to the other components, both `FluidPort_a.h_a`= $port_a.h^{\langle a\rangle}$ and `FluidPort_b.h_b`= $port_b.h^{\langle b\rangle}$ are set to equal the state variable `medium.h`. The two variables `FluidPort_a.h_b`= $port_a.h^{\langle b\rangle}$ and `FluidPort_b.h_a`= $port_b.h^{\langle a\rangle}$ are equally defined by neighboring components and complete the three-point stencil.



```
model Component
  FluidPort_a port_a;
  FluidPort_b port_b;
  Medium.BaseProperties medium;
  // ...
end Component;
```

$port_b.h^{\langle a\rangle}$
$port_a.h^{\langle b\rangle}$
$medium.h = port_a.h^{\langle a\rangle} = port_b.h^{\langle b\rangle}$

Figure 4: Three-point stencil interface for a dynamic control volume

In this concept, flow reversal is handled via a stencil with two specific enthalpies: `h_a` is the enthalpy of the fluid if it was flowing from a `FluidPort_a` to a `FluidPort_b`, and `h_b` is the enthalpy of the fluid if it was flowing from a `FluidPort_b` to a `FluidPort_a`. The "complete stencil" as commonly understood in Computational Fluid Dynamics is thus spread over two connectors of, e.g., a flow model (a connector contains the "own" value of the convected quantity and one "other" value). Casella et al. [29, 30] in turn describe the stencil interface pragmatically. Each component has to provide one equation for one of the two specific enthalpies on each connector. This is the "specific enthalpy if fluid was going out" and is usually prescribed as function of "specific enthalpy if fluid was going in" on other connectors or state variables. The other specific enthalpy on the connector has to be prescribed by the other component in the connection set of the port.

To see how this works practically, consider the following code of a control volume model describing the mass and energy storage of fluid in a vessel.

```
model CV
  FluidPort_a port_a;
  FluidPort_b port_b;
  replaceable package Medium = PartialPureSubstance;
  parameter Volume V;
  Medium.BaseProperties medium;
  Mass m;
  Energy U;
  Power H_flow_a, H_flow_b;
equation
  port_a.p = medium.p;
  port_a.h_a = medium.h;
  port_b.p = medium.p;
  port_b.h_b = medium.h;
  m = V*medium.d;
  U = m*medium.u;
  der(m) = port_a.m_flow + port_b.m_flow;
  H_flow_a = port_a.m_flow * (if port_a.m_flow > 0 then
                  port_a.h_b else port_a.h_a);
  H_flow_b = port_b.m_flow * (if port_b.m_flow > 0 then
                 port_b.h_a else port_b.h_b);
  der(U) = H_flow_a + H_flow_b;
end CV;
```

Listing 3: Dynamic control volume model using Stencil interface

Here, the port_a.h_a and port_b.h_b are both set to equal medium.h permanently. The reason is that the specific enthalpy in connector port_a under the assumption of mass flowing from a FluidPort_a to another FluidPort_b (not the instance in this dynamic volume) is indeed medium.h independently of the actual mass flow rate port_a.m_flow. The situation for port_b is analogous.

Additionally, consider the isenthalpic flow model with a detailed pipe friction correlation based on the upstream density ρ_{up} and dynamic viscosity η_{up}. To allow for smoothing of the wall friction correlation at flow reversal, both potential upstream properties are required. Assuming that the flow model is connected to a control volume on each side, the model is written using a function f(dp, rho_a, rho_b) and the densities on both sides in order to implement a smooth transition between positive flow and negative flow (the dependency on and the computation of the dynamic viscosity are omitted from now on for readability). The code of this flow model is the following.

```
model FM_pipeFriction
  replaceable package Medium = PartialPureSubstance;
  FluidPort_a port_a,
  FluidPort_b port_b;
  Density rho_a, rho_b;
  Pressure dp;
equation
  port_a.m_flow + port_b.m_flow = 0;
  port_b.h_b = port_a.h_b; // Energy balance in design direction
  port_a.h_a = port_b.h_a; // ... in non-design direction
  rho_a = Medium.density_ph(port_a.p, port_a.h_b);
  rho_b = Medium.density_ph(port_b.p, port_b.h_a);
  dp = port_a.p - port_b.p;
  port_a.m_flow = f(dp, rho_a, rho_b); // m_flow vs. dp correlation
end FM_pipeFriction;
```

Listing 4: Isenthalpic flow model using Stencil interface

If the fluid flows in design direction, then the specific enthalpy at port_b under the assumption of mass flowing from FluidPort_b to some FluidPort_a (i.e., port_b.h_b) equals port_a.h_b. Non-isenthalpic flow models are implemented similarly. Consider a basic isentropic component with an extra mechanical shaft to account for the extracted mechanical power.

```
model FM_stodolaTurbine
  replaceable package Medium = PartialPureSubstance;
  FluidPort_a port_a,
  FluidPort_b port_b;
  Temperature T_a;
  Modelica.Mechanical.Rotational.Interfaces.Flange_a shaft;
  Power P_mechanical;
  SpecificEnthalpy h_a_outflow;
  SpecificEnthalpy h_b_outflow;
  Power H_flow_a, H_flow_b;
  parameter Real K_t = 0.5 "Stodola turbine constant";
equation
  port_a.m_flow + port_b.m_flow = 0;
  h_a_outflow = Medium.isentropicEnthalpy(port_a.p,
    Medium.setState_ph(port_b.p, port_b.h_a));
  h_b_outflow = Medium.isentropicEnthalpy(port_b.p,
    Medium.setState_ph(port_a.p, port_a.h_b));
  port_a.h_a = h_a_outflow;
  port_b.h_b = h_b_outflow;
  T_a = Medium.density_ph(port_a.p, port_a.h_b);
  P_mechanical = shaft.tau * der(shaft.phi);
  H_flow_a = port_a.m_flow*(if port_a.m_flow > 0 then
                 port_a.h_b else port_a.h_a);
  H_flow_b = port_b.m_flow*(if port_b.m_flow > 0 then
                 port_b.h_a else port_b.h_b);
  port_a.m_flow = K_t * ((port_a.p^2 - port_b.p ^2)/T_a)^0.5;
  P_mechanical + H_flow_a + H_flow_b = 0;
end FM_stodolaTurbine;
```

Listing 5: Non-isenthalpic flow model using Stencil interface

The specific enthalpies are transformed according to the thermodynamic processes (here isentropically) and set to equal to the respective interface variables. Note that the equations involving the sign of the mass flow rate are only used to establish output variables.

2.5 Biased mixing volume

This thermo-fluid interface has been proposed by Tummescheit [181] and Eborn [53]. In particular, it was implemented in the ThermoFluid library by Tummescheit, Eborn and Wagner [182] and its predecessors. Also, this interface was later implemented in a first issue of the Modelica_Fluid library.

The original ThermoFluid library used overdetermined connectors to avoid unnecessary property inversions[2] as no language elements and tool support were available to avoid this in a different way. However, the concept can be translated to the non-overdetermined connector suggested by Elmqvist et al. [58]. As this connector definition is more concise, it will be used in place of the original connector in the following. Note that this is an universal connector (in contrast to the complementary connectors used for the stencil interface as described in sec. 2.4).

```
connector FluidPort
  Pressure p;
  flow MassFlowRate m_flow;
  SpecificEnthalpy h;
  flow EnthalpyFlowRate H_flow;
end FluidPort;
```

Listing 6: Fluid connector using Biased Mixing Volume interface

The interface uses a simple yet powerful approach to fulfill the requirement of numerical robustness at zero mass flow rate. Instead of utilizing a stencil to access all required properties, a single intermediate thermodynamic state is defined on the connector. This intermediate

[2]This topic is discussed later in detail in section 3.1.3 and chapter 4.

thermodynamic state does not conceptually represent the properties after ideal mixing in the connection set but equals the thermodynamic properties in the dynamic volume model attached to the connection set. This is done because the state variables representing the convected quantities of a dynamic control volume remain well-defined at zero mass flow rate. The properties in the connection set are thus "biased" toward the next dynamic control volume model.

When using this thermo-fluid interface, component models are implemented along a simple rule. The effort variable representing the intensive convected quantity (i.e., "specific enthalpy" for the present example) is set to equal that of the control volume model (which, for transient problems, is either a state variable itself or based on a state variable). The corresponding flow rate (e.g., "convected enthalpy flow") is established inside the "flow model"-type components as mass flow rate times this intensive convected quantity. Here, an upstream value has to be used. Using this concept, a control volume model can be implemented using the following code.

```
model CV
  FluidPort port;
  replaceable package Medium = PartialPureSubstance;
  parameter Volume V;
  Medium.BaseProperties medium;
  Mass m;
  Energy U;
equation
  port.p = medium.p;
  port.h = medium.h; // specific enthalpy on the connector
  m = V*medium.d;
  U = m*medium.u;
  der(m) = port.m_flow;
  der(U) = port.H_flow;
end CV;
```

Listing 7: Dynamic control volume model using Biased Mixing Volume interface

The isenthalpic flow model with a detailed pipe friction correlation based on the upstream density ρ_{up} and dynamic viscosity η_{up} is now implemented as follows.

```
model FM_pipeFriction
  replaceable package Medium = PartialPureSubstance;
  FluidPort port_a, port_b;
  Density rho_a, rho_b;
  Pressure dp;
equation
  port_a.m_flow + port_b.m_flow = 0;
  port_a.H_flow + port_b.H_flow = 0;
  // Enthalpy flow rate
  port_a.H_flow = semiLinear(port_a.m_flow, port_a.h, port_b.h);
  dp = port_a.p - port_b.p;
  rho_a = Medium.density_ph(port_a.p, port_a.h);
  rho_b = Medium.density_ph(port_b.p, port_b.h);
  port_a.m_flow = f(dp, rho_a, rho_b); // m_flow vs. dp correlation
end FM_pipeFriction;
```

Listing 8: Isenthalpic flow model using Biased Mixing Volume interface

While the original implementation of this concept used an `if then else` control structure, the code fragment listed above uses the `semiLinear()`-operator. Conceptually, `semiLinear(x,` p, n) is translated to if x>=0 then p*x else n*x. Among some other subtleties, the operator triggers an analysis to avoid unnecessary property inversions[3]. The operator is defined in the Modelica Language Specification (e.g., section 3.7.2.4 of [123]).

Putting the states on the interfaces works well whenever model topologies of alternating dynamic control volume and static flow models are built. The reason is that then the port enthalpies of each flow model are both a function of the state variables of the fluid stored in the respective dynamic control volume models, and thus known at each time step. The simulation problem can be solved at each time step without solving implicit equations.

The basic isentropic component with a mechanical shaft to account for the extracted mechanical power is implemented as follows.

```
model FM_stodolaTurbine
  replaceable package Medium = PartialPureSubstance;
  FluidPort port_a, port_b;
  Temperature T_a;
  Modelica.Mechanical.Rotational.Interfaces.Flange_a shaft;
  Power P_mechanical;
  SpecificEnthalpy h_a_outflow;
  SpecificEnthalpy h_b_outflow;
  parameter Real K_t = 0.5 "Stodola turbine constant";
equation
  port_a.m_flow + port_b.m_flow = 0;
  port_a.H_flow + port_b.H_flow + P_mechanical = 0;
  port_a.H_flow = semiLinear(port_a.m_flow, port_a.h,
    h_a_outflow);
  port_b.H_flow = semiLinear(port_b.m_flow, port_b.h,
    h_b_outflow);
  h_a_outflow = Medium.isentropicEnthalpy(port_a.p,
    Medium.setState_ph(port_b.p, port_b.h));
  h_b_outflow = Medium.isentropicEnthalpy(port_b.p,
    Medium.setState_ph(port_a.p, port_a.h));
  T_a = Medium.temperature_ph(port_a.p, port_a.h);
  port_a.m_flow = K_t * ((port_a.p^2 - port_b.p ^2)/T_a)^0.5;
  P_mechanical = shaft.tau * der(shaft.phi);
end FM_stodolaTurbine;
```

Listing 9: Non-isenthalpic flow model using Biased Mixing Volume interface

Note that the complete isentropic compression and expansion model supports flow reversal. As before, only Stodola's turbine equation is (as implemented) meaningful for the design mass flow direction only (from port_a to port_b).

2.6 *Unbiased mixing volume*

Out of the established thermo-fluid interfaces, the one proposed by Elmqvist et al. [58] is the most recent one. It was implemented in a second issue of the Modelica_Fluid library [28] between 2003 and 2008. Like the "Biased Mixing Volume"-interface, it defines a single intermediate thermodynamic state on the connection set. The semantics of the interface are designed such that the intermediate state represents the properties after ideal mixing in the connection set. This is considered one of the key advantages of the approach, as the connectors and their variables thus have a rather intuitive meaning. The properties in the connection set are thus "unbiased".

In the scope of this non-causal thermo-fluid interface, Elmqvist et al. [58] suggested the connector definition, which was applied to the biased mixing volume interface already. It was

[3]This topic is discussed later in detail in section 3.1.3 and chapter 4.

listed in 6 already and is repeated here for convenience.

```
connector FluidPort
  Pressure p;
  flow MassFlowRate m_flow;
  SpecificEnthalpy h;
  flow EnthalpyFlowRate H_flow;
end FluidPort;
```

Listing 10: Fluid connector using Unbiased Mixing Volume interface

When implementing component models, an equation for the enthalpy flow rate `H_flow` is included for each connector instance in most of the cases. The equation makes use of the `semiLinear()`-operator, which was described in section 2.5. Using this thermo-fluid interface, a control volume model is implemented using the following code.

```
model CV
  FluidPort port;
  replaceable package Medium = PartialPureSubstance;
  parameter Volume V;
  Medium.BaseProperties medium;
  Mass m;
  Energy U;
equation
  port.p = medium.p;
  port.H_flow = semiLinear(port.m_flow, port.h, medium.h);
  m = V*medium.d;
  U = m*medium.u;
  der(m) = port.m_flow;
  der(U) = port.H_flow;
end CV;
```

Listing 11: Dynamic control volume model using Unbiased Mixing Volume interface

Note that an equation for the enthalpy flow rate is given, which depends on the sign of the mass flow rate. It therefore implements an upstream discretization as discussed in section 2.2.1.3.

The isenthalpic flow model with a detailed pipe friction correlation based on the upstream density ρ_{up} and dynamic viscosity η_{up} is implemented as follows.

```
model FM_pipeFriction
  replaceable package Medium = PartialPureSubstance;
  FluidPort port_a, port_b;
  Density rho_a, rho_b;
  Pressure dp;
equation
  port_a.m_flow + port_b.m_flow = 0;
  port_a.H_flow + port_b.H_flow = 0;
  // Enthalpy flow rate
  port_a.H_flow = semiLinear(port_a.m_flow, port_a.h, port_b.h);
  dp = port_a.p - port_b.p;
  rho_a = Medium.density_ph(port_a.p, port_a.h);
  rho_b = Medium.density_ph(port_b.p, port_b.h);
  port_a.m_flow = f(dp, rho_a, rho_b); // m_flow vs. dp correlation
end FM_pipeFriction;
```

Listing 12: Isenthalpic flow model using Unbiased Mixing Volume interface

Note that for this *isenthalpic flow model*, only one equation involving the enthalpy flow rate and the `semiLinear()`-operator is formulated (i.e., for one connector instance). Instead of adding a second equation for the other connector instance, a static balance equation of the enthalpy flow rates is posed. The basic isentropic component with a mechanical shaft to account for the extracted mechanical power in turn is implemented as follows.

```
model FM_stodolaTurbine
  replaceable package Medium = PartialPureSubstance;
  FluidPort port_a, port_b;
  Temperature T_a;
  Modelica.Mechanical.Rotational.Interfaces.Flange_a shaft;
  Power P_mechanical;
  SpecificEnthalpy h_a_outflow;
  SpecificEnthalpy h_b_outflow;
  parameter Real K_t = 0.5 "Stodola turbine constant";
equation
  port_a.m_flow + port_b.m_flow = 0;
  port_a.H_flow + port_b.H_flow + P_mechanical = 0;
  port_a.H_flow = semiLinear(port_a.m_flow, port_a.h,
    h_a_outflow);
  port_b.H_flow = semiLinear(port_b.m_flow, port_b.h,
    h_b_outflow);
  h_a_outflow = Medium.isentropicEnthalpy(port_a.p,
    Medium.setState_ph(port_b.p, port_b.h));
  h_b_outflow = Medium.isentropicEnthalpy(port_b.p,
    Medium.setState_ph(port_a.p, port_a.h));
  T_a = Medium.temperature_ph(port_a.p, port_a.h);
  port_a.m_flow = K_t * ((port_a.p^2 - port_b.p ^2)/T_a)^0.5;
  P_mechanical = shaft.tau * der(shaft.phi);
end FM_stodolaTurbine;
```

Listing 13: Non-isenthalpic flow model using Unbiased Mixing Volume interface

Note that the exemplary component implementations using the unbiased mixing volume thermo-fluid interface appear to be somewhat similar to those using the biased thermo-fluid interface discussed in section 2.5. However, the underlying concept and the resulting properties of the interfaces are quite different.

CHAPTER 3

REQUIREMENTS ON INTERFACES

The objective of this chapter is to address the definition of metrics, which can be used to assess the quality of a thermo-fluid interface. The metrics or requirements posed in this chapter can then be used to rigorously assess non-causal thermo-fluid interface definitions.

3.1 Robustness and efficiency

In engineering, requirements for robustness and efficiency are nearly universal. These requirements are pivotal and are addressed at hand of representative metrics such as required computational time, and the handling of zero and reversing mass flows.

3.1.1 Computational time

The required computational time is an important metric to assess the efficiency of a thermo-fluid interface. With the definition of relevant test cases, it provides a quantitative metric.

REQUIREMENT 1: The computational time to solve reference problems shall be low.

3.1.2 Zero and reversing mass flow

System-level simulation of thermo-fluid dynamics often involves conceptual simplifications of physical phenomena such as thermal conduction and convection. For small mass flow rates and non-idealized fluid continua as described by the Navier-Stokes equations, conduction is the dominant phenomenon. For a flow model two idealizations are usually made. First, the flow model does not define any storage. As a consequence, all convected properties are propagated instantaneously at flow reversal. Second, it is usually decided to neglect conduction. Consequently, the thermodynamic properties are strictly speaking not defined at zero mass flow rate.

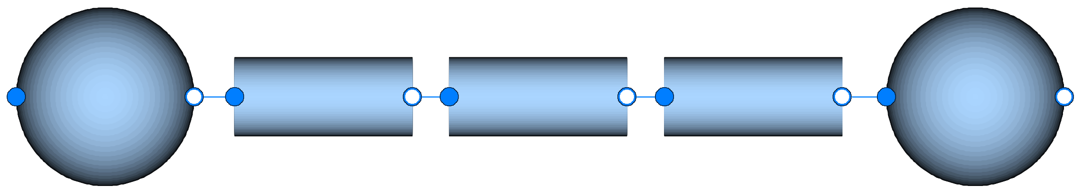


Figure 5: A chain of three static flow models between dynamic control volume models

The resulting problems can be illustrated at the example of *numerically* zero mass flow rate through a series of flow models using upstream properties in their equations. The topology is illustrated in figure 5. Depending on whether the numerical solver converges to $\dot{m} = 0 + \varepsilon$ or $\dot{m} = 0 - \varepsilon$, the upstream properties refer to the thermodynamic properties on one side or the other of the series of flow models. Only the control volumes with dynamic states remain well-defined and continuous.

An established approach to avoid such problems is to forbid such series of flow models and require alternating control volume and flow models. While this leads to more robust models, it restricts the user in his or her choice of model topology and increases the number of state variables, which possibly results in a stiff DAE.

REQUIREMENT 2: A thermo-fluid interface shall handle zero and reversing mass flow in a robust and efficient manner.

This requirement has further implications. Even if many applications apparently require only *upstream* fluid properties, several applications call for access to downstream properties, too. This may be required for specific types of models themselves or for smoothing of correlations at flow reversal. As an example of the latter, consider an algebraic flow model with a detailed friction correlation based on the upstream density ρ_{up} and dynamic viscosity η_{up}. Following the reasoning above, at mass flow reversal, the upstream properties change discontinuously and may show spurious oscillations at numerically zero mass flow rate between both potential upstream conditions. To avoid numerical difficulties, the correlation is usually smoothed at small mass flow rates using the properties on both sides of the flow model. This obviously requires access to *both* properties (upstream and downstream).

Requirement 3: A thermo-fluid interface shall provide both upstream and downstream properties to enable robust and efficient simulation code at zero and reversing mass flow.

As highlighted in the preceding section, convectively transported properties change their values discontinuously when the mass flow is reverted and, at the same time, conductance or dissipation is neglected. Similarly, the mixing enthalpy in an ideal mixing junction with n fluid flows is also discontinuous when mass flows are reverting[1]. Additionally, the component equations may be formulated using Boolean conditions. If an algebraic loop contains a condition on a variable that depends on the solution of said loop then the nonlinear equation system contains a Boolean unknown.

Requirement 4: For a thermo-fluid interface, all such properties (discontinuous iteration variables, discontinuous residuals, and mixed discrete continuous residual equations involving Boolean unknowns) shall be avoided.

The reason is that they violate the prerequisites of a relevant fraction of algebraic equation solvers and potentially result in poor robustness and efficiency if such variables or equations are contained in algebraic equation systems that have to be solved numerically.

3.1.3 Fluid property inversion at connection points

Following a general minimality principle, a thermo-fluid interface should be defined using a minimum set of variables that is required to describe the physical interactions taking place at the interface. Based on the governing equations of thermal fluid dynamics given in section 2.1, reasonable candidates for such a set are pressure and specific enthalpy. Consider a medium model using pressure and temperature as independent variables (e.g., a perfect gas). If, based on the suggested connector variables[2], two components are connected together, the following equations are obtained after the first pass of symbolic manipulation:

```
medium_a.h = Medium.h_pT(medium_a.p, medium_a.T);
medium_b.h = Medium.h_pT(medium_b.p, medium_b.T);
medium_a.h = medium_b.h;
medium_a.p = medium_b.p;
```

Assume now that `medium_a.p` and `medium_a.T` are state variables so that they are known at each time step, while the corresponding `medium_b` properties (e.g., within an adjacent temperature sensor component) must be computed from the system equations. After applying alias substitution for `medium_b.h` and `medium_b.p`, the Block Lower Triangular transformation allows to solve these equations sequentially.

```
medium_a.h := Medium.h_pT(medium_a.p, medium_a.T);
0 = medium_a.h - Medium.h_pT(medium_a.p, medium_b.T);
```

If no further symbolic manipulation is applied, then the last equation must be solved for temperature `medium_b.T` numerically, which is inefficient and unfortunate, since it is apparent that the thermodynamic properties of both medium models are the same.

To solve this problem, two different techniques have been employed.

[1] If $n-2$ flows are zero at some instance in simulation time.

[2] They are effort or potential variables and have thus to be set equal.

- Instead of using a minimum set of variables in the connector, additional (redundant) variables can be included. In the given example, temperature could be inserted. Then, the thermodynamic properties are computed once (in the control volume model for example) and provided via alias variables to the adjacent component (the temperature sensor in the given example). While this is a feasible and proven technique, it has one major drawback in that it allows only one-to-one connections. Consequently, it does not meet flexibility requirements and is not considered a modern approach.

- In order to maintain well-defined, non-redundant connector designs an alternative approach can be taken to avoid a numerical solution of the implicit equation. Using a standardized annotation, the function `Medium.h_pT` could be marked as monotonic with respect to the second input argument. Like this, any tool can automatically infer that

  $$x = f(y, v) \wedge x = f(y, w) \Rightarrow v = w \tag{26}$$

  This allows to substitute the implicit equation with the simple assignment

  ```
  medium_b.T := medium_a.T;
  ```

Requirement 5: A means to avoid property inversion shall be provided. This is an important requirement for all component models using thermodynamic property computations that do not use the connector effort variables as independent variables.

3.1.4 Semantics allow for meaningful simplifications

Efficient and robust handling of reversing flow is not required in all cases. For some applications, users are explicitly only interested in flow in design direction. Notably, such simplifications usually result in a relevant speed-up of computational time, which may be crucial for real-time performance of complex models for example.

Requirement 6: The semantics of a thermo-fluid interface shall thus include a means to prescribe assumptions on the flow direction of fluid through the interface. This may be prescribed constantly or for initialization and time simulation separately.

One approach to do so is to prescribe the `min` or `max` attribute of floating point variables on the connectors (e.g., the mass flow rate). Then, a Modelica translator can make such simplifications based on the semantics of the interface.

3.1.5 Efficient handling of ideal mixing

Ideal mixing is a frequent assumption. In case of a large number of fluid flows n mixed in a single ideal junction or media with many independent substances n_{Xi}, the ideal mixing may become inefficient if the equations were not properly designed.

Requirement 7: Ideal mixing shall be robust and efficient for a thermo-fluid interface. Ideally, the number of equations in the reduced algebraic equation system should be independent of both the number of mixed flows n and the length of the independent substance vector n_{Xi}.

While no approach was formulated yet to reach both goals concurrently, it is certainly possible to stay independent of *one* of the two parameters.

3.2 User Friendliness, end-user

A thermo-fluid interface should provide an intuitive mapping between the model topology exposed in a graphical user interface and the *mathematical* model it represents. For instance, a connection line is often interpreted as an infinitesimally short, idealized pipe segment between the connected components. This implies that, for example, two dynamic control volume models directly connected to each other, are not lumped into a single volume. While, following the given interpretation of the connection line, the volume pressures have to be identical, their temperature should, in general, be different.

Requirement 8: To assist the end-user, the model decomposition shall be intuitive. Furthermore, the thermo-fluid interface variables provided in simulation results shall have a clear meaning.

3.3 User Friendliness, developer

Even if a developer thoroughly masters a thermo-fluid interface definition, it may be easy or laborious to build models using the interface.

Requirement 9: To assist the the developer, the implementation of models using a given thermo-fluid interface shall be intuitive and simple.

3.4 Flexibility

Flexibility is demanded by both library developers and users. This has many advantages beyond sparing the need to develop and maintain several incarnations of the same model (e.g., when using complementary connectors A and B the models with connectors AAA, AAB, ABB, BBB). A thermo-fluid interface with such flexibility enables complex hierarchies, especially in conjunction with advanced language features such as parametric polymorphism (i.e., type parameters). Otherwise, the varying need for junction models or junction parametrization and potentially conflicting requirements on polarity conventions (e.g., design mass flow direction) impose burdens on the developer.

As a practical example, consider figure 6. The element with three ports illustrates a hierarchical model. Following some convention on component polarity (e.g., design mass flow direction), all possible polarity combinations are required even for simple models. In general it is not possible to accomplish this with vectorized interfaces (as in the case of junction models).

Requirement 10: A thermo-fluid interface shall utilize a single universal connector (instead of, for example, two complementary ones). Furthermore, it shall allow multiple way connections (as opposed to one-to-one connections, which require explicit junction models).

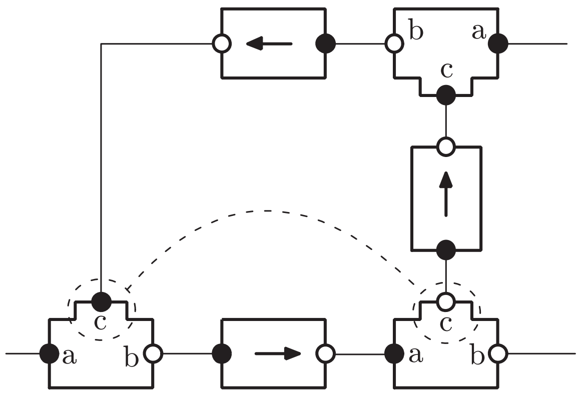


Figure 6: A notional hierarchical model

CHAPTER 4

PRELIMINARY EVALUATION OF ESTABLISHED INTERFACES

The objective of this chapter is to apply the requirements on thermo-fluid interfaces, which were defined in chapter 3. Like this, the established interfaces described in chapter 2 are assessed. In particular, a consistent line of argument shall trace the difficulties reported in the introduction to deficiencies of the interfaces with respect to the requirements.

Note that the assessment involves not only analysis of the semantics, but also actual computation on exemplary cases (due to, for instance, the requirement for low computational time). For this reason, the chapter first introduces methods and tools that were developed for the purpose of this comparison. Then, results are presented and discussed.

4.1 Methods and tools

In order to systematically compare thermo-fluid interfaces, dedicated testing tools were developed. These were compiled in a testing library FluidSandbox. The key concept was to treat the thermo-fluid interfaces (i.e., connector definitions and connection semantics) via parametric polymorphism [120]. The testing library was implemented in the equation-based, object-oriented modeling language Modelica. Using parametric polymorphism (a class with *replaceable* prefix [23]), it allows exchanging the non-causal thermo-fluid interface.

As a result, a plant model can be built independently of a specific thermo-fluid interface. At the same time, this enforces that the only difference between models that are compared is indeed the thermo-fluid interface. This allows to compare “apples to apples”.

4.1.1 The FluidSandbox library

Before discussing the concept and implementation, general constraints on model topologies that can be depicted using this library are introduced. These constraints are due to limitations of different thermo-fluid interfaces as described in chapter 2.

4.1.1.1 Topology constraints

Several implications on admissible model topologies result from the goal of allowing to swap the thermo-fluid interface implementations freely without making any changes to a test model and without changing what is modeled. After all, the test model has to meet all the restrictions of the different interfaces. The most important generic rules for test models follow.

1. Only volume models with `FluidPort_A` connectors are allowed (inherited from the biased mixing volume interface).
2. Only one-to-one connections are allowed (inherited from the stencil interface).
3. Only connections between `FluidPort_A` and `FluidPort_B` interfaces are allowed (inherited from the stencil interface, recommended for the biased mixing volume interface).
4. A flow model has to provide two `FluidPort_B` connectors; an asymmetric distributed pipe model using a staggered grid has to have a `FluidPort_A` and a `FluidPort_B` connector (inherited from the biased mixing volume interface).
5. Adapters such as AA or BB shall not be used, as they infer with the alternating pattern of control volumes and flow models (inherited from the biased mixing volume interface).

6. Boundary conditions can be used with both `FluidPort_A` and `FluidPort_B` interfaces as long as rule three is followed (connections are only made between `FluidPort_A` and `FluidPort_B` instances). Otherwise, the modeled system is not constant over the thermo-fluid interface implementations as using a `FluidPort_A` boundary condition connected to a dynamic volume results in a higher index problem (and possibly index reduction, which removes the specific enthalpy state when using the biased mixing volume interface). When using any of the other thermo-fluid interfaces, no higher index problem results.

4.1.1.2 Implementation

In FLUIDSANDBOX, each thermo-fluid interface is implemented using parametric polymorphism and a replaceable package, which has to be a sub-type of `PartialFluidInterface`. This virtual class defines the interface to which the thermo-fluid interface implementations have to conform. First and foremost, it contains an interface for the connector definition.

```
partial package PartialFluidInterface
  "Partial fluid interface implementation"

  replaceable partial connector FluidPort
    "Partial connector for FluidInterface"

    replaceable package Medium = Media.Interfaces.PartialMedium
      "Medium model";

  end FluidPort;

  replaceable partial connector FluidPort_a
    "Generic fluid connector at design inlet"

    replaceable package Medium = Media.Interfaces.PartialMedium
      "Medium model";

  end FluidPort_a;

  replaceable partial connector FluidPort_b
    "Generic fluid connector at design outlet"

    replaceable package Medium = Media.Interfaces.PartialMedium
      "Medium model";

  end FluidPort_b;

  // ...

end PartialFluidInterface;
```

Listing 14: Virtual fluid interface, fluid connectors

These connector definitions are provided in the packages implementing thermo-fluid interfaces. Below, an example is attached for the implementation of the stencil approach described in section 2.4.

```
package Stencil_A
  "Stencil on staggered grid (similar to the ThermoPower library)"

  extends Interfaces.PartialFluidInterface;

  redeclare connector extends FluidPort
    "Connector of thermo-fluid interface"

    // Pressure in the connection point
    Medium.AbsolutePressure p;
    // Mass flow rate from the connection point into the component
    flow Medium.MassFlowRate m_flow

  end FluidPort;

  redeclare connector extends FluidPort_a
    "Generic fluid connector at design inlet"

    extends FluidPort;

    // Upstream specific enthalpy if fluid flew out of this port
    output Medium.SpecificEnthalpy h_a;
    // Upstream specific enthalpy if fluid flew into this port
    input Medium.SpecificEnthalpy h_b;

  end FluidPort_a;

  redeclare connector extends FluidPort_b
    "Generic fluid connector at design outlet"

    extends FluidPort;

    // Upstream specific enthalpy if fluid flew into this port
    input Medium.SpecificEnthalpy h_a;
    // Upstream specific enthalpy if fluid flew out of this port
    output Medium.SpecificEnthalpy h_b;

  end FluidPort_b;

  // ...

end Stencil_A;
```

Listing 15: Implementation of fluid interface, fluid connectors

Additionally to the interface for connectors, `PartialFluidInterface` defines the interfaces for model classes such as boundary conditions or isenthalpic flow models (`SourceInterfaceA`, `SourceInterfaceB`, `LumpedVolumeInterface`, `FlowModelInterface`, `FlowModelInterfaceAA`, `FlowModelInterfaceAB` etc.). First, a simple example of the interface of a source model class using a `FluidPort_A` connector is given. In `PartialFluidInterface`, it is defined as follows.

```
partial package PartialFluidInterface
  "Partial fluid interface implementation"

  // ...

  replaceable partial model SourceInterfaceA
    "Partial fluid interface of a source model (one Port_a)"

```

```
 9       // Medium model
10       replaceable package Medium
11         = Modelica.Media.Interfaces.PartialMedium "Medium model";
12
13       // Internal connector for sources
14       Source.InternalBus.InterfaceToImplementation internal(
15            redeclare final package Medium = Medium);
16
17       // Sandbox settings
18       outer Settings.SandboxSettings sandboxSettings
19         "Globally prescribed sandbox settings";
20
21       // Connector declaration
22       FluidPort_a port(redeclare package Medium
23         = Medium) "Fluid connector";
24
25     end SourceInterfaceA;
26
27     replaceable partial model SourceInterfaceB
28       "Partial fluid interface of a source model (one Port_b)"
29
30       // ...
31
32     end SourceInterfaceB;
33
34     // ...
35
36   end PartialFluidInterface;
```

Listing 16: Virtual fluid interface, model classes

Using the unbiased mixing volume thermo-fluid interface described in section 2.6, the interface of the model class is implemented in a straight-forward manner. Obviously, a component model cannot be implemented using the interface of the model class alone. The connector variables for mass flow rate and pressure are defined via alias equations on variables on an internal bus connector. The enthalpy flow rate is established via an equation in such variables. The internal bus was declared in the previous listing 16 in line 14.

```
package Unbiased_A
  "Unbiased mixing volume (similar to the Modelica_Fluid library)"

  // ...

  redeclare model extends SourceInterfaceA
    "Fluid interface of a source model (one Port_a)"

  equation
    // Provide the component implementation with the required
    // variables from the connectors
    port.m_flow = internal.m_flow_in;

    // Put the variables provided by the component implementation
    // onto the connectors
    port.p = internal.p_source;
    port.H_flow = semiLinear(port.m_flow, port.h, internal.h_source);
  end SourceInterfaceA;

  // ...

end Unbiased_A;
```

Listing 17: Implementation of fluid interface, model classes

The internal bus establishes a link between the thermo-fluid interface implementation and the generic component implementation. Figures 7 and 8 show the outside view and inside view (icon and diagram) of a generic boundary condition model in FLUIDSANDBOX. The thermo-fluid interface of the model class "boundary condition with `FluidPort_A`" discussed so far is shown in figure 8 in the middle. The other model in this figure is the implementation of the physical behavior of the boundary condition (as opposed to the thermo-fluid interface). Both are connected via an orange line, which represents the connection between both internal bus connectors.

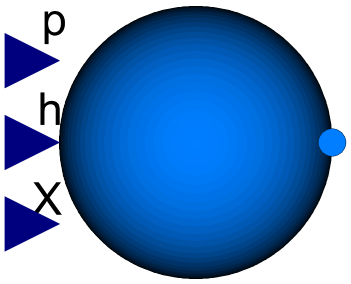


Figure 7: Outside view of a boundary condition model in FLUIDSANDBOX

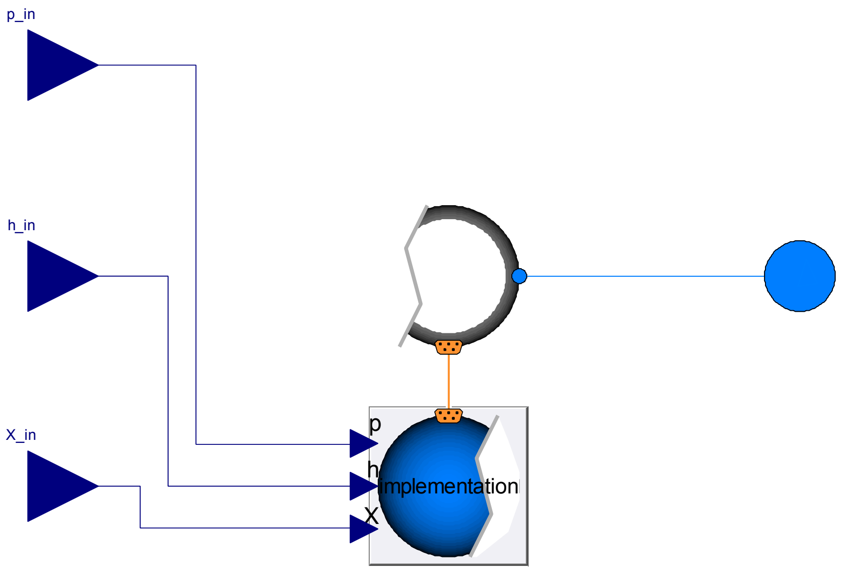


Figure 8: Inside view of a boundary condition model in FLUIDSANDBOX

The internal bus connector definition is specific to the type of component. This could be generalized, but in order to arrive at a library design that is still easy to understand and maintain, it was decided to not do so.

In order to implement new connection semantics for a new thermo-fluid interface, the thermo-fluid interface package provides a Boolean package constant that indicates whether or not to use a manual implementation of connection semantics. This is useful to test such semantics and to substantiate the claim that an interface has superior properties and that a regular implementation in a Modelica translator is required. Conceptually, a connection semantics object is an instance of a model class that has two ports. It has to replace each `connect()`

statement with one instance and two connections at either port. In the following example, a listing is provided that implements such an object for an obsolete interface iteration[1] [56].

```
redeclare replaceable model extends ConnectionSemantics
  "Implements unsupported connection semantics"

equation
/*
  Original form of equations generated from the
  unsupported connection semantics

  // Effort variables _without_ inside prefix: Equality
  port_a.p = port_b.p;
  port_a.h_outside = port_b.h_outside;

  // Effort variables _with_ inside prefix: Ignore

  // Flow variables: Sum to zero
  port_a.m_flow + port_b.m_flow = 0;
  port_a.H_flow + port_b.H_flow = 0;

  // Plus declaration equations inside connector (not listed)
*/

  // Manipulated form of connection equations using
  // new connection semantics
  port_a.p = port_b.p;
  port_a.m_flow + port_b.m_flow = 0;

  // Enthalpy flow
  port_a.h_inflow = port_b.h_outflow;
  port_a.h_outflow = port_b.h_inflow;

  port_a.h_outside = noEvent(if -port_a.m_flow > 0 then
          port_a.h_outflow else port_a.h_inflow);
  port_b.h_outside = noEvent(if -port_b.m_flow > 0 then
          port_b.h_outflow else port_b.h_inflow);

  port_a.H_flow = -semiLinear(
          -port_a.m_flow,
          port_a.h_inflow,
          port_a.h_outflow);
  port_b.H_flow = -semiLinear(
          -port_b.m_flow,
          port_b.h_inflow,
          port_b.h_outflow);
end ConnectionSemantics;
```

Listing 18: Implementation of fluid interface, connection semantics

[1]No example for such a connection semantics object for the stream connectors is given. First, this is the case because the concept is only introduced later in chapter 5. Furthermore, the connection semantics object is trivial for stream connectors, as it only has to flip the variables `h_outflow` and the ones representing the `inStream()`-operator in the test implementation.

Additionally to the thermo-fluid interface described so far, an interface for component implementations and internal buses is therefore required. These are compiled in packages, too. Each of them contains an internal bus definition, partial models with replaceable thermo-fluid interface *and* replaceable component implementation, as well as a interface for component implementations. The following listing provides an example of such an internal bus definition.

```
package Source "Interfaces for source models"

  package InternalBus "Internal interfaces for sources"
    connector InterfaceToImplementation
      "Internal bus: Connects from interface to implementation"

      // Medium model
      replaceable package Medium
        = Modelica.Media.Interfaces.PartialMedium "Medium model";

      // Mass flow rate
      output Medium.MassFlowRate m_flow_in "Mass flow rate into port";

      // Thermodynamic properties of reservoir
      input Medium.AbsolutePressure p_source "Pressure in source";
      input Medium.SpecificEnthalpy h_source
        "Specific enthalpy in source";

    end InterfaceToImplementation;

    connector ImplementationToInterface
      "Internal bus: Connects from implementation to interface"

      // Medium model
      replaceable package Medium
        = Modelica.Media.Interfaces.PartialMedium "Medium model";

      // Mass flow rate
      input Medium.MassFlowRate m_flow_in "Mass flow rate into port";

      // Thermodynamic properties of reservoir
      output Medium.AbsolutePressure p_source "Pressure in source";
      output Medium.SpecificEnthalpy h_source
        "Specific enthalpy in source";

    end ImplementationToInterface;

    connector Observer
      "Internal bus: Allows to provide the variables to sensors etc"
      // ...
    end Observer;
  end InternalBus;

  // ...

end Source;
```

Listing 19: Virtual components, internal bus

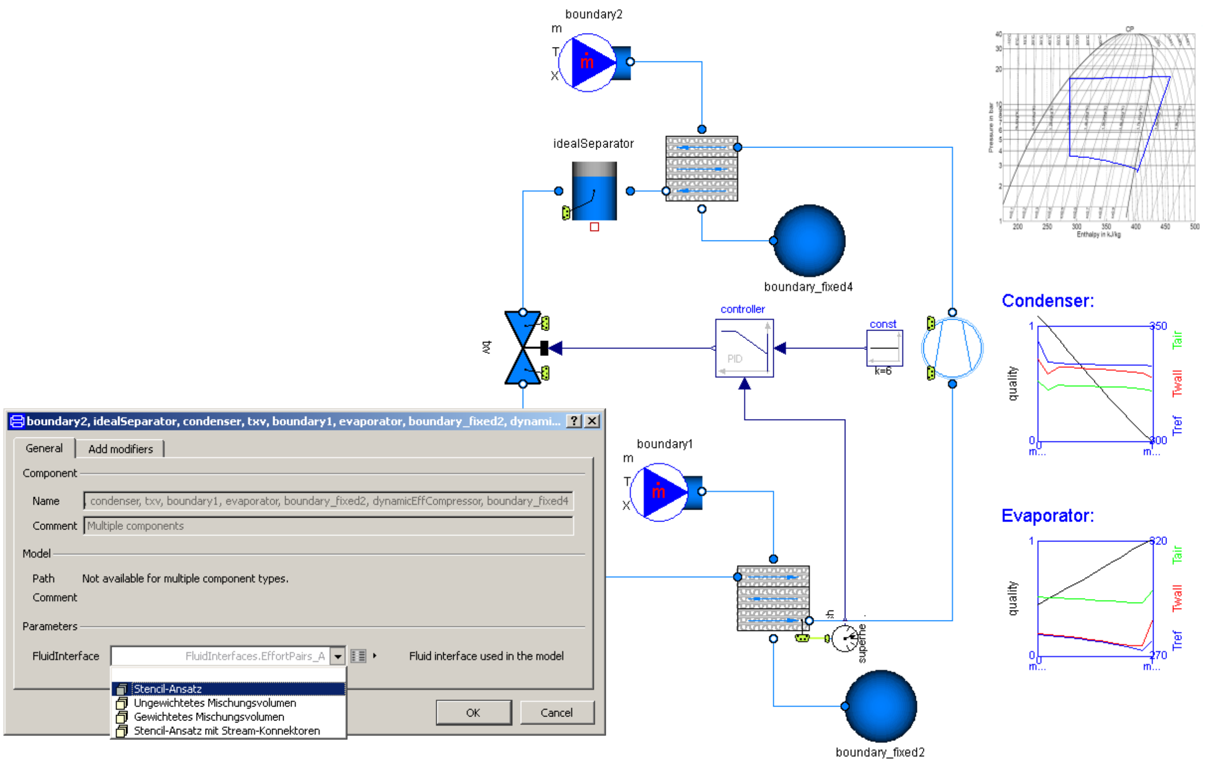
An example of partial models with replaceable thermo-fluid interface *and* replaceable component implementation can be inferred from figure 8. The definition of the interface for component implementations is simple in many cases, as it usually contains an internal bus connector and common implementation elements such as parameters or non-fluid connectors (e.g., heat transfer ports) only.

In order to control plant operation, sensors are often required. This is an important topic for the present comparison, as sensor models may affect the robustness and efficiency of a simulation model if not implemented properly. In FLUIDSANDBOX, the use of explicit sensor models is discouraged, as it is difficult to avoid varying effects on the evaluation metrics for all thermo-fluid interface implementations. Instead, component models provide additional sensor ports, on which readily available thermodynamic properties are exposed (in particular pressure and specific enthalpy). A sensor component may then take these values directly out of the component implementation (it is not connected via a fluid connector but a causal data connector) and compute derived quantities requested by the modeler such as saturation temperature. Like this, adverse effects of sensors both on the robustness as well as the efficiency are circumvented, as expensive computations of unnecessary thermodynamic properties are avoided.

4.1.2 Reference models

Several reference models were implemented for comparison of thermo-fluid interfaces. These include examples of important topology elements such as ideal mixing and industrial application models. Out of the latter type, a set of models was chosen as reference cases in this comparison.

These models implement a conventional vapor compression cycle using a sub-critical process. The model fidelity is similar to that used by Pfafferott [141]. The model topology is shown in figure 9. The illustration also shows how the thermo-fluid interface can be exchanged in a Modelica editor at the click of a mouse button.



boundary2, idealSeparator, condenser, txv, boundary1, evaporator, boundary_fixed2, dynami...

General | Add modifiers

Component

Name: condenser, txv, boundary1, evaporator, boundary_fixed2, dynamicEffCompressor, boundary_fixed4

Comment: Multiple components

Model

Path: Not available for multiple component types.

Comment

Parameters

FluidInterface: FluidInterfaces.EffortPairs_A — Fluid interface used in the model

Stencil-Ansatz
Ungewichtetes Mischungsvolumen
Gewichtetes Mischungsvolumen
Stencil-Ansatz mit Stream-Konnektoren

OK | Cancel

Figure 9: Reference model of vapor compression cycle

In particular, the heat exchangers use a standard Finite Volume Method with a fixed grid for discretization of the fluid passes. The discretization uses $n = 12$ cells per heat exchanger refrigerant side and air side. The heat transfer was implemented using the Nusselt number correlations of Shah [161] and Gungor and Winterton [75] for the refrigerant. For the air side, the Colburn correlation of Chang and Wang [34] was used. The refrigerant R134a was modeled following Tillner-Roth and Baehr [170]. This model of the thermodynamic properties was implemented outside of the present research by the company XRG Simulation GmbH. In the model, a topology of alternating flow model and control volume models is implemented. The model includes some simplifications, in particular, dry air is assumed and thus no condensation is included.

The reference models are obtained by simulating the vapor compression cycle under varying boundary and initial conditions. They all involve transient simulation only; initialization problems are addressed in chapters 7 to 10. In the end, the following reference models were defined.

1. Relaxation transient
2. Change in set-point after initialization in steady-state
3. Shut-down using pump-down control after initialization in steady-state

4.2 Results

First, the results of the comparison are presented in table 1 and then each score is explained in detail.

Criterion	Biased	Stencil	Unbiased
Robustness and efficiency	+	+	-
- Computational time	++	++	-
- Zero and reversing flow	+	++	--
- Property inversion	+	-	++
- Simplifications	+	+	+
- Ideal mixing	-	+	-
User fr., end-user	-	-	+
User fr., developer	+	-	+
Flexibility	-	-	++

Table 1: Preliminary evaluation of established thermo-fluid interfaces

4.2.1 Robustness and efficiency

4.2.1.1 Computational time

Reproducibility of the simulation timings was high. Still, each reference model was run ten times per thermo-fluid interface to establish the required computational time. The results were checked for outliers, which did not exist in any sample, and an average was established over all ten cases per interface.

Figure 10 shows the required computational time for the reference model "Relaxation transient". These and all other computations were carried out on a Windows PC with a Intel® Core™ 2 Duo T7700 CPU and 2 giga bytes of RAM. The figure shows how much computational time (ordinate) was required per simulation time (abscissa). A result below the angle bisector would signal real-time capability. Figures 11 and 12 show the required computational time for the reference models "shut-down" and "change in set-point" respectively.

In all three figures, the total required computational time is most relevant. It can be seen on the far right of the figures (at maximum simulation time). From the figures we infer that the required computational time is nearly identical for the biased mixing volume and stencil thermo-fluid interfaces. The unbiased mixing volume thermo-fluid interface in turn requires nearly twice the computational time in comparison to the other two approaches. This is why, based on requirement 1, the thermo-fluid interfaces using the stencil and the biased mixing volume were given a very good score and the unbiased mixing volume was given a poor score.

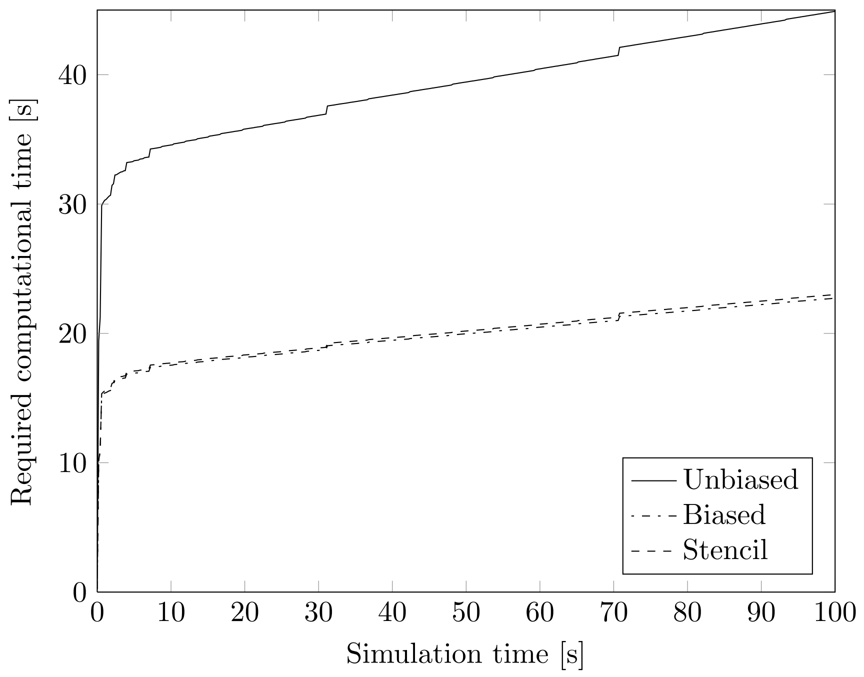


Figure 10: Required computational time, reference case "Relaxation transient"

4.2.1.2 Zero and reversing flow

Unbiased mixing volume: The semantics of this thermo-fluid interface are constructed to prescribe an infinitesimally small mixing volume in each connection set. Consider the case of a series of flow models between dynamic control volume models (as shown in figure 5 in section 3.1.2). After symbolic manipulation of the equations involving the `semiLinear()`-operator (see sections 2.5 and 2.6 for details), the specific enthalpy in the connection point between the first and the second flow model will be established based on the following expression

```
flowModelTwo.port_a.h = if flowModelTwo.port_a.m_flow > 0 then
  volumeOne.medium.h else volumeTwo.medium.h;
```

Listing 20: Energy balance, unbiased mixing volume interface

Consequently, the densities such as `flowModelOne.rho_b` and `flowModelTwo.rho_a` that are required for the wall friction correlation will depend on the sign of the flow rate. This introduces a discontinuity of density around zero mass flow rate (which is a critical case from a numerical point of view), and creates a system of non-smooth nonlinear equations whose unknowns are the port pressure, the port enthalpy, and the mass flow at the connection point between the two flow models. Furthermore, at numerically zero mass flow rate, the port-specific enthalpy shows spurious oscillations around both potential upstream values depending on which limit of zero the nonlinear equation solver converges to.

The same kind of problem arises when multiple-way connections are made between flow models and even for connections between flow models and control volumes (i.e., if a user sticks to an alternating model topology). The unbiased mixing volume interface thus does not fulfill requirement 2 adequately. Also, the interface does not provide both upstream and downstream values of convected properties, and therefore fails to meet requirement 3. This is why this thermo-fluid interface has a poor score for zero and reversing mass flow rates.

Stencil: For this thermo-fluid interface, the situation is different. By construction, it allows propagation of both upstream and downstream convected quantities and thus fulfills requirement 3. Additionally, the approach results in high robustness and efficiency. This is obvious from an analysis of the connection semantics on a series of flow models.

In case of such a topology, the specific enthalpies of the dynamic control volume models at each end of the series of flow models are propagated in both directions. The specific enthalpies

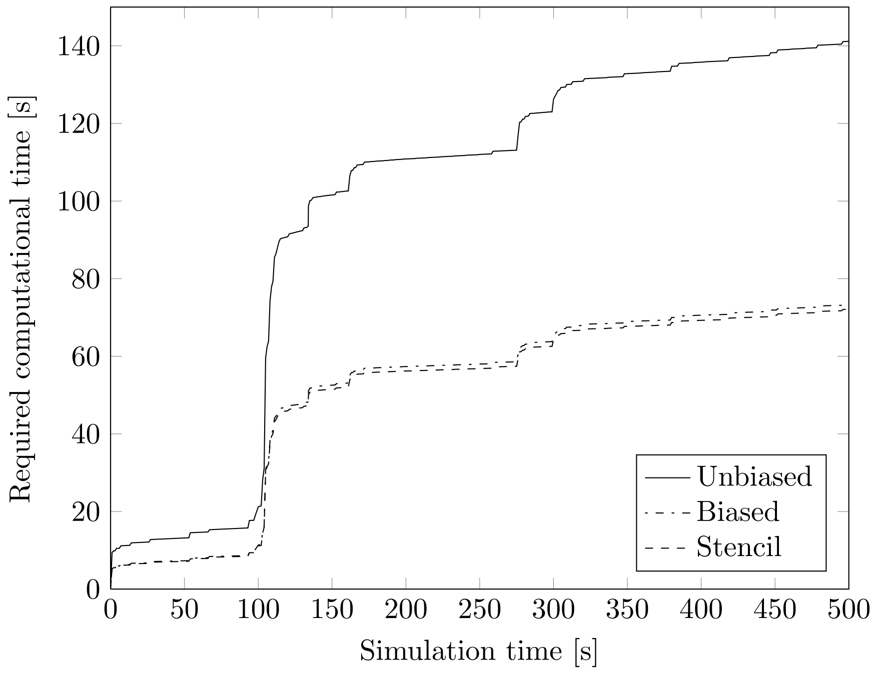


Figure 11: Required computational time, reference case "Shut-down"

of the control volume models are usually state variables themselves or computed based on state variables and thus well-behaved. Like this, the density for the wall friction correlation is permanently available for both potential flow directions and well-behaved. For isenthalpic flow models along the topology, the control volume specific enthalpy is propagated without any modification. In case of non-isenthalpic flow models, the specific enthalpy is explicitly transformed according to the thermodynamic process (for examples, see section 2.4). In either case, the two exemplary density variables are well-behaved for any value of the actual mass flow. Furthermore, they do not change when the sign of the flow rate is inverted. No discontinuities are generated in case of nonlinear algebraic equation systems as long as the component equations are smooth. Consequently, this thermo-fluid interface also fulfills requirement 4 and thus receives a very good score.

Biased mixing volume: When using this thermo-fluid interface, then the convected quantities on the connectors are always well-behaved as long as alternating topologies of flow models and volume models are used. The reason is that the convected quantities are state variables inside the volume models (or established using explicit algebraic equations based on state variables). Like this, even equations involving conditions on the sign of the mass flow rate such as ones involving the `semiLinear()`-operator will, in almost all sane applications, result in continuous solutions for iteration variables and residuals. If this specific model structure that this concept was designed for is followed, then it results in robust and efficient models. In case of a series of flow models between control volume models (such as the topology discussed in the previous paragraph), the semantics of this thermo-fluid interface also result in an infinitesimally small mixing volume like those of the unbiased mixing volume interface do. Just like before, after symbolic manipulation of the equations involving the `semiLinear()`-operator, the specific enthalpy in the connection point between the first and the second flow model will be established based on the following expression

```
flowModelTwo.port_a.h = if flowModelTwo.port_a.m_flow > 0 then
  volumeOne.medium.h else volumeTwo.medium.h;
```

Listing 21: Energy balance, biased mixing volume interface

Again, the densities such as `flowModelOne.rho_b` and `flowModelTwo.rho_a` that are required for the wall friction correlation will depend on the sign of the flow rate. This introduces

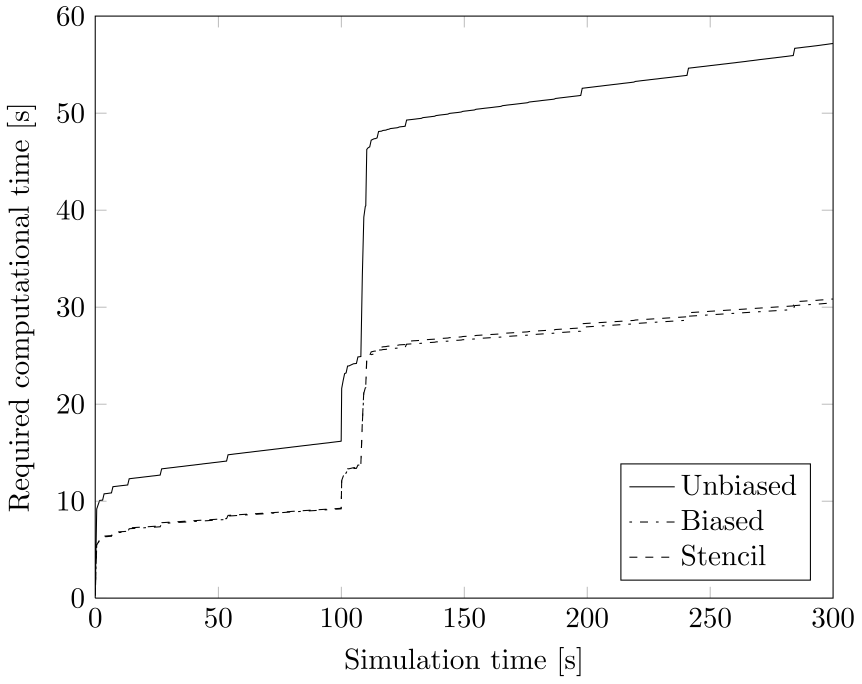


Figure 12: Required computational time, reference case “Set-point”

a same discontinuity of density around zero mass flow, and creates a system of non-smooth nonlinear equations. Furthermore the port-specific enthalpy shows spurious oscillations at numerically zero mass flow rate around both potential upstream values depending on which limit of zero the nonlinear equation solver converges to. In summary, the interface fulfills requirement 2 well for specific topologies. For these topologies (alternating flow models and control volumes), it also fulfills requirement 3[2]. In case of other topologies, robustness and efficiency are poor. This is why, in summary, this interface received a good score for this metric.

Based on the discussion above, it is obvious that the thermo-fluid interfaces fulfill requirement 4 (avoid discontinuous residuals) in a similar fashion. The unbiased mixing volume interface does not fulfill it at all, the biased mixing volume interface only for alternating topologies, and the stencil interface for all topologies.

4.2.1.3 Fluid Property Inversion at Connection Points

Conceptually, an algorithm in symbolic preprocessing to avoid property inversions at connection points is orthogonal to the definition of the thermo-fluid interface. At the time of writing, it is integrated into the semantics of the `semiLinear()`-operator however. Therefore, the unbiased mixing volume thermo-fluid interface fulfills requirement 5 and receives a very good score. The biased mixing volume interface in turn used overdetermined connectors in it original form, which is not considered an elegant solution to this problem. However, as shown in section 2.5, this thermo-fluid interface can also be used with non-overdetermined connectors and the `semiLinear()`-operator. Therefore, it can be modified to meet requirement 5 and receives a good score. On the other hand, the stencil thermo-fluid interface does not provide any means to avoid unnecessary property inversions. Theoretically, it could be used together with an overdetermined connector, too (and this would not result in additional constraints on possible connections, as the interface uses complementary connectors and one-to-one connections only anyway). Still, this is not considered an elegant solution as it is not based on symbolic preprocessing. This is why this interface does not properly fulfill requirement 5 and is given a poor score.

[2]Note that this interface fulfills requirement 3 only for flow model-like components. For control volumes access to both potential upstream properties is usually not required however and thus this requirement is considered fulfilled for the given type of topology.

4.2.1.4 Semantics Allow For Meaningful Simplifications

For all established thermo-fluid interfaces, simplifications in terms of a predefined flow direction can be made easily by adapting equations. For the stencil interface, only the equations relating the meaningful convected quantity variables (forward direction) need to be implemented, the other one can even be removed from the connector. For the biased and unbiased mixing volume, the conditional expression to establish the flow rate of the convected quantity is replaced by an unconditional product of quantity and mass flow rate.

If a Modelica translator provides such symbolic preprocessing, then the `min` or `max` attributes of floating point variables on the connector mass flow rate can alternatively be used to make such simplifications. Such attributes may be set based on Boolean parameters indicating whether to assume uni- or bi-directional flow. Then, the translator can, for example, select the appropriate branch of a conditional expression of the `semiLinear()`-operator.

To summarize, all three established thermo-fluid interfaces fulfill requirement 6 equally well and are thus given a good score.

4.2.1.5 Efficient Handling Of Ideal Mixing

For ideal mixing between three flow models and a flue gas with six substances, the unbiased and biased mixing volume interfaces result in an nonlinear algebraic equation system with twelve to 23 iteration variables (after symbolic reduction using tearing [57, 33, 25]). Obviously, this depends on the precise tearing heuristics of a Modelica tool. The given numbers apply for the Modelica translator Dymola® in versions up to 7.4. Using the stencil thermo-fluid interface, the coupled algebraic equation system had six to eight unknowns.

In summary, the efficiency and robustness of modeling ideal mixing together with fluids that contain several substances and either the unbiased or the biased mixing volume interface are poor. If these interfaces are used, it is usually advantageous to employ small dynamic mixing volumes, even if this may increase stiffness of the resulting equation system. This is why the latter interfaces were given a poor grade in this regard. The stencil interface in turn can be used for such ideal mixing problems and therefore this interface is understood to comply with requirement 7.

Note that the ideal mixing problem is analyzed in more detail in section 5.1.2.

4.2.2 User friendliness, end-user

Unbiased mixing volume: A goal of the development of the unbiased mixing volume thermo-fluid interface was to arrive at an intuitive meaning of the connection sets, which conceptually represents an infinitesimally small ideal mixing volume for this interface. For a simple model topology, the resulting system boundaries are illustrated in figure 13.

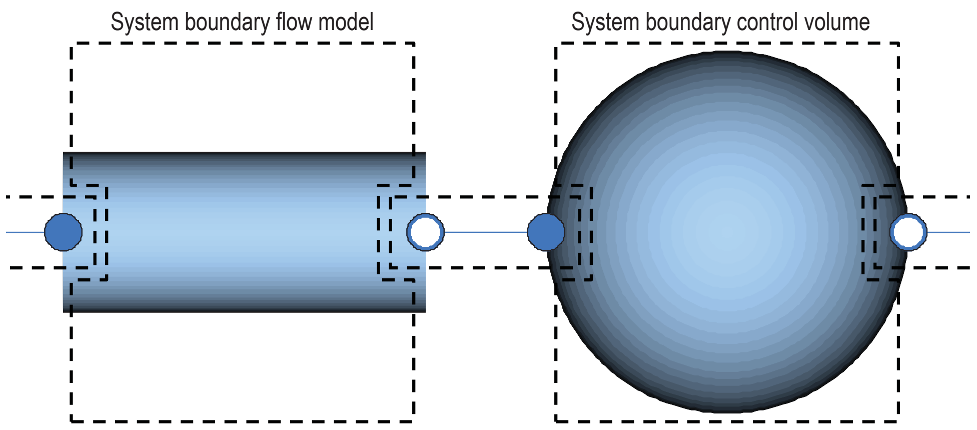


Figure 13: Control volume boundaries for example model topology (unbiased mixing volume interface)

For this thermo-fluid interface, interpreting the variables in the simulation results is easy. Furthermore, if connecting two control volume models, their pressure states are lumped by index reduction, their specific enthalpies remain separate however. This is a proper result. Altogether, requirement 8 is considered fulfilled and the interface is given a good score.

Biased mixing volume: The key element of this concept is that the specific enthalpy of a connection set is the specific enthalpy of the attached control volume model. This is illustrated via the system boundaries in figure 14 and leads to a simple description of the connector variables. Consider the case of mass flowing from left to right however. Then, the specific enthalpy at the outlet of the flow model will *not* reflect the specific enthalpy of the flow exiting said component in general. If, for example, the control volume model in the middle is attached to some heat source or sink, then the specific enthalpy at that connector will represent the thermodynamic properties subject to this heat transfer already. This behavior is not intuitive and may be confusing to users. Even if the connector variables have a simple definition, their meaning may not be always easy to understand for end-users. It might be necessary to declare extra variables reflecting the outlet conditions to clarify the results.

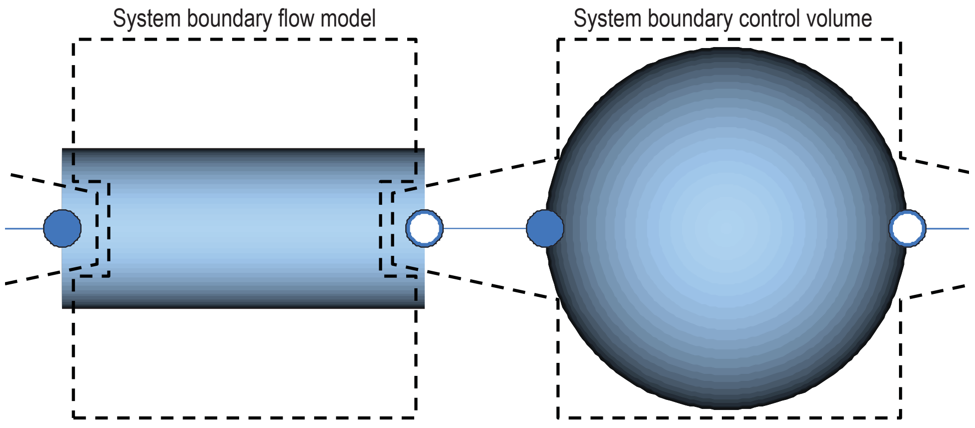


Figure 14: Control volume boundaries for example model topology (biased mixing volume interface)

The same can be stated about other topologies. If a control volume is part of a connection set, then the specific enthalpy in the connection set is always the specific enthalpy of the fluid within this control volume, while in a connection set spanning flow models only it actually depends on the direction of the flow. In fact, if a connection set included several control volumes and flow models then a virtual mixing volume was formed spanning all the control volumes and all the flow model connectors. This can be confusing for users, especially if this virtual mixing volume was spread among different composite models.

Consider the connection between two control volume models. In this case, both the pressure and the specific enthalpy of the fluid of the two volumes are the same. From a physical point of view, this means that the two volumes are *merged* into a single bigger volume, rather than being connected through a pipe of infinitesimal length. Modeling tools often employ index reduction in this case, which might be problematic in cases of complex media models, as the required derivatives may not be available at all or inefficient. Furthermore, this can be confusing to end-users, again especially if the two control volumes are hidden inside two hierarchically built models (two heat exchangers for example). In this approach, an ideal pipe model has to be used to “separate” two volume models.

Considering these facts, requirement 8 is not properly fulfilled and the biased mixing volume is awarded a poor score with respect to user friendliness for end-users.

Stencil: The physical meaning of the variables in the connector, which represent the convected quantities, is not intuitive as that for the unbiased mixing volume interface for example. Again, figure 14 illustrates the mapping of the components as shown in a graphical user interface (blue) to the mathematical models they represent (black). The stencil provides access to both system boundaries and thus upstream and downstream properties. The end-user has to understand under which conditions what variable refers to the actual, not the hypothetical case. If, for some reason, the exact compliance between interface instance name and class name is not given, inspection of the interface quantities may be confusing to end-users.

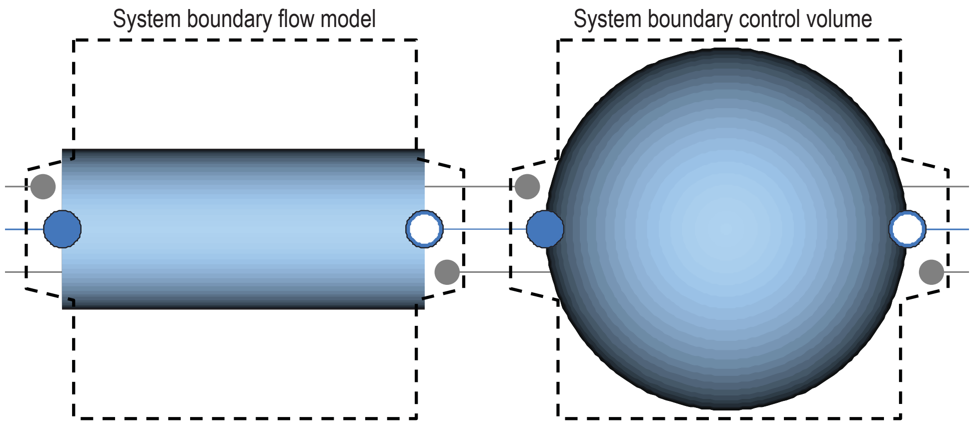


Figure 15: Control volume boundaries for example model topology (stencil interface)

Whenever a direct connection is made between two control volumes, the two volume pressures are bound to be equal, but the enthalpies are not. The generated model corresponds to two volumes that were connected by an infinitesimally short pipe and they are not merged into a single, bigger volume.

All in all, requirement 8 is not properly fulfilled and the thermo-fluid interface is given a poor score with respect to user friendliness for end-users.

4.2.3 User friendliness, developer

Unbiased mixing volume: In most cases, the thermo-fluid interface of models can be implemented easily. For most models, exactly one equation per convected quantity has to be implemented using the `semiLinear()`-operator. If the convected quantity is a vector, then this obviously has to be a vector equation. An exemplary exception to this rule is given by flow models conserving the convected quantity statically[3] (as mentioned in section 2.6 already, examples are isenthalpic and statically species conserving flow models). Here, such equations are provided for one connector only. For the other, a static flow balance equation is formulated for the flow rate variables of the convected quantities. Altogether, requirement 9 is fulfilled and the thermo-fluid interface is awarded a good score.

Biased mixing volume: If a developer thoroughly understands this concept, the thermo-fluid interface of model classes can be implemented easily, too. A developer has to understand the distinction between connectors, for which the intensive convected quantity has to be prescribed, and connectors, for which the convected flow rates are prescribed. For all connectors of components that represent simple lumped parameter approximations of dynamic control volumes, the former is the case. Also, for components, which represent discretized distributed domains on a staggered grid, all connectors that expose a control volume of a dynamic mass and energy balance are usually handled this way. Connectors of simple lumped parameter approximations of flow models and ones that expose a control volume of a static momentum balance on a staggered grid approximation of a distributed domain are handled differently. For them, the convected flow rates are usually prescribed. This interface therefore fulfills requirement 9 and is also given a good score.

Stencil: Even if a developer understands this concept, the thermo-fluid interface of a model class can be laborious to implement. The reason is that the conceptual cases of fluid flowing into or out of a connector need to be translated to the A-B or B-A pairing at the respective connectors. In order to do so, the correspondence between `h_a` for fluid leaving a `FluidPort_A`

[3]This is obviously a redundant (*statically* plus *flow model*) but emphasizes the point.

connector and `h_b` for fluid entering the `FluidPort_A` connector on one hand and `h_b` for fluid leaving a `FluidPort_B` connector and `h_a` for fluid entering the `FluidPort_B` connector on the other need to be present. With some practical experience this correspondence is obvious but it may still be required to look up the fluid connector class if the instance name does not properly reveal whether a type A or type B connector is used. In summary, this concept does not properly fulfill requirement 9 and is given a poor score in this regard.

4.2.4 Flexibility

Stencil: The Stencil concept does not comply with the flexibility requirements defined in section 3.4. The need of connecting complementary connectors introduces artificial constraints in how models can be connected, and requires usage of different models for the same physical object, which only differ by the type or polarity of their connectors. This also imposes constraints on complex model hierarchies, especially ones using advanced language features such as parametric polymorphism. Obviously, in this concept, multiple-way connections require explicit junction models, where the ideal mixing has to be modeled explicitly, i.e., it is not a result of the connection semantics. Therefore, requirement 10 is not fulfilled and the thermo-fluid interface performs poorly in this regard.

Biased mixing volume: This thermo-fluid interface does not fully comply with the flexibility requirement 10 defined in section 3.4. While it utilizes a universal connector that does allow any type of connection in theory, there are some limitations in practice. One is that the connection set should only contain one control volume model, otherwise they are lumped together (see section 4.2.2). A second limitation is that connection sets between two or more flow models without a control volume tend to deteriorate the robustness of this approach (see section 4.2.1.2). In consequence, to fully exploit the “good” robustness of this concept, the alternating use of control and flow models is required and thus the flexibility is limited.

Unbiased mixing volume: The unbiased mixing volume thermo-fluid interface utilizes a universal connector and allows multi-way connections. The connection semantics directly result in appropriate static balance equations for the connection set and explicit junction models are not needed. Therefore, this thermo-fluid interface fulfills requirement 10 and receives a very good score.

4.3 Summary

Altogether, either robustness and efficiency or user friendliness and flexibility have to be selected. This is considered a systematic error and explains the observations described in section 1.5.3.

CHAPTER 5

IMPROVEMENTS AND ALTERNATIVES

The objective of this chapter is to suggest improvements to fulfill the requirements defined in chapter 3 in a more satisfactory manner. Much of the development was triggered in the year 2007 when it became clear to the Modelica community that the existing thermo-fluid interfaces for equation-based, object-oriented modeling languages did not meet the high standards the languages set in other domains. Consistently treating the convected quantities as "effort variable" using the unbiased mixing volume interface as described in section 2.6 was essentially impractical around flow reversal for large-scale modeling. This is the case, because convected properties are not effort variables. The stencil interface in turn resulted in superior robustness but was not physical as it could only be implemented using signal semantics as shown in section 2.4.

This chapter thus discusses key contributions that advanced the state of the art in this regard. As it also contains results from a collaborative effort of the Modelica community[1], it is different to the other chapters and contains a separate section 5.3 on acknowledgments. With the information presented therein, the reader can understand who the contributors are.

5.1 *Stream connectors*

As indicated above, convective transport does not follow the semantics of conventional "effort" or "flow" variables. Thermo-fluid interfaces that attempt to ignore this fact fail as described in chapter 4. A break-through was thus achieved by casting the notion of the stencil, which was very successful with respect to robustness and efficiency in said chapter 4, into the connection semantics of a new third fundamental connector variable, which was eventually called stream variable.

5.1.1 A notional concept for semantics

Connection semantics should ideally be simple and take away possible pitfalls via complicated connector definitions or unnecessary operator arguments. Their definition is not a trivial problem and several iterations were required until Casella proposed [122] a first solution that met the spirit of the requirements for user friendliness in chapter 3. It was later simplified considerably and finalized by Franke et al. [64, 63], who also coined the term "stream variable".

A fluid connector utilizing stream variables is defined as follows. Here, a stream variable is defined for each quantity that is governed by convective transport, i.e., it is carried along via the flow variable declared in the connector, here `m_flow`.

```
connector FluidPort
  Pressure p;
  flow MassFlowRate m_flow;
  stream SpecificEnthalpy h_outflow;
end FluidPort;
```

Listing 22: Stream connector

Each connector containing stream variables is called a stream connector. A stream connector must have exactly one scalar variable with the flow prefix. The idea is that all stream variables of this connector are associated with this flow variable. A stream variable then defines a quantity that is transported by a flow through the stream connector and represents the value of the transported quantity for the case of outflow through the stream connector, i.e.,

[1]This is considered necessary, as the contributions of the author of this thesis in this subject area are closely related to the overall effort.

`m_flow < 0`. Therefore, the variable is declared as `h_outflow` in the exemplary connector. The value under assumption of fluid leaving the connector is, just like when using the stencil connector, known from internal storage or from inflow values of other connectors, independently of the flow direction.

A stream variable does not lead to the generation of any connection equations[2]. Instead, the value under assumption of fluid entering through the connector can be queried in a model on demand by using an operator `inStream()`, e.g., `inStream(h_outflow)`. This operator implements the static balance equation for each convected quantity.

In order to establish a definition of the operator, the approach Casella proposed to formulate the ideal mixing equations in the context of his library using the stencil thermo-fluid interface [29, 30] is reviewed and compared with an obvious alternative.

5.1.2 Efficient handling of ideal mixing

In order to introduce the approach of Casella [29, 30], consider an ideal junction with three ports (two of A-type and one of B-type).

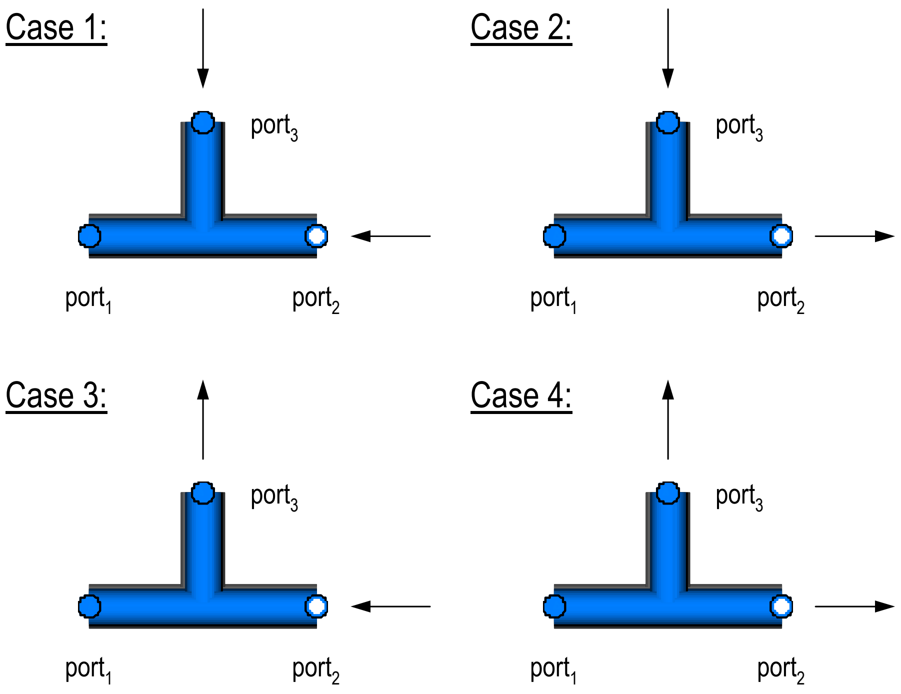


Figure 16: Four cases in computing `port_1.h_a`= $h_1^{\langle a \rangle}$

When computing `port_1.h_a` = $h_1^{\langle a \rangle}$ (i.e., the enthalpy that the flow would have if it was flowing out of the component), there are four possible cases, depending on the signs of the other two flow rates (see figure 16, `port_1` and `port_3` are of A-type).

1. Both other flow rates are entering the junction, so the mixing enthalpy must be computed via a static energy balance. If all the flow rates of the other ports become zero or negative (i.e., going out of the junction component), the mixing enthalpy is undefined. Casella [29, 30] thus proposes to introduce suitable small "epsilon flows" for regularization of this case so that the mixing enthalpy is always well-defined and continuous around zero mass flow.

$$h_1^{\langle a \rangle} = \frac{\max\left(\dot{m}_2, \varepsilon\right) h_2^{\langle a \rangle} + \max\left(\dot{m}_3, \varepsilon\right) h_3^{\langle b \rangle}}{\max\left(\dot{m}_2, \varepsilon\right) + \max\left(\dot{m}_3, \varepsilon\right)}$$

[2]For hierarchical component connections (called "outside" connections) the situation is different and handling is described by Franke et al. [63].

2. If only mass flow rate $\dot{m}_3$ is positive, then its specific enthalpy solely determines the mixing enthalpy

$$h_1^{\langle a \rangle} = \frac{\max(\dot{m}_3, \varepsilon)\, h_3^{\langle b \rangle}}{\max(\dot{m}_3, \varepsilon)}$$

3. If only mass flow rate $\dot{m}_2$ is positive then the equation for specific enthalpy at `port_1` under assumption of mass flow entering the latter is similar to that for case two

$$h_1^{\langle a \rangle} = \frac{\max(\dot{m}_2, \varepsilon)\, h_2^{\langle a \rangle}}{\max(\dot{m}_2, \varepsilon)}$$

4. If none of the flows through ports two and three are entering, the regularization using epsilon flows is used. It results in an arithmetic mean of the specific enthalpies at ports two and three under assumption of fluid entering.

$$h_1^{\langle a \rangle} = \frac{\varepsilon h_2^{\langle a \rangle} + \varepsilon h_3^{\langle b \rangle}}{\varepsilon + \varepsilon}$$

Obviously cases two to four are actually particular cases of the equation for case one, which covers all of them. Notice that the flow via `port_1` does not enter the balance equation. This is motivated by the assumption of negative mass flow via this port for the specific enthalpy `port_1.h_a`. This results in different hypothetical mixing enthalpies for each port and thus a "flow-specific mixing enthalpy".

By using this procedure for the remaining two connectors of the junction, similar equations are derived for the other port-specific enthalpies and the complete junction model is fully defined.

```
model JunctionAAB
   replaceable package Medium = PartialPureSubstance;
   FluidPort_a port_1;
   FluidPort_b port_2;
   FluidPort_a port_3;
   parameter MassFlowRate eps = 1e-8;
   Medium.AbsolutePressure p "Pressure";
equation
  port_1.p = p;
  port_2.p = p;
  port_3.p = p;
  port_1.m_flow + port_2.m_flow + port_3.m_flow = 0;
  port_1.h_a =
    (max(port_2.m_flow,eps)*port_2.h_a +
                              max(port_3.m_flow,eps)*port_3.h_b)/
    (max(port_2.m_flow,eps) +           max(port_3.m_flow,eps));
  port_2.h_b =
    (max(port_1.m_flow,eps)*port_1.h_b +
                              max(port_3.m_flow,eps)*port_3.h_b)/
    (max(port_1.m_flow,eps) +           max(port_3.m_flow,eps));
  port_3.h_a =
    (max(port_1.m_flow,eps)*port_1.h_b +
                              max(port_2.m_flow,eps)*port_2.h_a)/
    (max(port_1.m_flow,eps) +           max(port_2.m_flow,eps));
end JunctionAAB;
```

Listing 23: Static junction model, Stencil interface

These are the ideal mixing equations for the stencil interface of Casella [29, 30]. As they are not the only option to express ideal mixing for stencil-type thermo-fluid interfaces they are taken as starting point for an analysis of different options to define the operator to be used with stream connectors.

The following analysis is concerned with the algebraic equation system that different potential operator semantics imply for ideal mixing and considers two alternatives. Particular emphasis is put on the reductions in the dimension of the equation system that a particular substitution technique called tearing [57, 33, 25] can yield ideally. This is of interest, as a correlation is assumed between robustness and efficiency on one hand and the dimension of the nonlinear algebraic equation system on the other.

5.1.2.1 Candidate formulations

Based on the formulation for the stencil thermo-fluid interface above, the actual specific mixing enthalpy can be established via a static energy balance. It can be shown to obey the following equation [63, appendix A1].

$$h_{mix} = \frac{\sum_{j=1}^{n} h_j \cdot \max\left(-\dot{m}_j, 0\right)}{\sum_{j=1}^{n} \max\left(-\dot{m}_j, 0\right)} \tag{27}$$

A first option to define the `instream()` operator is such that it approximates the properties after a hypothetical flow reversal by the actual mixing properties. Such semantics neglect the effect of any hypothetical flow reversal but result in a “common” mixing enthalpy. Informally, we describe this first option (“method 1”) using (27) as follows.

```
instream(a.port.h_outflow)
 = instream(b.port.h_outflow)
 = instream(c.port.h_outflow)
 = h_mix
```

The approach postulated by Casella in turn is to set the convected properties after a hypothetical flow reversal equal to the mixing properties minus the “own contribution” (see section 2.4). This is close to the notional meaning of these connector variables but results in different “flow-specific” mixing enthalpies. Informally, this second option (“method 2”) is thus described as follows.

```
instream(a.port.h_outflow) = h_mix(a.port.m_flow=0)
instream(b.port.h_outflow) = h_mix(b.port.m_flow=0)
instream(c.port.h_outflow) = h_mix(c.port.m_flow=0)
```

Using equation (27), these expressions can be expanded. The result is summarized in the following listing.

```
instream(a.port.h_outflow) =
    (max(-b.port.m_flow,0)*b.port.h_outflow +
                 max(-c.port.m_flow,0)*c.port.h_outflow) /
    (max(-b.port.m_flow,0) + max(-c.port.m_flow,0))
instream(b.port.h_outflow) =
    (max(-a.port.m_flow,0)*a.port.h_outflow +
                 max(-c.port.m_flow,0)*c.port.h_outflow) /
    (max(-a.port.m_flow,0) + max(-c.port.m_flow,0))
instream(c.port.h_outflow) =
    (max(-a.port.m_flow,0)*a.port.h_outflow +
                 max(-b.port.m_flow,0)*b.port.h_outflow) /
    (max(-a.port.m_flow,0) + max(-b.port.m_flow,0))
```

Listing 24: Expanded candidate formulation 2

Note that this is not an academic problem. The two formulations were not only implemented in the FluidSandbox library for analysis but even later in the commercial Modelica translator Dymola® 7.0[3] to allow other people to consider the trade-off between both formulations.

5.1.2.2 Reference problem

In order to identify discontinuities in potential iteration variables of the nonlinear equation system or residuals and to assess tearing results, a reference ideal mixing point problem is analyzed for both alternatives. Three pipe models using a detailed pipe friction correlation for the laminar and turbulent flow regimes are joined in a single static mixing point. Thermodynamic properties are established based on equations for dry air. The topology is shown in figure 17.

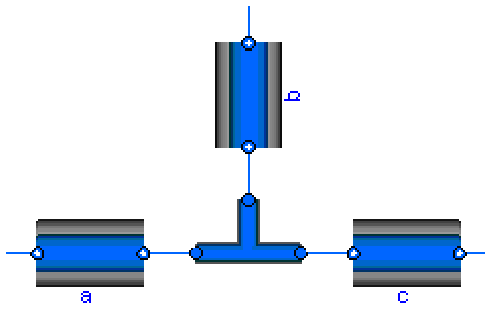


Figure 17: Ideal mixing reference problem

5.1.2.3 Size of the nonlinear equation system

For now, only single-substance media are considered and the minimum size of the nonlinear coupled algebraic equation system after symbolic reduction is established. For the common mixing enthalpy (method 1), the algebraic equation system that is created for the model shown above can theoretically be as small as two, which is independent of the number of flow components connected together in the ideal junction.

In this case, the pressure and the mixing enthalpy h_{mix} in the connection point are used as iteration variables for the connection point of n flow models. It is then possible to compute all `inStream()`-operator return values, then all densities and dynamic viscosities in the flow model components and finally all mass flow rates. The two residue equations are the equation for the mixing enthalpy and the equation for the mass balance.

In general, the two variables will only be the pressure (or a single Δp) and specific enthalpy if a thermodynamic property model with the independent variables pressure and specific enthalpy is used. If, for example, an ideal gas using pressure and temperature as independent variables is contained in the model, a Modelica translator will usually choose pressure (or a single Δp) and temperature in the mixing point as iteration variables.

For the flow-specific mixing enthalpy (method 2), it is theoretically possible to solve the algebraic system of equations created for the model shown above with three iteration variables: Two mass flow rates and the pressure in the mixing point. If two mass flow rates are given, the third mass flow rate can be computed via the mass balance. The equations for the mixing enthalpies depend all on `h_outflow`'s of the flow model components, i.e., of the (potentially transformed) control volume state variables not shown in figure 17. Therefore the `inStream()`-operator return values can be computed, once the mass flow rates are known. With the pressure in the mixing point, all pressure differences can be computed. With the pressure differences, together with all specific enthalpies, the mass flow rates can be computed from the pressure drop equations for all three flow models. These are the three residue equations for the nonlinear system of equations.

[3]In March 2008, Sven-Erik Mattsson implemented both forms for the now obsolete `upstream()` operator.

The given number of iteration variables (in the general case equal to n, the number of flow models connected in an ideal mixing point) is the minimum, which is only obtained if a thermodynamic property model with independent variables pressure and specific enthalpy is used. Only in this case the specific mixing enthalpies computed by the mass flow rates can be used directly to compute the densities and dynamic viscosities used in the pipe models. If, for example, an ideal gas with independent variables pressure and temperature is used, a Modelica translator will typically recognize that n property inversions would be necessary to obtain these quantities from the specific mixing enthalpies and therefore choose to use a single pressure (or Δp) and n temperatures as iteration variables.

In order to enforce the tearing of the equation system described in the penultimate paragraph also for thermodynamic property models using pressure and temperature as independent variables, flow models can be rewritten using pure function calls such as `density_ph()` to compute the density. Then, a local nonlinear equation system will be solved inside these functions. In general, also the viscosity must be computed and it is thus not meaningful to use pure function calls, as they required the inversion to compute the state (p, T) from (p, h) twice. This can be avoided by first computing the thermodynamic state record:

```
state = setState_phX(port_b.p, inStream(port_b.h_outflow));
```

For the moment, the situation can be summarized as follows. For the common mixing enthalpy (method 1) and general n way ideal mixing of single-substance media, the coupled nonlinear algebraic equation set can be reduced symbolically to a system of two residual equations and several forward evaluations. These two variables depend on the independent variables of the model of the thermodynamic properties. Typical examples are pressure (or Δp) and the mixing enthalpy h_{mix} for thermodynamic properties based on (p, h), and pressure (or a Δp) and the mixing *temperature* T_{mix} for thermodynamic properties based on (p, T).

For the flow-specific mixing enthalpy (method 2) and general n way ideal mixing of single-substance media using (p, h) as independent variables, the coupled nonlinear algebraic equation set can be reduced symbolically to a system of n residual equations and several forward evaluations. The iteration variables are $n - 1$ mass flow rates and the pressure in the mixing point (or a Δp). When using thermodynamic property models with independent variables (p, T) in turn, the problem has at $n + 1$ iteration variables and residuals. These are the mixing point pressure (or a Δp) and n temperatures (one per `inStream()`-operator). Alternatively, the model equations can be modified to yield $\{n, 1, \ldots, 1\}$ nonlinear equation systems. This is one equal to the equation system for single-substance media using (p, h) as independent variables and n property inversions from h to T using `setState_phX()` calls.

5.1.2.4 Characteristics of potential iteration variables

Arguing strictly from the size of the reduced equation system, the common mixing enthalpy (method 1) appears to be superior. However, the iteration variables pressure, mass flow rate, specific enthalpy, and temperature have different characteristics and these shall be considered additionally to the dimension of the equation system in order to arrive at a robust and efficient formulation.

Specific enthalpy and temperature as iteration variable: To illustrate the characteristics of specific enthalpy and temperature as iteration variables, the two types of flow reversal in a three-way ideal junction are considered. The first one is labeled "flow reversal in a regular junction" as there are two and one entering flows (both before and after flow reversal). In particular, mass flows enter the junction through two pipes a and b. Following continuity, the mass flow rate through pipe c has to be out of the junction (i.e., negative). At the time instance of interest, the flow through pipe a changes sign while the flow directions of pipe b and c remain unchanged.

In this case the common mixing enthalpy (method 1) reduces to the following expressions before flow reversal.

```
instream(a.port.h_outflow) = h_mix
 = (max(-a.port.m_flow,0)*a.port.h_outflow +
          max(-b.port.m_flow,0)*b.port.h_outflow) /
         (max(-a.port.m_flow,0) + max(-b.port.m_flow,0))
```

After flow reversal the common mixing enthalpy reduces to a simple expression.

```
instream(a.port.h_outflow) = h_mix = b.port.h_outflow
```

Without regularization, this expression is discontinuous in the first derivative as shown in figure 18.

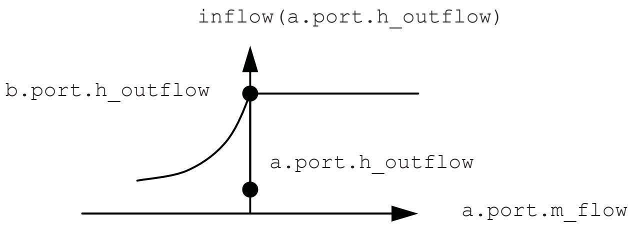


Figure 18: Common mixing enthalpy for case 1

Additionally to flow reversal in a regular junction, the second case of "flow reversal in a degenerate junction" is considered. In this case, flow reversal with strictly one entering flow is studied. In particular, mass enters the junction only through one pipe a. The mass flow through pipe b is out of the junction (negative). To construct a case that is different from case 1, zero flow through pipe c has to be considered (for non-zero flow through pipe c the setup corresponds to case 1). Therefore, a degenerate junction is considered, which also frequently occurs in practice. At the time instance of interest, the flows through pipe a and b change sign (while `c.m_flow=0`). After flow reversal only flow through pipe b enters the junction.

In this case, manipulation of the common mixing enthalpy (method 1) results in the following expression before flow reversal.

```
inflow(a.port.h_outflow) = h_mix = a.port.h_outflow
```

After flow reversal the common mixing enthalpy reduces to the following expression.

```
inflow(a.port.h_outflow) = h_mix = b.port.h_outflow
```

As shown in figure 19 this expression is discontinuous at flow reversal.

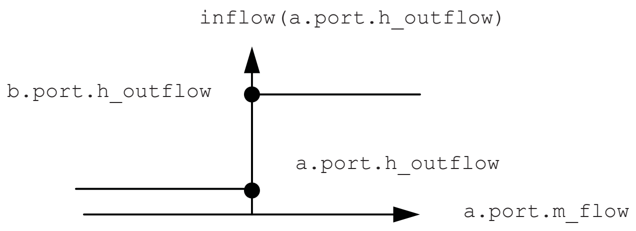


Figure 19: Common mixing enthalpy for case 2

When using the flow-specific mixing enthalpy (method 2) the results are as follows. Again, consider case 1 first (regular junction). The mass flows enter the junction through two pipes a and b. Following continuity, the mass flow through pipe c has to be out of the junction (negative). At the time instance of interest, the flow through pipe a changes sign. The flow directions of pipes b and c remain unchanged. In this case, the following expression holds before flow reversal.

```
instream(a.port.h_outflow) = h_mix(a.port.m_flow=0) = b.port.h_outflow
```

After flow reversal, the flow-specific mixing enthalpy reduces to the same expression.

```
instream(a.port.h_outflow) = h_mix(a.port.m_flow=0) = b.port.h_outflow
```

This expression is continuous as shown in figure 20. Therefore, for this relevant case, the nonlinear equation system is continuous and differentiable in the critical point whereas for the common mixing enthalpy (method 1), the nonlinear equation system is continuous but no longer differentiable. In most cases, a nonlinear solver will therefore have less difficulties to compute the solution when using the flow-specific mixing enthalpy (method 2) instead of the common mixing enthalpy (method 1).

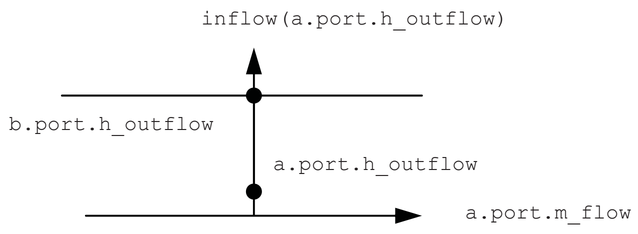


Figure 20: Flow-specific mixing enthalpy for case 1

For case 2 (flow reversal in a degenerate junction) and the flow-specific mixing enthalpy (method 2), the following expression holds before flow reversal.

```
instream(a.port.h_outflow) = h_mix(a.port.m_flow=0) =
(max(-b.port.m_flow,0)*b.port.h_outflow +
  max(-c.port.m_flow,0)*c.port.h_outflow)
/ (max(-b.port.m_flow,0) + max(-c.port.m_flow,0))
```

To avoid division by zero the 0/0 term is approximated by the average of the two `h_outflows` (using epsilon flows). Therefore, the following equation holds.

```
instream(a.port.h_outflow) = (b.port.h_outflow + c.port.h_outflow)/2
```

After flow reversal, this results in

```
instream(a.port.h_outflow) = h_mix = b.port.h_outflow
```

This expression is discontinuous as shown in figure 21. If the deviation of `c.port.h_outflow` from `b.port.h_outflow` is less than *twice* that of `a.port.h_outflow` from `b.port.h_outflow` then the discontinuity is smaller than for the common mixing enthalpy (method 1). This case is depicted in figure 21.

Mass flow rate as iteration variable: Mass flow rate $\dot{m}$ is assumed to be a more well-behaved iteration variable than temperature T or specific enthalpy h of inflowing fluid, because a discontinuity occurs at $\dot{m} = 0$ in convected quantities such as T and h only. If mass flow rate is computed via $\dot{m} = f(\Delta p, \rho_a, \rho_b, \eta_a, \eta_b)$, then $\dot{m}$ is continuous[4] at this point, even if ρ_a, ρ_b, η_a, η_b change discontinuously. Stated differently, if T and h are plotted over time, discontinuities can be present. However, when plotting mass flow rate over time, it is always continuous. A

[4]Depending on the quality of regularization, mass flow rate and pressure may also be continuously differentiable at flow reversal.

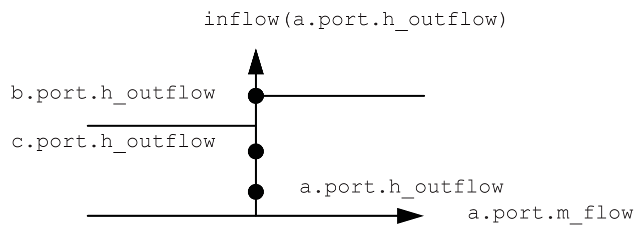


Figure 21: Flow-specific mixing enthalpy for case 2

nonlinear solver will have less difficulties to solve the nonlinear system over time, if all iteration variables are continuous over time, instead of having discontinuities. Therefore, mass flow rates are preferred iteration variables.

Pressure as iteration variable: Pressure or pressure difference Δp are included in either formulation and are thus not optional iteration variables for ideal mixing as analyzed herein. Fortunately, however, a similar argument as the one on mass flow rate in the last paragraph leads to the conclusion that pressure is also continuous over flow reversal.

Taking these results into consideration, it is concluded that the dimension of the symbolically reduced coupled nonlinear algebraic equation system is not the only metric to consider. For robustness and efficiency, it is also important to consider the characteristics of the iteration variables. In this regard, the flow-specific mixing enthalpy (method 2) leads always to a smoother nonlinear equation system in case of a proper junction (which may be the most often occurring case). In the case of a degenerate junction, both alternatives lead to discontinuous equations and it depends on the situation for which of the alternatives the discontinuity is smaller. However, it would be still better if a thermo-fluid interface did not use either one of these specific enthalpies or temperature but exclusively pressures and mass flow rates, as they are at least continuous in the general case.

5.1.2.5 Characteristics of potential residual equations

Flow reversal in a regular junction: For flow reversal in a regular junction, common mixing enthalpy (method 1), and a model of the thermodynamic properties, which uses pressure and temperature as independent variables, the following problem is posed using FluidSandbox. The nonlinear algebraic equation block is reduced from 16 to two dimensions. Typical remaining iteration variables are

- `a.dp`: Pressure drop over pipe a
- `a.medium_nonDesignDirection.state.T`: Temperature of the medium model instance reflecting the upstream conditions for the (potentially hypothetical) reverted mass flow direction. Via the thermodynamic properties model, h_{mix} can be computed explicitly from these two variables. Therefore, this variable can be understood as a T_{mix}.

A typical set of residual equations as obtained for this formulation is given in the following listing.

```
0 = a.m_flow - massFlowRate_dp(a.dp, ...);
0 = (if pipeFriction1.m_flow > 0 then a.m_flow else 0
      + if -b.m_flow > 0 then -b.m_flow else 0
      + if -c.m_flow > 0 then -c.m_flow else 0)
     * a.medium_nonDesignDirection.state.h
     - (273504*(if pipeFriction1.m_flow > 0 then a.m_flow else 0)
         + 1364510*(if -b.m_flow > 0 then -b.m_flow else 0)
         + 684916*(if -c.m_flow > 0 then -c.m_flow else 0));
```

Listing 25: Torn residual equations for regular junction, common mixing enthalpy

The residual equations include either the mass balance or an equation comparing the pressure difference using the mixing point pressure at the current iterate vs. the wall friction correlation. Above, the latter is shown. The second residual equation is the energy balance.

In the listing, some details were simplified for readability (e.g., use of the `noEvent()` operator is not shown, floating point numbers of the inflowing specific enthalpies are shown without decimal digits). The residual equations can be identified in the figures in this section via their color. The given residual equations are illustrated in orange, blue, green and red going from first to last (the latter two colors are used in figures below only).

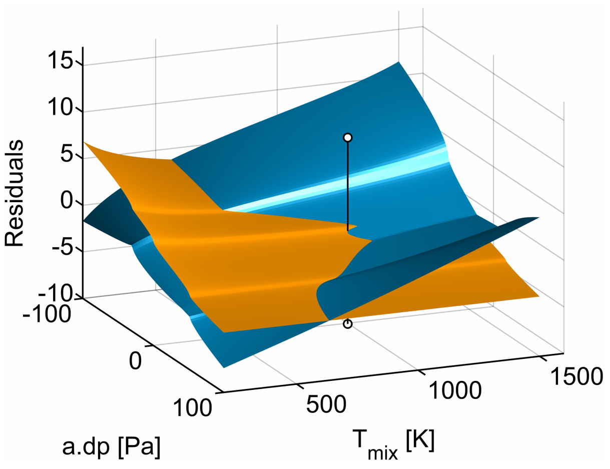


Figure 22: Residuals over iterates for common mixing enthalpy, regular junction, properties explicit in p, T

For this case, the residuals as a function of the iterate values are shown in figure 22. Here, `a.medium_nonDesignDirection.state.T` is abbreviated as T_{mix}. In this and several of the following figures each residual has been scaled differently, i.e., the order of magnitude of the first residual (a mass flow rate residual) is smaller than that of the second residual (a enthalpy flow rate residual) and thus, for figure 22, the first residual was scaled with a factor 1.0 and the second one with a factor $5 \cdot 10^{-6}$. The black vertical line points at the solution of the problem.

For the same problem but the flow-specific mixing enthalpy (method 2) and thermodynamic properties explicit in pressure and specific enthalpy, the equation set of the reference model using FLUIDSANDBOX is reduced from a block of 18 dimensions to one of three dimensions. Typical remaining iteration variables are

- `a.dp`: Pressure drop over pipe a,
- `-b.m_flow`: Mass flow rate through pipe b, and
- `c.m_flow`: Mass flow rate through pipe c.

Typical residual equations are either all equations comparing the pressure difference using the current iterate of the mixing point pressure vs. the wall friction correlations. Or they include two such equations and the mass balance. Then, the third equation of the former type is used in a torn forward assignment. A typical example using the latter option is as follows. Here, only one argument of the wall friction correlation `massFlowRate_dp()` is shown, the difference in total pressure. Usually, densities, dynamic or kinematic viscosity, pipe length, diameter and roughness are additionally required.

```
0 = a.m_flow - b.m_flow - c.m_flow;
0 = b.m_flow - massFlowRate_dp(b.dp, ...);
0 = c.m_flow - massFlowRate_dp(c.dp, ...);
```

Listing 26: Torn residual equations for regular junction, flow-specific mixing enthalpy

The solution space is three dimensional for this problem. In order to visualize the topology of the residuals, two plots are presented with a pair of tearing variables as x and y axes each. The z axis is, as before, the residual. Due to the third independent tearing variable for each plot a value has to be prescribed for that, too. In the figures shown here, it is the value for which the solution of the nonlinear equation system is obtained. That is, in the left plot of figure 23, `c.m_flow` $= -0.44$ and in the right plot `-b.m_flow`$= 0.63$.

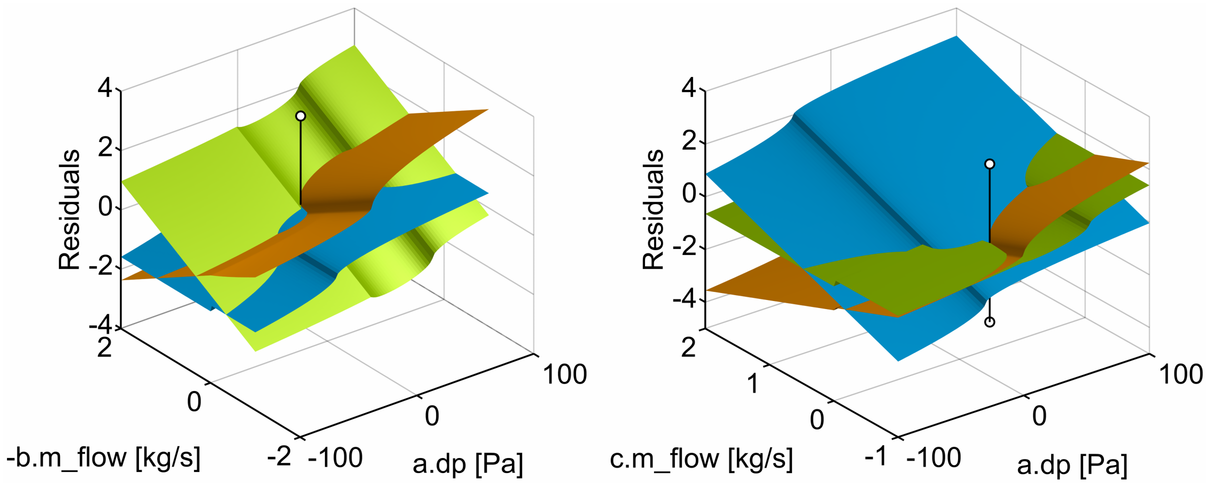


Figure 23: Residuals over iterates for flow-specific mixing enthalpy, regular junction, properties explicit in p, h

Due to the specific problem formulation, the solution to the problem evolves in the quadrant of negative `c.m_flow`. However, due to the increased number of tearing variables, the solution space contains regions (for positive `c.m_flow`), in which the discontinuous characteristics of the degenerate junction case can be observed. For instance, in the right graph of figure 23, the green residual surface for residual 3 is discontinuous for negative `a.dp` (i.e., flow out of the junction through pipe a) based on the following forward substitution.

```
a.m_flow := massFlowRate_dp(a.dp, ...);
```

That is, the `b.m_flow` required to fulfill residual equation 3 changes discontinuously (from one negative value to another, i.e., flow through b out of the junction in both cases) when `c.m_flow` changes sign. Residual equation 3 is the compliance of the mass flow rate vs. pressure loss correlation though pipe b with the tearing variable `b.m_flow`. As both `a.m_flow` and `b.m_flow` are negative (out of the junction) for the flow-specific mixing enthalpy the following equation holds.

```
instream(c.port.h_outflow)=(a.port.h_outflow + b.port.h_outflow)/2
```

Then, the sign of the mass flow rate through pipe c determines whether the boundary condition properties or `instream( c.port.h_outflow)` are returned and the observed discontinuity arises. Obviously, for the present problem (flow reversal in regular junction), the nonlinear equation system does contain quadrants that correspond to flow reversal in the degenerate junction due to the increased number of tearing variables.

For the same problem but thermodynamic properties explicit in pressure and temperature, the equation set of the reference model using FluidSandbox is reduced from a block of 21 dimensions to one of four dimensions. The following is a list of exemplary iteration variables.

- `a.dp`: Pressure drop over pipe a,
- `a.medium_nonDesignDirection.state.T`: Flow specific mixing temperature in pipe a,

- `b.medium_nonDesignDirection.state.T`: Flow specific mixing temperature in pipe b, and
- `c.medium_nonDesignDirection.state.T`: Flow specific mixing temperature in pipe c.

```
0 = a.m_flow + b.m_flow + c.m_flow;
0 = a.medium_nonDesignDirection.h
      - h_T(a.medium_nonDesignDirection.state.T);
0 = b.medium_nonDesignDirection.h
      - h_T(b.medium_nonDesignDirection.state.T);
0 = c.medium_nonDesignDirection.h
      - h_T(c.medium_nonDesignDirection.state.T);
```

The solution space is obviously four dimensional for this problem. The data is visualized at a constant value of `a.medium_nonDesignDirection.state.T` in order to illustrate the topology of the residuals. As fluid always flows though pipe b into the junction and through pipe c out of the junction this variable is always set to the value of the attached dynamic volume or boundary condition, say, `source_b.medium.T`. As before, both plots are based on a pair of tearing variables as x and y axes each. The z axis is the residual for both remaining tearing variables at the solution value.

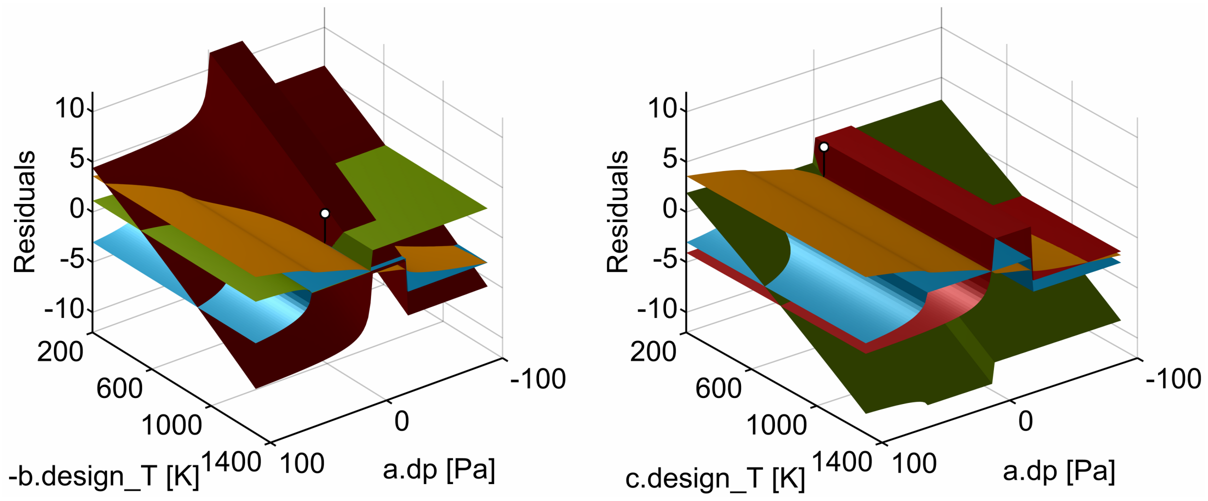


Figure 24: Residuals over iterates for flow-specific mixing enthalpy, regular junction, properties explicit in p, T

The first residual (orange, mass balance) is dominated by the pressure difference over pipe a that drives the mass flow rates and is well behaved. The other three residuals have gradients with respect to the mixing point pressure and one of the remaining tearing variables only. They do have several discontinuities and this formulation should therefore be avoided. In case of thermodynamic properties explicit in, e.g., pressure and temperature, the tearing procedure using $\{n, 1, \ldots, 1\}$ coupled systems described at the end of section 5.1.2.3 should be used instead.

Flow reversal in a degenerate junction: Flow reversal in a degenerate junction, common mixing enthalpy (method 1), and a model of the thermodynamic properties, which uses pressure and specific enthalpy as independent variables[5], results in the following problem when implemented in FluidSandbox. The nonlinear algebraic equation block is again reduced from 16 to two dimensions. Typical remaining iteration variables are

[5]For the common mixing enthalpy, the characteristics of the residuals for the cases of thermodynamic properties being explicit in pressure and temperature on one hand and explicit in pressure and specific enthalpy on the other are equivalent. Therefore, we illustrate the latter case in this example and the former at the beginning of section 5.1.2.5.

- `a.dp`: Pressure drop over pipe a
- `a.medium_nonDesignDirection.state.h`: Common specific mixing enthalpy h_{mix}.

The residual equations are equal to those in the first case analyzed in this section. The forward assignments differ however due to the different iteration variables for the thermodynamic property computations. Figure 25 illustrates the residuals as function of the iteration variables.

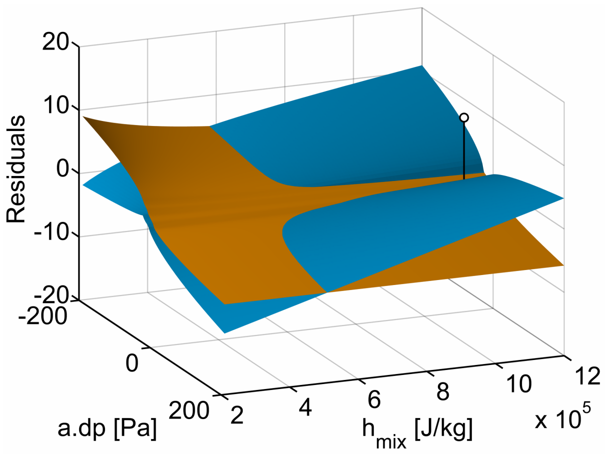


Figure 25: Residuals over iterates for common mixing enthalpy, degenerate junction, properties explicit in p, h

For the flow-specific mixing enthalpy (method 2) and thermodynamic properties explicit in pressure and specific enthalpy, the same residual equations and iteration variables are obtained as before.

The solution space is similar to the one for flow reversal in a regular junction. Again, the results are illustrated using two plots with a pair of tearing variables as x and y axes each. In figure 26, the z axis is, as before, the residual. The third independent tearing variable is set to the value for which the solution of the nonlinear equation system is obtained (these values are `-b.m_flow`= 1.34 for the left graph and `-c.m_flow`= 0.0 for the right one). Note that even though more than one intersection of the residual surfaces can be seen in the figure, only the marked solution is an intersection at zero residual.

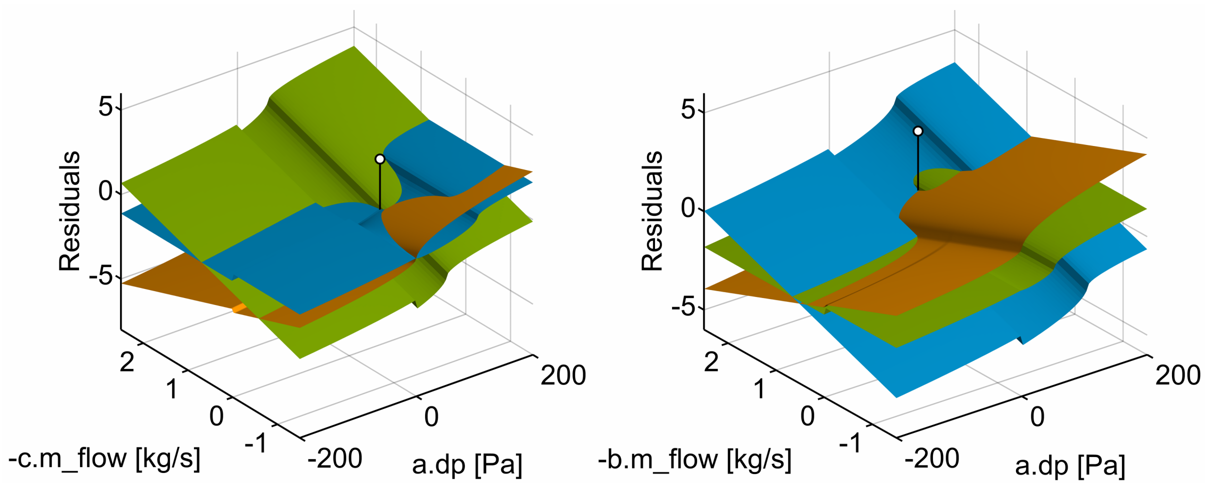


Figure 26: Residuals over iterates for flow-specific mixing enthalpy, degenerate junction, properties explicit in p, h

To summarize, the characteristics of the residual equations depend on the quantity, in which they formulate a residual. Residuals in temperature or specific enthalpy are, just like iteration variables in these quantities, generally less well-behaved (see also discussion in section 5.1.2.4). Note that one residual in figure 22 is enthalpy *flow rate* and thus well-behaved. Residuals in pressure or mass flow rate have usually better characteristics with respect to continuity and differentiability, too.

5.1.2.6 Multiple-substance media

Common mixing enthalpy: It was shown earlier in this section that the size of the nonlinear equation system was two for n connected flow models in an ideal junction for a single substance fluid. This tearing procedure used the pressure p and the mixing enthalpy h_{mix} in the connection point as iteration variables.

To generalize the concept of the common mixing enthalpy (method 1) to multiple substance media, n_{Xi} mixing mass fractions are required analogously to the mixing enthalpy. Based on these quantities all densities and dynamic viscosities in the flow model components can be calculated and then all mass flow rates based on the pressure differences. The residue equations are the equation for the mixing enthalpy, the equation for the mass balance, and the independent substance mass balances. In this case the size of the equation system is $2 + n_{Xi}$.

If the independent mass fraction vector becomes large, an alternative procedure results in a smaller problem, namely using $n - 1$ mass flow rates and the mixing pressure. The upper limit on the size of the equation system is therefore n.

The tearing works as follows: If all but one mass flow rates are known the remaining mass flow rate can be established based on the continuity equation. With all mass flow rates known, the mixing enthalpy and the independent mass fractions in the mixing point can be established from the pipe outflows. Based on this, the densities and dynamic viscosities are calculated for the pipe models, and the mass flow rates can be established according to the pressure loss correlations. The n residual equations are the comparison of the mass flow rate tearing variables with the mass flow rate vs. pressure loss correlations.

For multiple substance fluids and the common mixing enthalpy (method 1), the size of the equation system can consequently be reduced to a minimum of $\min(2 + n_{Xi}, n)$.

Flow-specific mixing enthalpy: Based on the analysis of the discontinuities, the flow-specific mixing enthalpy (method 2) using a coupled system with $n+1$ dimensions was rejected earlier. Consequently, the remaining alternative is the formulation using n tearing variables and, in case of thermodynamic property computations explicit in other variables than pressure and specific enthalpy, $\{n, 1, \ldots, 1\}$ coupled systems.

Recalling the tearing procedure suggested for this formulation, it immediately becomes obvious that the size of the algebraic loop is n, independently of the number of independent mass fractions n_{Xi}. Furthermore, the procedure is identical to the one alternatively introduced for the common mixing enthalpy (method 1). $n - 1$ mass flow rates and the pressure in the mixing point are utilized as tearing variables. The remaining mass flow rate can be computed via the mass balance. The flow specific mixing enthalpies are calculated differently but have similar dependencies and can be computed. With the pressure in the junction and the thermodynamic properties based on the mixing properties, all pressure differences and mass flow rates can be computed. These are the n residue equations for the nonlinear system of equations.

For multiple substance fluids and the flow-specific mixing enthalpy (method 2), the size of the equation system is consequently n.

5.1.2.7 Conclusions

The common mixing enthalpy (method 1) and the flow-specific enthalpy formulation (method 2) resulting in either a single equation system of dimension n or several systems of size $\{n, 1, \ldots, 1\}$ are the two alternatives still considered at this point. The flow-specific mixing enthalpy (method 2) using a coupled system with $n + 1$ dimensions was rejected earlier. The two remaining alternative compare as follows.

Based on the previous sections, the reasoning for these scores and additional differences between the two remaining alternatives are discussed.

Criterion	Common	Flow-spec. n	Flow spec. $n+1$
Problem dimension	++	+	-
Char. it. variables	--	++	-
- Regular junction	-	++	+
- Degenerate junction	--	++	--
Char. it. residuals	+	+	--

Table 2: Evaluation of ideal mixing formulations

- In several cases, the size of the equation system is smaller for the common mixing enthalpy (method 1), i.e., for $n > 2+n_{Xi}$. The most relevant case is probably $n = 3$ and $n_{Xi} = 0$. In this case, the common mixing enthalpy (method 1) results in a system of two dimensions while the flow-specific mixing enthalpy (method 2) requires three iteration variables.
- The iteration variables are discontinuous for the common mixing enthalpy (method 1) and continuous or continuously differentiable for the flow-specific mixing enthalpy (method 2). This is the case because the pressure in mixing point and mass flow rates are used as iteration variables.
- For mass flow reversal in a degenerate junction, the approximation of the `inStream()` specific enthalpy using the flow specific mixing enthalpy is superior than for the common mixing enthalpy (method 1).
  `inStream(a.port.h_outflow) = (b.port.h_outflow + c.port.h_outflow)/2` vs.
  `inStream(a.port.h_outflow) = a.port.h_outflow`
- The discontinuities and nonlinearities of the residual surfaces are comparable. It has, however, to be stated that close to the solution point, the flow-specific mixing enthalpy (method 2) is smooth (for the regular junction) whereas the common mixing enthalpy (method 1) can be continuous with discontinuous first derivative.
- For the calculation of the thermodynamic properties in the flow models attached to the junction, the common mixing enthalpy (method 1) utilizes one state record while the flow-specific mixing enthalpy (method 2) requires n state records (the flow-specific mixing enthalpies are different for each branch). Each of them may lead to one local nonlinear equation system.

As a result, the *flow-specific mixing enthalpy* (method 2) was judged more favorable. Based on this analysis, the semantics of the `inStream()`-operator were defined.

During implementation, it is important to write flow models using pure function calls. First, the thermodynamic state is computed (potentially requiring the solution of a local nonlinear equation system in case of thermodynamic properties explicit in, e.g., pressure and temperature). A notional example is given next.

```
model FM
  replaceable package Medium = PartialMedium;

  // ...

equation
  port_a.h_outflow     = inStream(port_b.h_outflow);
  port_b.h_outflow     = inStream(port_a.h_outflow);
  port_a_state_inflow = Medium.setState_phX(
    port_a.p, inStream(port_a.h_outflow),...);
  port_b_state_inflow = Medium.setState_phX(
    port_b.p, inStream(port_b.h_outflow),...);
  d_a_inflow = Medium.density(port_a_state_inflow);
  d_b_inflow = Medium.density(port_b_state_inflow);

  // ...

end FM;
```

Listing 27: Component model using function calls

Then, the necessary properties are established from the state record (dynamic viscosity, density etc.). Note that because the `setState_phX()` function implies a specific causality the state record variables will not appear as iteration variables of the nonlinear equation system. The given function solves internally a nonlinear equation system using Brent's algorithm [22], which is fast and reliable.

5.1.3 Stream connector semantics

In this section, the `inStream()` and other related operators are defined. Also, suggested regularizations of the `inStream()`-operator are described. The information presented in this section is based on the work by Franke et al. [64, 63].

Following the analysis presented in section 5.1.2, the `inStream()`-operator was supposed to implement the flow-specific mixing enthalpy (method 2). Using the original regularization of Casella [29, 30], the operator is thus defined as follows.

$$\text{inStream}(h_i) := \frac{\sum\limits_{j=1\dots n, j\neq i} h_j \max(-\dot{m}_j, \varepsilon)}{\sum\limits_{j=1\dots n, j\neq i} \max(-\dot{m}_j, \varepsilon)} \tag{28}$$

Note the condition $j \neq i$ in the summation in contrast to (27). ε are the epsilon flows introduced by Casella and described in section 5.1.2. In case of one-to-one connections and series of flow models, this definition leads to propagation of specific enthalpy variables as the stencil thermo-fluid interface. Additional provisions are taken to handle sensors and uni-directional flow efficiently. This works by setting the `min` and `max` attributes on the flow variable of the stream connector. A Modelica translator can deduce symbolic simplifications from such information. Furthermore, an operator `actualStream()` is introduced to provide access to the actual value of the convected quantity on the stream. It can be used to formulate the enthalpy flow rate across a connector conveniently. This is illustrated in the following code example.

```
FluidPort c;
actualStream(c.h_outflow) :=
  if c.m_flow > 0 then inStream(c.h_outflow)
    else c.h_outflow;
EnthalpyFlowRate H_flow =
  c.m_flow * actualStream(c.h_outflow);
```

Listing 28: Use of the `actualStream()`-operator

As already sketched in section 5.1.2, the `inStream()`-operator return value cannot be uniquely determined in case of zero mass flow rate. An exact version of the `inStream()`-operator without any approximation is illustrated in figure 27. The region with mass flow rates $\dot{m}_1$ and $\dot{m}_2$ zero or positive is left open as the value of the `inStream()`-operator is arbitrary by definition here.

The regularization suggested by Casella using epsilon flows and described in section 5.1.2 in turn is illustrated in figure 28. The use of epsilon flows results in use of the arithmetic mean for the arbitrary region (shown in blue).

The regularization does not result in C^1 or higher continuity; instead, the operator is still discontinuous in its first derivative due to the `max()` operator. Furthermore, the regularization is

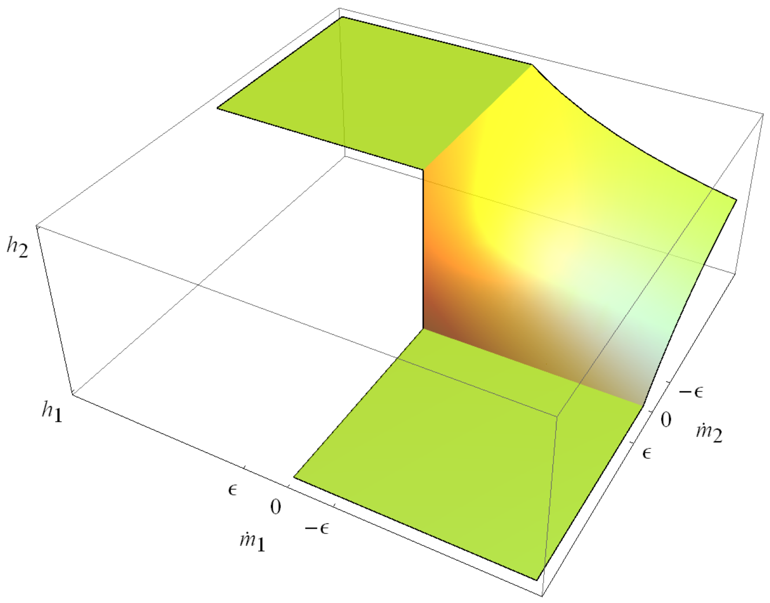


Figure 27: Exact version of the `inStream()`-operator without regularization

obviously not restricted to the ε neighborhood of flow reversal. As can be inferred from figure 28, this formulation does not identically fulfill equation (27) outside an open ε neighborhood around flow reversal. Therefore, an alternative regularization was proposed in [63].

First, the sum of the mass flow rates considered in the `inStream()`-operator when applying it on connector i, σ_i, is introduced.

$$\sigma_i = \sum_{j=1\ldots n, j\neq i} \max(-\dot{m}_j, 0) \tag{29}$$

Then, the simple regularization using $\max(0, \varepsilon)$ in equation (28) is substituted by a custom operator `positiveMax(x,` σ_i`)`, which is defined to always return a positive, non-zero value.

$$\text{inStream}(h_i) := \frac{\sum_{j=1\ldots n, j\neq i} h_j \cdot \text{positiveMax}(-\dot{m}_j, \sigma_i)}{\sum_{j=1\ldots n, j\neq i} \text{positiveMax}(-\dot{m}_j, \sigma_i)} \tag{30}$$

The custom operator `positiveMax(x,` σ_i`)` is defined as a linear combination of $\max(x, 0)$ and $\varepsilon \in \mathbb{R}^+$ along a variable α.

$$\text{positiveMax}\left(x, \sigma_i\right) := \alpha \cdot \max\left(x, 0\right) + (1 - \alpha) \cdot \varepsilon \tag{31}$$

Auxiliary variable α is defined as a C^1 smooth approximation to the step function of σ_i.

$$\alpha\left(\sigma_\text{i}\right) := \begin{cases} 1 & \textit{if } \sigma_i > \varepsilon \\ \left(3 - 2\frac{\sigma_i}{\varepsilon}\right)\left(\frac{\sigma_i}{\varepsilon}\right)^2 & \textit{if } 0 < \sigma_i \leq \varepsilon \\ 0 & \textit{if } \sigma_i \leq 0 \end{cases} \tag{32}$$

Figure 29 shows an illustration of this regularization. Again, the arithmetic mean value for the arbitrary region is shown in blue. Note that outside of the regularization domain points remain, which are not continuously differentiable. This is necessary due to a requirement that the approximation shall be exact whenever the sum of the absolute values of all flow variables is greater than the given small value ε.

One important property of this approach is the propagation of state variables for convected quantities along chains of flow models. This is illustrated in figure 30. The left dynamic volume model prescribes its state variable `medium.h` on its connector `b`. This variable is propagated via explicit algebraic equations through the chain of flow models, independently of whether they

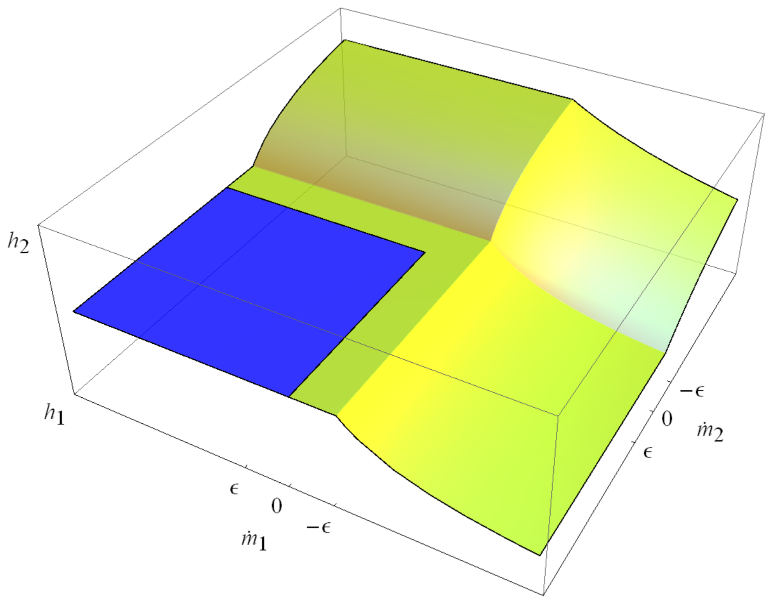


Figure 28: Simple version of `inStream()`-operator regularization

are isenthalpic or not. The specific enthalpy state variable of the right dynamic control volume is propagated similarly in the reverse direction.

5.1.4 Exemplary component models

As the streams thermo-fluid interface is essentially a stencil thermo-fluid interface, the model code when using these interfaces is similar. In this section, examples of such models are given. First, consider the model of a dynamic control volume.

```
model CV
  FluidPort_a port_a;
  FluidPort_b port_b;
  replaceable package Medium = PartialPureSubstance;
  parameter Volume V;
  Medium.BaseProperties medium;
  Mass m;
  Energy U;
  Power H_flow_a, H_flow_b;
equation
  port_a.p = medium.p;
  port_a.h_outflow = medium.h;
  port_b.p = medium.p;
  port_b.h_outflow = medium.h;
  m = V*medium.d;
  U = m*medium.u;
  der(m) = port_a.m_flow + port_b.m_flow;
  H_flow_a = port_a.m_flow * actualStream(port_a.h_outflow);
  H_flow_b = port_b.m_flow * actualStream(port_b.h_outflow);
  der(U) = H_flow_a + H_flow_b;
end CV;
```

Listing 29: Dynamic control volume model, Streams interface

Here, `port_a.h_outflow` and `port_b.h_outflow` are both set to equal `medium.h` permanently. The reason is that the specific enthalpy under the assumption of mass flowing out

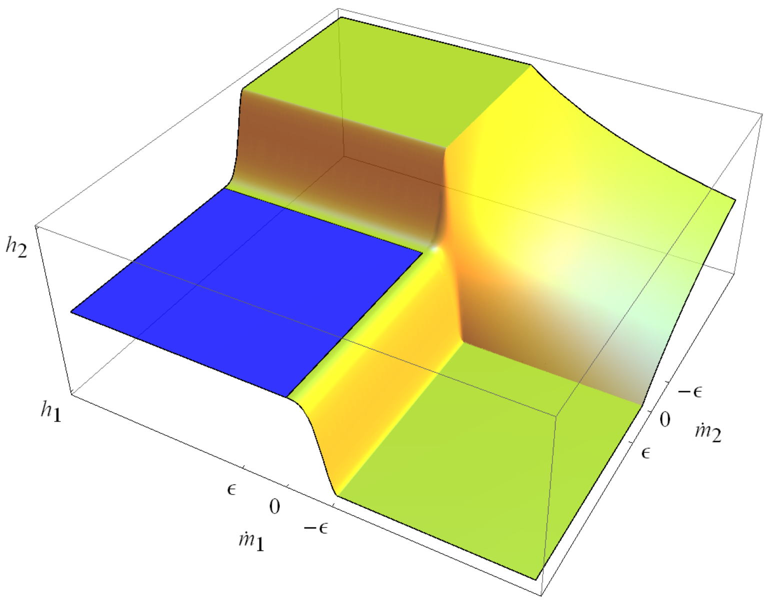


Figure 29: Recommended version of `inStream()`-operator regularization

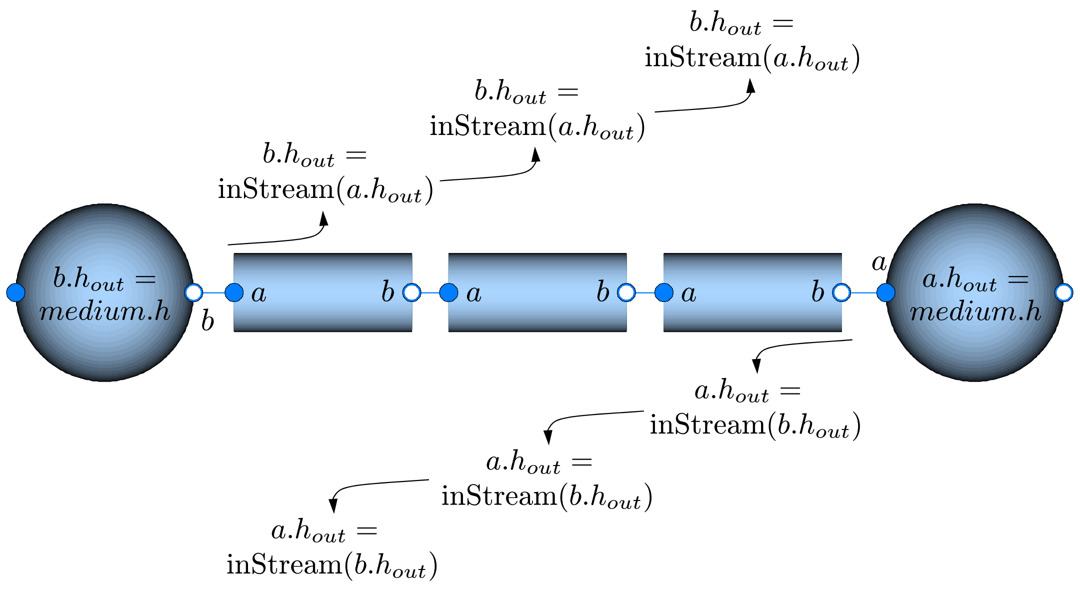
of the connectors `h_outflow` is indeed `medium.h` independently of the actual mass flow rates `port_a.m_flow` and `port_b.m_flow`.

Additionally, consider the isenthalpic flow model with a detailed pipe friction correlation based on the upstream density ρ_{up} and dynamic viscosity η_{up}. To allow for smoothing of the wall friction correlation at flow reversal, both potential upstream properties are required. Assuming that the flow model is connected to a control volume on each side, the model is written using a function `f(dp, rho_a, rho_b)` and the densities on both sides in order to implement a smooth transition between positive flow and negative flow. The code of this flow model is the following.

```
model FM_pipeFriction
  replaceable package Medium = PartialPureSubstance;
  FluidPort_a port_a,
  FluidPort_b port_b;
  Density rho_a, rho_b;
  Pressure dp;
equation
  port_a.m_flow + port_b.m_flow = 0;
  // Energy balance in design direction
  port_b.h_outflow = inStream(port_a.h_outflow);
  // ... in non-design direction
  port_a.h_outflow = inStream(port_b.h_outflow);
  rho_a = Medium.density_ph(port_a.p, inStream(port_a.h_outflow));
  rho_b = Medium.density_ph(port_b.p, inStream(port_b.h_outflow));
  dp = port_a.p - port_b.p;
  port_a.m_flow = f(dp, rho_a, rho_b); // m_flow vs. dp correlation
end FM_pipeFriction;
```

Listing 30: Isenthalpic flow model, Streams interface

If the fluid flows in design direction, then the specific enthalpy of fluid leaving the component through `port_b` equals `inStream(port_a.h_outflow)`. Non-isenthalpic flow models are implemented similarly. Consider a basic isentropic component with an extra mechanical shaft to account for the extracted mechanical power.



$b.h_{out} =$ inStream$(a.h_{out})$
$b.h_{out} =$ inStream$(a.h_{out})$
$b.h_{out} =$ inStream$(a.h_{out})$
$b.h_{out} =$ $medium.h$
b
a
b
a
b
a
b
a
$a.h_{out} =$ $medium.h$
$a.h_{out} =$ inStream$(b.h_{out})$
$a.h_{out} =$ inStream$(b.h_{out})$
$a.h_{out} =$ inStream$(b.h_{out})$

Figure 30: Forward and backward propagation of specific enthalpy along a chain of flow models

```
model FM_stodolaTurbine
  replaceable package Medium = PartialPureSubstance;
  FluidPort_a port_a,
  FluidPort_b port_b;
  Temperature T_a;
  Modelica.Mechanical.Rotational.Interfaces.Flange_a shaft;
  Power P_mechanical;
  Power H_flow_a, H_flow_b;
  parameter Real K_t = 0.5 "Stodola turbine constant";
equation
  port_a.m_flow + port_b.m_flow = 0;
  port_a.h_outflow = Medium.isentropicEnthalpy(port_a.p,
    Medium.setState_ph(port_b.p, inStream(port_b.h_outflow)));
  port_b.h_outflow = Medium.isentropicEnthalpy(port_b.p,
    Medium.setState_ph(port_a.p, inStream(port_a.h_outflow)));
  T_a = Medium.density_ph(port_a.p, inStream(port_a.h_outflow));
  P_mechanical = shaft.tau * der(shaft.phi);
  H_flow_a = port_a.m_flow*actualStream(port_a.h_outflow);
  H_flow_b = port_b.m_flow*actualStream(port_b.h_outflow);
  port_a.m_flow = K_t * ((port_a.p^2 - port_b.p ^2)/T_a)^0.5;
  P_mechanical + H_flow_a + H_flow_b = 0;
end FM_stodolaTurbine;
```

Listing 31: Non-isenthalpic flow model, Streams interface

The specific enthalpies are transformed according to the thermodynamic processes (here isentropically) and set to equal the respective interface variables. Again, the equations involving the sign of the mass flow rate are only used to establish output variables. Note that the code is exemplary and could be made more efficient using intermediate thermodynamic state records.

5.2 Fluid property inversion at connection points

In order to avoid unnecessary fluid property inversions without resorting to overdetermined connectors, an annotation was introduced in Modelica 3.1 to define inverses of functions. The following listing gives an example of use for a thermodynamic property model using pressure and temperature as independent variables.

```
function h_pT
  input  Real p "Pressure";
  input  Real T "Temperature";
  output Real h "Specific enthalpy";
algorithm
  // ...
end h_pT;

function T_ph
  input  Real p "Pressure";
  input  Real h "Specific enthalpy";
  output Real T "Temperature";
  annotation(inverse(h = h_pT(p,T));
algorithm
  // ...
end T_ph;
```

Listing 32: Inverse function annotation for thermodynamic properties

For such a thermodynamic property model, the specific enthalpy h is computed explicitly via a function `h_pT()`. Additionally, the inverse function `T_ph()` is needed in the general case, which will in many cases require a call to a nonlinear algebraic equation solver in order to invert function `h_pT()`. If function `T_ph()` has the annotation `inverse(h = h_pT(p,T))` then the modeler states that an inverse of `T_ph()` is function `h_pT`. The essential requirement is that the inverse function must have the same input and output arguments as the direct function, but with possibly permuted order. Especially, a variable that was declared as `input`, is used as `output` in the inverse function.

This allows to avoid property inversions not only when using the streams thermo-fluid interface but also when using any other thermo-fluid interface.

5.3 Acknowledgments

In 2002, Casella [29, 30] suggested the stencil thermo-fluid interface with the ThermoPower library. This is the foundation of the streams thermo-fluid interface. With the evolution of the library, Casella suggested the corresponding formulation of ideal mixing with epsilon flow regularization, which became the basis of the `inStream()`-operator. Also in 2002, the unbiased mixing volume thermo-fluid interface was proposed by Elmqvist [58]. Together with the Modelica_Fluid library, it was refined until 2008 but the interface never became sufficiently reliable. In January 2008, Elmqvist proposed a new operator (for an overview, see [122]), which eventually triggered further development and thinking outside the box. Several thermo-fluid interfaces were considered. Most notably, Casella [122] proposed simpler semantics for an interface based on the stencil thermo-fluid interface in January 2008. It used two operators. In March 2008 it was simplified considerably by Franke et al. [64], who proposed to use one essential operator only and coined the term "stream variable" based on a new prefix in connector definitions. These results were presented in section 5.1.1.

Beyond the definition of improved semantics, the thermo-fluid interface included three relevant improvements, which are all related to the numerical aspects. The first one was a reformulation to improve the numerical aspects of ideal mixing, which was contributed by Sielemann and Otter. A summary was provided in [63] already but the details were laid out in section 5.1.2. The second and third relevant improvements were an improved regularization of the `inStream()`-operator and the inverse function annotation, which were both proposed by Otter. They were presented in sections 5.1.3 and 5.2 respectively. All these analyses were established around April 2008.

Test implementations were developed by Mattsson in Dymola® and by Sielemann in FluidSandbox. Franke and Sielemann tested the concept and contributed to the test implementations. Olsson contributed an improved definition of inside and outside connectors. Otter introduced stream connectors to the Modelica_Fluid library.

Sielemann evaluated the streams thermo-fluid interface against established interfaces using a rigorous set of requirements. The results of this comparison justified the introduction of this thermo-fluid interface. The requirements on and comparison of thermo-fluid interfaces were given in detail in chapters 3, 4, and 6.

In May 2008, a final proposal was developed by the Modelica Fluid Working Group based on these contributions at the 57^{th} Modelica Design Meeting. At this Design Meeting, Modelica Association members voted for acceptance of this proposal and inclusion of the concept in the Modelica language specification version 3.1. The results were summarized and put into context by Franke et al. [63].

Note that [124] contains contributions by Sielemann, which have been included in chapters 3 and 4 in refined form. All illustrations shown in this chapter were created by Sielemann.

CHAPTER 6

FINAL EVALUATION OF INTERFACES

The objective of this chapter is to give evidence how the improvements suggested in chapter 5 lead to a superior non-causal thermo-fluid interface with respect to the requirements given in chapter 3. The methods and tools described in section 4.1 are re-used and applied to the streams thermo-fluid interface.

Criterion	Biased	Stencil	Unbiased	Streams
Robustness and efficiency	+	+	-	++
- Computational time	++	++	-	++
- Zero and reversing flow	+	++	--	++
- Property inversion	++	++	++	++
- Simplifications	+	+	+	+
- Ideal mixing	-	++	-	++
User fr., end-user	-	-	+	+
User fr., developer	+	-	+	+
Flexibility	-	-	++	++

Table 3: Final evaluation of thermo-fluid interfaces

Notably, the scores with respect to two metrics and requirements changed for established thermo-fluid interfaces in comparison to table 1. This is the case for “ideal mixing” and “property inversion”. The changed scores are due to the contributions presented in sections 5.1.2 and 5.2. Their application to other thermo-fluid interfaces than streams is discussed below.

6.1 Robustness and efficiency

6.1.1 Computational time

As before, reproducibility of the simulation timings was high. Still, each reference model was run ten times per thermo-fluid interface to establish the required computational time, and an average was established over all ten cases per interface.

Figure 31 shows the required computational time for the reference model “relaxation transient”. Here, the required computational time when using the streams thermo-fluid interface is also shown. As before, the figure shows how much computational time (ordinate) was required per simulation time (abscissa). Figures 32 and 33 show the required computational time for the reference models “shut-down” and “change in set-point” respectively.

As before, the total required computational time is of particular interest. The required computational time for models using the streams thermo-fluid interface is nearly identical to that of models using the biased mixing volume and stencil thermo-fluid interfaces. The unbiased mixing volume thermo-fluid interface in turn requires nearly twice the computational time. This is why, based on requirement 1, the thermo-fluid interfaces using the streams, stencil and the biased mixing volume were given a very good score and the unbiased mixing volume was given a poor score.

6.1.2 Zero and reversing flow

The streams interface is a type of stencil interface. Therefore, by construction, it allows propagation of both upstream and downstream convected quantities and thus fulfills requirements 2 and 3. Like the stencil interface, the approach results in high robustness and efficiency. The thermo-fluid interface also fulfills requirement 4 for continuous residuals and iterates and thus receives a very good score.

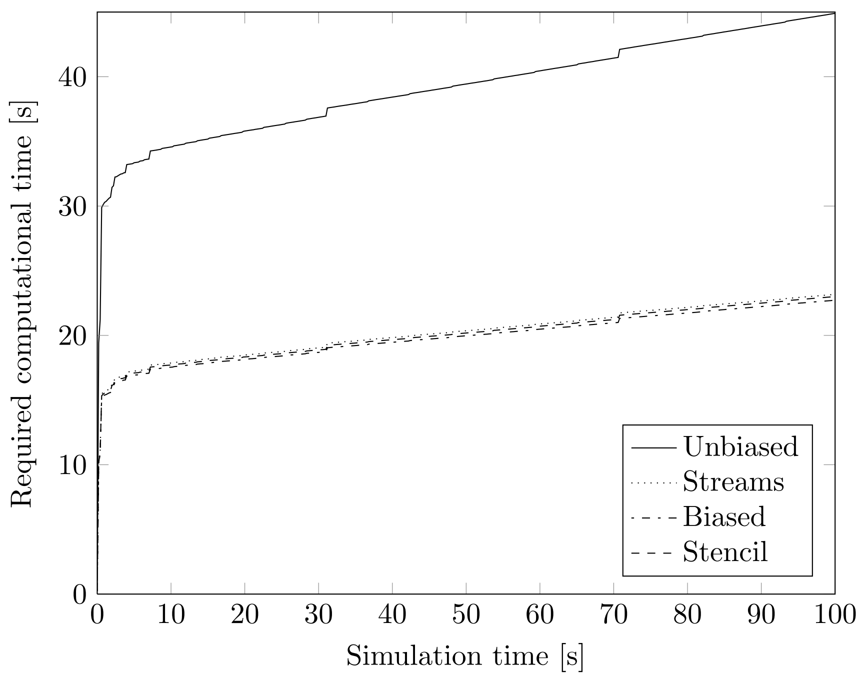


Figure 31: Required computational time , reference case “Relaxation transient”, including streams interface

6.1.3 Fluid property inversion at connection points

Using the inverse function annotation, all relevant information can be expressed for an algorithm in symbolic preprocessing to avoid property inversions at connection points. As the implementation is orthogonal to a specific thermo-fluid interface, all thermo-fluid interfaces fulfill requirement 5 and receive a very good score.

6.1.4 Semantics allow for meaningful simplifications

For all thermo-fluid interfaces, simplifications in terms of a predefined flow direction can be made easily by adapting equations. If a Modelica translator provides such symbolic preprocessing, then the `min` or `max` attributes of floating point variables on the connector mass flow rate can alternatively be used to make such simplifications. Then, the translator can, for instance, simplify the `inStream()`-operator.

To summarize, all established thermo-fluid interfaces fulfill requirement 6 equally well and are thus given a good score.

6.1.5 Efficient handling of ideal mixing

The formulation suggested in section 5.1.2 can be applied both to the stream interface and the stencil interface. With this modification, these interfaces thus result in a robust and efficient formulation of ideal mixing they are understood to comply with requirement 7.

The efficiency and robustness of ideal mixing together with fluids that contain several substances and either the unbiased or the biased mixing volume interface are poor. Again, small dynamic mixing volumes can be suggested for models using these interfaces, even if this may result in stiffness of the equation system. This is why the interfaces were given a poor grade in this regard.

6.2 User friendliness, end-user

The physical meaning of the variables in the streams connector, which represent the convected quantities, is not as intuitive as that for the unbiased mixing volume interface. However, it is much simpler than that of the variables in the stencil interface, as the subscript always refers

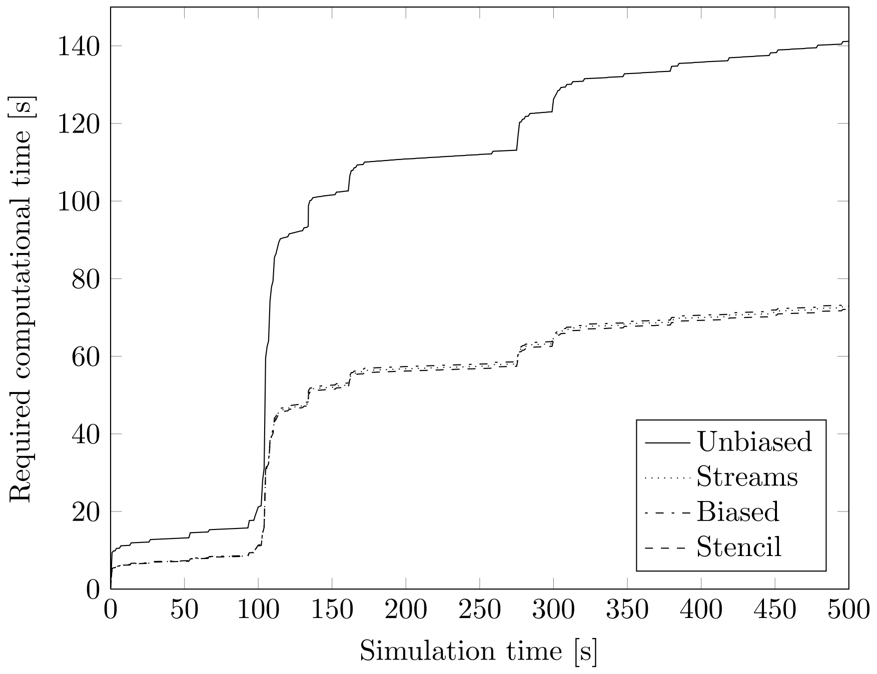


Figure 32: Required computational time, reference case "Shut-down", including streams interface

to a plain flow direction instead of a flow direction between `FluidPort_a` and `FluidPort_b` instances. Furthermore, if connecting two control volumes, their pressure states are lumped by index reduction, their specific enthalpies remain separate however. This is the proper result. Altogether, requirement 8 is considered fulfilled and the interface is given a good score.

Again, figure 34 illustrates the mapping of the components as shown in a graphical user interface (blue) to the mathematical models they represent (black). The system boundaries are identical to those when using the stencil interface.

6.3 User friendliness, developer

In most cases, the streams thermo-fluid interface of models can be implemented easily. Exactly one equation has to be provided for each convected quantity on the connector definition. Altogether, requirement 9 is fulfilled and the thermo-fluid interface is awarded a good score.

6.4 Flexibility

The streams thermo-fluid interface utilizes a universal connector and allows multi-way connections. The connection semantics directly result in appropriate static balance equations for the connection set and explicit junction models are not needed. Therefore, this thermo-fluid interface fulfills requirement 10 and receives a very good score.

6.5 Summary

With the improvements and alternatives introduced in chapter 5, a thermo-fluid interface emerged that concurrently fulfills the requirements for robustness and efficiency on one hand and user friendliness and flexibility on the other. The systematic error identified in section 4.3 is solved in a reasonable way. This concludes the contributions to the state of the art in non-causal thermo-fluid interfaces.

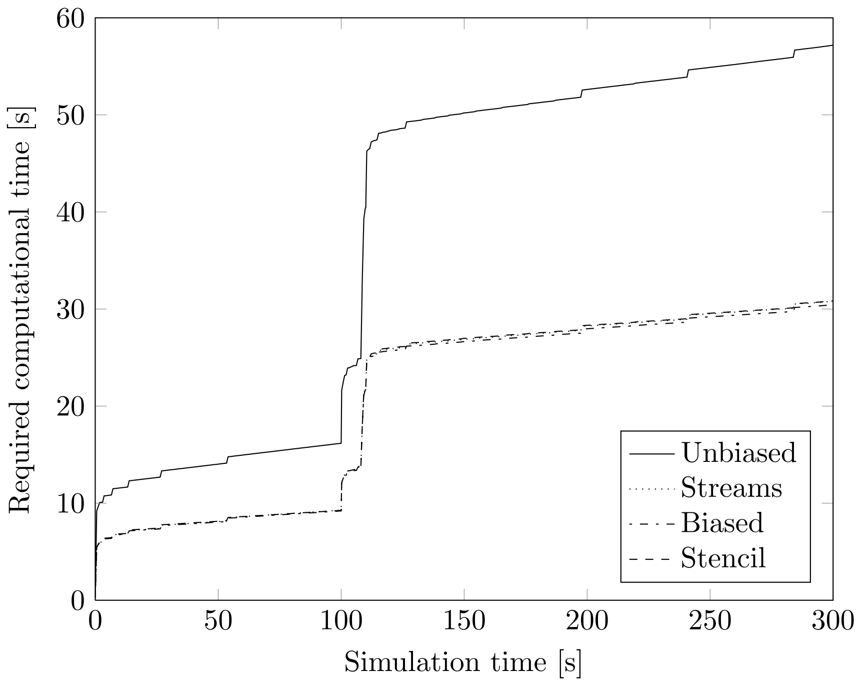


Figure 33: Required computational time, reference case "Set-point", including streams interface

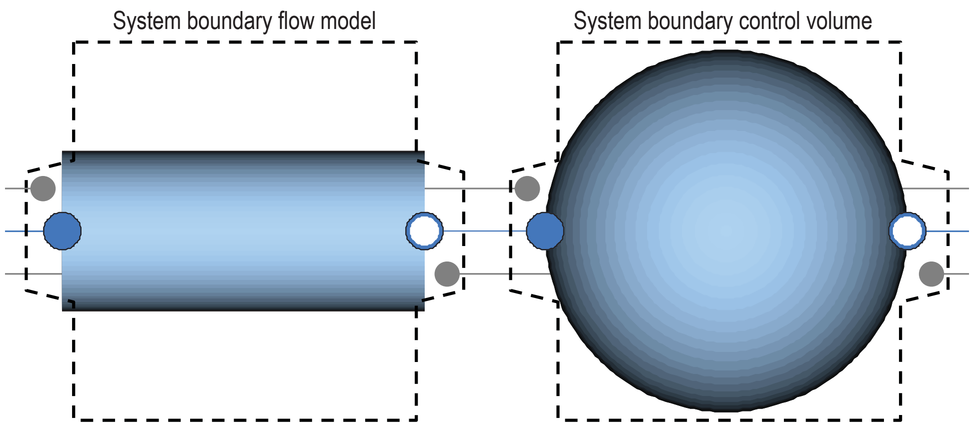


Figure 34: Control volume boundaries for example model topology (streams interface)

CHAPTER 7

ROBUST SOLUTION OF NONLINEAR ALGEBRAIC EQUATION SYSTEMS: PROBLEM ANALYSIS

The objective of this chapter is to rigorously analyze the observation of robustness issues with steady-state initialization stated in the introduction. In order to allow for a meaningful comparison, a rigorous quantitative metric is defined and a means for a concise graphical comparison of solver robustness for a given test problem is presented. A set of test problems is defined, which includes benchmarks that are relevant from an industrial perspective. Different state of the art solvers for nonlinear algebraic equation systems are compared with the solver provided in a commercial object-oriented modeling environment. Finally, conclusions are presented and it is argued whether the reported difficulties with steady-state initialization form a legitimate objection to current practice in object-oriented modeling and simulation.

7.1 Introduction

Equation-based, object-oriented modeling languages allow for a *declarative* high-level description. This is written by the modeler and processed by symbolical manipulation algorithms and transformed into a form that can be solved by standard numerical integration methods. A fundamental element of this concept is the separation of the code of the model from the numerical solver. Obviously, this increases readability and maintainability of both the model code and the solver implementation, but it also implies that no knowledge of the mathematical characteristics of the problem equations can be infused into the solver. Instead, a solver has to consider a general mathematical problem. In many domain-specific approaches, this is different. In the area of Computational Fluid Dynamics, for instance, problem-specific, yet rather simple, algorithms such as SIMPLE [139] or PISO [87] are applied to nonlinear problems with vast success.

As mentioned in section 1.5.3, the author of this thesis found that the large majority of state of the art algorithms for the processing steps involved in declarative languages are highly mature and robust for such general systems. There is however at least one sub-problem, for which difficulties were observed at times, that is steady-state initialization. In the context of system-level simulation, steady-state initialization is an initial value problem for a DAE with $\dim(F) = n_z + n_w$ equations, $0 = F(\dot{z}, z, w, t)$. Here, $z(t) \in \mathbb{R}^{n_z}$ is the vector of state variables and $w(t) \in \mathbb{R}^{n_w}$ is the vector of algebraic unknowns. As before, variable $t \in \mathbb{R}$ denotes time. For simplicity of the discussion, the DAE is assumed to be index-reduced, i.e., it has index 1, which means that the following expression be regular

$$\left[\begin{array}{cc} \frac{\partial F}{\partial \dot{z}} & \frac{\partial F}{\partial w} \end{array}\right]$$

Initialization means to provide consistent initial values for $\dot{z}_0$, z_0, w_0 such that the differential algebraic equation system is fulfilled at the initial time t_0. Since these are $2n_z + n_w$ unknowns and the equation system has $n_z + n_w$ equations, additional n_z equations must be provided, $0 = G(\dot{z}, z, w, t)$ with $\dim(G) = n_z$. For example, steady-state initialization implies $G(\dot{z}, z, w, t) = \dot{z} = 0$. Formally, this usually results in a system of nonlinear algebraic equations, which has to be solved numerically. This is a particular instance of an algebraic loop. Note that the quoted problem is present in legacy block diagram simulators as well and is usually more severe as no symbolic processing takes place.

Practitioners employ various approaches to circumvent these difficulties. A short overview on ways to avoid them is provided for completeness.

Start iterates A reason for the inability to find a solution may be the local convergence properties of the underlying algebraic equation system solver. Obviously, providing a start iterate, which is close to the solution, may solve the problem. For industrial problems, this is tedious and error-prone work however, as the solution is not known a-priori. Especially for automated optimization, this can be a major obstacle. The merit of this approach may further deteriorate if the modeler is unable to maintain a fixed set of variables as unknowns, either because the tool does not offer such a feature or because this set cannot be kept due to evolutionary changes in model topology.

Transient formulation Instead of solving the initial value problem, one may also start integration at some value of the state variables and continue until the transient behavior is dissipated and only the steady-state solution remains ("relaxation transient"). For industrial problems, this may however take prohibitively large wall time. Furthermore, practitioners report that imprecise initial state values may result in unrealistic behavior, such that, for example, in the area of thermo-fluid dynamics, the initial mass and heat flows are out of the valid range of the models themselves. Then, practitioners have to tweak initial state values instead of initial iterates, which clearly does not solve the problem properly.

Pseudo-transient formulation Many intermediate solutions between the original problem and the transient formulation have been used in practice. Additional state variables can be infused into algebraic loops by first-order differential equations, as their value is always known from the last step of the integrator and they therefore "break" the algebraic loops. Alternatively, the computational expense of the transient formulation may be reduced by modifying the time constants of underlying physics. Obviously, only the steady-state solution is meaningful then.

Reformulation In several cases, the problem equations may be reformulated using a different set of variables as unknowns, which can lead to more benevolent properties of an algebraic loop, or ideally, a convex problem. For industrial problems, this often requires many engineering hours and may be beyond the time scale of project work.

Approximate solution In several applications, it may be sufficient to solve the problem only approximately, for instance in optimization. For such applications, the problem is solved several times. Therefore, the initialization problem can be relaxed and solved only approximately. An additional equality constraint is introduced and the optimizer ensures consistency of the original DAE system [18].

7.2 Concise illustration of solver robustness

7.2.1 Literature review

Given the moderate number of published comparisons between nonlinear algebraic equation solvers, it is noted that the majority of them focuses on performance (i.e., the number of residual evaluations, Jacobian evaluations, wall time and so on). Several potential problems arise.

Tabular data Many authors focus on tabular data (e.g., [160, 83, 42, 43]), which usually include Boolean information on whether a given solver failed on a test problem. For large data sets, such tables are hard to read. Furthermore, they always leave room for interpretation.

Low number and choice of start iterates In general, the choice of starting points is arbitrary. In several published comparisons this is not reflected properly, as the number of starting points is low. Dent et al. [43] for example compare solver robustness by multiplying all components of the known solution with multiples of 10% and using them as start iterate. Ideally, the number of starting points investigated should be large and properly distributed.

Data analysis In some contributions, an interval of convergence is defined (e.g., [43]). For problems in several dimensions this seems simplistic. Basins of attraction for, e.g., plain Newton Method have been shown to have complex geometric character (fractals) and cannot be expressed in such intervals [31]. Other authors use medians and quartiles to process the data or create rankings. In any case, information on the relative size of improvement from one solver to the other is lost. Also, interpreting such data may be no easier than reading the raw data.

In the literature on benchmarking efforts of optimization problems, a more sophisticated approach was suggested. Dolan and Moré [48] suggested using a cumulative distribution function of a solver performance metric to assess and compare solvers. The result is called a performance profile. Typically, the ratio of the wall time of a solver over the lowest wall time of all solvers is used as performance metric. Given a factor $\tau \in \mathbb{R}$, the performance profile reports the probability for a solver that the performance metric is within this factor τ of the best possible value. For example, given that a user wants to wait at most twice the wall time of the fastest solver ($\tau = 2$), he or she can extract the probability that a given solver finds the solution during the prescribed wall time of one of the problems in the problem set.

While performance profiles capture the solver performance relative to other solvers on a given problem set, this information may not be sufficient for computationally expensive problems. Griffin and Kolda [74] and Moré and Wild [127] therefore suggested an alternative called data profiles in the latter reference. These metrics answer the question of the percentage of problems that can be solved (for a given tolerance) within a given number of function evaluations.

Ultimately, the performance and data profiles had to be rejected for the objective of the present chapter, because they focus mostly on performance. While it is possible to read a single "total" probability of convergence off a performance profile (in the limit of $\tau \rightarrow r_M^-$), the abscissa is irrelevant for the question of robustness, which, from a practitioner's point of view, is the interesting question as the cost of additional computational time is low in comparison to that of the engineering hours to solve a convergence failure.

To summarize, while performance and data profiles are a powerful method for the trade-off between performance and robustness, a more meaningful illustration of the robustness data is possible.

7.2.2 Robustness profiles

In this section, a rigorous approach to quantify robustness is suggested. In the context of nonlinear algebraic equation systems, robustness is defined as the ability of a solver to find a solution independently of the quality of the starting point. For the definition of robustness profiles, it is suggested to quantify the "ability of a solver to find a solution" by a single significant metric, namely, the probability to convergence, P_{conv}. Additionally, it is presumed that a relevant yet simple metric for the quality of the starting point is its distance from a solution, an observation that was already discussed above. When starting iteration far away from a solution, then convergence is usually harder to achieve than if starting close to a solution.

Obviously, it is impossible to establish such metrics for industrial problems using analytical methods. Therefore, the stochastic method of Monte Carlo simulation [119] is adapted to the present problem. The problem is sampled several thousand times and the results are condensed into a single data set on P_{conv} as a function of the lowest distance of the starting point to a solution according to some norm. According to the best knowledge of the author, no such systematic comparison of robustness has been published so far.

Assume that a set of solvers $\mathcal{S}$ shall be compared on a set of test problems $\mathcal{P}$. In this and all following definitions, a generic problem with a vector of unknowns $x = [\dot{z}_0; z_0; w_0]$ and residual equations $f = [F; G]$ will be used in place of the variables introduced before. Each problem in $\mathcal{P}$ is given as a vector of residual equations f and lower and upper bounds l, u such that the vector of unknowns be $x \in [l, u]$. Additionally, the solutions s_j are given with $j = 1 \dots n_s$ and n_s as the number of solutions.

For most technical problems upper and lower bounds are given naturally, be it by the definition of the variables (e.g., negative pressure or concentrations beyond $[0\%, 100\%]$ are not reasonable) or by the models themselves (e.g., elaborate two-phase models of thermodynamic properties have strict bounds of validity).

In most technical problems, the order of magnitude of the unknowns varies greatly (e.g., pressure in Pascal is on the order of 10^5, mass fractions are often on the order of 10^{-3}). In order to avoid biasing the suggested metric "distance from a solution" toward the scalar components of the vector of unknowns with larger magnitude, the interval defined by l, u is used to define an affine transformation of the vector of unknowns x to coordinates in the unit hypercube $\tilde{x}$, that is $\tilde{x}_i = (x_i - l_i)/(u_i - l_i)$. In the unit hypercube, the distance d is defined as the Euclidean norm between a starting point $\tilde{x}$ and the closest solution $\tilde{s_j}$.

$$d = \min\left(\|\tilde{x} - \tilde{s_j}\|_2\right) \tag{33}$$

This distance is used as abscissa for the robustness profiles. The ordinate is the probability of convergence, P_{conv}. According to the principles of Monte Carlo simulation it is the number of observed successful samples over the total number of experiments.

Given that a problem $p \in \mathcal{P}$ is sampled a finite number of times at random starting points, each observation has to be assigned to a bin or band in order to establish the number of observed successful samples. This is much like creating a histogram; non-overlapping intervals are defined in the distance d of same size. Then, the number of samples is counted for each bin.

Following the law of large numbers, the quality of a robustness profile depends on the number of samples per bin. In order to condense the relevant information into a single figure, a robustness profile shall therefore come with a small bar chart below the actual graph, which presents the number of samples in each of the bins. Only if the number of samples per bin is reasonably high, the quality of the robustness profile is sufficient (in the following section 7.2.3, an indication of the approximate order of error is given for different numbers of samples per bin). As this may take some wall time for completion, a (more noisy) robustness profile with fewer samples may be considered, but with the information in the bar chart the quality is immediately visible in the profile itself (if the number of samples per bin is constant however, it may be advantageous to omit the bar chart in order to utilize space more efficiently).

7.2.3 Exemplary robustness profile

In order to illustrate the concept of robustness profiles, a simple two-dimensional example is discussed next. For readability, the scaling step to the unit hypercube is excluded from this example. Furthermore, the problem is treated analytically, which avoids the need for Monte Carlo simulation.

The interval is defined by the lower and upper limits, $l = (-20, -20)$ and $u = (20, 20)$. Furthermore, the residual equations are as follows.

$$f(x) = \begin{pmatrix} \frac{\exp\left(-1/\left(x_1^2/100 + x_2^2/10\right)\right)}{x_1^2/100 + x_2^2/10} \\ \sin\left(\pi x_2/40\right) \end{pmatrix} \tag{34}$$

The residual equations are illustrated in figures 35 and 36. The interesting one in this problem is the first one. Note that this residual is non-convex. A local (damped) iterative solver using only gradient information will, whenever it is outside of the "bucket" around the origin, diverge from the solution $s_1 = (0, 0)$. This is the feature of the exemplary problem which will lead to zero probability of convergence far away from the solution (outside the bucket).

In order to treat the problem analytically, a simplified convergence criterion is defined. Given a starting iterate x, if the Newton direction $d_N(x) = -J(x)^{-1} f(x)$ points to the inside of the circle with the solution $(0, 0)$ as center and through x, then it is assumed that a properly damped algorithm succeeds to find the solution. If it points to the outside, then failure is expected. Mathematically, this is tested using the dot product of x and the Newton direction d_N. If it is positive, then the Newton direction points to the outside of the circle and failure of

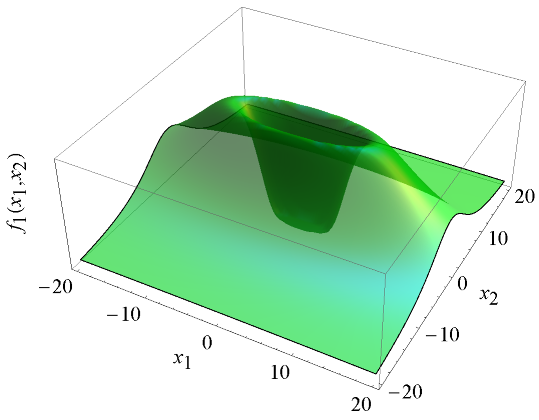


Figure 35: Exemplary residual 1

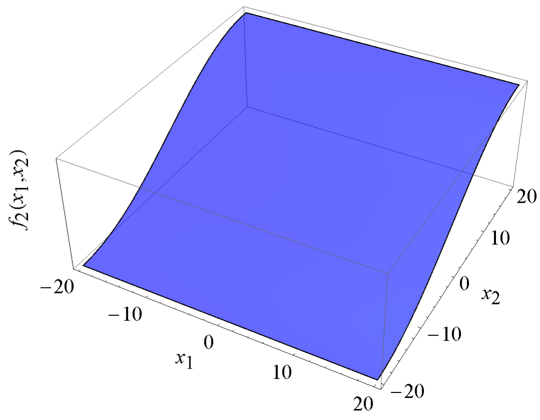


Figure 36: Exemplary residual 2

convergence is assumed. If it is negative then it points to the inside and successful iteration to a solution expected.

$$x \cdot d_N(x) < 0 \Rightarrow \text{convergence} \tag{35}$$

This criterion is illustrated in 37. As expected, the outer limit of this region coincides with the tip of the "bucket". In the inside of this region, there are parts which do not fulfill the simplified criterion. A properly damped method will however control the norm of the residuals and will make sure that the solver does not leave the bucket. Therefore, the inner part of the region is included in the convergence region, too. The residuals for this revised simple criterion are shown in figure 38, from which it is clearly visible that the criterion is appropriate.

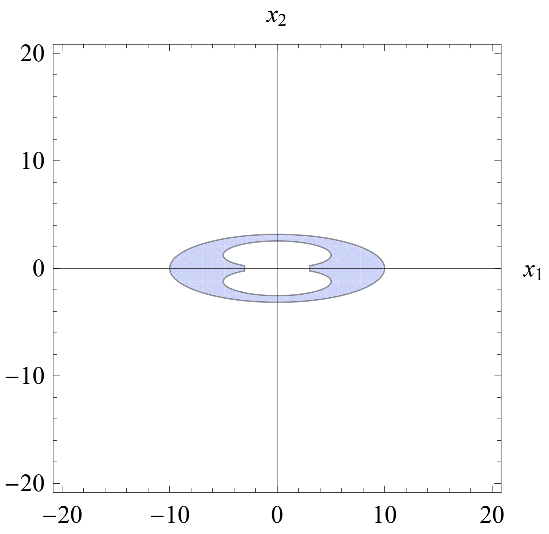


Figure 37: Convergence acc. to eq. (35)

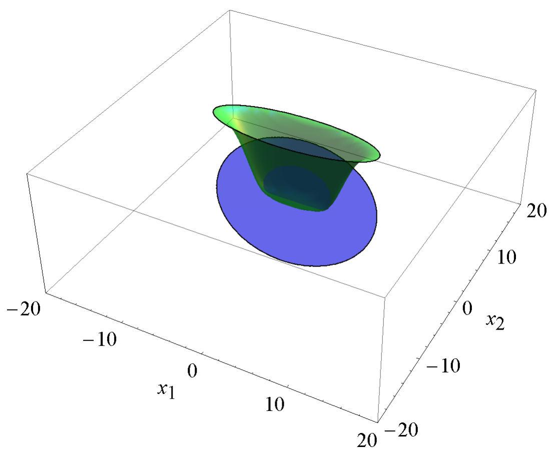


Figure 38: Residuals for revised criterion

Assuming that each starting point x within the bounds is equally likely, the probability of convergence can be expressed analytically as the volume in the n-dimensional Euclidean space (area for the example in two dimensions) for which convergence is observed over the total volume (area). Both can be expressed as a function of the Euclidean distance r_1 from the known solution s_1.

$$P_{conv}\left(r_1\right) = \frac{A_{conv}\left(r_1\right)}{A_{total}\left(r_1\right)} = \frac{\int_{r_1}^{r_1+\Delta r}\int_0^{2\pi}\chi_{conv}\cdot\chi_{bounds}d\varphi dr}{\int_{r_1}^{r_1+\Delta r}\int_0^{2\pi}\chi_{bounds}d\varphi dr} \tag{36}$$

Here, indicator functions χ_{bounds} are used to test whether a point is within the bounds l, u and χ_{conv} to test whether the starting point yields convergence.

For the reasons discussed above, the bin width Δr cannot be reduced to zero when using Monte Carlo simulation. For this analytically treated simple example, the limit $\Delta r \to 0$ can however be considered. The resulting robustness profile is shown in figure 39. If starting close to the solution (inside the "bucket"), then the probability of convergence is 100%. This is in accordance with the observation of the topology. As soon as the Euclidian distance from the solution passes the value $r_1 = \sqrt{10} \approx 3.162$ the starting point may be beyond the region of convergence with finite probability. Consequently, P_{conv} starts to decline. At an Euclidian distance of $r_1 = 10$, all starting points will lead to divergence and P_{conv} becomes 0%. As the scaling step was omitted for readability, these values directly correspond to the limits in figures 37 and 38. Note that this particular shape is a purely theoretic result for the arbitrarily chosen convergence properties of an imaginary solver. It serves as illustration. Real world impact, e.g., of scaling issues, is not included.

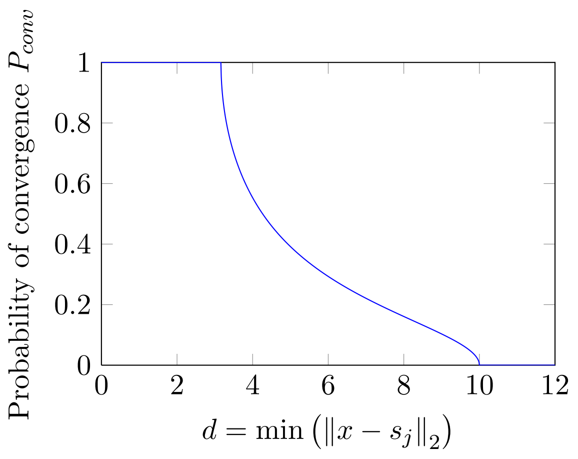


Figure 39: Exemplary robustness profile

In figure 40, the numerical method is applied to obtain robustness profiles. The figure illustrates the error in the robustness profiles for different numbers of samples per bin. This allows to establish a notion of the order of magnitude of the error, which is $O(10^0)$ for 1 sample per bin, approximately $O(10^{-1})$ for both 10 and 100 samples per bin and about $O(10^{-2})$ for 1000 samples per bin. In order to yield a reasonable approximation, 1000 samples per bin are therefore recommended. One has however to recognize that, as always in statistics, the (computational) cost of doing this may be prohibitive. Furthermore, it is highlighted that a robustness profile obtained by the numerical method always is an estimate and will generally be subject to sample-to-sample variation.

7.3 Comparison of state of the art gradient-based iterative solvers

With robustness profiles, a proper tool is available to analyze the observation stated in the introduction. A large number of different algorithms has been suggested for the solution of nonlinear algebraic equation systems in literature. Probably the best-known and most utilized algorithms are of the type of gradient-based iterative solvers, such as the Newton Method [41, 96, 45, 135]. These algorithms usually rely on a local linear or tensor model of the residuals and are fast but converge only locally. Two other classes of algorithms shall be mentioned. Continuation or homotopy methods [5] form a less widespread but commonly known alternative to gradient based iterative solvers. In the context of nonlinear algebraic equation systems, the idea is to start with a simple problem and continuously deform it to the difficult problem of interest. While the concept sounds simple, the details of these methods and algorithms are intricate. Unless proper provisions are taken, the existence of the homotopy trace between the start and a solution, finite length, nonexistence of singularities along the path and other important requirements are not guaranteed. A third exemplary class of algorithms is called interval methods [131]. The strong advantage of these algorithms is that they *guarantee* to find *all* solutions to a problem. They pose strict requirements however. Namely, they require access to the symbolic expressions of the equations via directed acyclic graphs or the like (only

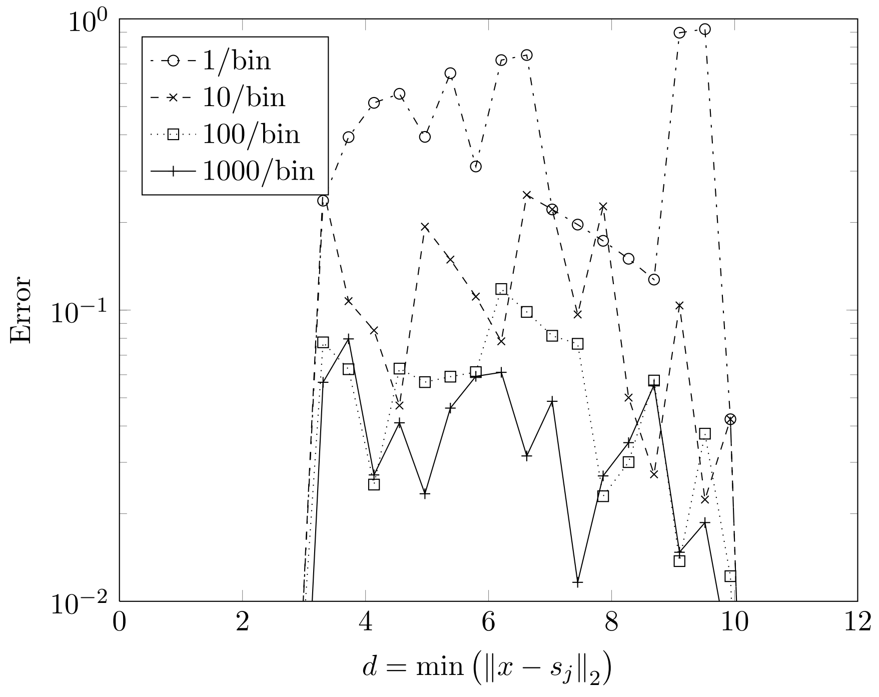


Figure 40: Absolute value of error for different numbers of samples per bin

the *values* of the residuals and possibly the Jacobian are not enough). Furthermore, these algorithms can be computationally expensive, as these algorithms use exhaustive search and therefore can suffer from combinatorial explosion.

In order to investigate the question raised in the introduction, algorithms of the most common class of gradient-based iterative solvers are considered and compared to the robustness of a commercial software implementation.

7.3.1 Solvers

To achieve a meaningful assessment of the mentioned observation and to allow for a fair comparison, requirements for quality of solver implementations were defined. They are either required to be able to run the problems in the set of test cases or to compare "apples to apples".

Inter-process communication with different platforms Inter-process communication via Berkeley sockets is used in order to be able to freely combine solvers and problems. Like this, residual equations can be implemented manually in C++ code (using CppAD [15] for the automatic generation of derivative information), which is useful for testing. Furthermore, this approach is used to interface with commercial modeling environments. Like this, any solver can be used to run industrial applications and, at the same time, exploit the symbolic processing made by such tools. For this work, such an inter-process communication interface was created for the Modelica tool Dymola [54] in versions 7.3 and 6.1.

Variable solver configuration via XML files Instead of hard coding solver parameters, they are read from XML configuration files. Like this, a user can modify such settings with a text editor and adapt the solver without repeated compilation.

Automatic scaling In system-level simulation, the Jacobian matrix is often ill-conditioned. Due the automatic reduction of the size of the algebraic loops (tearing [57]), conditioning may become even worse. Therefore, automatic scaling can become highly important to achieve convergence.

Proper reaction to failure in residual evaluation In industrial test cases it happens routinely that the residual equations cannot be evaluated. Possibly, the solver suggested an iterate, which lies on a boundary and results in division by zero, or inherent limits of the model validity were violated (e.g., thermodynamic property models). In any case, the solver shall not immediately quit but react properly to this situation (often, the solver simply reduces step size). At the same time, infinite cycles have to be avoided.

Numeric approximation of Jacobian Some representative test cases are provided without analytic Jacobian matrices. While this has per-se negative impact on both performance and robustness, these problems frequently occur in industrial practice. Some detailed power plant models for example utilize the detailed IF97 model of the thermodynamic properties of water and steam [188]. This model contains polynomial equations of order well above 40, which lead, if differentiated using automatic differentiation methods, to very inefficient code. Therefore, if the user opts to use this model, no analytic Jacobian is used. In this case, a numeric finite difference approximation of the Jacobian has to be created.

Robust yet fast configuration of solver parameters Obviously, the solver configuration plays an important role in the performance and robustness of a solver. Unless ample proof was given that a change was mandatory, the author stuck to the original parameter set provided by the algorithm authors. As such, the quality of this parameterization is difficult to assess. In the present chapter, the performance of a commercial Modelica tool is used as reference for what "good" robustness is.

Convergence criteria Each solver uses a particular convergence criterion. They often involve the norm of the step. Only if it drops below a prescribed limit, no further improvement can be achieved. In order to implement a fair comparison, it was decided to neglect these results however (i.e., the return codes of the solvers are not considered to decide whether a problem was solved or not). Instead, each solver is run as is, and after it finished, a constant convergence criterion is used to categorize the result. This convergence criterion checks whether the infinity norm of the unscaled residuals is below a given threshold. The set of allowable solutions is therefore defined by

$$\left\{x^* : \|f\left(x^*\right)\|_\infty < 10^{-5}\right\}$$

The comparison is conducted between the following six algorithms. As mentioned above, they all belong to the class of gradient-based iterative solvers.

7.3.1.1 Minpack

The Minpack [126] algorithm `HYBDJ` developed in the late 1970s at Argonne National Laboratory is the first solver considered herein. It is a modified Powell hybrid method. By itself, the algorithm only provides automatic scaling of the unknowns x. Therefore, the algorithm was equipped with automatic scaling in unknowns and residuals using row sums and nominal value information. Care is taken to update the scaling factors every time the Jacobian is refreshed and that all internal variables are updated properly, too. Furthermore, in its original form, the solver cannot react properly to a failure in evaluating the residuals. Therefore, additional logic was incorporated into the solver to reduce step size when this situation occurs. Even though an alternative solver calculating a numeric Jacobian approximation is provided with Minpack, the authors chose to not use the latter but to re-implement the finite difference logic in `HYBDJ` due to its interaction with the automatic scaling mechanism.

7.3.1.2 Nleq1 and Nleq2

The algorithms Nleq1 and Nleq2 [45, 132] developed in the early 1990s at Zuse Institut Berlin implement an affine-invariant Newton Method with and without rank reduction for poorly conditioned Jacobian matrices. These algorithms provide scaling of the unknowns and the linear system, which the author considers superior to what can be done in an interface layer as developed in the present effort, which is outside of the solver itself. See the technical report [132] for details. In its original form, the solver cannot react properly to a failure in evaluating the residuals. Therefore, additional logic was incorporated into the solver to reduce the damping factor when this situation occurs.

7.3.1.3 Tensolve

The algorithm Tensolve [20] implements a tensor method, which uses a quadratic instead of a linear local model. It was developed in the 1990s at Argonne National Laboratory. It provides two globalization strategies, namely a line search method and a tensor trust region method. Herein, the former is called Tensolve-LS, and the latter Tensolve-TR. The implementation provides a rather simple scaling mechanism only. The user can provide constant scaling vectors for the unknowns x and the residuals f. While nominal information may be used to establish the former, automatic rescaling of the equation system during the solving process is considered important for both the unknowns and the residuals. Also, the other implementations employ such scaling and in order to compare "apples to apples" an automatic scaling method was added to Tensolve. Care has been taken to ensure that an update of the scaling factors does not turn the internal state of the solver inconsistent. Similar to the others, this solver did not react properly to a failure in evaluating the residuals. Therefore, additional logic was incorporated into the solver to reduce the step size when this situation occurs. For this solver additional modifications were required to catch infinite cycles (e.g., in the standard line search implementation `TSLSCH`).

7.3.1.4 Dymola Solver

The commercial Modelica environment Dymola [54] uses a proprietary solver for algebraic loops, which is called the Dymola solver in this work. It is included in this comparison as reference for commercial implementations to make sure that the conclusions drawn in this work properly scale up to the situation practitioners face. The Dymola solver uses a method, which is not further specified. It provides an automatic scaling mechanism which, due to more than a decade of software development and industrial use, is considered mature. The solver also comes with proper logic to react to failure to evaluate residual equations. Furthermore, the solver configuration is considered mature.

7.3.2 Definition of test cases

The author reviewed a large number of models of technical systems. Out of them, a small number of models was not solved with constantly full probability of convergence. These models were considered relevant to the objective of this work and thus selected for inclusion in this study. These 12 models are summarized in tables 4 to 7.

Name	n_{ut}	n	n_s	χ_{aj}	Description
Aircraft ECS 1	427	51	1	1	Aircraft air conditioning system using an air cycle architecture
Aircraft ECS 2	493	29	1	0	Aircraft air conditioning system using a vapor cycle architecture
Air distribution	550	57	1	1	Aircraft air distribution system

Table 4: Test cases "Thermo-fluid dynamics"

Note that the Inverter Chain problem listed in table 7 can be solved sequentially. In the test case, a fake dependency was injected into the Block Lower Triangular transformation to require solving the problem in a single block.

In these tables, n_{ut} is the number of equations and unknowns *before* symbolic reduction and n is the number of equations and unknowns *after* reduction. The symbolic reduction technique is called tearing [57] and was mentioned in the subsection on Automatic Scaling of section 7.3.1

Name	n_{ut}	n	n_s	χ_{aj}	Description
Counter Current Reactors [109, 118]	6	5	1	1	Model of counter current reactors with $n = 6$
Liquid Phase Split [11]	6	4	5	1	Separation problem of two components in two phases using the NRTL equations as liquid phase activity coefficient model (with exclusion of trivial solutions)
Homogeneous Azeotropic Distillation [77, 12]	794	88	1	1	Steady state of a distillation column (ternary system)

Table 5: Test cases "Process engineering"

Name	n_{ut}	n	n_s	χ_{aj}	Description
Racing Car 1 [40]	2616	15	1	1	Detailed model of a Formula 1 racing car, straight steady state
Racing Car 2 [40]	2647	17	1	1	Detailed model of a Formula 1 racing car, cornering steady state
Direct Kinematics [118]	11	11	2	1	Determine the pose parameters of the platform of a parallel robot
General Stewart-Gough Platform [198]	18	8	8	1	Forward kinematics of the general Stewart-Gough platform

Table 6: Test cases "Mechanics"

already. It is implemented within the modeling software Dymola [54]. n_s is the number of solutions to the problem. The value of χ_{aj} indicates, whether an analytic Jacobian is available for this problem.

7.3.3 Results

The given test problems were analyzed using the methodology outlined so far. Following the principles of Monte Carlo simulation, the abscissa was cut into 30 slices. Then, 1,000 starting points per slice were analyzed for whether they yield convergence. Therefore, a total of 30,000 starting points were run for each solver on each problem. On average, wall time was between 23 hours and 7 days per problem for all solvers on a single processor computer. The robustness profiles are shown in figures 42 to 53. Note that only figure 42 presents a complete profile; in the remaining figures the bar chart was omitted due to the constant number of samples per bin.

7.3.3.1 Test cases "Thermo-fluid dynamics"

On both Aircraft ECS cases the robustness of the Dymola solver and the Minpack algorithm is highest. Furthermore, the results for both algorithms are very similar. For Aircraft ECS 1, Minpack is superior and for Aircraft ECS 2 Dymola is. On Aircraft ECS 1, the robustness of all other solvers is only a fraction of that of Dymola and Minpack respectively. On Aircraft ECS 2, the algorithms Nleq1 and Nleq2 are worse than Dymola and Minpack but still competitive.

Name	n_{ut}	n	n_s	χ_{aj}	Description
OA 741 [117]	52	11	2	1	Operational amplifier 741
Inverter Chain	151	50	1	1	Inverter Chain of $n = 50$ inverters

Table 7: Test cases "Analog circuits"

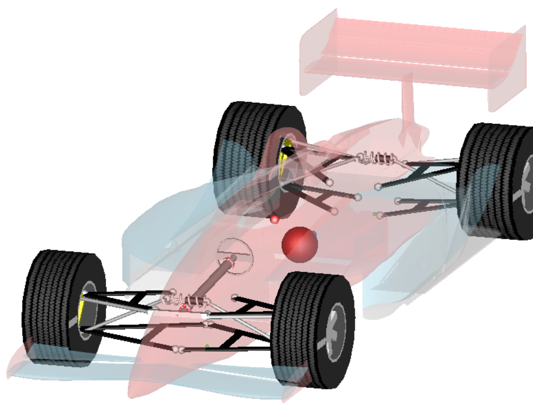


Figure 41: Graphical representation of the Racing Car test model

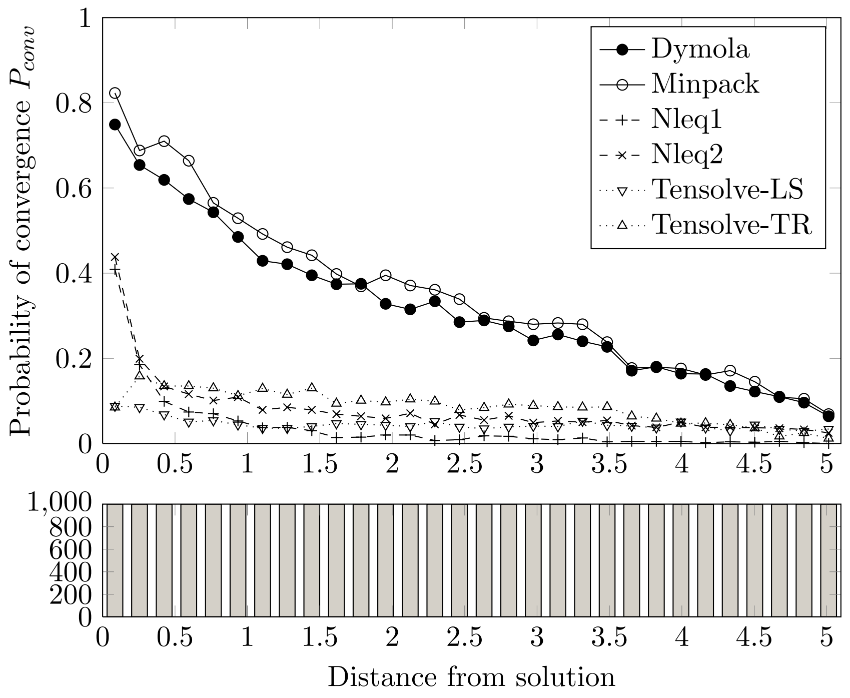


Figure 42: Robustness profiles for Aircraft ECS 1

The Tensolve algorithms are not competitive.

7.3.3.2 Test cases "Process engineering"

The solver Nleq2 performs best on the test case Counter Current Reactors. When relating this to the robustness of Nleq1, which is worst for this case, the potential of the rank reduction device of Nleq2 becomes apparent (this is the only difference between Nleq1 and Nleq2). The Tensolve algorithms are second only to Nleq2. On this test case, the Tensolve globalization using the line search technique performs slightly better. Again, the Dymola solver and the Minpack algorithm perform very similarly. Their robustness is still competitive.

The solvers that perform best on the Liquid Phase Split test case are Nleq2 and Tensolve with line search globalization. The robustness of Nleq1 is nearly as good as that of the leading duo, highlighting that the rank reduction device does not alleviate difficulties on this test case. The Dymola solver and the Minpack algorithm are not performing best but again providing competitive robustness. The same holds for the Tensolve algorithm with trust regions globalization, even though the robustness is low when starting far away from the solution.

For Homogeneous Azeotropic Distillation, Nleq2 again performs best. Nleq1 in turn is again

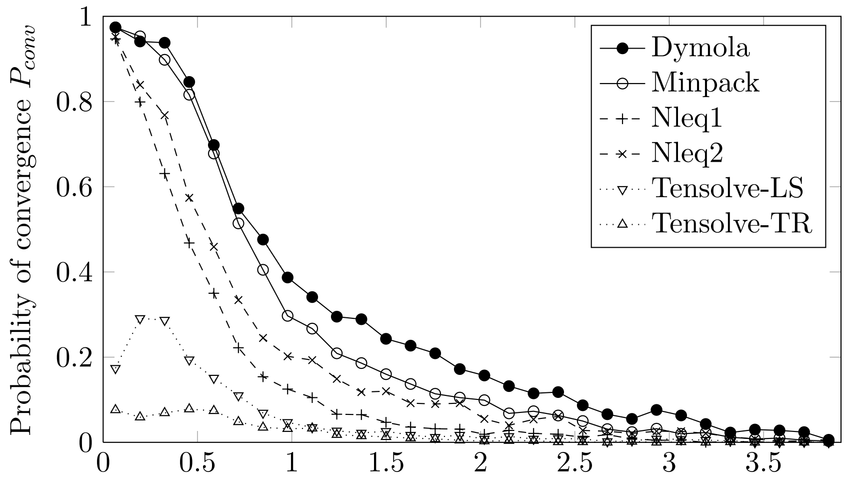


Figure 43: Robustness profiles for Aircraft ECS 2 (1000 samples per bin)

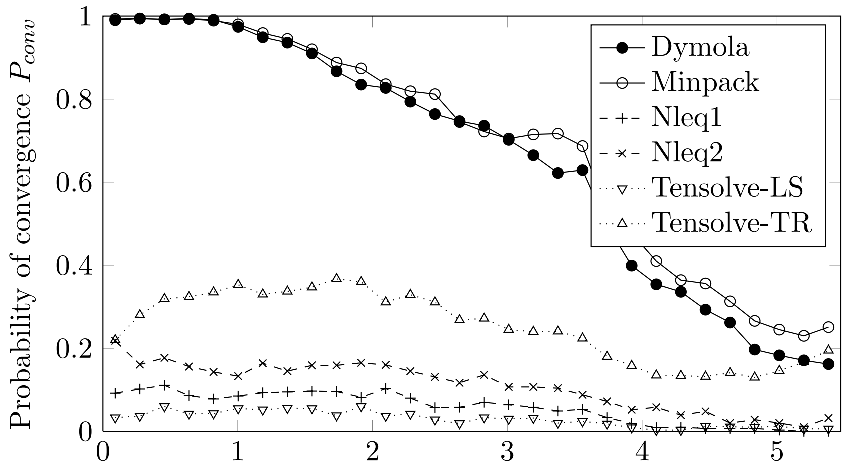


Figure 44: Robustness profiles for Air Distribution (1000 samples per bin)

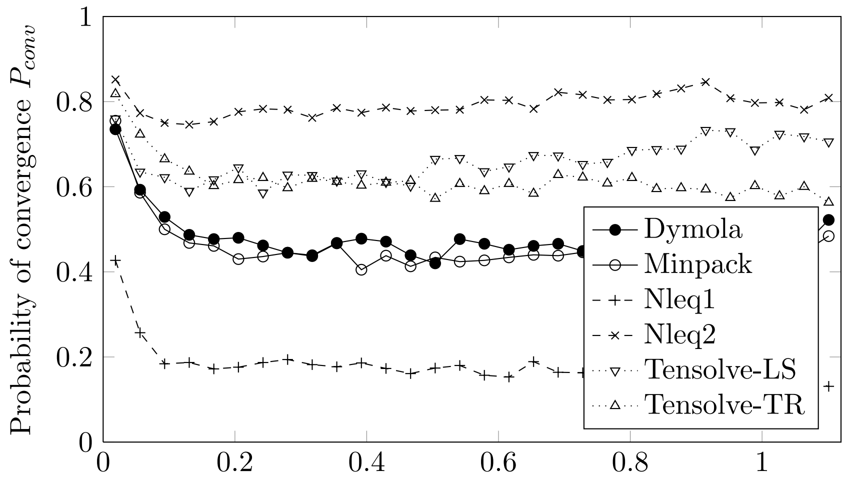


Figure 45: Robustness profiles for Counter Current Reactors (1000 samples per bin)

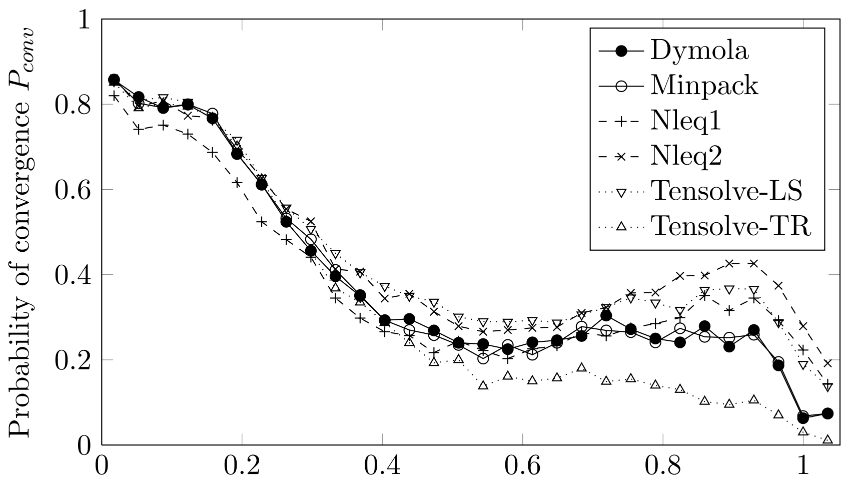


Figure 46: Robustness profiles for Liquid Phase Split (1000 samples per bin)

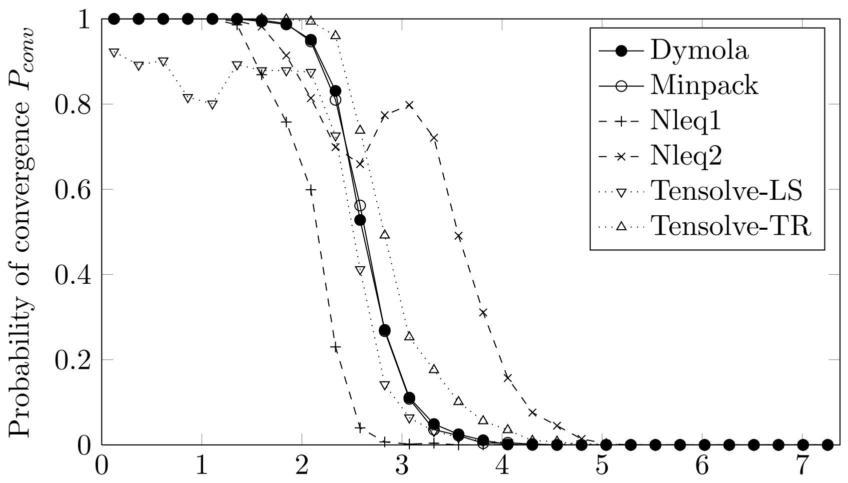


Figure 47: Robustness profiles for Homogeneous Azeotropic Distillation (1000 samples per bin)

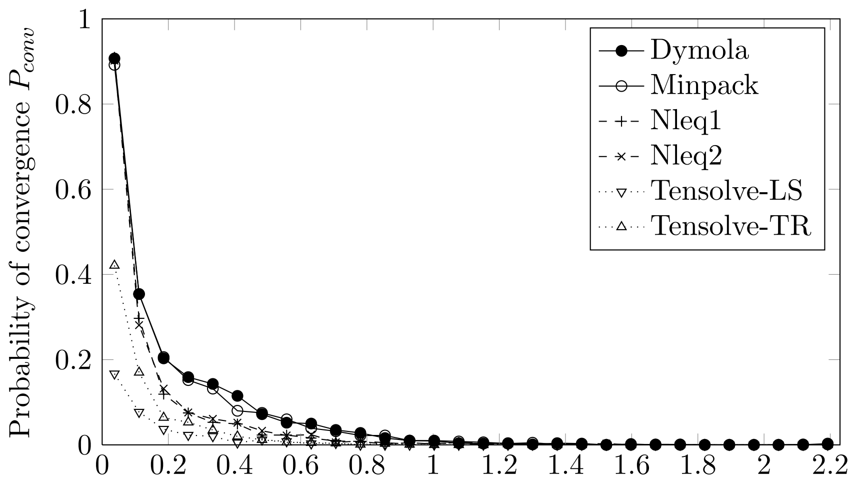


Figure 48: Robustness profiles for Racing Car 1 (1000 samples per bin)

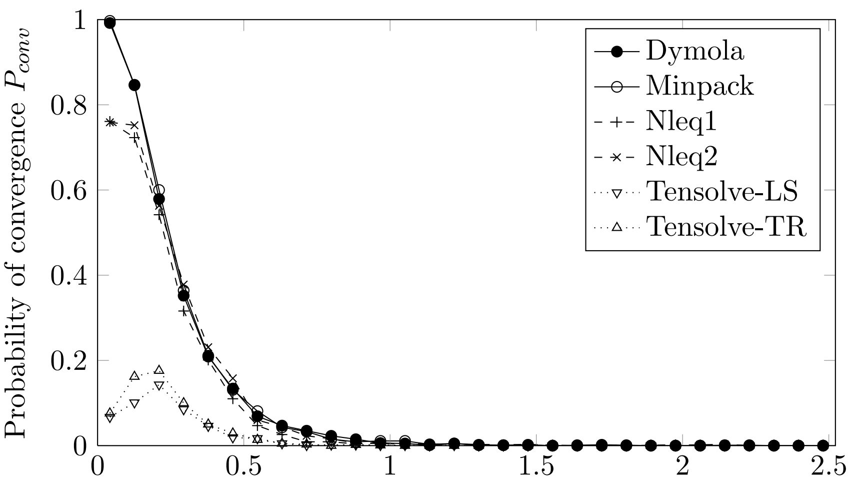


Figure 49: Robustness profiles for Racing Car 2 (1000 samples per bin)

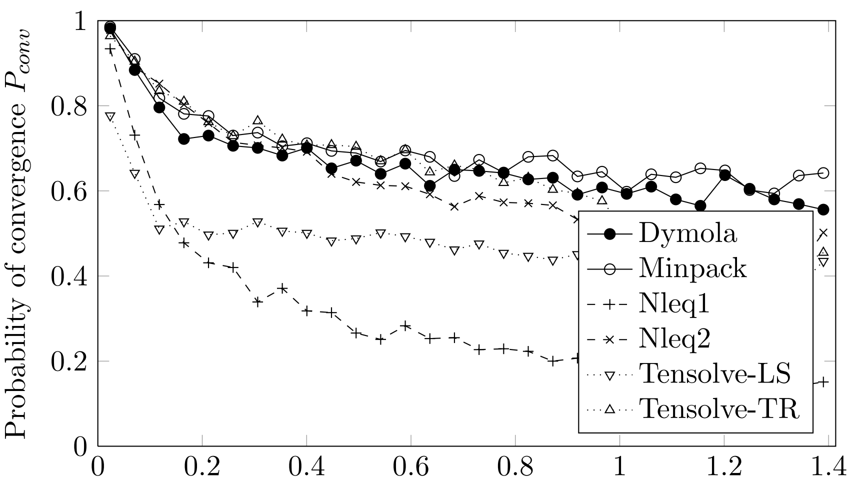


Figure 50: Robustness profiles for General Stewart-Gough Platform (1000 samples per bin)

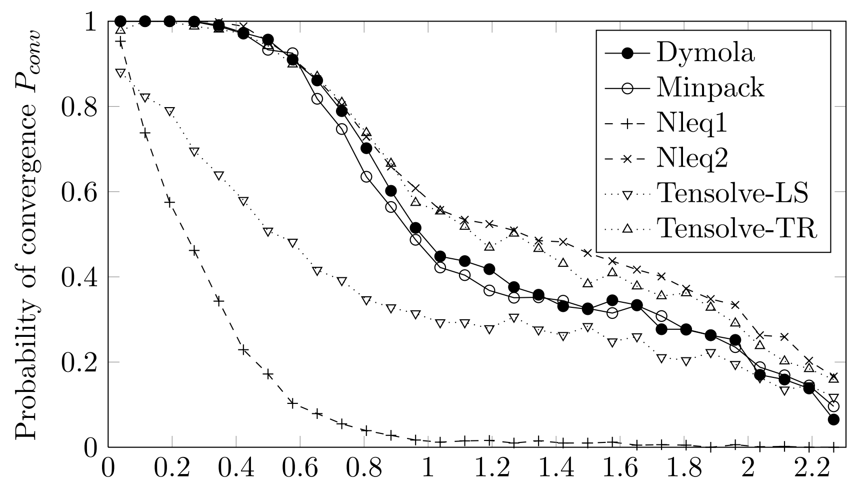


Figure 51: Robustness profiles for Direct Kinematics (1000 samples per bin)

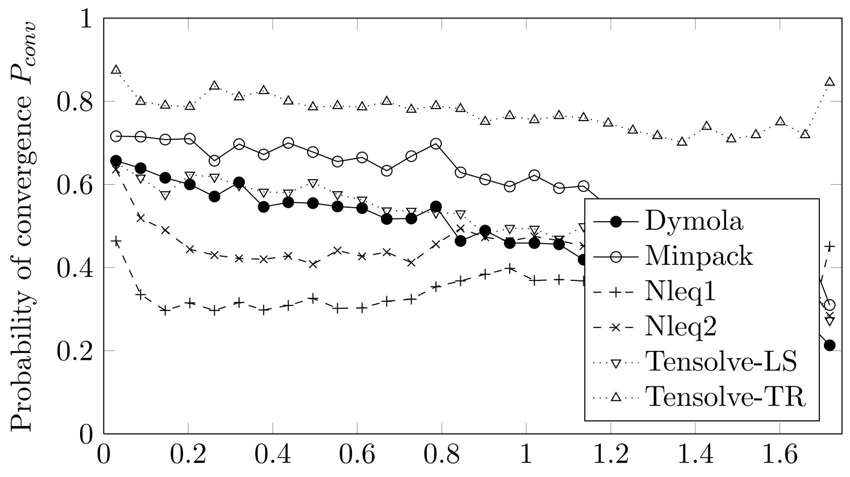


Figure 52: Robustness profiles for Operational Amplifier 741 (1000 samples per bin)

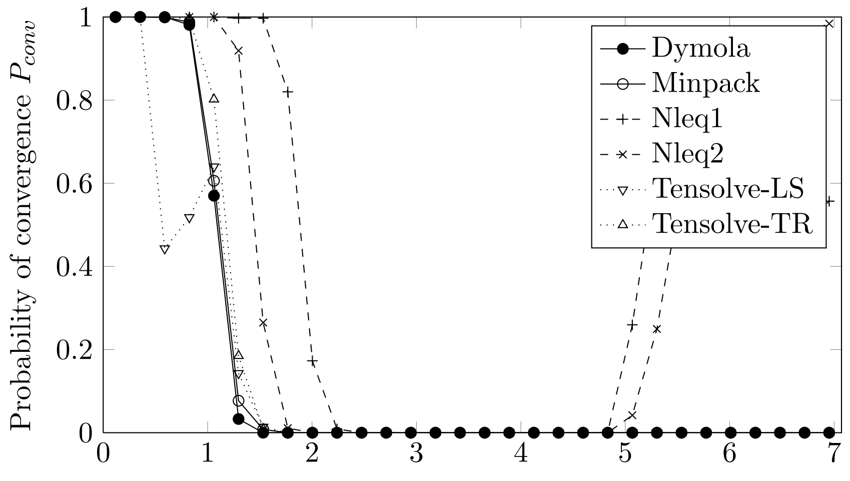


Figure 53: Robustness profiles for Inverter Chain (1000 samples per bin)

less robust. All other algorithms provide competitive robustness.

7.3.3.3 Test cases "Mechanics"

On both Racing Car test cases, the Dymola solver, the Minpack algorithm and the two Nleq algorithms perform best. In both cases, the Dymola solver and the Minpack algorithm are slightly superior however. For Racing Car 1, the Tensolve algorithm with trust region globalization is competitive, the one using the line search globalization is not.

On the remaining test cases (General Stewart-Gough Platform, Direct Kinematics), the Dymola solver, the Minpack algorithm, Nleq2, and Tensolve with trust region globalization provide most robustness. Tensolve with line search globalization is still mostly competitive, only Nleq1 is not fully competitive on these cases (on Direct Kinematics in particular).

7.3.3.4 Test case "Analog circuits"

On the Operational Amplifier 741 test case the Tensolve algorithm with trust region globalization performs most robustly. Minpack is second and both Nleq algorithms provide moderate robustness only. These algorithms exited mostly with either too low damping factor or reduced step size to extremely low values such that no progress was obtained in any reasonable number of steps (5000 for a problem with 11 unknowns).

The Inverter Chain test case results in zero probability of convergence unless iteration starts within about 20% of the maximum distance in the unit hyper cube from the solution. Algorithms Nleq1 and Nleq2 provide spurious convergence when starting far away from the solution, which is considered a coincidence due to the symmetry of the problem as probability of convergence is zero when starting *closer* to the solution.

7.4 Conclusions

In this chapter, a rigorous approach to quantify robustness in the context of nonlinear algebraic equation systems was presented. The definition of robustness profiles was derived from a clear, comprehensible statement on the meaning of robustness. The result is a straightforward graphical means to assess the latter on different test problems.

This tool was utilized in a study of six gradient-based iterative solvers. All of them were adapted to meet well-defined quality of implementation requirements. The results of this study are summarized in the following paragraphs.

The first important conclusion is that the solvers used in this comparison are competitive alternatives to the implementations used in state of the art commercial tools for declarative modeling. Robustness may be even higher using one of the alternative solvers. The consequence is that the conclusions drawn from this work are representative and relevant for practitioners.

The second conclusion concerns the robustness of the different solvers. As it differs largely between the various test cases, it is difficult to make simple statements about which solver performs best. Essentially, such a statement may not even be possible. It is highlighted however that for the twelve industrial test cases, the robustness of the Dymola solver and the Minpack algorithm on one hand and Nleq2 on the other is best. The former algorithms are better on some cases (e.g., Aircraft ECS 1 and 2) and the latter is on others (e.g., Counter Current Reactors, Homogeneous Azeotropic Distillation). Nleq1 is not competitive. Similarly, Tensolve-LS and Tensolve-TR *may* perform either very well or poorly on some problems. Likely, the question of whether the line search globalization or the trust region globalization is more robust for the tensor algorithm Tensolve cannot be answered in general. In some cases (e.g., Liquid Phase Split, Counter Current Reactors) the line search globalization is more robust. In others (e.g., General Stewart-Gough Platform, Direct Kinematics), the trust region globalization performs better.

The third and most fundamental conclusion is that the observed difficulties with convergence of algebraic loops are real. As seen in several examples (e.g., Racing Car 1 in figure 48), the probability of convergence can be effectively 0% if not starting in close vicinity of the (initially unknown) solution. It is noted however that the test cases analyzed herein are the worst cases encountered in this study. The large majority of example cases were solved with constantly full probability of convergence. For obvious reasons they were discarded for this work however.

It is therefore concluded that the initially stated observation on difficulties with steady-state initialization forms a legitimate objection to current practice in object-oriented modeling and simulation. Alternative approaches such as the ones mentioned in the first paragraph of section 7.3 have to be considered in order to improve the current state of affairs in the context of general declarative modeling languages.

CHAPTER 8

INTRODUCTION TO HOMOTOPY METHODS

Several groups of alternative solution methods have been proposed in literature to numerically solve the root finding problem defined in section 7.2.2. The objective of this chapter is to introduce a well-known class of alternative solution methods called homotopy methods in a comprehensible, informal fashion. In the context of nonlinear algebraic equation systems, the idea is to start with a simple problem and continuously deform it to the difficult problem of interest. Unless proper provisions are taken, the existence of the homotopy trace between the start and a solution, its finite length, nonexistence of singularities along the path and other important requirements are not guaranteed. Further examples of alternative approaches are interval methods [131] and terrain methods [108].

Continuation is a technique for numerically approximating a solution curve which is implicitly defined by an under-determined system of equations [4]. The system of equations, $\rho(x, \lambda) = 0$, is the homotopy, a continuous deformation from one map to another [10]. In the context of solving a system of algebraic equations the homotopy is constructed such that one equation set is easy to solve and the other is the one of interest, $f(x) = 0$. The additional dimension is the new variable λ, which is called the homotopy or continuation parameter and is restricted to some range, e.g., $[0, 1]$. Then, the augmented system $\rho(x, 0) = 0$ is solved with ease and reduces to the system of interest at $\lambda = 1$, i.e., $\rho(x, 1) = f(x)$. "Homotopy" is a common abbreviation for the exact term "homotopy map". An overview of continuation algorithms and homotopy in general is given by Allgower and Georg [5].

8.1 *Homotopy methods and the root finding problem*

Starting at $\rho(x, 0) = 0$, a curve (x, λ) is followed along $\rho(x, \lambda) = 0$ until $\lambda = 1$. Formally, this is $\{x|\rho(x, \lambda) = 0\}$. In traditional *continuation or embedding methods* a monotonically increasing parameter λ is assumed [69]. This leads to a unique $x(\lambda)$. The underlying assumption is not generally fulfilled. Consider figure 54 that illustrates different types of traces. A continuation method is not able to follow the bottom track with a turning point.

This limitation led to the wider use of pseudo-arc-length continuation methods, in which both x and λ are considered functions of some arc length s. The continuation algorithm then follows $\rho(x(s), \lambda(s)) = 0$ until $\lambda(s) = 1$. In the context of the root finding problem, this type of method is called a *homotopy method* [69]. Using such methods, the bottom track with the turning point can be established.

For non-trivial problems, issues arise when tracing either of the other tracks however. These issues are either qualitative (in case of divergence to $\pm\infty$ or closed loops called "isolae") or numerical (in case of bifurcations). All these issues do not arise when using *probability-one homotopy* due to the construction of the homotopy map. This methodology is based on a theory on convergence and is due to Chow et al. [36] and Watson et al. [190]. Alexander and Yorke [3] and Garcia and Zangwill [68] provide additional information on degree arguments and homotopy. According to the judgment of the author, Chow et al. [36] provide a body of theory to *guarantee* global convergence under rather mild assumptions. As the resulting coercivity conditions inherently also imply the *existence* of a solution it is not trivial to prove that a specific problem fulfills them.

To illustrate how such singularities arise, an example that appeared in [117] is given. Continuation methods, both monotonic and pseudo-arc-length, have been used widely in analog circuit simulators to calculate DC operating point problems. A typical approach is to multiply the voltage of each source with the continuation parameter λ. Then, at zero excitation, an obvious solution is $0V$ in the complete circuit. A continuation method is then used to increase λ

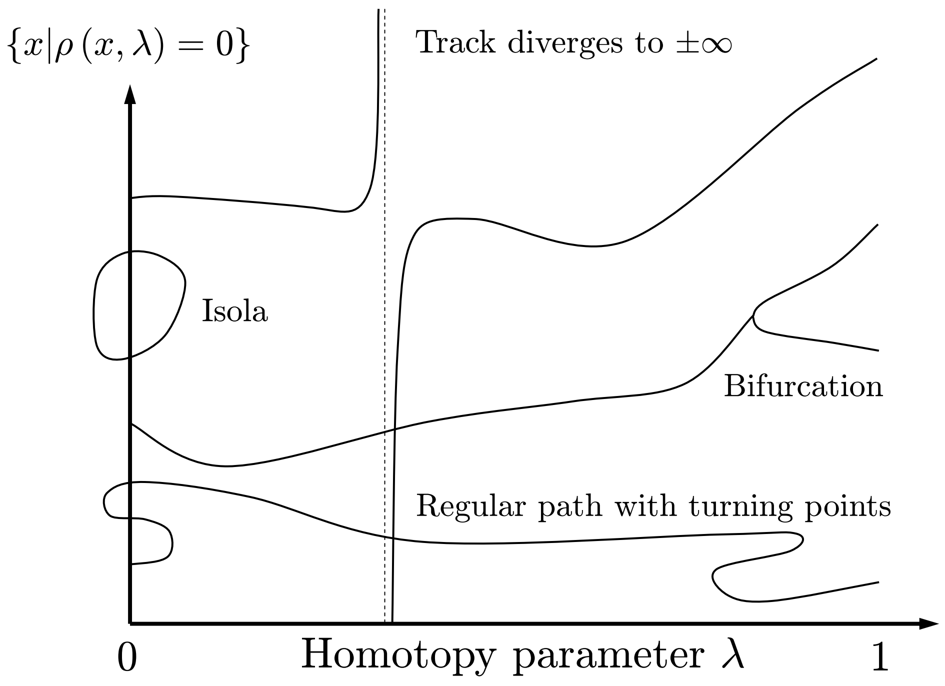


Figure 54: Different types of solutions to a homotopy map

to one, which yields the solution to the problem of interest. This type of ramping of the boundary conditions (called source stepping [130]) is a representative example of natural parameter continuation methods and continuation methods in general.

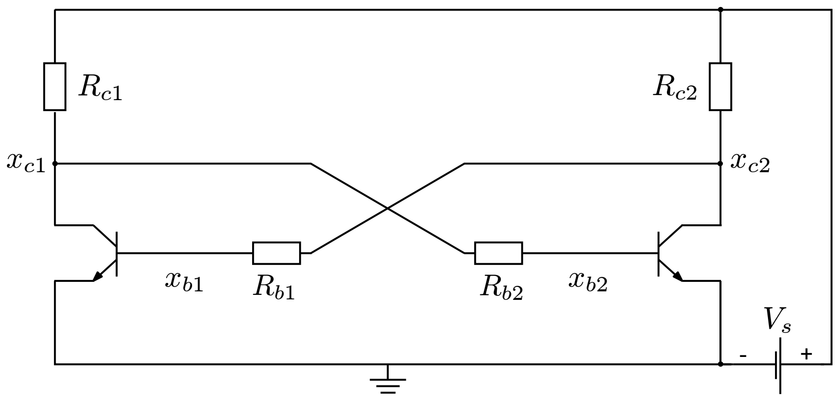


Figure 55: Simple flip-flop circuit [116]

Consider the flip-flop shown in figure 55. The approach of ramping the boundary conditions is applied to this exemplary problem. For this purpose, assume for the source voltage $V_s^* = \lambda V_s$. The collector resistances are $R_{c1} = R_{c2} = 1\text{k}\Omega$, the base resistances $R_{b1} = R_{b2} = 20\text{k}\Omega$. Using the potentials of the connection sets as unknowns and the sum of the currents out of a connection set as residual equations, the following equation set is established, which represents the embedding of the source voltage into the circuit equations.

$$\rho(x,\lambda) = \begin{bmatrix} \frac{x_{c1}-\lambda V_s}{R_{c1}} + \frac{x_{c1}-x_{b2}}{R_{b2}} + I_c(x_{b1}, x_{c1}, 0) \\ \frac{x_{b1}-x_{c2}}{R_{b1}} + I_b(x_{b1}, x_{c1}, 0) \\ \frac{x_{c2}-\lambda V_s}{R_{c2}} + \frac{x_{c2}-x_{b1}}{R_{b1}} + I_c(x_{b2}, x_{c2}, 0) \\ \frac{x_{b2}-x_{c1}}{R_{b2}} + I_b(x_{b2}, x_{c2}, 0) \end{bmatrix} \tag{37}$$

Herein, $I_c(x_b, x_c, x_e)$ and $I_b(x_b, x_c, x_e)$ are the collector and base currents. They are functions of the base, collector, and emitter voltages according to some transistor model, for instance

Ebers-Moll EM1 [70]. Similarly, $x = [x_{c1}, x_{b1}, x_{c2}, x_{b2}]^T$ is the vector of unknowns.

During operation, the flip-flop has three DC operating points, out of which two are stable. In contrast to this, for the case of zero source potential, the circuit has only a single solution. The results of the continuation or homotopy approach are given in figure 56.

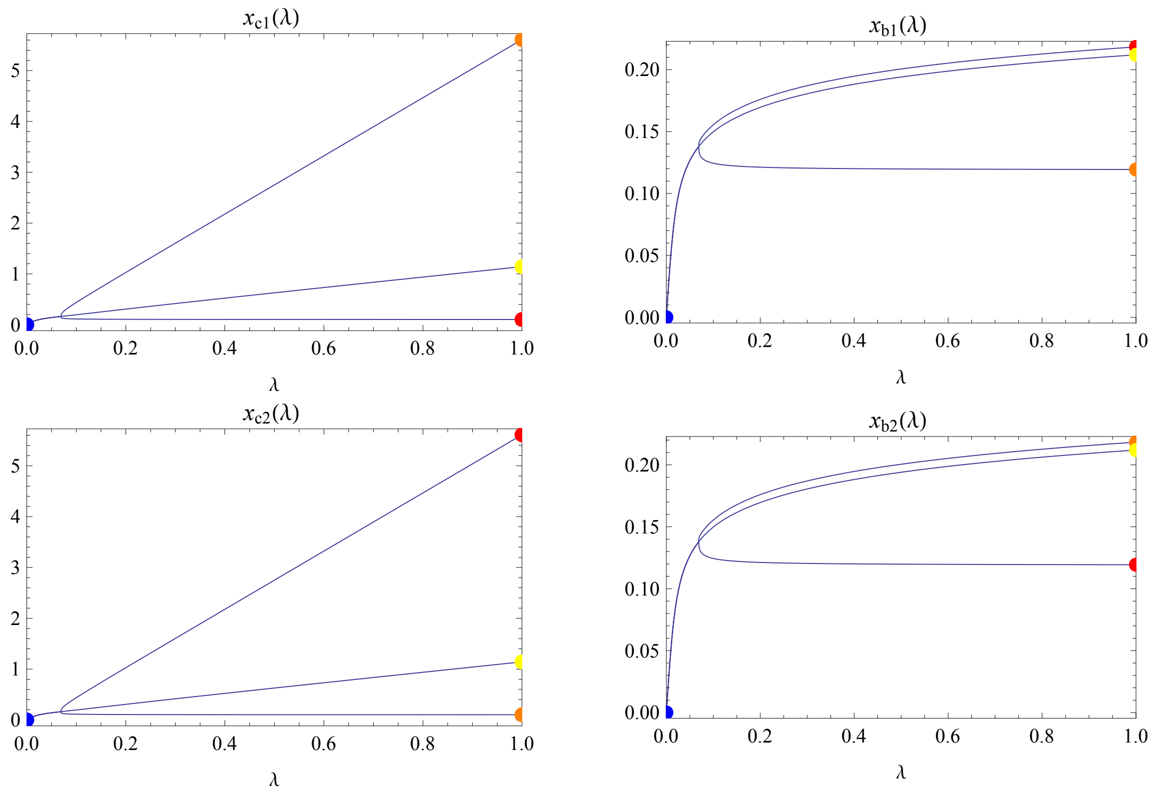


Figure 56: Natural parameter continuation for the flip-flop circuit

Here, the starting point at zero excitation is marked with a blue dot. From this point, the continuation algorithm starts increasing the supply voltage. At a specific value of the supply voltage the base-emitter junctions of the two transistors are turned on. From this point three branches emerge to the direction of higher supply voltage (higher λ). This point is a bifurcation. Bifurcations are numerically challenging and coincide with a singular Jacobian matrix. If a continuation algorithm manages to continue at all, it will often follow the middle track that leads to the unstable solution. In reality, noise will drive the flip-flop into one of the two stable states. The stable steady-state solutions are marked with a red and orange marker respectively. The instable one is highlighted with a yellow one.

Note that even if a physical system has a unique solution at the *final* value of the boundary conditions, it may have several steady-state solutions at the intermediate values.

Several generic homotopy maps have been constructed to solve a root finding problem $f(x)$ without embedding λ in f. Three successful generic maps from literature are introduced next. The first one is the fixed point homotopy map. It was used by many authors including Chow et al. [36], who suggested to solve for fixed points and zeros using this map. The version for finding roots is of interest herein,

$$\rho(x, \lambda) = \lambda f(x) + (1 - \lambda)(x - x_0) \tag{38}$$

A second map is the Newton homotopy map (also called global homotopy map), which, in the form common today, is suggested by Keller [95],

$$\rho(x, \lambda) = \lambda f(x) + (1 - \lambda)(f(x) - f(x_0)) = f(x) - (1 - \lambda) f(x_0) \tag{39}$$

As Allgower and Georg [4] report, earlier formulations are due to Branin [21] and Smale [164]. Keller's theoretical work is supported by Percell [140], who applied Sard's theorem to this type of homotopy.

Finally, the affine homotopy map is attributed to Wayburn and Seader [192]. Here, $f'(x)$ denotes the Jacobian of f with respect to x at x.

$$\rho(x,\lambda) = \lambda f(x) + (1-\lambda) f'(x_0)(x - x_0) \tag{40}$$

This homotopy map combines certain advantageous features of the fixed point homotopy map (single solution of easy function) and of the Newton homotopy map (scale invariance). Interestingly, Garcia and Gould [67] already mention this map as modified Merrill homotopy map and attribute it to Fisher et al. [62]. Recent applications primarily cite the former work however, and consequently this one is referenced herein, too.

The affine homotopy map can be considered a type of fixed point homotopy map. For the latter, the use of a scaling matrix G is common,

$$\rho(x,\lambda) = \lambda f(x) + (1-\lambda) G(x - x_0)$$

Then, for the special case $G = f'(x)$ the affine homotopy map results from the fixed point homotopy map with scaling matrix. The author considers the affine homotopy an independent map however, because of its scale-invariance and because G is in several cases a diagonal matrix [117, 197]. Also this distinction appears in literature [192].

These three generic homotopy maps have different characteristics and may (unless they are used in a probability-one homotopy method) prescribe ill-posed traces as already illustrated in figure 54. As Wayburn and Seader [192] point out, the Newton homotopy map has the advantage of being scale invariant and that homotopy traces may be contained in an open bounded set. However, as the simple problem $\rho(x,0) = 0$ may have more than one zero, more than one trace may exist for a given start iterate. In this case it cannot be guaranteed that a given track, which is selected via the choice of the start iterate, leads toward a solution of the actual problem (i.e., $\lambda = 1$). Instead, it may return toward a solution of the simplified problem (i.e., $\lambda = 0$). As Choi and Book [35] report, it is possible that such tracks become infinite loops, which do not cross $\lambda = 1$. Such tracks are called isolae. The fixed point homotopy map and affine homotopy map in turn have the advantage of allowing only a single zero of the easy problem $\rho(x,0) = 0$. Therefore, it is impossible to start tracking inside an isola. The fixed point homotopy map has a poor scaling property however, which is due to the connection of the independent and the depended variables [192]. The affine homotopy map in turn is scale invariant like the Newton homotopy map [192]. Fixed point and affine homotopy maps may in turn produce tracks, which diverge toward an infinite value of components of the vector of unknowns z. Such tracks cannot be traced numerically.

Some authors suggest to use complex arithmetic to connect homotopy tracks that are unconnected in real space. In particular, Wayburn and Seader [192] and Kuno and Seader [100] provide examples in which all real solutions are connected via complex space. The hypothesis that all real solutions can be found using complex arithmetic using fixed point homotopy maps and arbitrary starting points is shown to be invalid by the already cited work of Choi and Book [35], example 2. For completeness, note that Gustafson and Beris [76] show that this particular example can be solved by using one of the variables (i.e., one element of the vector of unknowns x) as a continuation parameter and allowing the rest of the variables as well as the homotopy parameter to be complex and / or infinite. Independently, Wolf and Sanders [195] analyze multi-parameter homotopy continuation methods utilizing complex continuation parameters in analog electronic circuit simulation. The convergence properties still pose an open issue. Such methods are not considered here. Instead, the focus is on the described probability-one homotopy method.

Due to the described characteristics of the tracks prescribed by homotopy methods, additional means were proposed in literature to make the procedure more efficient or more robust. First, as the traces may strike the boundaries of definition of the residual equations, or even diverge to $\pm\infty$, homotopy bounding methods are introduced following Paloschi [137, 138]. Here, the homotopy $\rho(x,\lambda)$ is used in the following bounding equation ($\pi(x)$ and $v(x)$ are auxiliary functions and $z^b(x)$ is a specific map of the present iterate x to some open and bounded subset

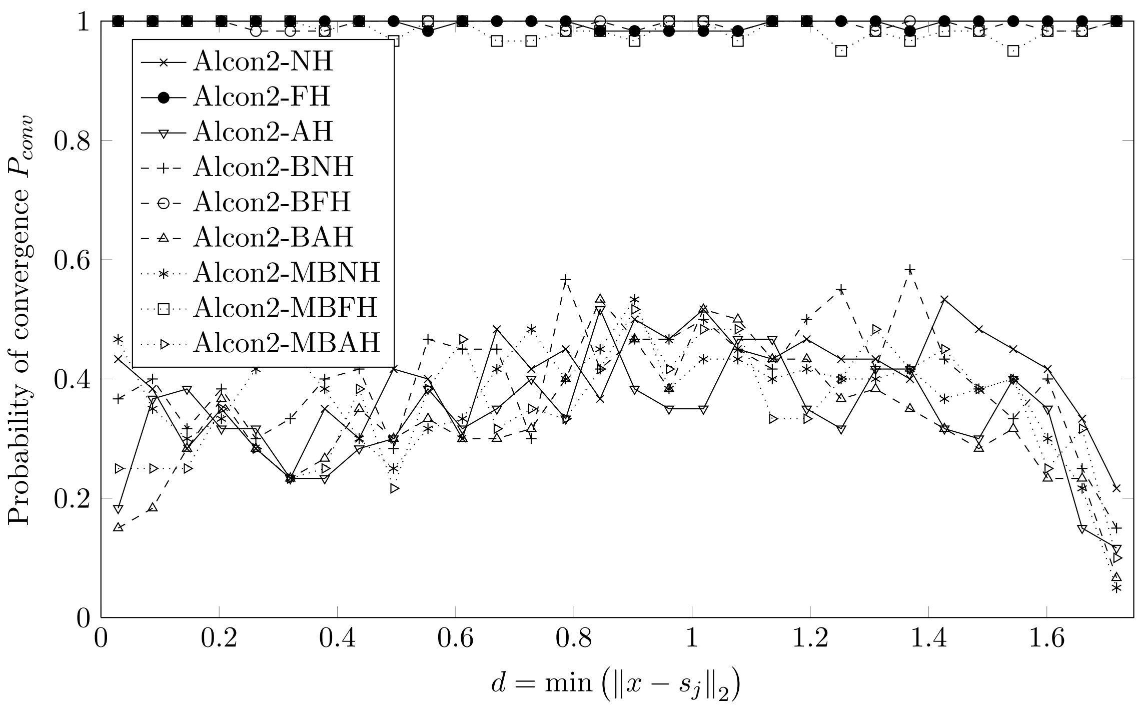


Figure 57: Robustness profiles for Operational Amplifier 741, generic homotopy maps (60 samples per bin)

of $\mathbb{R}^{2n_z+n_w}$).

$$\rho_b(x,\lambda) = \pi(x)\,\rho(x,\lambda) + v(x) - v(x^b)$$

Second, the bounding method of Malinen and Tanskanen [110] is considered. Here, the same bounding equation is used but the numerical tracing algorithm is applied in mapped variables. The authors claim some advantages such as superior robustness (in particular in connection with tight bounding zones). They introduce the infinity norm in their version of the auxiliary function $\pi(x)$ however, which destroys the property of C^2 smoothness of the original approach. For this reason, both bounding procedures are considered.

8.2 Illustrative examples

For illustration, a test case analyzed in chapter 7 is considered, the semi-conductor operational amplifier. Probability-one homotopy and plain homotopy methods are applied to this example. The continuation algorithm used was Alcon2 [44], which used an implicit change of parametrization with QR-decomposition and an automatic step length control for tangent continuation derived from theoretical considerations.

The results of the semi-conductor operational amplifier with respect to robustness are compiled in figure 57. In this robustness profile all generic homotopy maps are abbreviated (FH for fixed point homotopy map, NH for Newton homotopy map, AH for affine homotopy map with prefixes B for the plain homotopy bounding after Paloschi [137, 138] and MB for the modified homotopy bounding after Malinen and Tanskanen [110]).

For this particular test case, the algorithm results can be split into two groups. One group of algorithms provides poor robustness, partially even worse than the local gradient-based algorithms considered in section 7.3.1. The second group of algorithms provides nearly 100% probability of convergence independently of the quality of the start iterate. This is the group of algorithms using probability-one homotopy. Note that the homotopy maps are similar; the difference lies in the underlying theory. For the operational amplifier the residual equations were formulated such that coercivity, a relevant condition, is fulfilled [117]. Definitions of coercivity and corresponding theorems are introduced in the following chapter 9.

Furthermore, figure 57 indicates that for this particular test case the homotopy bounding algorithms do not influence the robustness much. In fact, all differences lie within measurement accuracy. Based on this result and similar findings for other cases, such bounding mechanisms are not addressed any further in this thesis.

The robustness profiles indicate the importance of coercivity in the context of homotopy methods. Theory guarantees convergence for problems satisfying the relevant theorems with probability-one. Figure 57 illustrates that indeed slightly less than 100% total probability of convergence is obtained for this particular test case. Additionally to the theoretic limitation in probability of convergence, challenges in the area of scientific computation are the reason for this observation. As Roychowdhury and Melville [152] report, practical probability of convergence may be deteriorated via ill-conditioned numerics leading to failure of path following and homotopy paths that continue forever (i.e., impractically long) without reaching $\lambda = 1$.

Therefore, the need for homotopy maps that are well-conditioned except near the solution becomes obvious from the example. Ill-conditioning is illustrated for the present example via the introduction of a diagonal scaling matrix G in the fixed-point homotopy map.

$$\rho\left(x, \lambda\right) = \lambda f\left(x\right) + \left(1 - \lambda\right) G\left(x - x_0\right) \tag{41}$$

The results are shown in figure 58. Each row i shows a plot of x_i over λ for different values of the diagonal entries on the scaling matrix G. From left to right, these start at 1 and are reduced with j from left to right to 10^{1-j}, i.e., 0.0001 on the right of the figure. Therefore, a column in the graphic shows all unknowns for a fixed value of the scaling matrix. Obviously, the scaling matrix can spread out the gradients over the continuation path. If this simple homotopy map was used in practice however, some heuristic was required to establish all individual entires on the scaling matrix, as the shown values are too high for variables x_2, x_5, x_{10} but too small for x_6, x_8. An example of such an heuristic is given in [65]. An alternative to setting parameters of generic homotopy maps to problem-specific values to arrive at well-behaved trajectories is the definition of problem-specific homotopy maps. This is addressed in more detail in chapter 10.

As seen on this illustrative example, probability-one homotopy can work well in practice. In the following chapters, approaches to implement such methods are analyzed in the context of general multi-domain physical modeling and equation-based, object-oriented modeling languages.

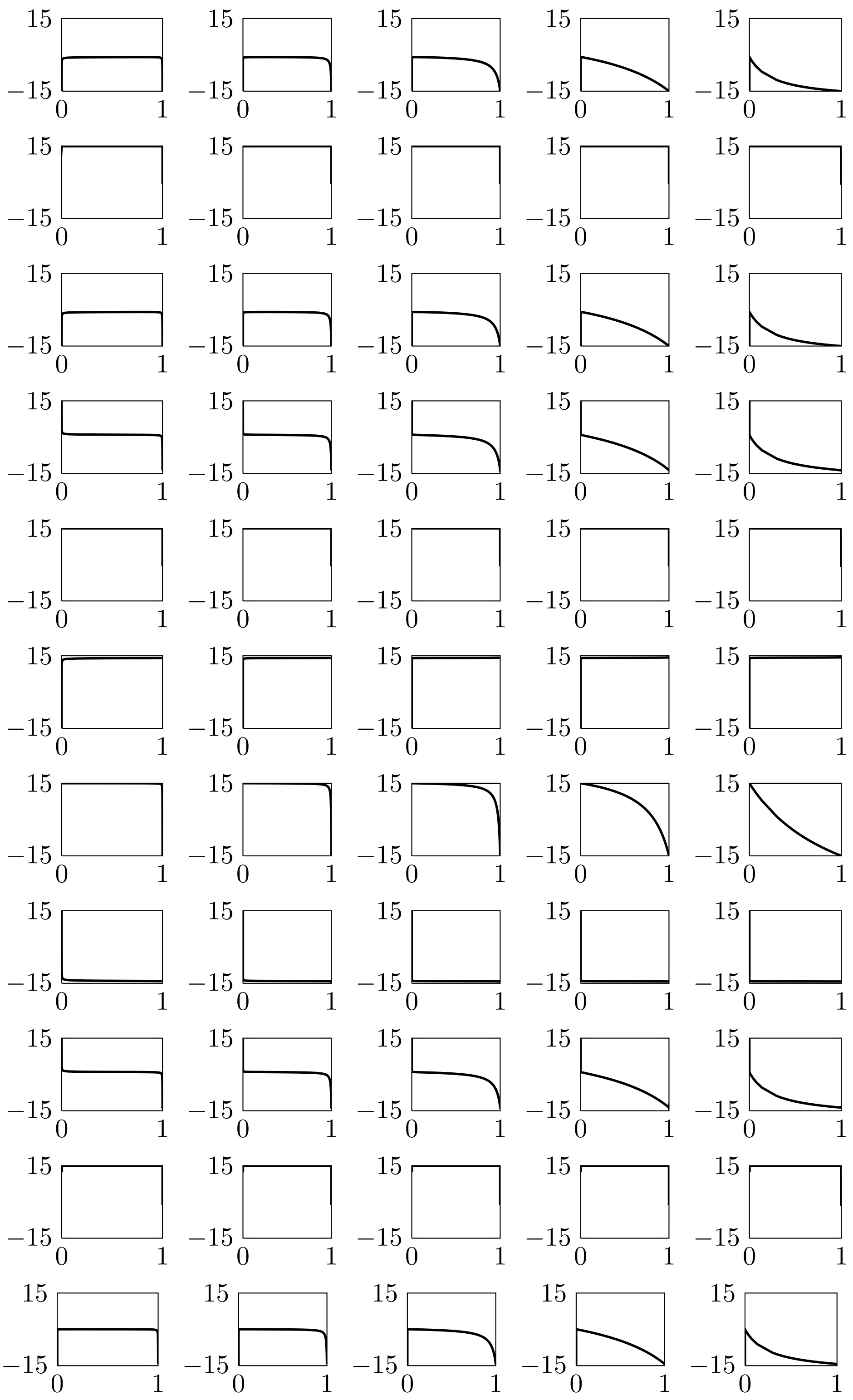


Figure 58: Influence of scaling matrix G on Fixed Point Homotopy Map and Operational Amplifier 741: Columns of homotopy traces for all 15 unknowns are shown for five different values of G from 1 to 10^{-4}.

CHAPTER 9

PROBABILITY-ONE HOMOTOPY FOR GENERIC MULTI-DOMAIN MODELING

Based on the introduction on homotopy methods in the previous chapter 8, it is concluded that ideally a probability-one homotopy should be implemented for multi-domain physical modeling and equation-based, object-oriented modeling languages. While homotopy in general has been used in system-level simulation of several physical domains (e.g., Wayburn and Seader [192] in chemical engineering), the results from analog electronic circuit simulation are particularly well-developed [177, 175, 176, 117, 115, 86, 73, 179, 111, 151, 197, 152].

In particular, Melville et al. [117] provide a rigorous application of the theory of Chow et al. [36] to analog electronic circuit simulation. Their work appears to indicate a potential route for such a proof in multi-domain physical modeling. Melville et al. [117] present two alternative arguments on coercivity conditions. The first one is based on the no-gain property of a large set of analog electronic components, the second one on passivity, an energy argument.

The no-gain property and related conditions were defined for several relevant cases by Willson [193]. An element has this property if "any connected network containing that element [...] possesses the [...] no-voltage-gain and no-current-gain properties". Passivity was formally defined by Wyatt et al. [196]. A short definition is given in [178, p. 78] as "at any operating point, the net power delivered to the element is non-negative".

While it is unlikely to find an analogous property to no-gain in all domains possibly covered by multi-domain physical modeling, energy is a general concept. Furthermore, conservation of energy is a fundamental principle and thus the second argument appears to hold considerable potential for a widened scope. The objective of this chapter is therefore to present an analysis of this proof involving energy arguments.

This analysis reveals that passivity is not sufficient for analog circuits due to boundary conditions. The first argument in [117] (no-gain) remains intact however and furthermore the typical circuit components happen to be sufficiently dissipative, so no harm is done to the theory for circuit applications. After all, as seen in chapter 8, the theory works well in practice. However, several physical domains exist in which the amount of dissipation is much lower. Therefore, the chapter eventually highlights that the energy argument cannot be used to generally proof coercivity for general multi-domain physical modeling.

9.1 Theory

Several authors contributed to global convergence theory for homotopy methods; in particular Scarf [155], Eaves [52], and Saigal [154]. Allgower and Georg [4, 5] provide an overview. Herein, the focus is set on the work of Chow et al. [36], which requires considerably weaker assumptions. To start, a number of theorems for probability-one homotopy is reviewed based on their work.

Following the problem statement in chapter 7, a root $x^* \in \mathbb{R}^n$ shall be found such that $f(x^*) = 0$ with $f : \mathbb{R}^n \to \mathbb{R}^n$. Alternatively, algorithms for fixed point problems $F(x^*) = x^*$ with $F : \mathbb{R}^n \to \mathbb{R}^n$ can be considered, because one type of problem can be easily transformed into the other, $F(x) - x = f(x)$.

In order to present the supporting theory, transversality to zero [191] is defined next.

Definition 1. *Let $U \subset \mathbb{R}^m$ and $V \subset \mathbb{R}^n$ be open sets, and let $\rho : U \times [0, 1) \times V \to \mathbb{R}^n$ be a C^2 map. ρ is said to be transversal to zero if the Jacobian matrix $\partial\rho$ has full rank on $\rho^{-1}(0)$.*

The map $\rho(a, \lambda, x)$ represents the embedding of the equations f to be solved for a root x^* as introduced informally in chapter 8. It is a key element in all homotopy methods. On top of the arguments listed before, it has an additional parameter dependency on a vector $a \in \mathbb{R}^m$ (in the remainder of this work $m = n$ and thus $a \in \mathbb{R}^n$). The Jacobian matrix $\partial\rho$ is a $n \times (2n+1)$ matrix and can be written as concatenation of three sub-matrices.

$$\partial\rho = \left[\begin{array}{ccc} \frac{\partial\rho}{\partial a} & \frac{\partial\rho}{\partial\lambda} & \frac{\partial\rho}{\partial x} \end{array}\right] \tag{42}$$

In the informal terms of chapter 8, the expression $\rho^{-1}(0)$ could be described as the set of roots of ρ given some λ. Formally, it is defined as follows.

$$\rho^{-1}(0) = \{(a, \lambda, x)\,| a \in \mathbb{R}^n, 0 \leq \lambda < 1, x \in \mathbb{R}^n, \rho(a, \lambda, x) = 0\} \tag{43}$$

The following theorem, which is based on differential geometry and the Parametrized Sard's Theorem, is a generic formulation of probability-one homotopy methods.

Theorem 1. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a C^2 map, $\rho : \mathbb{R}^n \times [0,1) \times \mathbb{R}^n \to \mathbb{R}^n$ a C^2 map, and $\rho_a(\lambda, x) = \rho(a, \lambda, x)$. Suppose that*

1. *ρ is transversal to zero, and, for each fixed $a \in \mathbb{R}^n$,*
2. *$\rho_a(0, x)$ has a unique nonsingular solution x_0,*
3. *$\rho_a(1, x) = f(x)$.*

Then, for almost all $a \in \mathbb{R}^n$, there exists a zero curve Γ_a of ρ_a emanating from $(0, x_0)$, along which the Jacobian matrix $\partial\rho_a$ has full rank. If, in addition,

4. *$\rho_a^{-1}(0)$ is bounded, then Γ_a reaches a point $(1, x^*)$ such that $f(x^*) = 0$. Furthermore, if $\partial f(x^*)$ has full rank, then Γ_a has finite arc length.*

This theorem is due to Watson [191] and is therefore called Watson's Theorem in this work. In order to apply this theorem, homotopy maps are constructed to meet prerequisites (2) and (3) by design. Prerequisite (1) may be trivial to verify for some homotopy maps and harder for others, in which λ and a are involved nonlinearly. According to [189], prerequisite 4 may be hard to verify and often is a "deep result" as (1)–(4) holding implies the *existence* of a solution to $f(x) = 0$.

A remark is in order on the statement of probability one. This characteristic of the theorem is inherited from the Parametrized Sard's Theorem and is motivated by probability of failure being 0 in the sense of a Lebesgue measure. Figuratively speaking, this means that the set of points leading to failure forms at most an $n-1$ dimensional manifold inside n-dimensional space, that is, it does not occupy any "volume".

Informally, Watson's Theorem can be understood as a statement on the probability of singularities along a continuation path. A bifurcation for instance may occur on a problem fulfilling this theorem. But a random variation of the parameter vector a will be sufficient to avoid the singularity on a following attempt (with probability one). On problems with more than one solution (such as the flip-flop circuit introduced earlier), this choice of a determines what solution the homotopy map converges to.

Next, different specialized theorems on probability-one homotopies are reviewed, which are considered for the application to the problem at hand.

9.1.1 Brouwer Fixed Point Theorem

Historically, solutions to difficult fix point problems were among the first successes of probability-one homotopy methods. As a root finding problem can easily be converted into a fix point problem, this theorem is considered for the problem at hand. Before stating the theorem some notation is introduced. Additionally to the real n-dimensional space $\mathbb{R}^n$ with norm $|\cdot|$, the interior of the unit ball is $\mathfrak{B}^n = \{x \in \mathbb{R}^n : |x| < 1\}$. Furthermore, the closed unit ball is $\overline{\mathfrak{B}^n} = \{x \in \mathbb{R}^n : |x| \leq 1\}$ and its surface $\mathfrak{S}^{n-1} = \{x \in \mathbb{R}^n : |x| = 1\}$.

Theorem 2. *Let $F : \overline{\mathfrak{B}^n} \to \overline{\mathfrak{B}^n}$ be a smooth map C^2 of the closed unit ball $\overline{\mathfrak{B}^n} \subset \mathbb{R}^n$ into itself. Define $\rho : \mathbb{R}^n \times (0,1) \times \mathfrak{B}^n \to \mathbb{R}^n$ as follows.*

$$\rho\left(a, \lambda, x\right) = \left(1 - \lambda\right)\left(x - a\right) + \lambda\left(x - F\left(x\right)\right) \tag{44}$$

For $a \in \mathfrak{B}^n$ we define $\rho_a\left(\lambda, x\right) = \rho\left(a, \lambda, x\right)$. Let Γ_a be the component of $\rho_a^{-1}\left(0\right) \cap \left(0,1\right) \times \overline{\mathfrak{B}^n}$ whose closure contains $\left(0, a\right)$. Furthermore, let $I - \partial F\left(x^*\right)$ be regular for every fixed point x^* (with the identity matrix I). Then, for almost every $a \in \mathfrak{B}^n$, the curve Γ_a is a smooth curve connecting $\left(0, a\right)$ to $\left(1, x^*\right)$ where x^* is a fixed point of F, $F(x^*) = x^*$. In particular, Γ_a lies inside $\left(0,1\right) \times \mathfrak{B}^n$ and has finite length.

This theorem is called the Brouwer Fixed Point Theorem. It was given as theorem 2.4 by Chow et al. [36]. This theorem is now adapted to the root finding problem. $f(x) = 0$ holds for the zero finding problem, which can be translated as follows.

$$F(x) = f\left(x\right) + x = x \tag{45}$$

The principal prerequisite of the theorem is that F be a map into itself, $F : \overline{\mathfrak{B}^n} \to \overline{\mathfrak{B}^n}$. A key challenge for the application of this method to the context of equation-based, object-oriented languages is therefore to provide a procedure to pose the equation system such that the sum of the residuals $f\left(x\right)$ and the unknowns x remains inside $\overline{\mathfrak{B}^n}$ for any unknown $x \in \overline{\mathfrak{B}^n}$.

For completeness, it is described how the prerequisites of this theorem fulfill the conditions of Watson's Theorem. With respect to assumption (1), it is noted that the Jacobian matrix $\partial\rho$ has been defined in equation (42). If it has rank n on $\rho^{-1}\left(0\right)$ then ρ is transversal to zero. This can be verified easily as $\frac{\partial\rho}{\partial a} = -(1-\lambda)\text{diag}\left(n\right)$ is a diagonal matrix, which, by definition, has full rank.

Prerequisite (2) of Watson's Theorem is fulfilled trivially. It is obvious that $\rho_a(0, x) = G(x) = x - a = 0$ has a unique solution $x_0 = a$.

Prerequisites (3) and (4), covering the boundedness and thus the existence of a solution, are a result of the properties of the map $f : \overline{\mathfrak{B}^n} \to \overline{\mathfrak{B}^n}$, i.e., the fact that it maps into itself. This is a fundamental result[1], which, in the context of probability-one homotopy methods, is derived in [36].

9.1.2 Inner Product Theorem

In their 1976 work, Chow et al. [36] also considered root finding problems directly. The corresponding theorem (theorem 4.1 in [36]) contains a condition on the inner product, which is why it is labeled Inner Product Theorem herein.

Theorem 3. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a smooth map C^2. Suppose that for $0 < r \in \mathbb{R}$ the following holds.*

$$x^T f\left(x\right) \geq 0 \quad \text{when } |x| = r \tag{46}$$

Again, let Γ_a be the component of $\rho_a^{-1}\left(0\right) \cap \left(0,1\right) \times \mathbb{R}^n$ whose closure contains $\left(0, a\right)$. The homotopy map $\rho\left(a, \lambda, x\right)$ is defined as follows.

$$\rho\left(a, \lambda, x\right) = \left(1 - \lambda\right)\left(x - a\right) + \lambda f\left(x\right) \tag{47}$$

Be $\partial f(x^*)$ regular in every root x^*. Then, for almost any $|a| < r$, we have a root x^* in f with $|x^*| \leq r$. Furthermore, the smooth homotopy track Γ_a connects $\left(0, a\right)$ to $\left(1, x^*\right)$ with finite length.

Note that $x^T f\left(x\right) \geq 0$ has to be fulfilled for any x with $|x| = r$, not only for some root x^*. Therefore, the residual equations $f\left(x\right) = 0$ cannot be utilized, as they are only valid for roots, not all x.

[1]Note that for the fix point perspective $\rho_a(1, x) = -F(x)$, i.e., not $\rho_a(1, x) = F(x)$ as when using one of the other homotopy maps considered in this thesis.

Consider again the four prerequisites for Watson's Theorem. Assumption (1) is fulfilled, because the sub-matrix $\frac{\partial \rho}{\partial a}$ of the Jacobian equation (42) is identical to the one using the homotopy map of the Brouwer Fix Point Theorem and has rank n. Assumption (2) is fulfilled as $\rho_a(0, x) = G(x) = x - a = 0$ has a unique solution $x_0 = a$. The same holds for prerequisite (3); substitution of $\lambda = 1$ into equation (47) yields $\rho_a(1, x) = f(x)$. Finally, the boundedness and therefore existence of solution is a fundamental result of the inner product condition equation (46). A proof of the latter is beyond the scope of this thesis and the reader is referred to [36].

9.1.3 Watson's Theorem using a standard homotopy map

Alternatively to using the one of the two anterior theorems, one may choose to apply Watson's Theorem directly. For this purpose a suitable homotopy map $\rho(a, \lambda, x)$ has to be defined. One obvious option is to use the standard convex combination of $f(x)$ and $g(x)$ with $g(x)$ linear as defined in equation (47).

In this case, prerequisite (1) is fulfilled because the sub-matrix $\frac{\partial \rho}{\partial a}$ of the Jacobian equation (42) has rank n. Similarly, prerequisite (2) is fulfilled as $\rho_a(0, x) = x - a = 0$ has a unique solution. Furthermore, applying $\lambda = 1$ in equation (47) results in $\rho_a(1, x) = f(x)$, which fulfills prerequisite (3). Prerequisite (4), i.e., $\rho_a^{-1}(0)$ is bounded, practically requires that the solution x to $\rho_a(\lambda, x)$ does not grow beyond a specific limit[2], which need not be known explicitly—only its existence is relevant. This prerequisite has to be considered in context with the particular set of residual equations $f(x)$.

9.1.4 Polynomial systems

For any nonlinear equation system consisting of polynomials only, an established solution method exists based on probability-one homotopy methods [194, 165]. Due to the the characteristics of the application at hand, this body of theory cannot be utilized but is mentioned here for completeness.

9.1.5 Summary

Three suitable theorems were identified for the application of globally convergent probability-one homotopy methods to general systems of nonlinear algebraic equation systems. They are:

Brouwer Fix Point Theorem Application of this theorem requires the following prerequisites to be met:

1. The homotopy map ρ and thus the residual equations $f(x) + x$ are C^2,
2. The residual equations are posed such that $f(x)+x : \overline{\mathfrak{B}^n} \rightarrow \overline{\mathfrak{B}^n}$ be map of the closed unit ball into itself.

Then, the fix point homotopy equation (44) will be successful for almost any $a \in \mathfrak{B}^n$. Furthermore, if $I - \partial F(x^*)$ is regular for every fixed point x^*, then Γ_a is of finite length. All other prerequisites of Watson's Theorem are fulfilled by the construction of the homotopy map or result from the given two assumptions.

Inner Product Theorem Application of this theorem requires the following assumptions to hold:

1. The homotopy map ρ and thus the residual equations $f(x)$ are C^2,
2. The residual equations are posed as a map $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$, for which $x^T f(x) \geq 0$ holds for any $|x| = r$ with $0 < r \in \mathbb{R}$.

[2] Notably, there are similarities with the Brouwer Fixed Point Theorem. As both may employ different homotopy maps, they are treated as separate options in this work.

Then, the root finding homotopy equation (47) will be successful for almost any $a \in \mathbb{R}^n$ with $|a| < r$. Furthermore, if $\partial f(x^*)$ is regular in every root x^*, then Γ_a will be of finite length. All other prerequisites of Watson's Theorem are fulfilled by the construction of the homotopy map or result from the given two assumptions.

Watson's Theorem using a standard homotopy map Application of the theorem requires the following assumptions to be fulfilled:

1. The homotopy map ρ and thus the residual equations $f(x)$ are C^2,
2. The zero set $\rho_a^{-1}(0)$ is bounded.

Then, the root finding homotopy equation (47) will be successful for almost any $a \in \mathbb{R}^n$. Furthermore, if $\partial f(x^*)$ is regular in every root x^*, then Γ_a will be of finite length. All other prerequisites of Watson's Theorem are fulfilled by the construction of the homotopy map.

Once more, it is highlighted that this list of theorems is not complete and that other theorems are in use today for probability-one homotopy. For example, Yamamura et al. [197] use a theorem of Garcia and Zangwill [199].

9.2 Application to multi-domain modeling

In the previous section, different theorems for probability-one homotopy have been introduced. From the preceding discussion it became evident that such methods cannot be applied directly to randomly posed equation systems. Instead, they have to be formulated carefully in order to fulfill the conditions of one of the given theorems (or any other suitable one). In the remainder of this chapter, procedures are analyzed that are suitable to do so for generic models of physical systems. Herein, the proofs utilized by the analog electronic circuit simulator community receive particular attention.

A common prerequisite is the C^2 smoothness of the residual equations. This has to be met in any case and is therefore not elaborated any further.

9.2.1 The Inner Product Theorem and passivity

When utilizing the Inner Product Theorem, a key condition on the problem is that $x^T f(x) \geq 0$ holds for any $|x| = r$ and $0 < r \in \mathbb{R}$. When considering generic models of physical systems, a single overarching principle comes to mind, which may be leveraged in order to fulfill this theorem: conservation of energy. Exclude boundary conditions for a moment. Then, if a component is formulated properly, the sum of the powers delivered to any component has to be zero (conservation of energy) or positive (dissipation of energy). If the residual equations and the vector of unknowns are set up such that $x^T f(x)$ is a sum of powers delivered to the components, then the sum of them will be zero or positive, i.e., $x^T f(x) \geq 0$ for any $|x| = r$.

Melville, Trajkovic and co-workers [117, 177] provided such a proof for analog electronic circuit simulators. In this domain, the energy criterion is called passivity, which was defined in the introduction of this chapter 9. Therefore, this section can be understood as an analysis of whether this type of coercivity argument can be generalized to physical modeling in different domains.

9.2.1.1 Topologies without boundary conditions

Consider the connection set illustrated in figure 59, which can be considered a part of some general model topology. To each connection set, a vector of node potentials is assigned, whose length is the number of pairs of potential and flow variables in the corresponding connector definition. In the figure, three components are connected in connection set i, each designated with a number 1 to 3. Each of these components has a vector of flow variables, whose length is

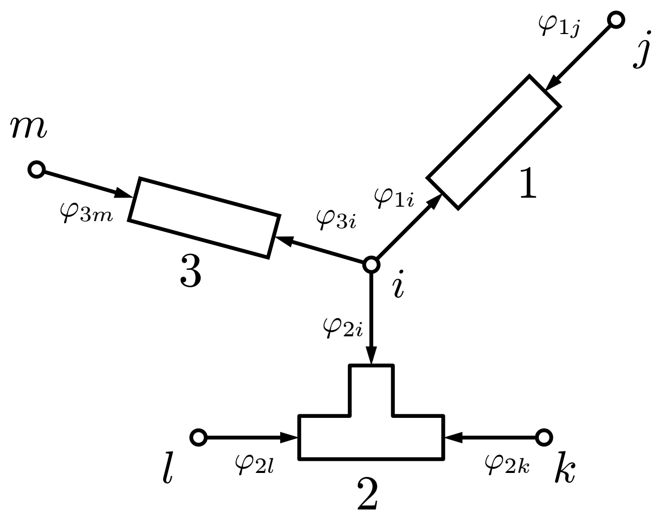


Figure 59: A general connection set without boundary condition

again equal to the number of pairs of potential and flow variables in the connection set. These flow variables have indices referring to the component (e.g., 3) and the node (e.g., i), e.g., φ_{3i}.

In a multi-domain physical model each of the nodes can implement interactions in a different physical discipline (e.g., thermal conduction, mechanical connection). Independently of this, each of these interactions can be described by pairs of potential and flow variables in such a way that the product of potential and flow is the power delivered to a component at the corresponding connector. For several exemplary physical domains, such pairs are given in table 8 below. It is admitted that these pairs are, in some cases, unconventional. For translatory mechanics, the given pair is only reasonable for a subset of mechanical systems for which position is not relevant. Otherwise, the position vector or transformation matrix has to be used as potential. In that case, the product of potential and flow is not power anymore. Likely, the convective transport of energy (i.e., enthalpy flow) is split for the unidirectional fluid flow into the sum of flow work and convective transport of inner energy. It is highlighted that this work considers these pairs as working hypotheses in order to assess the applicability of certain homotopy methods. They are not understood as universally preferable pairs of potential and flow variables. Furthermore, it is noted that it is not necessary to actually define these variables on the connectors or component interfaces; instead, these types of pairs may be highlighted using annotations.

Physical domain	Potentials x	Flows φ
Translational mechanics	Velocity v	Force F
Rotational mechanics	Angular velocity ω	Torque τ
Electrical circuits	Voltage V	Current I
Heat transfer	Temperature T	Rate of entropy change $\dot{S}$
Unidirectional thermo-fluid	Pressure p, inner energy u	Volumetric flow rate $\dot{V}$, mass flow rate $\dot{m}$

Table 8: Potential and flow variable pairs whose product is power or heat flow

Next, the equation system is posed. As unknowns x the node potentials are chosen. The residual equations $f(x)$ shall each be the sum of the flows for the different connection sets, which have to be zero according to potential and flow semantics. Then, for node i, $\varphi_i(x) = \varphi_{1i} + \varphi_{2i} + \varphi_{3i}$. Note that this equation for node i may be a vector equation if the number of pairs of potential and flow variables in the connector employed in node i is larger than one.

This leads to the following value of the inner product of the vector of unknowns with the transpose of the residual equations.

$$xf(x)^T = \overbrace{x_i\varphi_{1i} + x_i\varphi_{2i} + x_i\varphi_{3i}}^{x_i f_i(x)^T} + \\ x_j\varphi_{1j} + x_k\varphi_{2k} + x_l\varphi_{2l} + x_m\varphi_{3m} + \ldots \tag{48}$$

Based on the rationale outlined above, P_r is introduced, the power delivered to a component r. For physical models these powers have to be non-negative, e.g., $P_1 = x_i\varphi_{1i} + x_j\varphi_{1j} \geq 0$. Rearranging equation (48), the following equation is obtained.

$$xf(x)^T = P_1 + P_2 + P_3 \ldots \tag{49}$$

Here, $P_1 + P_2 + P_3 \geq 0$. Therefore, the inner product condition is fulfilled for the topological elements shown in figure 59. As this argument can be repeated for all possible connections sets, the inner product condition is fulfilled for general topologies without boundary conditions.

9.2.1.2 Topologies with boundary conditions

In order to cover practically relevant topologies, boundary conditions have to be addressed. Consider figure 60, which shows a part of some topology that features an independent potential boundary condition at connection set m. Independent potential boundary conditions are the only boundary conditions analyzed at this point.

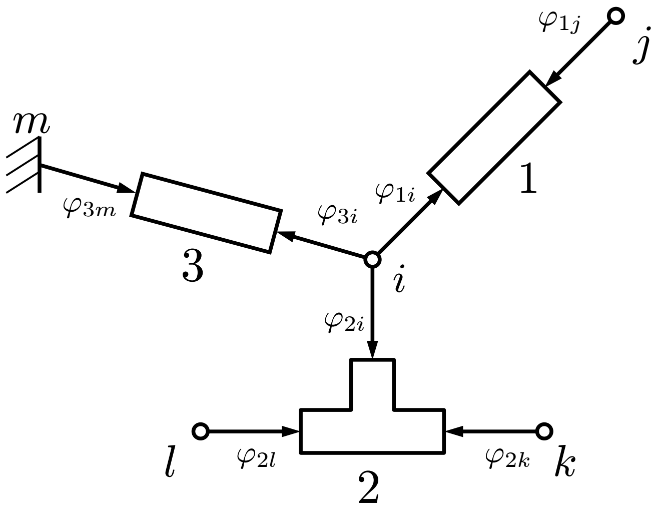


Figure 60: A general connection set with boundary condition

As the potentials in connection set m are not unknown anymore, no corresponding x_m is included in x and no residual $\varphi_m(x)$ is included in $f(x)$. Consequently, the inner product changes as follows[3].

$$xf(x)^T = x_i\varphi_{1i} + x_i\varphi_{2i} + x_i\varphi_{3i} + \\ x_j\varphi_{1j} + x_k\varphi_{2k} + x_l\varphi_{2l} + \ldots \tag{50}$$

If it is again assumed that the power delivered to any component is conserved or (partially) dissipated, the inner product condition can be reduced to the following inequality for a single independent potential boundary condition (using the same algebra as in the previous section 9.2.1.1, and $P_i = 0$, the case imposing the strictest conditions).

$$x_i\varphi_{3i} \geq 0 \tag{51}$$

[3]At a first glance, one might be tempted to maintain x_m as an additional, artificial unknown in x. However, the corresponding residual equation would then be $\varphi_m(x) = \varphi_{BC,m} + \varphi_{3m} = 0$. Due to the presence of $\varphi_{BC,m}$ the imbalance in eqs. (50) and (51) is not solved. Furthermore, no equation is available for the flow $\varphi_{BC,m}$ by an independent potential boundary condition.

Melville, Trajkovic and co-workers [117, 177] assume that only resistive elements are connected to potential boundary conditions. For resistive elements, the following equation holds with $k \in \mathbb{R}^+$.

$$\varphi_{3i} = k\,(x_i - x_m) \tag{52}$$

In particular, for resistors in electronic circuits, $k = 1/R$ is constant. At this point, general resistive components according to equation (52) are assumed to be connected to the boundary conditions and thus the latter equation is substituted into equation (51).

$$x_i k\,(x_i - x_m) \geq 0 \tag{53}$$

Figure 61 illustrates for which x_i inequality (53) is fulfilled. For these regions the inner product condition is fulfilled. Note that the shown domains hold independently of the particular value of k as long as $k > 0$.

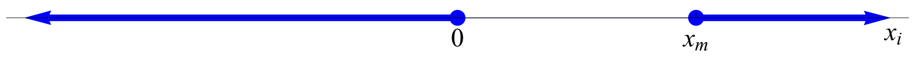


Figure 61: Inner product condition with independent potential source (inequality (53))

Melville, Trajkovic and co-workers [117, 177] finish their proof with inequality (53), which, according to their understanding, states that the inner product condition is satisfied. For the one-dimensional case illustrated in figure 61 the value of r can be made sufficiently large in order to fulfill the inner product condition. For the multi-dimensional case, the assumptions given in this article so far and employed by the cited authors do not suffice. Obviously, the inner product condition need not only hold for *some* x but for *all* $|x| = r$ with $r \in \mathbb{R}^+$ (see section 9.1.2). Consider for a moment the topology shown in figure 60 and assume that, in addition to component 3, n other components are connected forming loops such that each connection set links at least two components. Posing the equation system as suggested in the preceding paragraphs leads to an equation system with a vector of unknowns that includes x_i, x_j, and other unknowns for the other connection set potentials. For the moment, the focus is on x_i and x_j. All other potentials influence the power delivered to the n components, for which the inequality $P_i \geq 0$ holds. For this topology, the inner product condition leads to the following inequality.

$$x_i k\,(x_i - x_m) + \sum_n P_i \geq 0 \tag{54}$$

According to Melville, Trajkovic and co-workers [117, 177], assuming $P_i = 0$ does not violate the inner product condition. The regions in which this conditions is fulfilled under this assumption are illustrated in figure 62. Obviously, the radius r can be increased to any value without fulfilling the inner product condition. Only if an additional non-zero contribution is available via the P_i in equation (54) then the inner product condition can be fulfilled for all $|x| = r$.

Therefore, the proof given by Melville, Trajkovic and co-workers [117, 177] that passivity establishes the prerequisites of the inner product condition and thus the success of the probability-one homotopy method is not complete. Based on this, the question for the success of the method for analog electronic circuit simulators arises. There are at least two straight forward answers. First, investigations of electronic circuit devices show that several common electronic circuit elements are rather dissipative. Therefore, the power delivered to their ports is larger than zero, $P_i > 0$. Moreover, the dissipated power happens to increase with the potential difference at the component, which allows to compensate the negative contribution to the inner product when equation (53) is not fulfilled. For electronic circuits, this dissipated power eventually allows to satisfy the inner product condition for some (potentially larger) radius r. Second, the first argument of Melville, Trajkovic and co-workers [117, 177] (no-gain) remains intact, so no harm is done to the theory for circuit applications. After all, as seen in chapter 8, the theory works well in practice.

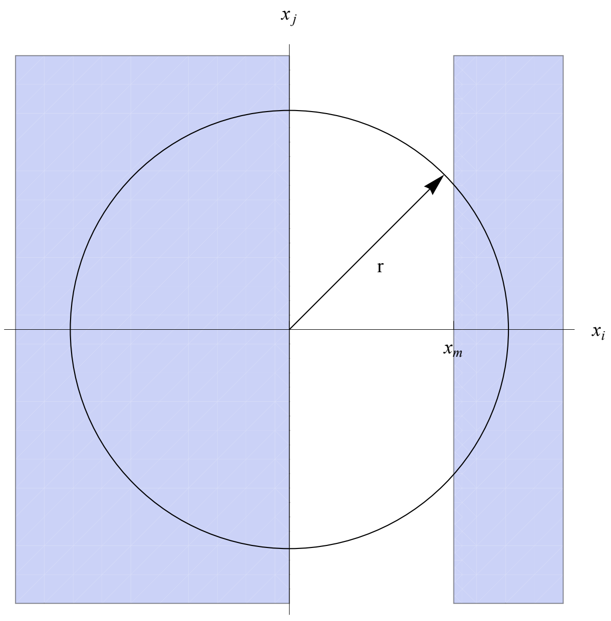


Figure 62: Inner product condition with independent potential source for two unknowns

To summarize, for the application of the energy conservation perspective to the inner product condition of probability-one homotopies and multi-domain physical modeling, energy conserving components are *not* sufficient. Similarly, general energy dissipating components that dissipate, e.g., an infinitely small amount of energy may not be sufficient either. Instead, a well-defined amount of power has to be dissipated (the net power entering the connection sets through components connected to boundary conditions).

Furthermore, it is not possible to fulfill this requirement for dissipative components in general multi-domain physical modeling. In several physical domains it is not possible to create such dissipative components (an example is given in the following section). As a consequence, the present energy argument for the inner product condition is considered not applicable to general multi-domain physical modeling.

9.2.1.3 Example

In order to illustrate the findings with respect to electronic circuits and general physical modeling, an example of the failure of the inner product condition is provided. Consider a serial network of three electrical or, respectively, thermal resistors shown in figure 63. At each end, independent potential boundary conditions prescribe the electric potentials V_L and V_R or, respectively, temperatures T_L and T_R.

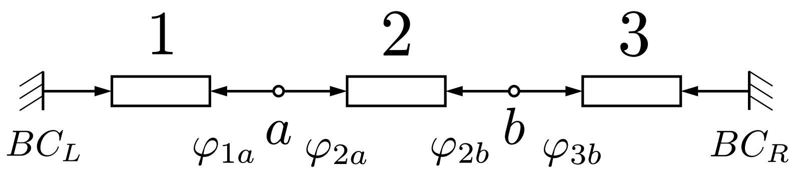


Figure 63: Exemplary network topology

The vector of unknowns is the vector of potentials at the connection sets without boundary condition.

$$x = \begin{pmatrix} x_a \\ x_b \end{pmatrix} \tag{55}$$

Similarly, the residual equations are the sums of the flows for each connection set.

$$f(x) = \begin{pmatrix} \varphi_{1a} + \varphi_{2a} \\ \varphi_{2b} + \varphi_{3b} \end{pmatrix} \tag{56}$$

First, consider the electrical circuit interpretation. The potentials and flows are substituted (according to table 8, set $x \equiv V$ and $\varphi \equiv I$). The inner product of the unknowns and the transpose of the residual equations is therefore as follows.

$$xf(x)^T = V_a I_{1a} + \underbrace{V_a I_{2a} + V_b I_{2b}}_{P_2} + V_b I_{3b} \tag{57}$$

Assuming resistances R_i for components $i = 1 \ldots 3$ and applying Ohm's Law, one arrives at the following equation for the inner product with $P_2 = (V_a - V_b)^2 / R_2$.

$$xf(x)^T = V_a \frac{V_a - V_L}{R_1} + P_2 + V_b \frac{V_b - V_R}{R_3} \tag{58}$$

The implications on the inner product condition are illustrated in figures 64 and 65. On the left figure, the value of the inner product is shown assuming $P_2 = 0$, which is admissible for passive components according to Melville, Trajkovic and co-workers [117, 177]. In order to identify the regions in which the inner product condition is fulfilled, the domain in which a negative value results is shown in gray (the darker, the larger the absolute value). Here, an inner product condition would be false. In the green regions the value is positive with the absolute value increasing from light green to green. To summarize, the inner product condition is fulfilled in the domain marked with colors ranging from white to green. In the given figure, a circle with an exemplary radius $r = 15$ is shown. From that, one can infer that the inner product condition can be fulfilled with a conservative component 2 (i.e., $P_2 = 0$). The value of P_2 for this particular resistor circuit is, as required for a conservative or dissipative component, always larger than or equal to zero. The correct inner product condition (with the actual non-zero value of P_2) is shown in figure 65. Again, the condition can be fulfilled assuming some finite radius r. So far, this example is in line with the findings reported by Melville, Trajkovic and co-workers [117, 177].

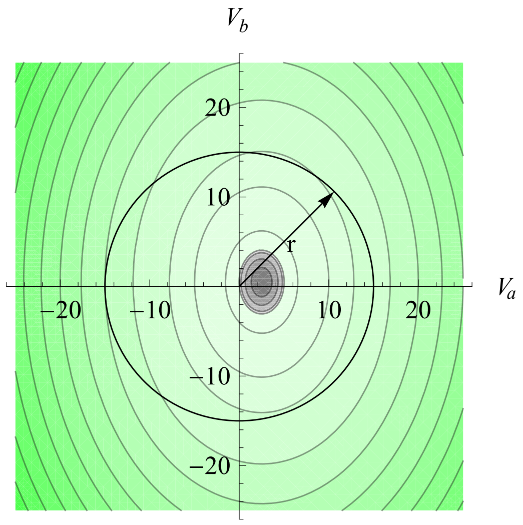


Figure 64: IPC assuming $P_2 = 0$

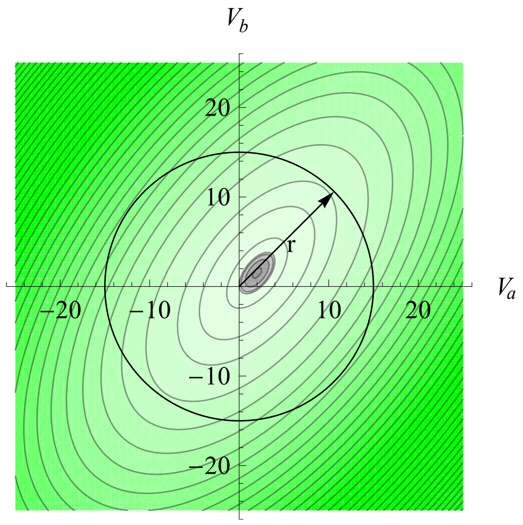


Figure 65: IPC with actual P_2

This is mostly the case for the thermal network interpretation of figure 63, too. In this case, $P_2 = 0$. In the following equations, the expression $\lambda_i A_i / (\Delta x_i)$ with thermal conductivity λ, area A and thickness Δx is abbreviated with the letter χ_i.

$$xf(x)^T = T_a \frac{\chi_1}{T_a}(T_a - T_L) + T_b \frac{\chi_3}{T_b}(T_b - T_R)$$

This inner product condition for the thermal conductor network is shown in figure 66. Pathologic behavior is observed in that no radius r can be chosen which leads to fulfilling the inner product condition. The reason is that the condition on resistive elements $k > 0$ (e.g., $k = \chi_1/T_a$) in equation (53) is not fulfilled if $T_a \leq 0$. To illustrate the key difficulty with the proof of Melville, Trajkovic and co-workers [117, 177], this complication is skipped by arguing that for temperatures in Kelvin, the values of zero and below are not practically relevant and therefore introduce a small minimum temperature $T_{min} \approx 10^{-9}$ for regularization. The generalized resistance is then regularized as

$$\chi_1/T_a \approx \begin{cases} \chi_1/T_a & \text{if } T_a \geq T_{min} \\ \chi_1/T_{min}\,(2 - T/T_{min}) & \text{else} \end{cases}$$

This leads to the inner product condition depicted to the right, for which a radius r exists to fulfill the condition. The parameter values used for figures 64 and 65 are $R_1 = R_2 = 1\Omega$, $R_3 = 2\Omega$, $V_L = 5V$, $V_R = 1V$. The ones used for figures 66 and 67 are $\chi_1 = 1W/K$, $\chi_3 = 2W/K$, $T_L = 12K$, $T_R = 0K$.

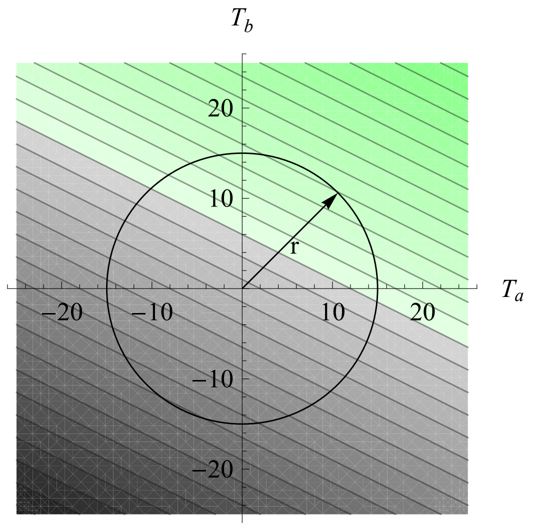


Figure 66: IPC as is

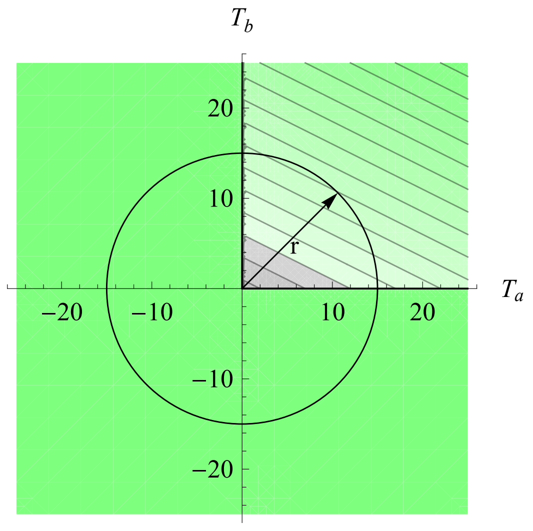


Figure 67: Regularized IPC

In order to illustrate the key weakness of the proof of Melville, Trajkovic and co-workers [117, 177], consider a serial network similar to the one shown in figure 63 but with *four* instead of three electrical or, respectively, thermal resistors in series. Again, independent potential boundary conditions prescribe the potentials.

Consider the electrical circuit interpretation first. The inner product of the unknowns and the transpose of the residual equations is therefore as follows.

$$xf(x)^T = V_a I_{1a} + \underbrace{V_a I_{2a} + V_b I_{2b}}_{P_2} + \underbrace{V_b I_{3b} + V_c I_{3c}}_{P_3} + V_c I_{4c} \tag{59}$$

According to the proof by Melville, Trajkovic and co-workers [117, 177], assuming passive components and thus $P_2 = 0$ and $P_3 = 0$ does not violate the inner product condition. An analogous illustration to figure 64 for the present case of four resistive elements is given in figure 68. Here, the two-dimensional circle with radius r is replaced by three-dimensional spheres with different radii r. On the surface of the spheres, information is given on whether the inner product condition is fulfilled or not by either drawing the surface (formerly green regions) or using exclusions (formerly gray regions). A cut perpendicular to the V_b axis results in an elliptical exclusion similar to the gray region shown in figure 64.

As can be inferred from the following figure 68, the contrary of what Melville, Trajkovic and co-workers [117, 177] suggest is the case. Even for large values of r, no compensation for the negative contribution to the inner product is available and thus the inner product condition is *not* valid for all $|x| = r$.

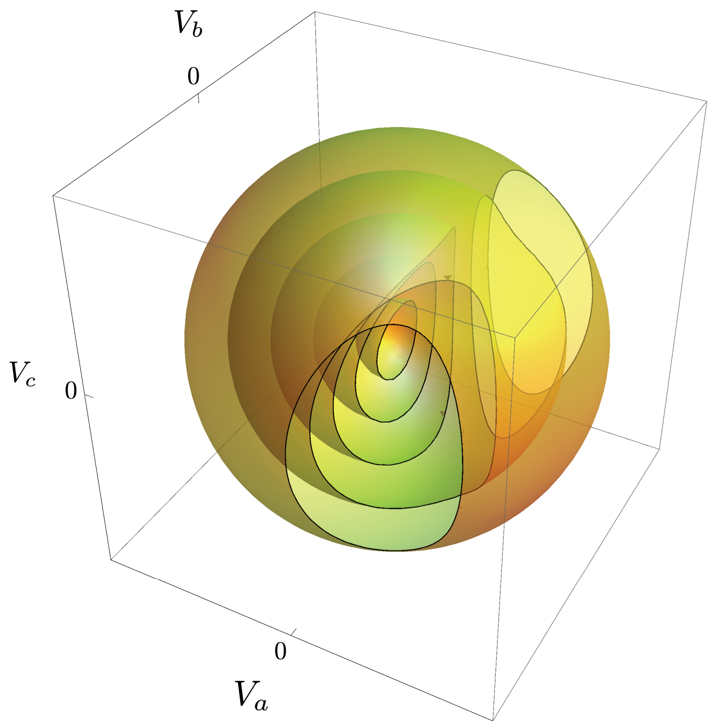


Figure 68: Inner product condition for example topology (electrical network, passive components 2 and 3)

As mentioned earlier, the power dissipated in actual electronic circuit components increases with the potential difference over the components. Therefore, as r is increased, the values of P_2, P_3 increase. In particular the following type of equality holds for the resistors.

$$P_2 = (V_a - V_b)^2 / R_2 \tag{60}$$

Only when considering these dissipative properties of the electronic circuit components, the inner product condition can be fulfilled. This is illustrated in figure 69 (here, an additional green surface marks the boundary between the regions, in which the condition is fulfilled, and regions, in which this is not the case). For the maximum value shown ($r = 50$), the inner product condition is finally fulfilled. The values used for figures 68 and 69 are $R_1 = 0.3\Omega$, $R_2 = R_3 = R_4 = 1\Omega$, $V_L = 35V$, $V_R = 0V$. The value of the radius was $r = \{10, 20, 30, 40, 50\}$.

Next, consider the thermal network interpretation of the four resistor topology similar to the one shown in figure 63. Again, the potentials and flows are substituted ($x \equiv T$ and $\varphi \equiv \dot{S}$). The inner product of the unknowns and the transpose of the residual equations is therefore as follows.

$$xf\left(x\right)^T = T_a\dot{S}_{1a} + \underbrace{T_a\dot{S}_{2a} + T_b\dot{S}_{2b}}_{P_2} + \underbrace{T_b\dot{S}_{3b} + T_c\dot{S}_{3c}}_{P_3} + T_c\dot{S}_{4c} \tag{61}$$

Fourier's Law allows to establish expressions for the rate of entropy change, e.g., $\dot{S}_{2a} = -\chi_2\left({}^{T_a}\!/\!{}_{T_b} - 1\right)$ with thermal conductivity times area divided by thickness χ_2. Substituting such expressions into the power delivered to the components, P_i, the result $P_2 = P_3 = 0$ emerges. With this and similar expressions for components 1 and 4, equation (61) is manipulated to arrive at the following final inequality for the inner product condition.

$$xf\left(x\right)^T = -\chi_1 T_a\left({}^{T_L}\!/\!{}_{T_a} - 1\right) - \chi_4 T_c\left({}^{T_R}\!/\!{}_{T_c} - 1\right) \geq 0 \tag{62}$$

The contribution of component 1 to the inner product will be negative whenever $T_a < T_L$. Analogously, the contribution of component 4 to the inner product will be negative whenever $T_c < T_R$. The sphere in three dimensions (due to three unknowns in the counterpart of equation (55) for four resistive components) includes points, for which both $T_a < T_L$ and

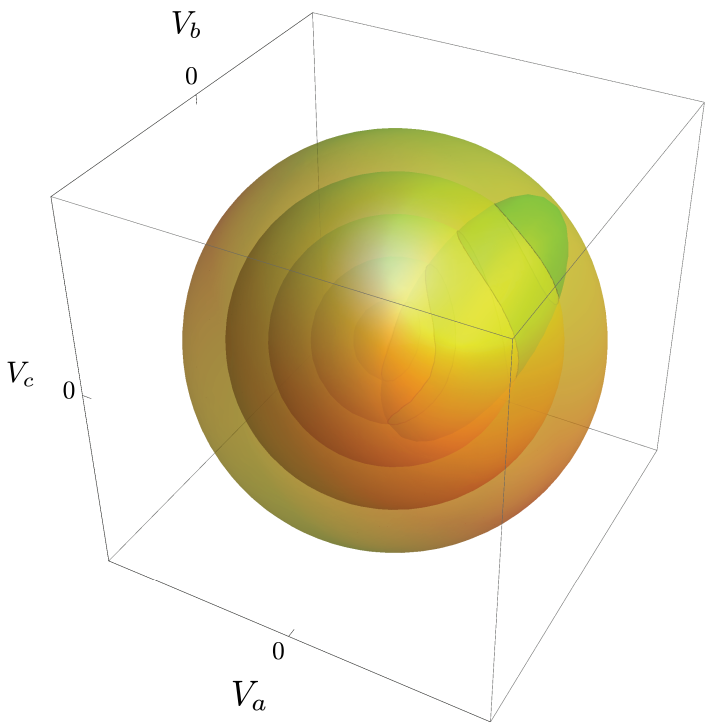


Figure 69: Inner product condition for example topology (electrical network, dissipative components 2 and 3)

$T_c < T_R$. As can be seen from equation (62) no positive contribution due to variation of T_b can be obtained. Therefore, the Inner Production Condition does *not* hold for all $|x| = r$. This is independent of the value of r and similar to the result observed for the electronic circuit interpretation with *passive* components[4].

Figure 70 illustrates where the inner product condition holds for different values of radius $r = \{10, 20, 30, 40, 50\}$ (this figure is the three-dimensional counterpart of figure 67). At $T_a = T_L$ and $T_c = T_R$, the condition is fulfilled; if either one of them is reduced the other has to be increased (the actual amount depends on the ratio of the coefficients χ_1/χ_4). If, however, $T_a < T_L$ and $T_c < T_R$, then the inner product condition is not fulfilled; furthermore, this is independent of the value of T_b. The figure uses the given regularizations for small and negative temperatures. Without it, the region, in which the inner product condition is fulfilled, is even smaller. The particular parameter settings used in figure 70 are $\chi_1 = 1W/K$, $\chi_4 = 1.5W/K$, $T_L = 20K$, and $T_R = 20K$.

9.2.2 Brouwer Fixed Point Theorem

Within the scope of the present work, no general physical principle was identified, which could be utilized to prove that $F(x)+x : \overline{\mathfrak{B}^n} \to \overline{\mathfrak{B}^n}$ in the context of multi-domain physical modeling.

9.2.3 Watson's Theorem and boundedness

No generic principle to fulfill coercivity in multi-domain physical modeling was identified. This holds for the Inner Product Theorem, the Brouwer Fixed Point Theorem and also Watson's Theorem. The latter, however, can be utilized to prove coercivity for individual physical domains. As already mentioned, this has been done by Melville, Trajkovic and co-workers [117, 177] for analog electronic circuits using the no-gain property.

[4]For completeness, note that while these trivial examples do not fulfill the inner product condition, they do meet the conditions of Watson's Theorem via the Second Law of Thermodynamics or the no-gain property of electronic resistors.

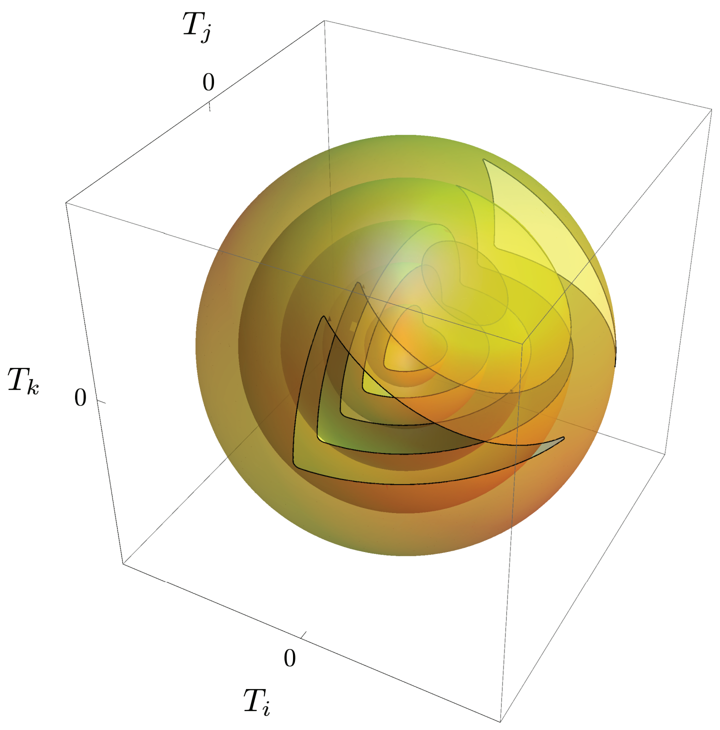


Figure 70: Inner product condition for example topology (thermal network)

9.2.3.1 Analog Electronic Circuits

Boundedness of the solution to a technical system may be utilized to fulfill the prerequisites of probability-one homotopy. In their articles, Melville, Trajkovic and co-workers [117, 177] do this in the context of analog electronic circuit simulators. By excluding a small set of components (components that do not have the so called no-gain property [193] such as amplifiers), the authors are able to guarantee that the vector of unknowns (the node potentials) remain within the sum of the source potentials and ground potential. This is a statement on boundedness, which can be utilized to fulfill the boundedness condition of Watson's Theorem for these applications.

If components shall be considered that do not fulfill the no-gain property, then properly modeling saturation is an approach to still meet the boundedness criterion. Saturation is a limiting behavior and may or may not be modeled for a component. Melville, Trajkovic and co-workers [117, 177] consider analog electronic components that do *not* fulfill the no-gain property, e.g., an amplifier. The amplifier receives a reference potential, which is amplified by some gain. In a simple model, the gain could be a constant value. The behavior would however be realistic only for small values of the reference potential. As soon as the amplified signal reaches the potential of the supply voltage of the amplifier, the amplifier starts to saturate. Therefore, independently of the value of the reference potential, the amplified potential is bounded if the component properly models saturation. Thus, the boundedness criterion of Watson's theorem may be fulfilled for analog electronic circuits with properly modeled no-gain components or ones that implement saturation.

9.2.3.2 Thermo-Fluid Dynamics

Willson [193] provided theorems to test whether two-port and three-port elements fulfill the no-gain property. These theorems can be generalized to other physical domains such as thermo-fluid dynamics. This allows proving that open networks consisting of wall friction elements and heat exchanger models fulfill the boundedness criterion of Watson's Theorem. A short example is given below for an isothermal hydraulics wall friction element. Here, pressure p serves as potential and volumetric flow rate $\dot{V}$ serves as flow variable. Then, the thermo-fluid dynamics counterpart of theorem 1 of [193] is as follows.

Theorem 4. *A two-terminal element is a no-gain element if and only if its behavior is characterized by a* Δp*–*$\dot{V}$ *relation for which* $\Delta p\dot{V} \geq 0$*, with* $\Delta p = 0$ *if and only if* $\dot{V} = 0$*.*

Assume that a pressure loss coefficient ζ is used to describe the wall friction. Therefore, total pressure loss due to wall friction is $\Delta p = 1/2\zeta\rho \cdot v \cdot |v|$. Here, ρ is density, v is fluid velocity. Immediately, $\Delta p = \alpha\dot{V} \cdot \left|\dot{V}\right|$ emerges as Δp–$\dot{V}$ relation for this element with $\alpha = 1/\left(2A^2\right)\zeta\rho > 0$ with cross-section area A. Obviously, $\Delta p\dot{V} \geq 0$ with $\Delta p = 0$ if and only if $\dot{V} = 0$.

Other components such as turbo compressors and turbines do not fulfill such a generalized no-gain property and have thus to be modeled, if at all, with saturation effects. The second law of thermodynamics provides a boundedness argument for thermal conduction[5].

9.2.3.3 Mechanics

Similarly to the other domains considered so far, models in mechanics may contain some elements that do fulfill boundedness criteria and some that do not. Quaternions are usually bounded because $q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$. Rotational angles may be considered bounded during initialization, i.e., $-\pi \leq \varphi \leq \pi$ if 2π is added or subtracted when reaching the given limits. After all, it is only possible to specify an angle in the range of 2π if it is not known how a configuration was formed. Translatory coordinates are, in case of non-mobile applications, bounded by finite length components, which are attached to some reference position in space. Some components may require modeling of saturation behavior (e.g., spring).

The boundedness criterion of Watson's Theorem provides a means to prove coercivity for problems in technical domains beyond electronic circuits. Such proofs have to be constructed for each physical domain and possibly technical problem separately.

At a first glance, this appears to be a limitation. After all, no "one fits all"-probability-one homotopy for multi-domain physical modeling has been devised in this chapter. Notably however, the most successful probability-one homotopy applications (e.g., [117, 152]) are reported in conjunction with problem-specific approaches to coercivity and the construction of homotopy maps. This was already hinted at by the initial results of the introduction to homotopy methods in section 8.2. In particular, successful maps capture the physics of the problem, are well-conditioned except near the solution, and possibly have λ deeply and nonlinearly embedded. With such problem-specific maps, probability-one homotopy methods hold potential to yield a relevant improvement over the robustness of local gradient-based algorithms.

[5]The second law of thermodynamics is not used explicitly in simulation models, but imposes the direction of heat flux.

CHAPTER 10

PROBABILITY-ONE HOMOTOPY AND PROBLEM-SPECIFIC MAPS

The objective of this chapter is to formulate a proposal for homotopy methods using problem-specific maps. For this purpose, the work of Melville et al. [117] on analog electronic circuits is considered to establish coercivity in multi-domain physical modeling. Melville et al. [117] present two alternative arguments. The one utilizing energy arguments such as passivity was analyzed in detail in chapter 9. It was not possible to apply it to generic multi-domain physical modeling. The other one is based on a domain-specific argument on boundedness (the no-gain property). In this chapter, problem-specific approaches to coercivity and the construction of homotopy maps are considered.

10.1 General problem-specific homotopy

10.1.1 Concept definition

In a number of articles [86, 197], *generic* homotopy maps are utilized for probability-one homotopy. Some references [117, 152, 69, 17] indicate however that homotopy maps that capture the physics of a problem, are well-conditioned except near the solution, and possibly have λ deeply and nonlinearly embedded may provide substantially higher potential for efficiency and robustness. Methods utilizing such maps are called general problem-specific homotopy methods in this thesis. Ideally, they are based on coercivity and are thus probability-one homotopy methods.

For a general problem-specific probability-one homotopy, it is proposed to

1. Derive the simplified system from the actual system of interest, and

2. Prove global convergence using a suitable theorem such as Watson's Theorem (see section 9.1).

The formulation of the homotopy map including a simplified system is problem-specific and allows modelers to infuse their knowledge about the physics of the problem into the way the equation system is solved. After all, as Roychowdhury and Melville [152] report, theoretical guarantees of probability-one homotopy on global convergence may still be deteriorated via at least two failure mechanisms, ill-conditioned numerics leading to failure of path following and homotopy paths that continue forever (i.e., impractically long) without reaching $\lambda = 1$. Therefore, the construction of general problem-specific homotopies is considered a question of engineering skill and to some extend an art.

Numerous approaches can be followed to derive the simplified system. For example, the residual equations may be linearized at some nominal operating point. Alternatively they can be split into a linear and a nonlinear part. Two approaches to do so have been considered by Mathis and Trajkovic [111, 179]. They assume that the problem is described by $Lx + N(x) = b$. Then, a simplified linear system may be constructed as $Lx = b$. Mathis and Trajkovic [111, 179] found that a homotopy map $f(x, \lambda) = (Lx - b) + N(\lambda x)$ results in a more efficient homotopy than $f(x, \lambda) = (Lx - b) + \lambda \cdot N(x)$ for their problems. Obviously, the former homotopy map can only be implemented if the nonlinear operator N satisfies $N(0) = 0$. According to the experience of the author, such formalistic approaches are often less powerful however than ones that deeply address the specifics of the actual problem (cf. first paragraph).

It is up to the user to implement homotopy maps, which adhere to the second step given above (prove global convergence). It is important to understand that such reasoning may yield formally similar homotopy maps but either a conventional homotopy method or a modern probability-one homotopy method.

It is highlighted that the proposal is compatible with the concept of object-orientation. Based on this, homotopy should be considered as a feature introduced by domain experts to selected key equations and models.

10.1.2 Application to equation-based, object-oriented modeling languages

As mentioned in section 7.1, equation-based, object-oriented modeling languages are meant to allow a declarative problem description. That is one in which no information has to be provided on how to solve the problem. Instead, the problem itself is described. The solution algorithms are encapsulated in the language compilers and simulators.

In order to be useful for practitioners, the notion of problem-specific homotopy maps has to be integrated into simulation tools. As before, the integration of the generic problem-specific homotopy is discussed in context of equation-based, object-oriented modeling languages. As described above, such languages enable a declarative description of the problem. The goal was thus to extend the declarative description to homotopies.

Using object-oriented modeling languages, one structures a model in terms of classes and objects. Therefore, it is proposed to specify a homotopy map $\rho_a(x, \lambda)$ on the level of the equation set of the model classes, too.

In order to implement the suggested approach, it is proposed to utilize two built-in operators, `homotopy()` and `lambda()`[1]. The operator `homotopy()` has one argument, an expression involving λ, which describes the problem-specific homotopy map. This expression is written using the second operator `lambda()` for each occurrence of variable λ. This operator may return a value in $[0, 1]$ during the numeric solution of algebraic equation systems and strictly 1 during the generation of simulation results. The operator `lambda()` may only be used inside an argument to `homotopy()`. The operator `homotopy()` has no functionality but to mark homotopy expressions. If the operator `lambda()` is used without an argument then a *single-phase homotopy map* is implemented. If integer arguments are used then a homotopy map is implemented, which consists of n phases, where n is the maximum over all arguments of the operator `lambda()`. For example, when using `lambda(1)` and `lambda(2)`, then a homotopy map is implemented in which λ_1 values are first swept from 0 to 1. After this is finished, λ_2 values are swept from 0 to 1. λ has to be swept from 0 to 1 during these sequential continuation runs of λ_i in order to infuse the random element required by theory[2].

In order to simulate a given model efficiently, simulation software may apply symbolic preprocessing steps. A step that has to be considered in the context of homotopy is equation sorting. A typical example of a sorting algorithm used for equation-based, object-oriented modeling languages is the Block Lower Triangular (BLT) transformation [55], using a graph-theoretical algorithm by Tarjan [169]. This algorithm re-orders and partitions the equations in a system so that they can be solved as much as possible in a sequential fashion. At the same time, algebraic loops are identified, i.e., the nonlinear algebraic equation systems, which are subject of this chapter.

Consider the following representative example, where equations and variables have already been re-ordered into BLT form, and the dependency on the homotopy parameter λ is made

[1]Note that this proposal is different from an earlier proposal of an operator `homotopy()` by Sielemann et al. [163]. In the earlier proposal, the operator was supposed to depend on two arguments, namely `actual`, an expression of the actual problem, and `simplified`, an expression corresponding to the simple problem. Following the earlier proposal, the compiler was then able to expand this operator according to some predefined homotopy map. As this original operator is less powerful for modern problem-specific probability-one homotopy, the present proposal is made.

[2]When using the homotopy operator with integer arguments, several distinct continuation runs have to be started as the trajectories will in general not be smooth at the joining point of traces in any λ_i and λ_{i+1} . In general, the trajectories will be continuous but not differentiable. Even if a continuation algorithm manages to "hop over" such a joining point, starting continuation separately will be more efficient.

explicit[3]:

$$\begin{aligned} 0 &= f_1(x_1, \lambda) \\ 0 &= f_2(x_1, x_2, x_3) \\ 0 &= f_3(x_2, x_3) \\ 0 &= f_4(x_4) \\ 0 &= f_5(x_4, x_5) \end{aligned}$$

Conceptually speaking, the continuation must be applied to the equation set as a whole. If the homotopy operator was not considered in the sorting process, then it would be possible to initially solve the first equation alone for x_1, then solve the second and the third for x_2, x_3, by using the already computed value of x_1, then the fourth for x_4, and finally the fifth for x_5, by using the already computed value of x_4. It is apparent that this strategy is not equivalent to applying the continuation to the entire system, since the second and third equations would be solved without any homotopy, by only considering the value of x_1 obtained for $\lambda = 1$. Such a blind application of the BLT partitioning must be avoided, as it leads to an unwanted result. The correct approach is instead to simultaneously solve via continuation all the equations that are either directly or indirectly influenced by the homotopy operator, in this case the first three ones.

Note that if any of the theorems introduced in section 9.1 is fulfilled, then a large fraction of potential problems is avoided. For example, no singular Jacobian matrix at $\lambda = 0$ can arise.

10.1.3 Test implementation

In order to validate the methodology, a test implementation was developed. It was based on the equation-based, object-oriented modeling language Modelica and the compiler Dymola® in versions 7.3 and 6.1. This test implementation utilized the LOCA continuation algorithms of Trilinos [82] and had the following properties.

- It provided three options for the treatment of the suggested homotopy operator. Normally, it was expanded according to a homotopy map. Alternatively, reduced equation sets were obtained by inlining the homotopy expression assuming $\lambda_i = 1.0$ or $\lambda_i = 0.0$. In the latter case, maximum structural simplification of the equation system resulted.

- The user was able to manually prescribe whether to use homotopy initialization or not. This was an important feature for library development and debugging, and may be useful for users, too (e.g., if a local gradient-based solver converges to a mathematically valid, but physically unreasonable or unstable solution or when a local gradient-based solver does not converge and a user does not want to wait at the start of each simulation until the software realized this).

- Verbose information on the homotopy was optionally provided, which was useful for library development and debugging. In particular, the homotopy traces were visualized. Like this, it was possible to reconstruct what happened during the solution of the simplified problems and the homotopy transformation.

Several implementation aspects such as automatic scaling and solver configuration via XML files have been described in section 7.3.1 and equally apply to this solver implementation.

10.2 Applications in analog electronic circuits

As mentioned in the introduction to chapter 9, the use of probability-one homotopy is particularly well-developed in the area of analog electronic circuit simulation [177, 175, 176, 117, 115, 86, 73, 179, 111, 151, 197, 152]. Before establishing a probability-one homotopy for thermo-fluid dynamic applications, results from this area are therefore reviewed.

[3]This analysis can be trivially extended to cases with more than one λ.

10.2.1 Variable Stimulus

Melville et al. [117] proposed the Variable Stimulus Probability-One Homotopy. Its homotopy map is as follows.

$$\rho(x, \lambda) = (1 - \lambda) G (x - a) + f(x, \lambda) \tag{63}$$

Here, the residual equations $f(x, \lambda)$ are posed in the nodal analysis form [46] and the node voltages of the nonlinear elements are multiplied by λ. Therefore, the influence of the nonlinear elements is removed from the circuit at $\lambda = 0.0$ and a linear circuit has to be solved. The matrix G defines the leakage from voltage sources of value a. These voltage sources and the associated vector a provide the random element needed in the probability-one approach. The leakage matrix G is a diagonal matrix with coefficients `Gleak`.

In order to substantiate that the Variable Stimulus Homotopy is globally convergent, Melville et al. [117] utilize Watson's Theorem as stated on page 96. Their arguments are as follows.

- The homotopy map (63) is twice continuously differentiable if and only if the device models used to assemble the residual equations in nodal form $f(x)$ are sufficiently smooth. It is assumed that this is fulfilled.
- The homotopy map ρ is transversal to zero as $\partial\rho/\partial a$ in (42) is a diagonal matrix with entries $-(1-\lambda) \cdot$ `Gleak`. For $\lambda < 1$, this matrix has full rank.
- $\rho_a(0, x)$ has a unique non-singular solution, because for $\lambda = 0$ the circuit consists of resistors and voltage sources only. Such a linear problem has a unique non-singular solution.
- $\rho_a(1, x) = f(x)$ because the leakage circuitry is removed completely at $\lambda = 1$ and each nonlinear device model is stimulated by the actual voltage.
- The zero set $\rho_a^{-1}(0)$ is bounded due to the no-gain property of the actual circuit and any partially stimulated circuit with leakage circuitry.

Additionally, Melville et al. [117] make the engineering assumption that the Jacobian of ρ_a has full rank at the solution x^*.

This Variable Stimulus Homotopy can be implemented on analog circuits using the proposed homotopy operator. First, a model of a NPN bipolar junction transistor is provided.

```
model NPN
  // Connectors
  Modelica.Electrical.Analog.Interfaces.Pin C "Collector";
  Modelica.Electrical.Analog.Interfaces.Pin B "Base";
  Modelica.Electrical.Analog.Interfaces.Pin E "Emitter";

  // Parameters
  parameter Real af = 0.995 "Forward current gain";
  parameter Real ar = 0.5 "Reverse current gain";

equation
  C.i = homotopy(iCollectorNpn(
            lambda()*B.v, lambda()*C.v, lambda()*E.v, af, ar));
  E.i = homotopy(iEmitterNpn(
            lambda()*B.v, lambda()*C.v, lambda()*E.v, af, ar));
  B.i = homotopy(iBaseNpn(
            lambda()*B.v, lambda()*C.v, lambda()*E.v, af, ar));
end NPN;
```

Listing 33: NPN transistor model using variable stimulus

Here, three functions `iCollectorNpn()`, `iEmitterNpn()`, and `iBaseNpn()` are used to establish the collector, emitter, and base currents respectively. In order to implement the leakage circuitry, a model instance of the following class is attached to each connection set.

```
model ElectricalNode
  // Connectors
  parameter Integer n=0 "Number of pins"
    annotation(Evaluate=true, Dialog(connectorSizing=true));
  Modelica.Electrical.Analog.Interfaces.Pin pin[n] "Pin array";

  // Parameters
  parameter Real Gleak "Leakage";
  parameter Real a "Random source voltage";

equation
  0 = -sum(pin[:].i) + homotopy((1.0 - lambda())*Gleak*(pin[1].v-a));
  for i in 1:n-1 loop
    pin[i].v = pin[i+1].v;
  end for;
end ElectricalNode;
```

Listing 34: Electrical node class

Note the negative sign in front of the summation of the currents of the pins. This is necessary as the nodal analysis form [46] summarizes the currents going into the components attached to a node.

According to the experiments of Melville et al. [117], the solution trajectories of this homotopy are "much smoother" than those of the generic homotopy map (38). Additionally, "the action is spread out evenly over all values of λ". The experiments conducted in the scope of this thesis arrived at similar results and support these findings.

10.2.2 Variable Gain

Melville et al. [117] also proposed the Variable Gain homotopy, which is similar to the Variable Stimulus homotopy but addresses bipolar transistors differently. Instead of multiplying the terminal voltages of all nonlinear elements by λ, the forward current gain α_F and the reverse current gain α_R are multiplied by λ. The simplified problem with $\alpha_F = 0$ and $\alpha_R = 0$ therefore consists of resistors, voltage sources, and diodes only.

$$\rho(x, \lambda) = (1 - \lambda) G (x - a) + f(x, \lambda\alpha) \tag{64}$$

Again, the residual equations $f(x, \lambda\alpha)$ are posed in the nodal analysis form [46]. Due to the diodes, the leakage circuitry is not necessary to avoid floating nodes. However, it is still included in this homotopy to provide the random element to avoid bifurcations [117].

Originally, the Variable Gain homotopy was implemented as a two-stage procedure. First, the Variable Stimulus homotopy was used to solve the $\lambda = 0$ problem of the Variable Gain homotopy. Then, continuation was started on the Variable Gain homotopy map (64) and the actual problem was solved. Today, Variable Gain Homotopy is commonly understood as what was originally labeled the "hybrid approach" in reference [117]. A local gradient-based algorithm is used to solve the $\lambda = 0$ problem and the continuation is applied directly on the Variable Gain homotopy map. The robust convergence of a local gradient-based algorithm on the $\lambda = 0$ problem is justified by Melville et al. [117] in case of norm-reducing algorithms (algorithms using so-called globalizations) by the work of Duffin [51]. The single-stage procedure is "two to three times faster than using homotopy alone" [117].

In order to show that the Variable Gain Homotopy is globally convergent, Melville et al. [117] again utilize Watson's Theorem. Their arguments are as follows.

- As before, the homotopy map (64) is twice continuously differentiable if and only if the device models used to assemble the residual equations in nodal form $f(x)$ are sufficiently smooth. Again, it is assumed that this is fulfilled.
- The homotopy map ρ is transversal to zero as $\partial\rho/\partial a$ in (42) is a diagonal matrix with entries $-(1-\lambda)\cdot$ `Gleak`. For $\lambda < 1$, this matrix has full rank.
- $\rho_a(0, x)$ has a unique non-singular solution, because for $\lambda = 0$ the circuit consists of resistors, voltage sources, and diodes only. Duffin [51] proved that such a problem has a unique solution.
- $\rho_a(1, x) = f(x)$ because the leakage circuitry is removed completely at $\lambda = 1$ and each nonlinear device model uses the nominal forward and reverse current gains.
- The zero set $\rho_a^{-1}(0)$ is bounded as Melville et al. [117] showed. This is due to the results of [193], who showed that bipolar transistors exhibit the no-gain property as long as the absolute values of the current gains remain less than or equal to one.

This Variable Gain Homotopy can be implemented on analog circuits using the proposed homotopy operator. Again, a model of a NPN bipolar junction transistors is given.

```
model NPN
  // Connectors
  Modelica.Electrical.Analog.Interfaces.Pin C "Collector";
  Modelica.Electrical.Analog.Interfaces.Pin B "Base";
  Modelica.Electrical.Analog.Interfaces.Pin E "Emitter";

  // Parameters
  parameter Real af = 0.995 "Forward current gain";
  parameter Real ar = 0.5 "Reverse current gain";

equation
  C.i = homotopy(iCollectorNpn(
            B.v, C.v, E.v, lambda()*af, lambda()*ar));
  E.i = homotopy(iEmitterNpn(
            B.v, C.v, E.v, lambda()*af, lambda()*ar));
  B.i = homotopy(iBaseNpn(
            B.v, C.v, E.v, lambda()*af, lambda()*ar));
end NPN;
```

Listing 35: NPN transistor model using variable gain

As before, three functions `iCollectorNpn()`, `iEmitterNpn()`, and `iBaseNpn()` are used to establish the collector, emitter, and base currents respectively. Instead of the terminal voltages, the current gains are multiplied with λ. The leakage circuitry can be implemented using model instances of the class listed in section 10.2.1 and is not repeated here.

According to Melville et al. [117], this is their fastest converging homotopy map. In particular, "the time required to solve a system of operating point equations with this homotopy [map] is not more than two to three times slower than the time required to solve the same equations by less widely convergent methods".

10.2.3 Example using Bipolar Junctions Transistors

Results are presented on probability-one homotopy using bipolar junction transistors and both the variable stimulus and the variable gain method on the operational amplifier example listed in table 7 of chapter 7. See figure 71 for results on using probability-one homotopy methods and figure 52 on using local gradient-based algorithms in comparison. In terms of computational

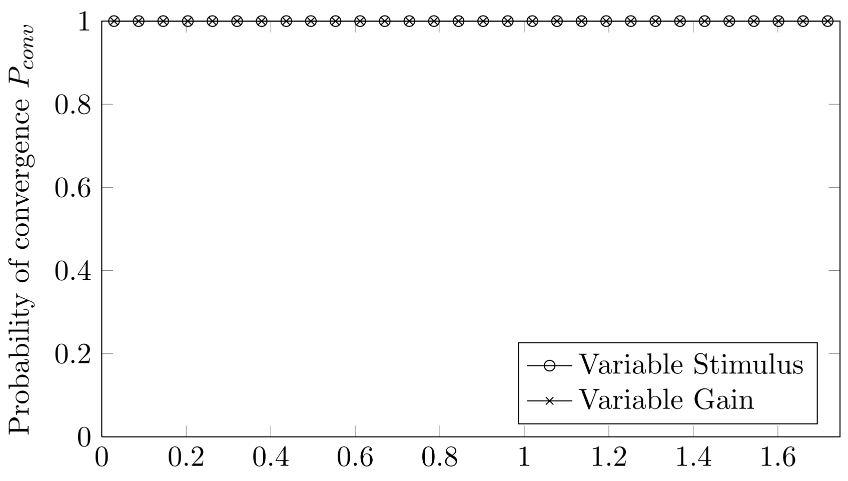


Figure 71: Robustness profiles for Operational Amplifier 741, problem-specific homotopy maps (60 samples per bin)

time, the variable gain homotopy map was superior to the variable stimulus homotopy. These results are in line with the findings of Melville et al. [117].

10.2.4 Arc-Tangent Shichman-Hodges

The Arc-Tangent Shichman-Hodges or ATANSH model was proposed by Roychowdhury and Melville [152, 151] for probability-one homotopy and large-scale integrated circuits of metal-oxide semiconductor field-effect transistors. Conceptually, it is similar to the Variable Gain homotopy in that it varies key nonlinearity in component models. The ATANSH model uses two homotopy parameters λ_1 and λ_2. Parameter λ_1 influences the drain–source driving point characteristic without affecting the gain. Parameter λ_2 in turn controls the transfer characteristic, i.e., the gain, without affecting the driving point characteristic.

$$\rho\left(x, \lambda, \lambda_1, \lambda_2\right) = (1-\lambda)\, G\left(x-a\right) + f\left(x, \lambda_1, \lambda_2\right) \tag{65}$$

The ATANSH MOS homotopy model is a single-piece model. The drain–source current I_{ds} is given via the following equation [152].

$$I_{ds} = \frac{\beta}{2}\left[V'_{gs}\left(V_{gb}, V_{db}, V_{sb}, \lambda_2, \lambda_1\right)\right]^2 h\left(V_{db}-V_{sb}, \lambda_1\right) \tag{66}$$

Roychowdhury and Melville [152, 151] remark that their probability-one homotopy map is a heuristic. In an attempt to justify its success, Watson's Theorem as stated in section 9.1.3 is considered.

- The homotopy map (65) is twice continuously differentiable if and only if the device models used to assemble the residual equations in nodal form $f(x)$ are sufficiently smooth. For the given MOS model this is fulfilled.
- The homotopy map ρ is transversal to zero as $\partial\rho/\partial a$ in (42) is a diagonal matrix with entries $-(1-\lambda)\cdot \texttt{Gleak}$. For $\lambda < 1$, this matrix has full rank.
- $\rho_a(0, x)$ has a unique non-singular solution, because for $\lambda = 0$ the circuit consists of resistors, voltage sources, and simplified MOS transistors only. At $\lambda_1 = 0$ and $\lambda_2 = 0$ the simplified MOS devices become two-terminal almost-linear resistors. It is a reasonable engineering assumption to assume that such a problem has a unique non-singular solution.
- $\rho_a(1, x) = f(x)$ because the leakage circuitry is removed completely and each MOS device model is restored to its original form.

- The zero set $\rho_a^{-1}(0)$ is bounded due to the no-gain property of the actual circuit and the simplified one with leakage circuitry and simplified MOS device models.

Additionally, one can make the engineering assumption that the Jacobian of ρ_a has full rank at the solution x^*.

This MOS model for probability-one homotopy can be implemented using the proposed homotopy operator. The following listing illustrates this on an n-channel MOS transistor.

```
model NMOS
  // Connectors
  Modelica.Electrical.Analog.Interfaces.Pin G "Gate";
  Modelica.Electrical.Analog.Interfaces.Pin D "Drain";
  Modelica.Electrical.Analog.Interfaces.Pin S "Source";
  Modelica.Electrical.Analog.Interfaces.Pin B "Bulk";

equation
  // Drain-source current according to ATANSH
  D.i = homotopy(idsNchannel(G.v-B.v, D.v-B.v, S.v-B.v,
            lambda(1), lambda(2)));
  S.i = -D.i;
  // Gate, source
  G.i = 0;
  B.i = 0;
end NMOS;
```

Listing 36: MOS-FET model using ATANSH

Function `idsNchannel()` implements equation (66) for this type of transistor. Note how the `lambda()` operator is used as described in section 10.1.2 with an integer argument. As Roychowdhury and Melville [152] first ramp λ_2 and then λ_1, their homotopy is implemented using λ_2 =`lambda(1)` and λ_1 =`lambda(2)`. The leakage circuitry can be implemented using model instances of the class listed in section 10.2.1 and is not repeated here.

Roychowdhury and Melville [152, 151] report that local gradient-based algorithms are two to three times faster than the ATANSH homotopy on average *if they converge*. They additionally provide data to show however that the ATANSH homotopy took “considerably less time to obtain the DC operating point of the circuit than conventional methods took to give up” on their test cases. This illustrates that the extra wall time is an acceptable price to pay for robust convergence on large-scale problems.

10.2.5 Example using Metal-Oxide-Semiconductor Field-Effect Transistors

The ATANSH method is applied on the inverter chain test case listed in table 7 of chapter 7. See figure 72 for results on using probability-one homotopy methods[4] and figure 53 on using local gradient-based algorithms in comparison.

10.3 Applications in thermo-fluid dynamics

In this section, a basic but robust probability-one homotopy for thermo-fluid dynamic applications with unidirectional flow is introduced and applied to the Air Distribution test case (see table 4 in chapter 7). This is a thermo-hydraulic example with pipes transporting gases under wall friction and heat transfer, heat loads in cabin volumes, fans and so on. A graphical overview is given in figure 73.

[4]The ATANSH homotopy map cannot be compared to the variable gain or variable stimulus homotopy maps. The reason is that they are specific to a type of transistor, either the MOSFET or the BJT.

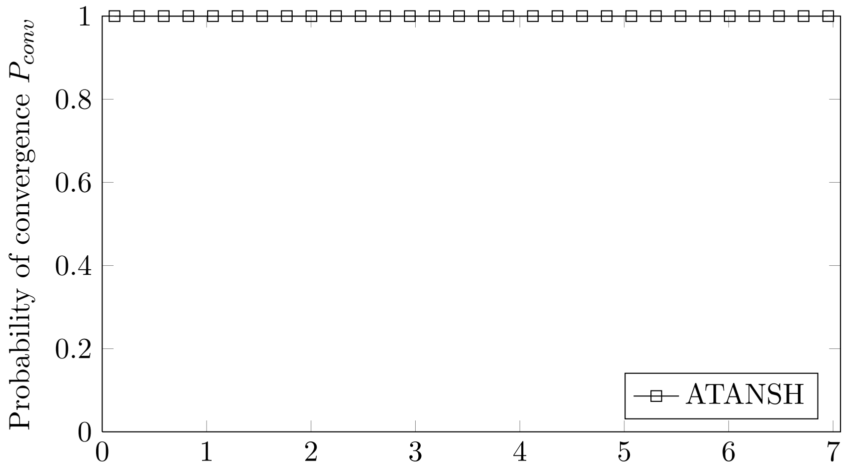


Figure 72: Robustness profiles for Inverter Chain, problem-specific homotopy map (60 samples per bin)

The notion of a nodal approach for probability-one homotopy is adopted. Therefore, the mass and energy balances are addressed in this context[5]. Pressure is a potential variable and thus the established approach of leakage circuitry used in sections 10.2.1, 10.2.2, and 10.2.4 can be applied trivially. Therefore, the components implementing the mass balance in the homotopy map are written in nodal form as follows.

$$\rho_{hyd}\left(x_{hyd}, x_{th}, \lambda\right) = (1-\lambda)\, G_{hyd}\left(a_{hyd} - x_{hyd}\right) + f_{hyd}\left(x_{hyd}, x_{th}, \lambda\right) \tag{67}$$

The subscript in $\rho_{hyd}\left(x_{hyd}, x_{th}, \lambda\right)$ refers to the mass balance as hydraulic part. Consequently, $x_{hyd} = p$, i.e., the vector of unknowns of this part of the homotopy map is the vector of unknown pressures. G_{hyd} is the hydraulic leakage, a_{hyd} is the vector of pressure values introducing the random element required by probability-one homotopy. The vector of residual equations $f_{hyd}\left(x_{hyd}, x_{th}, \lambda\right)$ for the hydraulic part are the mass balances, that is the sums of the connection set mass flow rates. Of course these residual equations also depend on x_{th}, the vector of thermal unknowns. These can be either temperatures or specific enthalpies. As it does only matter to the model of thermodynamic properties which one is used and all equations can be transformed accordingly, it is assumed without loss of generality that they correspond to temperature, i.e., $x_{th} = T$.

For the thermal part the situation is more involved. As the temperatures or specific enthalpies x_{th} are not potentials (note that their values are not equal over all connectors in a connection set in the general case), a mechanistic application of the concept to the energy balance will fail. Using such a simplistic approach to the components implementing the energy balance in the homotopy map would look like this.

$$\rho_{th,1}\left(x_{hyd}, x_{th}, \lambda\right) = (1-\lambda)\, G_{th}\left(a_{th} - x_{th}\right) + f_{th,1}\left(x_{hyd}, x_{th}, \lambda\right) \tag{68}$$

The subscript in $\rho_{th,1}\left(x_{hyd}, x_{th}, \lambda\right)$ refers to a first alternative of the energy balance as thermal part. G_{th} would be the thermal "leakage", namely a conductance. a_{th} would be a vector of temperature values introducing a random element. The vector of residuals equations $f_{th,1}\left(x_{hyd}, x_{th}, \lambda\right)$ for the thermal part are the plain energy balances, that is the sums of the connection set enthalpy flow rates.

As discussed in section 5.1.2, the energy balance cannot be used to uniquely determine a temperature or specific enthalpy in case of zero mass flow rate. As long as $\lambda < 1$ this problem can be handled conceptually by the conductance but as λ approaches 1 the homotopy map becomes increasingly ill-conditioned and finally singular at $\lambda = 1$.

[5]Note that in this section the node flow rates are opposites to the ones in classic nodal analysis. They are equal to the flow rates in a node model class.

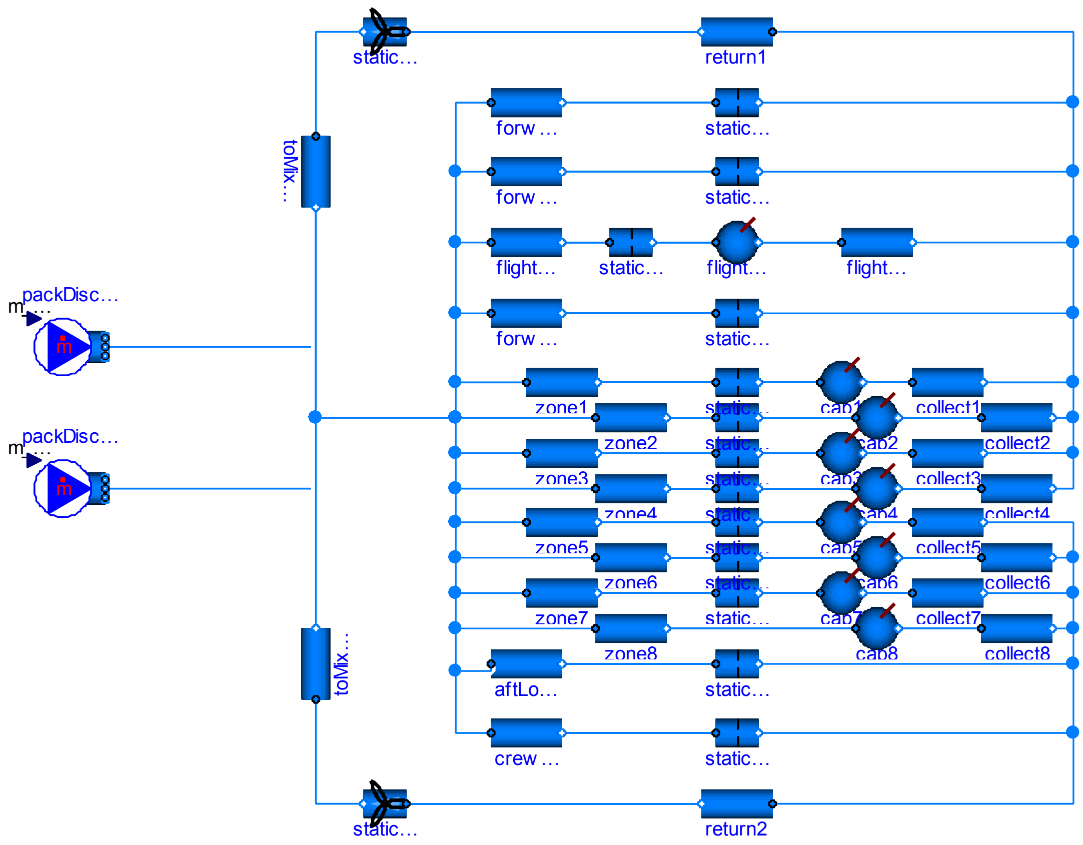


Figure 73: Air Distribution test case

Additionally, this homotopy map component does not balance the leakage enthalpy flow resulting from equation (67), i.e., carried by the mass flow rate $(1-\lambda)\,G_{hyd}\,(x_{hyd}-a_{hyd})$. Therefore, if for any component of x_{hyd} the pressure is low and therefore all enthalpy flow rates are positive then the energy balance requires that the conductive heat flow rate and thus the node temperature be unnecessarily high.

To address these problems, two alternative homotopy map components for the energy balance are considered. Both build on information implied by unidirectional flow, that is a classification of the connectors into inlets and outlets[6].

$$\rho_{th,2}\,(x_{hyd},x_{th},\lambda) = (1-\lambda)\,G_{th}\,(a_{th}-x_{th}) + f_{th,2}\,(x_{hyd},x_{th},a_{th},\lambda) \tag{69}$$

Here, a set of residual equations $f_{th,2}$ is used that also provides the random element. Therefore, the homotopy map can be used also without conductance, i.e., $G_{th}=0$.

$$\begin{aligned} f_{th,2}\,(x_{hyd},x_{th},a_{th},\lambda) =&(1-\lambda)\sum_{inlets}\left(\dot{m}_i^{+}\cdot h_{pT}\,(a_{hyd},a_{th})\right)\\ &+\lambda\sum_{inlets}\left(\dot{m}_i^{+}\cdot h_i\right)\\ &+h_{pT}\,(x_{hyd},x_{th})\left((1-\lambda)\,G_{hyd}\,(x_{hyd}-a_{hyd})+\sum_{outlets}\dot{m}_i^{-}\right)\end{aligned} \tag{70}$$

The first line prescribes an assumed enthalpy flow entering the connection set via the inlets at a specific enthalpy based on the random parameters a_{hyd} and a_{th}. The function $h_{pT}(p,T)$ returns

[6]Note that still the general case is considered in that the vector a_{hyd} is random over some interval. Therefore, the mass flow rates do change sign during initialization. But the energy balance is written based on assumed signs. The presented homotopy can be extended to bidirectional flow. Note that for this the Stream connector semantics do not have to be changed to a C^2 regularization of the `inStream()` operator, as the node models have one-to-one connections only. This extension to bidirectional flow is beyond the scope of this manuscript.

specific enthalpy based on pressure and temperature. As λ increases, a convex combination of this enthalpy flow rate and the one prescribed by the second line is used. This second line is the actual enthalpy flow rate based on the model topology. The last line in turn implements the enthalpy flow rate of fluid leaving the connection set. Here, the thermal node value x_{th} is used in the specific enthalpy computation. The mass flow rate leaving the connection set is the sum of the mass flow rates over the outlet connectors plus the mass flow rate due to leakage in equation (67). The superscripts $\pm$ on the mass flow rates indicate that they have been limited to a positive or negative epsilon flow using a C^2 regularization.

The third alternative homotopy map component for the energy balance is written in the dimension of a specific enthalpy. It can equally be used with and without conduction. Note that the physical dimension of G_{th} is changed in comparison to equations (68) and (69).

$$\rho_{th,3}\left(x_{hyd}, x_{th}, \lambda\right) = (1-\lambda)\, G_{th}\left(a_{th} - x_{th}\right) + f_{th,3}\left(x_{hyd}, x_{th}, a_{th}, \lambda\right) \tag{71}$$

The residual equations involving λ are as follows.

$$f_{th,3}\left(x_{hyd}, x_{th}, a_{th}, \lambda\right) = (1-\lambda)\, h_{pT}\left(a_{hyd}, a_{th}\right) + \lambda \frac{\sum\limits_{inlets} \dot{m}_i^+ \cdot h_i}{\sum\limits_{inlets} \dot{m}_i^+} - h_{pT}\left(x_{hyd}, x_{th}\right) \tag{72}$$

The homotopy map has been established in terms of the connection set equations. Optionally, one may create embeddings in the device models. For wall friction correlations, a convex combination of a secant approximation through some operating point and the actual wall friction correlation was successfully tested. Heat transfer may be established equally based on secant approximations or even zero heat transfer at $\lambda = 0$.

A note is in order on the second form of the thermal component of the homotopy map according to equation (69). If a random vector a_{hyd} over some interval is used, then the mass flow directions will in general not follow the prescribed flow direction assumed for unidirectional flow. For this case this form of the homotopy map is not as robust as form 3, equation (71). If a random vector a_{hyd} is established in a way that respects the required total pressure gradient to arrive at the assumed flow directions then this robustness issue does not arise. What however happens with the homotopy map of form 2, equation (69), in the former case and how can it be modified to improve robustness? Consider the simple case of a node model between higher upstream and downstream pressures according to a_{hyd} (for $\lambda = 0$). All $\dot{m}_i^+$ for the inlets will be finite and all $\dot{m}_i^-$ for the outlets will be small epsilon flows. Then, however, the sum of $\dot{m}_i^+$ for the inlets, all $\dot{m}_i^-$ for the outlets and the leakage flow will *not* sum up to zero. This imbalance will adversely affect the specific enthalpies in equations (69) and (70) and deteriorate robustness. A means to improve robustness of these corresponding homotopy maps is immediately obvious. If the energy balance is reformulated such that the mass flow rates used in it are always balanced, then it will be more robust. Obviously, this is the case if a modified total outlet flow is assumed.

$$\left((1-\lambda)\, G_{hyd}\left(x_{hyd} - a_{hyd}\right) + \sum_{outlets} \dot{m}_i^-\right) := -\sum_{inlets} \dot{m}_i^+ \tag{73}$$

Then, however, it immediately becomes obvious that this modified form 2 is equivalent to form 3.

In order to substantiate that the thermo-fluid homotopy is globally convergent, theorem 1 (Watson's Theorem) is applied. The arguments are as follows.

- The homotopy map based on components (67) and (69) or (71) is twice continuously differentiable if and only if the device models used to assemble the residual equations in nodal form $f(x)$ are sufficiently smooth. It is assumed that this is fulfilled.
- The homotopy map ρ is transversal to zero as $\partial\rho/\partial a$ with $a = [a_{hyd}; a_{th}]$ in (42) contains a diagonal matrix with entries $-(1-\lambda) \cdot G$ with $G = [G_{hyd}; G_{th}]$ if a conductance is used. If the conductance is not used, i.e., $G_{th} = 0$, then $\partial\rho/\partial a$ contains $-(1-\lambda) \cdot G_{hyd}$ for the

hydraulic part. For the thermal part, $\partial\rho/\partial a$ contains $(1-\lambda)\sum_{inlets} m_i^+ \cdot c_p$ in case of form 2 in equation (69) or $(1-\lambda)\,c_p$ in case of form 3 in equation (71). In any case $\partial\rho/\partial a$ and the Jacobian (42) have full rank for $\lambda < 1$.

- The homotopy map $\rho_a(0,x)$ has a unique non-singular solution, because for $\lambda = 0$ the circuit consists of adiabatic linear pressure loss models and boundary conditions only. Such a problem has a unique non-singular solution.

- $\rho_a(1,x) = f(x)$ because the balance equations are restored completely at $\lambda = 1$ and each device model exposes the actual behavior.

- For the hydraulic part, the zero set $\rho_a^{-1}(0)$ is bounded due to the no-gain property of the pressure loss correlations. See theorem 4 in section 9.2.3.2. Note that fans prescribe a bounded pressure difference Δp and can thus be considered as boundary conditions for the hydraulic part. The exact value of Δp obviously depends on the specific operating conditions and need not be known explicitly. It only matters that it is finite. For the thermal part, the zero set is bounded due to the Second Law of Thermodynamics.

For the remainder of this section, form 3 and no conductance are used. The homotopy map was implemented using the proposed homotopy operator. The code for a model class to be instantiated in each connection set is as follows.

```
model ThermoFluidDynamicsNode
  replaceable package Medium = PartialPureSubstanceMedium;

  // Connectors
  parameter Integer nInlets = 0 "Number of inlets"
    annotation(Evaluate=true, Dialog(connectorSizing=true));
  parameter Integer nOutlets = 0 "Number of outlets"
    annotation(Evaluate=true, Dialog(connectorSizing=true));
  Modelica.Fluid.Interfaces.FluidPort_a inlet[nInlets](
    redeclare package Medium = Medium);
  Modelica.Fluid.Interfaces.FluidPort_b outlet[nOutlets](
    redeclare package Medium = Medium);

  // Parameters
  parameter Medium.AbsolutePressure a_hyd "Random pressure";
  parameter Medium.Temperature a_th "Random temperature";
  parameter Real G_hyd "Leakage in hydraulic part"

  // Variables
  Medium.AbsolutePressure p "Pressure in node";
  SI.MassFlowRate m_flow_plus[nInlets] "Limited inlet flow";
equation
  // Hydraulic part
  for i in 1:nInlets loop
    inlet[i].p = p;
  end for;
  for i in 1:nOutlets loop
    outlet[i].p = p;
  end for;
  0 = homotopy((1-lambda())*G_hyd*(a_hyd - p) +
        sum(inlet[:].m_flow) + sum(outlet[:].m_flow));

  // Thermal part, form 3, no conductance
```

```
34   for i in 1:nInlets loop
35     // Hypothetical case
36     inlet[i].h_outflow = Medium.h_default;
37   end for;
38   for i in 1:nOutlets loop
39     // Actual case
40     outlet[i].h_outflow = Medium.h_pT(p, T);
41   end for;
42   0 = homotopy((1-lambda())*Medium.h_pT(a_hyd, a_th) +
43         lambda() * sum({
44             m_flow_plus[i]*
45             inStream(inlet[i].h_outflow) for i in 1:nInlets}
46           )/sum({m_flow_plus[i] for i in 1:nInlets}) -
47         Medium.h_pT(p, T));
48   m_flow_plus[:] = f(inlet[:].m_flow, ...);
49 end ThermoFluidDynamicsNode;
```

Listing 37: Thermo-fluid dynamics node class

This node model implements the homotopy map on the thermodynamic balance equations of mass and energy. In particular, equation (67) in lines 30 and 31 and equation (71) in lines 42 to 47. The implementation of the device models is straight-forward. In the following listing, the steady-state part of a simple dynamic pipe model is presented (the transient equations do not matter for initialization and are thus omitted for readability).

```
model Pipe
  replaceable package Medium = PartialPureSubstanceMedium;

  // Connectors
  Modelica.Fluid.Interfaces.FluidPort_a port_a[nInlets](
    redeclare package Medium = Medium);
  Modelica.Fluid.Interfaces.FluidPort_b port_b[nOutlets](
    redeclare package Medium = Medium);

  // Parameters
  parameter SI.Length diameter "Pipe inside diameter";
  parameter SI.Length length "Pipe length";
  parameter SI.Length Delta "Surface roughness";
  final parameter SI.Area heatTransferArea =
    Modelica.Constants.pi*diameter*length;
  parameter SI.Temperature T_amb "Ambient temperature";
  parameter SI.Pressure dp_nominal "Nominal dp";

  // Variables
  SI.SpecificEnthalpy dh "Change of h over device"
  SI.CoefficientOfHeatTransfer kc;
  Real effectiveness "NTU effectiveness";
  SI.Density rho "Upstream density";
  SI.DynamicViscosity eta "Upstream dynamic viscosity";
  SI.SpecificHeatCapacity cp "At constant pressure";
  SI.ThermalConductivity lambda "Thermal conductivity";

equation
  // Static mass balance
  port_a.m_flow + port_b.m_flow = 0;

  // Static energy balance
  port_b.h_outflow = inStream(port_a.h_outflow) + dh;
  port_a.h_outflow = Medium.h_default;

  // Static momentum balance
  m_flow = homotopy(
    lambda()*wallFriction_mflow_dp(dp, ...) +
    (1-lambda())*dp/dp_nominal*
      wallFriction_mflow_dp(dp_nominal, ...));

  // Heat transfer
  kc = heatTransfer_kc_mflow(m_flow, ...);
  effectiveness = 1-exp(-(kc*heatTransferArea/(cp*m_flow)));
  dh = homotopy(lambda()*effectiveness*cp*(T_amb - state.T));

  // Auxiliary equations for thermodynamic,
  // transport properties
  // ...
end Pipe;
```

Listing 38: Pipe model using UTP

The following table 9 lists the simplifying assumptions for the device models employed for $\lambda = 0$ in this test case. The implementation of the ramped heat transfer and the secant approximation to the wall friction correlation for the pipe with heat transfer is described in listing 38.

Device	Simplifications
Pipe with heat transfer	Ramp of heat transfer, secant approximation of wall friction correlation
Orifice	Secant approximation of wall friction correlation
Cabin	Ramp of heat loads
Fan	Ramp of heat dissipated to fluid

Table 9: Implemented device models with simplifications for $\lambda = 0$ problem

Figure 74 shows a robustness profile for the resulting unidirectional thermo-fluid dynamics probability-one homotopy. The results illustrate that the proposed homotopy map and the probability-one homotopy method provide robust convergence, even in light of large variations of the start iterate and random vector (temperatures range from 200K to 400K and pressures from 0.5bar to 2bar).

Figures 75 and 76 finally present homotopy traces that illustrate the global convergence properties. Even if the start iterates and random vector a move further away from the solution (see ranges mentioned in the last paragraph) the algorithm converges robustly. Note that the pressure range is smaller than the given range of start iterates and random values due to leakage flow. As the gradients of the traces are concentrated toward λ values near unity, scaling factors can be used to spread out these gradients (see figure 57 for an illustration). The "dents" in the temperature traces at higher λ values each stem from a flow reversal in one of the device models (these dents are particularly visible in the lower plot of figure 75 near $\lambda = 1$).

10.4 Conclusions

The probability-one homotopy method works well in practice. The proposed integration of the method into the framework of equation-based, object-oriented modeling languages works well as the results and the concise code listings illustrate.

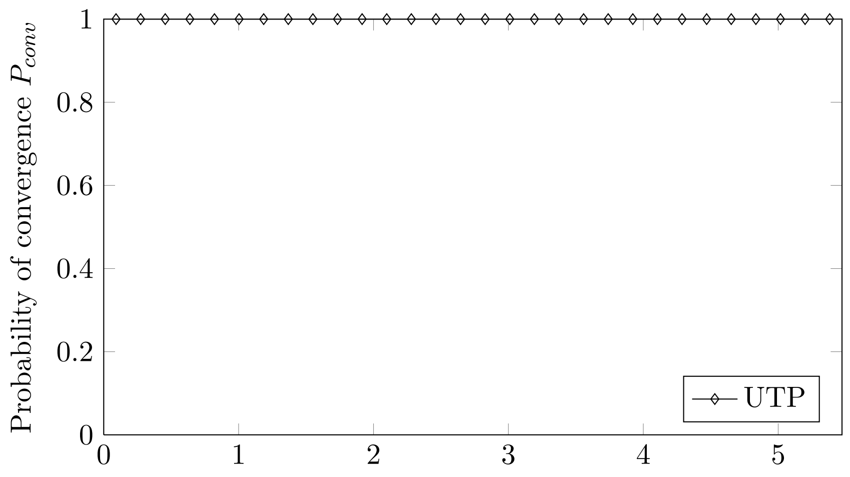


Figure 74: Robustness profiles for Air Distribution, problem-specific homotopy map (60 samples per bin)

Additionally to the suggested operators, which are the minimum tools required for use of this method, some language elements, annotations and tool support are considered helpful in order to facilitate the use by model developers.

- An annotation for connector definitions, which leads to an automatic instantiation of a node model such as the one presented in section 10.2.1 in connection sets.
- An enumeration to select iteration variables (e.g., `IterateSelect` similar to `StateSelect`) to demand specific unknowns as iteration variables of nonlinear algebraic equation systems, for which boundedness arguments hold.
- Proper support by tools via a "results browser" that allows to investigate results not only over simulation time but also over λ for development and debugging.

If the method was adopted by the community, then such enhancements should be considered. This concludes the contributions to the start of the art in steady-state initialization using equation-based, object-oriented modeling and simulation.

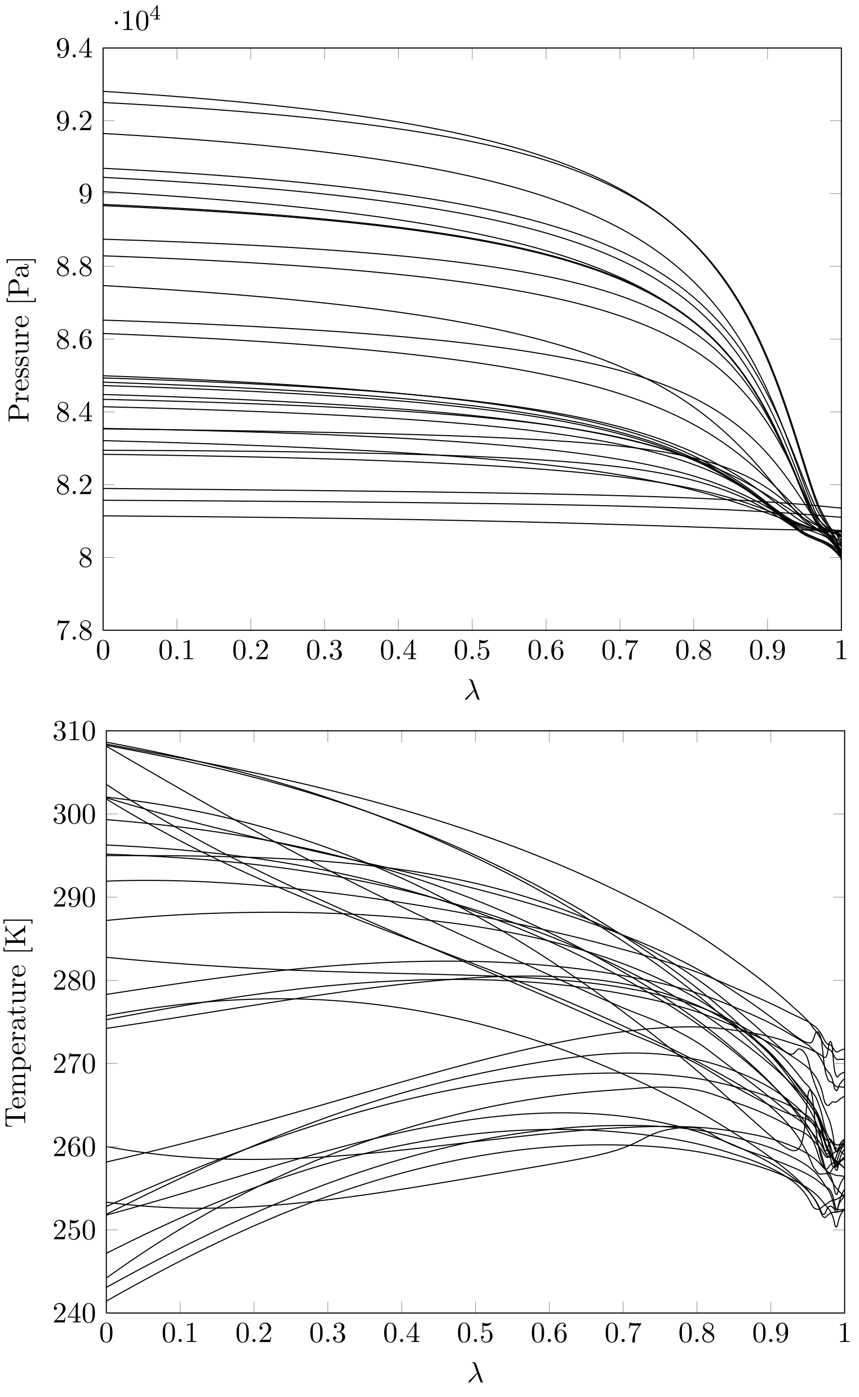


Figure 75: Homotopy traces for air distribution test case, medium distance from solution

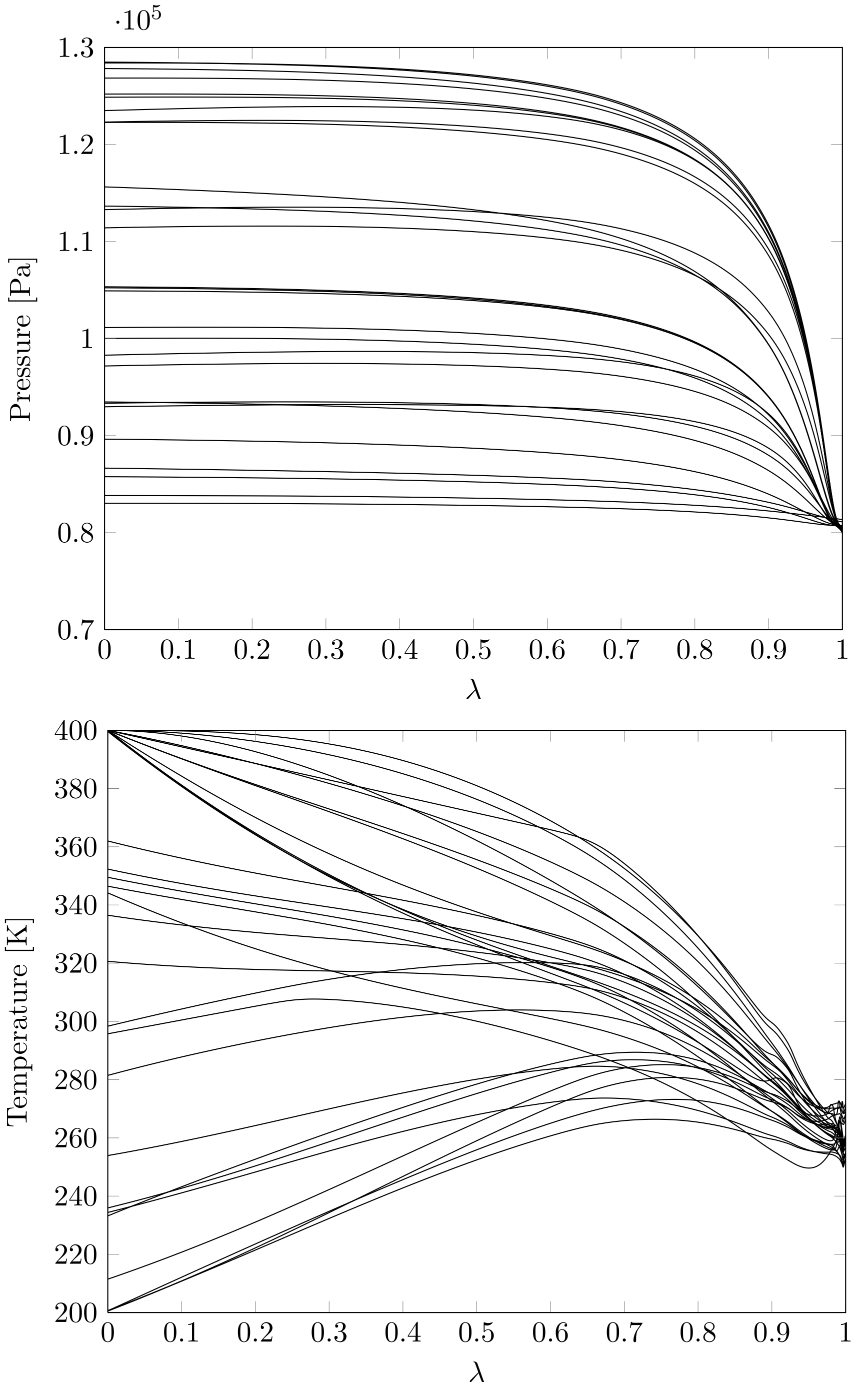


Figure 76: Homotopy traces for air distribution test case, high distance from solution

CHAPTER 11

MODELING AND SIMULATION OF ENVIRONMENTAL CONTROL SYSTEMS

The objective of this chapter is first to describe the ECS to some extend and second to present how to address the requirements of design methods for physics-based plant models at this example. Particular attention is paid to the aspects mentioned in section 1.3.3 such as variable causality and off-design performance simulation as well as to a general requirement for flexibility.

11.1 The Environmental Control System

A conventional ECS consists of the following subsystems (see figure 77): Air conditioning system (ACS), temperature control system (TCS), ventilation control system (VCS), air distribution system, and cabin pressure control system (CPCS). Additionally, optional systems such as combined ozone / VOC converters (VOZC), humidification systems (HUM), or dry air generation systems (DAGS) can be installed on an ECS.

The primary function of the ECS is to establish an environment that addresses the physiological needs and comfort requests of the passengers and crew. These tasks require the following functions: Control temperature and humidity and regulate cabin pressurization. Ensure sufficient ventilation and fresh air, and remove pollutants.

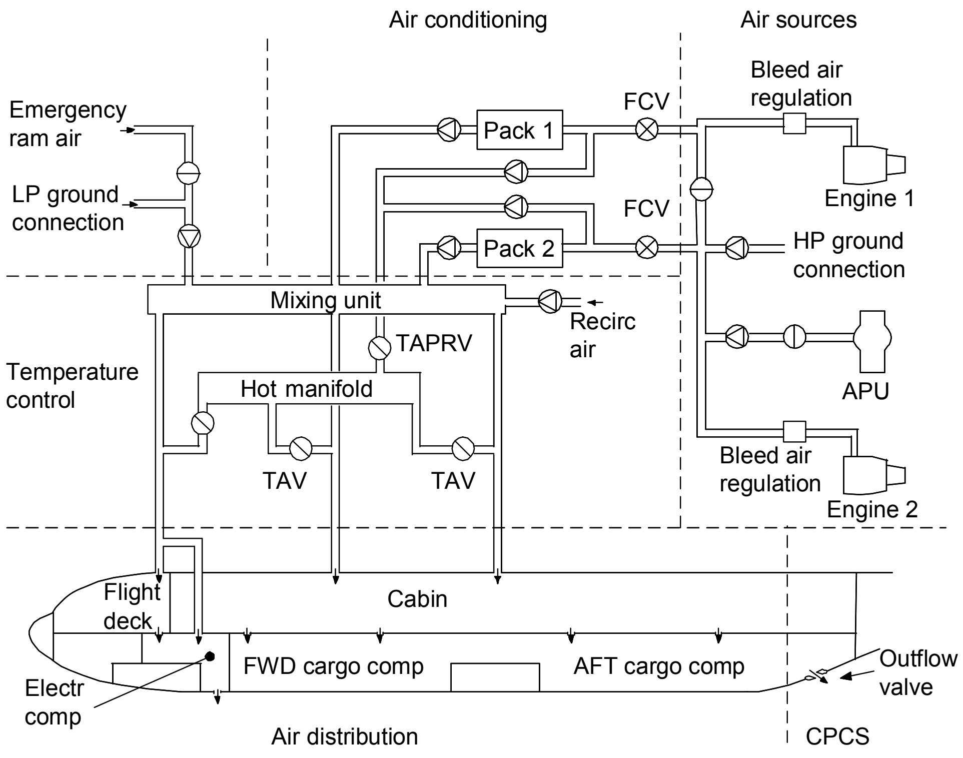


Figure 77: Conventional ECS architecture [24]

11.1.1 Overview

The central component of the ECS is the air generation unit (also called pack), which is the device used to condition the air flow. Usually, two air generation units are installed in an aircraft. Traditionally, they use engine bleed air as power source. At the start of the jet engine age, thrust was created by high exit velocity. The total mass flow passed through the core of the engine and the engine cycle itself was not very efficient. Consequently, the cost of bleeding air for use in systems such as the ECS was small. The compressor stage, at which bleed air is extracted, is selected based upon the maximum pressure requirement of the ECS and the wing ice protection system (WIPS). Due to the disparate segments of the aircraft mission, the pressure at the different stages varies remarkably and a single bleed port design would be inefficient. Therefore, two bleed ports are typically installed in the engine compressor (high and intermediate pressure). The system implementing the bleed air off take is called the engine bleed air system (EBAS). It provides bleed air primarily to the ECS and pneumatic WIPS.

The pack provides conditioned air to the flight deck, cabin, and cargo compartment. In the different classes of the cabin, in the flight deck, and in the cargo compartments, the number of passengers and the amount of payload vary more and more as the commercial aircraft become larger and more sophisticated. In order to provide conditioned air to all of those compartments, the conditioned air flows from the packs are mixed in a dedicated volume. This is called the mixing unit. From there, supply ducts carry the conditioned air to the compartments. Herein, the design principle is to maintain a constant volumetric flow rate. Therefore, every supply duct is calibrated with a fixed orifice. Large commercial aircraft divide the cabin in up to eight zones per deck. The packs are controlled such that the mixing unit temperature fulfills the most demanding cooling requirement of the cabin zones. To meet the exact temperature requirements of all zones, hot trim air is mixed to the supply air in tappings. The system that implements and controls this process is the TCS.

The CPCS maintains the cabin pressurization at a value, which depends on altitude and mission phase. It controls the outflow in the under-floor by means of dedicated outflow valves (OFVs). Here, a certain time derivative of pressure may not be exceeded in order to avoid irritation of the human ear.

Another important consumer of pneumatic power is the WIPS mentioned earlier. Currently, hot bleed air is used to heat the wing leading edge which prevents ice build-up. Starting at the end of the 1960s, turbojet engines were superseded by turbofan engines. That is, the paradigm to create thrust by high exit velocities was surpassed by the concept to move larger mass flows using a bypass to the engine core. Engines drawing on this concept provided considerably improved propulsion efficiency but increased the cost of extracting bleed air from the engine core. In an effort to reduce these drawbacks, recirculation systems were designed. The approach is to use highly efficient filters to reduce the bleed air off take from the engine without compromising the cabin air quality. As a result, the fresh air and not the ventilation requirements drive the amount of pack air flow in modern ECS designs. Nowadays, a general trend toward even higher bypass ratio engines is ongoing. This development, together with the well-established inherent inefficiencies of bleeding air from the compressor stages (throttling of pressure and reducing the temperature below the auto-ignition temperature of fuel in the pressure regulating valve and the pre-cooler respectively) provides the main argument in favor of electric ECS and reduced usage of pneumatic power on energy efficient aircraft in general.

To implement an electric ECS, two different approaches can be employed or combined. One option is to use ambient air instead of bleed air in air cycles and drive the associated turbo machinery electrically. This involves so called motorized compressors and motorized turbine compressors. The natural competitor to the air cycle is the well-established vapor compression cycle, as it almost generally has a higher efficiency (coefficient of performance, COP). Traditionally, vapor cycle systems are however associated with larger weights. In addition, condenser inlet air temperature variation results in reduced COPs. Therefore, vapor cycle systems are currently not used extensively in commercial aviation.

11.1.2 Air cycles

As mentioned above, the air generation unit is the core component of the ECS. This section provides information on different conventional approaches (thermodynamic cycles) to implement an air cycle in such a device.

All air cycles are inspired by the reverse Joule cycle (also called reverse Brayton cycle) for

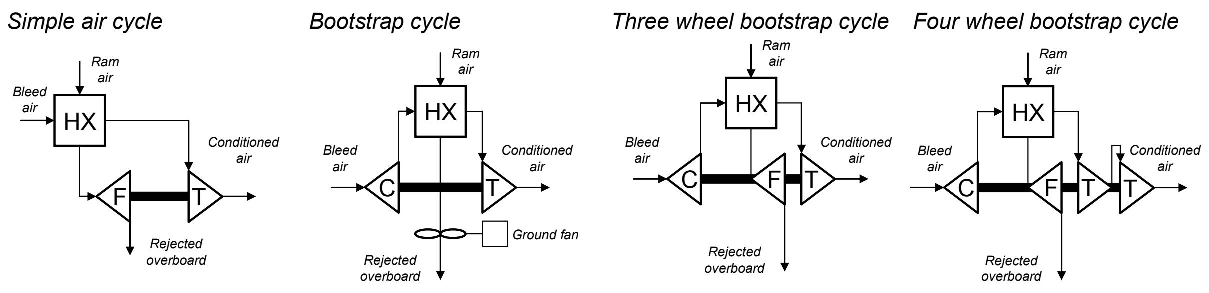


Figure 78: Typical air cycles used to condition air aboard commercial aircraft [103]

open systems. They employ air compression (at least in the engine compressor), heat rejection at high temperature, and air expansion to cool below the ambient temperature. Figure 78 illustrates four classic air cycles. They are or were all used in conventional ECS consuming bleed air and represent different levels of sophistication. To the left, the simple air cycle is shown. It features a heat exchanger (HX), in which bleed air is cooled by ram air (this is generally reasonable for subsonic commercial aircraft). Then, the bleed air flow is expanded in a turbine (T) to meet the temperature requirements imposed on the pack. The mechanical power obtained in the expansion process is used to drive a fan (F), which propels the ram air flow across the heat exchanger, which is vital for ground operation. The simple air cycle requires high bleed pressure. At the same time the turbine is not well loaded, unless the fan is made inefficient. From today's point of view, the simple air cycle is inefficient. This approach is used for the ECS aboard the Fokker 100, for example.

Next, the bootstrap cycle is shown. Before cooling the bleed air flow using ram air as heat sink, the hot engine air is compressed (C). Due to the higher temperature on the hot side of the heat exchanger (HX), the cycle becomes quite efficient. After cooling the bleed air, it is expanded in the turbine (T) and cooled below the ambient temperature. On ground, a fan is used to drive the ram air over the heat exchanger. It is usually driven electrically (e.g., Boeing 727) or pneumatically (e.g., Boeing 737 Classic).

Before continuing the discussion of the air cycles shown in figure 78, the fundamental effects of water content in ambient air shall be addressed. As the ECS is concerned with conditioning air, which, in many cases, involves cooling, the temperature can sink below the saturation point. This results in condensation and free water in the moist air flow, which can theoretically happen in any ECS component (e.g., ice build up) or the cabin (e.g., fogging). Consequently, the function to control the humidity level mentioned at the beginning of this section 11.1 is vital for the reliable operation of the aircraft.

In determining candidate locations for the extraction of water, the question for the location of condensation comes up. It cannot be answered generally, as it depends on the specific cycle and exact operating conditions. One prediction can be made, however. As the lowest temperature is encountered downstream of the turbine, free water can certainly be found there. The placement of a water extractor at this location led to so called low pressure water separation designs. Usually, in the case of condensation, ice build up cannot be prevented by other means than limiting the turbine discharge temperature to 0°C. The cooling capacity of such pack designs is therefore restricted. The apparent alternative is to install a water extractor upstream of the turbine. These designs are called high pressure water separation loops. The cooling capacity of a pack is not limited by ice build up and the higher pressure level supports the condensation of sufficient free water in the air flow. A drawback of this layout is that a pair of additional heat exchangers is required to ensure high pack performance (the reheater and condenser).

An important technique to recover performance lost in the separation of water is to spray the extracted water in front of the heat exchangers into the ram air channel. The latent heat to evaporate the liquid water is extracted from the surrounding material, lowers the temperature of the ram air and thus increases pack performance.

This takes the discussion back to the air cycles shown in figure 78. The three wheel bootstrap cycle combines the simple air cycle and the bootstrap cycle in that it features three wheels, the compressor (C), turbine (T), and ram air fan (F) on a single shaft. Even though the efficiency of this arrangement is slightly lower than that of the bootstrap cycle, this design has the advantage of being self-contained and not dependent on any other power source. The three wheel bootstrap cycle was first used on the Boeing 747 with low pressure water separation and is, using high pressure water separation, probably the most common pack configuration today (it is in use on the Airbus A320 and A330 / A340 families, the Boeing 757, 737NG, and on later versions of the Boeing 747 and early versions of the 767).

The four wheel bootstrap cycle (also called condensing cycle) shown to the right of figure 78 is probably the most sophisticated conventional design to date. The general arrangement of components is similar to that of a three wheel bootstrap cycle. The main difference stems from conflicting requirements on the turbine of a three wheel bootstrap cycle. The high altitude cruise conditions mandate a different turbine design than the ground cases, in which the water separation is pivotal. In a four wheel bootstrap cycle, these two requirements can be emphasized in one turbine each. The first turbine is laid out such that it expands the air flow to a temperature a little above the freezing point (minimizing the required de and anti-icing provisions). In the condenser heat exchanger, the latent heat of vaporisation required to condense water content on the hot side is injected to the cold air flow. The energy not recovered in the first turbine plus the latent heat of vaporisation is then retrieved in the second turbine. Additionally, the reheater heat exchanger is not required anymore. The four wheel bootstrap cycle provides different bypass paths to adapt the cycle to different ambient and operating conditions. Most prominently, at high altitudes, low pressure and humidity, an altitude valve can be opened to bypass both the high pressure water separator and the first turbine. This operates the cycle as a three wheel bootstrap cycle with an increased turbine nozzle area and decreases the restriction of the flow in the air cycle machine (ACM).

Even though the four wheel bootstrap cycle can be more efficient under certain conditions, it seems to be a matter of philosophy whether to use a four or three wheel bootstrap cycle. The advantages of one cycle have to be traded off against the ones of the other on a case to case basis. This cycle is used today on the Airbus A380, the Boeing 777 and later 767 versions, and the Embraer EMB 170 / 190 family.

11.1.3 Three wheel bootstrap Air Generation Unit

The general concepts of different air cycles have been introduced. To illustrate the components of an actual air generation unit, figure 79 shows an exemplary three wheel bootstrap pack with high pressure water separation.

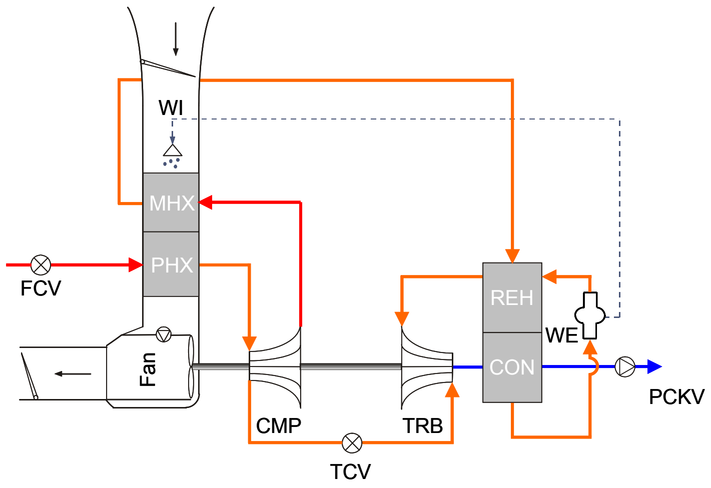


Figure 79: Typical three wheel bootstrap air generation unit with high pressure water separation

A flow control valve (FCV) controls the amount of bleed air going to the packs and the temperature control system (not shown in figure 79). In the primary heat exchanger (PHX), the bleed air flow is cooled by ram air. In the compressor (CMP) it is then compressed. The primary heat exchanger keeps the compressor outlet temperature within the material constraints of aluminum alloys. After compression the air flow enters the main heat exchanger (MHX) where it is cooled further by ram air. The air flow then enters the high pressure water separation loop, which consists of the reheater and condenser heat exchangers (REH and CON) and the water extractor (WE). In both heat exchangers the air flow is cooled down such that the water in the moisture laden air condensates before entering the water extractor. In order to lower the adverse effect on ACM performance of cooling the air flow with the turbine outlet air flow, the reheater is used to recover energy. The air flow is expanded in the turbine (TRB) and passes the cold side of the condenser. Through the pack check valve (PCKV) the conditioned air flow is discharged toward the mixing unit. The pack discharge temperature is controlled via the temperature control valve (TCV).

11.2 Modeling and simulation

When designing systems, it is not always instrumental to strictly simulate the performance of this system. The reason is, for example, that this may be easier to assume that system requirements are fulfilled or that certain system components will be devised such that they behave in a specific way. An example is the design of the air distribution system of commercial aircraft. Here, the pressure drop due to wall friction through the air distribution ducts shall be minimized. However, it is not practical to use excessively large duct cross-section areas, which is why such distribution systems are usually designed in order to arrive at a well-defined compromise pressure drop. It is incurred via the duct to the cabin zone, which is farthest away from the mixer. All other ducts are then calibrated using orifices to carry a well-defined amount of air to their cabin zones.

When simulating strictly the performance only, all precise orifice parameters had to be established. Such detailed data may not be available during early phases of system design and is even not required. After all, the pressure drop to the cabin zone farthest away from the mixer could be calculated normally and the correct mass flow rates could be prescribed for each duct. At a later stage in system design, it is more interesting to model the performance of a fixed system however, which is why the causality is then inverted.

The described approach of idealized and inverse models, which is used in several areas of the present tool, implies challenges for the steady-state initialization of such models. This is the case especially if they are used on systems, whose thermodynamic cycle involves a loop. This may be the case for the cycle itself (e.g., vapor compression cycle) or due to the system design (e.g., the reverse Joule cycle, which is an open thermodynamic cycle but includes a loop via the rigid inertia-free shaft of the air cycle machine in case of steady-state initialization). Then, steady-state initialization may be challenging using local gradient-based algorithms such as damped Newton Methods. Alternative solution methods have been presented in the previous chapters.

In the present simulation tool, models are classically split into plant and control, which is not always possible when using inverse models with ideal control logic (algebraic control laws). If this separation can be made, then the resulting models are easier to understand for practitioners, as they more closely resemble the systems implemented in reality. Additionally, the component equations become simpler themselves. This is the case, because a plain, say, integral control is usually simpler than a complex ideal algebraic control law. An example of the latter can be given for the mixer temperature demanded by trim air valve logic. In this case, non trivial equations have to be used in order to, e.g., compensate the demanded temperature with respect to condensation of moist air associated with the pressure drop in ducts. Such equations are not required when using conventional control and the resulting models are easier to understand for practitioners.

The modeling and simulation tool roughly spans three core domains. These are *Extended*

Cabin, *Air Conditioning*, and *Ram Air Channel*. On top of them, miscellaneous component models of the Engine Bleed Air System, Cabin Air Compressors and the like are provided.

The models cover off-design performance. The author believes that this is useful, because design is intrinsically linked to trade-offs and without properly covering off-design performance such considerations cannot be captured appropriately.

All components can be modeled using roughly three different hypotheses. Either, a model uses characteristic maps. They are typically created using high-fidelity codes and may be scaled using appropriate scaling factors. Alternatively, geometry-based performance estimators or constant efficiencies can be used. Geometry-based performance estimators are often based on handbook methods or established correlation schemes and use geometric data to establish component performance.

The Extended Cabin domain covers the cabin itself (control volumes for cabin and flight deck; under floor; heat loads via convection, radiation, and conduction; loads dissipating moisture), air distribution (ducts, fans, pressure regulating valves), mixing (mixer of fresh and recirculating air), and temperature control. The Air Conditioning package covers components of conventional air cycles such as air-to-air heat exchangers, turbo machinery, and water extractors on one hand. On the other, it also includes components to model vapor compression cycle systems (refrigerant-to-air heat exchangers, compressors, expansion valves). Like this unconventional and hybrid systems can be assembled. Finally, the Ram Air Channel provides component models of devices used to capture ambient ram air at the aircraft skin (ram air inlets and outlets in different geometrical shapes). This is typically required to reject excess heat to the ambient or to capture fresh air to be provided to the cabin.

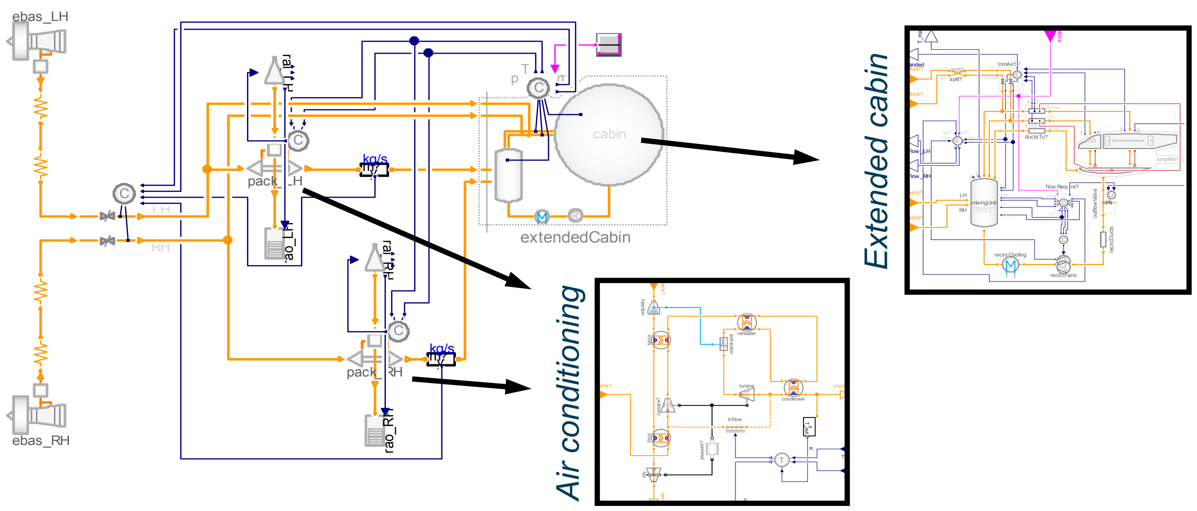


Figure 80: Model of a conventional ECS Architecture

CHAPTER 12

DISCRETIZATION SCHEMES FOR HIGH-SPEED COMPRESSIBLE FLOW

Many of the modeling and simulation problems outlined in section 11.2 of the previous chapter involve low-speed compressible flow as defined in section 1.4. For them, state of the art methods are applied. A detailed presentation of the implementation details is academically not of highest interest and thus omitted.

However, the modeling and simulation applications also include a class of problems, for which no state of the art methods in equation-based, object-oriented modeling languages can be applied. It involves high-speed compressible flow. A typical example is the pneumatic air supply including the bleed air regulation and flow control valve (FCV) shown in figure 77. The bleed air regulation typically controls the pressure level in the pneumatic air supply, which should ideally be held constant. A flow control valve controls the mass flow rate, which is fed into an air conditioning pack. Obviously, the bleed air regulators and the flow control valves influence each other. Therefore, dynamic momentum and high-speed compressible flow are relevant phenomena for the design of the complete system and the associated control laws. A similar problem in this area involves the APU check valve (also shown in figure 77). When the engine bleed air supply becomes available, then the APU check valve closes. Such processes may, if designed improperly, be abrasive and even involve shocks, which may excessively stress both the check valve and the ducts. Proper understanding of high-speed compressible flow is thus required.

For this reason, the objective of this chapter is to review relevant concepts of the theory in numerical solution methods for high-speed compressible flow. Furthermore, a goal is to translate them from the algorithmic perspective taken in literature to the non-causal concepts of equation-based, object-oriented modeling languages. Finally, the elements of generic discretization schemes shall be decomposed in an object-oriented fashion and implemented in a generic library.

12.1 The governing equations in compact flux form

To address high-speed compressible flow, the formulation of the governing equations in primitive variables as introduced in section 2.1 is discarded. Instead, a compact flux formulation as described by Toro [174] is considered. It is posed using conserved variables u and flux f.

$$u_t(x,t) + f\left(u(x,t)\right)_x = s\left(u(x,t)\right) \tag{74}$$

$$u(x,t) = \begin{pmatrix} \rho \\ \rho v \\ \rho u_0 \end{pmatrix}, \qquad f\left(u(x,t)\right) = \begin{pmatrix} \rho v \\ \rho v^2 + p \\ v\left(\rho u_0 + p\right) \end{pmatrix} \tag{75}$$

If the cross-sectional area A is supposed to vary smoothly with time and position, then the following source term including heat transfer and viscous wall friction can be used [174].

$$s\left(u(x,t)\right) = \begin{pmatrix} 0 \\ \Delta p_{fr} \\ \rho \dot{q}_e \end{pmatrix} - \frac{1}{A}\frac{dA}{dt}\begin{pmatrix} \rho \\ \rho v \\ \rho u_0 + p \end{pmatrix} \tag{76}$$

12.2 Conservative methods

An alternative approach to discretize the governing equations of thermo-fluid dynamics is now introduced based on Toro [174]. It is formulated in conserved variables and therefore called a conservative method. Such methods are particularly suited for high-speed compressible flow.

The use of conservative methods is motivated by the presence of discontinuities such as shock waves in the solution of certain problems such as gas dynamics. Hou and LeFloch [84] have shown that formulations based on variables other than the conserved ones fail to correctly predict the solution at shock waves. They result in wrong jump conditions and thus wrong shock strength, speed, and location. The theorem of Lax and Wendroff [101] in turn states that conservative methods, if convergent, do converge to the weak solution of the conservation law.

Consequently, conservative methods are an obvious choice if shock waves are potentially contained in the solution. Alternative approaches are the somewhat outfashioned shock-fitting [128] and adaptive primitive-conservative schemes [174].

In this section, the compact formulation of the conservation laws introduced in equation (74) is used. The vector of conserved quantities is denoted by $u(x,t) = (\rho, \rho v, \rho u_0)$. The conserved quantities are used in place of the primitive variables considered before.

The governing equations introduced in section 2.1 were derived from the integral relations on control volumes and their boundaries. The differential forms of these equations such as equations (1) to (3) or equation (74) in turn are based on the assumption of smoothness of the flow variables, which is not fulfilled for the applications considered in this section. In order to include weak solutions of (74), an integral form of the equations is used, a finite volume method. Therefore, equation (74) is integrated over the interval I_i to obtain

$$\frac{d\overline{u}(x_i,t)}{dt} = s\left(\overline{u}(x_i,t)\right) - \frac{1}{\Delta x_i}\left(f\left(u\left(x_{i+1/2},t\right)\right) - f\left(u\left(x_{i-1/2},t\right)\right)\right) \tag{77}$$

Herein, a cell average is used

$$\overline{u}(x_i,t) = \frac{1}{\Delta x_i}\int_{x_{i-1/2}}^{x_{i+1/2}} u(\xi,t)\,d\xi$$

As before, equation (77) is approximated by a semi-discretized conservative scheme, which results in a differential algebraic equation,

$$\frac{d\overline{u_i}(t)}{dt} = s\left(\overline{u_i}(t)\right) - \frac{1}{\Delta x_i}\left(f_{i+1/2} - f_{i-1/2}\right) \tag{78}$$

Herein, $\overline{u_i}(t)$ is a numerical approximation of the exact cell average $\overline{u}(x_i,t)$, and $f_{i\pm1/2}$ is a numerical flux, an approximation of the physical flux $f\left(u\left(x_{i\pm1/2},t\right)\right)$.

The remainder of this section is concerned with the construction of numerical fluxes. All these fluxes consist of a monotone flux and a reconstruction. Practically, a monotone flux is a flux free of spurious oscillations. Due to Godunov's Theorem such linear fluxes are however first-order accurate only. Therefore, these monotone fluxes are often used together with reconstructions in order to build higher-order schemes. The reconstruction provides an approximation of the vector of conserved variables u (or any other variable of interest) based on the cell averages. Its higher-order accuracy yields, together with a first-order monotone flux, higher-order numerical flux.

12.2.1 Monotone flux and first-order schemes

A monotone numerical flux is defined using a function g,

$$f_{i+1/2} = g\left(u^-_{i+1/2}, u^+_{i+1/2}\right) \tag{79}$$

Here, $u^-_{i+1/2}$ is in general an approximation of the vector of conserved variables at $x_{i+1/2}$ in the left limit, and $u^+_{i+1/2}$ in the right limit. If g satisfies the following conditions then (79) is a monotone flux [162].

1. $g(a,b)$ is Lipschitz continuous in both arguments
2. $g(a,b)$ is a non-decreasing function in a and a non-increasing function in b.

3. $g(a, b)$ is consistent with the physical flux f, i.e., $g(a, a) = f(a)$.

Each monotone flux can be used without reconstruction with the approximation $u^-_{i+1/2} \approx u_i$ and $u^+_{i+1/2} \approx u_{i+1}$. The results are first-order schemes. Alternatively, any more sophisticated approach may be used to reconstruct $u^\pm_{i+1/2}$.

In the following presentation of monotone fluxes, q_r will refer to the right limit $q^+_{i+1/2}$ of a quantity q. Similarly, $q^-_{i+1/2}$ is abbreviated as q_l.

Monotone fluxes are classified as either upwind methods or central methods. Upwind methods discretize equations on a mesh according to the direction of propagation of information on that mesh. Central methods do not make a distinction based on the direction of information propagation. Within the upwind methods, both Godunov-type methods and flux vector splitting methods are presented based on [174].

12.2.1.1 Godunov-type Upwind Methods

These methods are also called flux difference splitting methods or Riemann approach methods. In the general case, $u^-_{i+1/2} \neq u^+_{i+1/2}$, i.e., at position $x_{i+1/2}$ a discontinuity is present. The original Godunov monotone flux therefore interpreted this as Riemann problem and provided the conserved variables at $x_{i+1/2}$, $u_{i+1/2}$. This is the state that will be present instantly at this position and will remain constant over a time step. Then, the flux can be evaluated at this position, $f\left(u_{i+1/2}\right)$. The result is the Godunov monotone flux.

As the Godunov monotone flux uses the exact solution to the Riemann problem, the resulting method is computationally relatively expensive and is rarely used for practical computations. Godunov-type monotone fluxes follow the approach of the Godunov monotone flux but employ an approximate Riemann solver. This reduces the computational expense significantly and results in rather accurate monotone fluxes.

Roe's Monotone Flux: This Godunov-type flux uses one of the most well-known approximate Riemann solvers. The approximate Riemann solver is due to Roe [149] and works as follows. The original Riemann problem is replaced by an approximate Riemann problem, which is solved exactly. The approximate problem is based on linearized conservation laws, $u_t + A_{lr} u_x = 0$.

The Jacobian of the approximate problem A_{lr} has to fulfill the following conditions [149, 174].

1. Consistency with exact Jacobian: $A_{lr}\left(u, u\right) = A(u)$ with the Jacobian $A(u)$ of the nonlinear problem (74), $A\left(u\right) = \partial f\left(u\right)/\partial u$

2. Hyperbolicity of the system: A_{lr} can be diagonalized.

3. Conservation across discontinuities: $f\left(u^+_{i+1/2}\right) - f\left(u^-_{i+1/2}\right) = A_{lr}\left(u^+_{i+1/2} - u^-_{i+1/2}\right)$

The linearized problem has to be established for each combination of governing equations (e.g., Euler equations) and thermodynamic property model (e.g., ideal gas).

Roe [149] established a methodology using averaged values such that $A_{lr}\left(u^+_{i+1/2} - u^-_{i+1/2}\right) = A_{lr}\left(\overline{u}\right)$ fulfills the given conditions. The vector $\overline{u}$ is the vector of Roe average values. For the one-dimensional Euler equations and ideal gas, the Roe average values are as follows.

$$\overline{\rho} = \frac{\rho_r + \rho_l}{\sqrt{\rho_r} + \sqrt{\rho_l}}$$

$$\overline{v} = \frac{\sqrt{\rho_r} v_r + \sqrt{\rho_l} v_l}{\sqrt{\rho_r} + \sqrt{\rho_l}}$$

$$\overline{h_0} = \frac{\sqrt{\rho_r} h_{0,r} + \sqrt{\rho_l} h_{0,l}}{\sqrt{\rho_r} + \sqrt{\rho_l}}$$

and

$$\overline{c}^2 = (\kappa - 1)\left(\overline{h_0} - \frac{1}{2}\overline{v}^2\right)$$

Due to the above mentioned properties, the linearized system can be transformed into a system of independent transport equations. The data difference $\Delta u = u_r - u_l$ is projected onto the right eigenvectors of A_{lr}. This establishes the wave strengths α_i. Proper integral relations allow to establish the numerical flux as

$$g_{Roe}\left(u_l, u_r\right) = \frac{1}{2}\left(f_l + f_r\right) - \frac{1}{2}\sum_{i=1}^{3} \alpha_i \left|\lambda_i\right| K_i$$

with eigenvalues λ_i and right eigenvectors K_i.

For the problem of interest, the wave strengths are

$$\alpha_1 = \frac{1}{2\overline{c}}\left[\Delta m - \Delta\rho\left(\overline{v} + \overline{c}\right)\right] - \frac{1}{2}\alpha_2$$

$$\alpha_2 = -\frac{\kappa - 1}{\overline{c}^2}\left[\Delta\rho\left(\overline{v}^2 - h\right) - \overline{v}\Delta m + \Delta\bar{e}\right]$$

$$\alpha_3 = \Delta\rho - \alpha_1 - \alpha_2$$

Here, the data difference Δm for example refers to the difference in momentum. This approximate Riemann solver calculates an expansion shock, which is obviously unphysical. Fixes for this and other weak points of this monotone flux are discussed in [174].

HLLE Monotone Flux: The Harten, Lax and van Leer [81] monotone flux simplifies the approximate Riemann problem even further. It neglects the contact surfaces and consequently assumes that between the shock and the expansion fan only a single homogeneous state is present. For hyperbolic systems of two equations this is correct, but for the Euler equations addressed herein this is a rough approximation. Even if the resolution of contact surfaces is poor, this monotone flux is still a robust and efficient one, whose accuracy is, on global level, often sufficient.

An advantage of this flux is that it can be applied easily to different thermodynamic property models. The approximate Riemann solver of Roe for example is not straight-forward to apply to several problems such as ones involving real gas equations. It is therefore a relevant candidate for equation-based, object-oriented modeling languages applications, as the specific thermodynamic property models are often factored out of the component models, in which the discretized Euler equations are implemented.

The scheme is implemented via an a-priori estimation for the fastest signal speeds and its monotone flux is defined as

$$g_{HLLE}\left(u_l, u_r\right) = \frac{c_r^+ f\left(u_l\right) - c_l^- f\left(u_r\right)}{c_r^+ - c_l^-} + \frac{c_r^+ c_l^-}{c_r^+ - c_l^-}\left(u_r - u_l\right)$$

Here, the signal speeds are $c_r^+ = \max\left(0, v_r + c_r, \overline{v} + \overline{c}\right)$ and $c_l^- = \min\left(0, v_l - c_l, \overline{v} - \overline{c}\right)$ respectively. In these equations the Roe average velocity $\overline{v}$ and the Roe average speed of sound $\overline{c}$ have been used.

12.2.1.2 Flux Vector Splitting Upwind Methods

In chapter 2.2, the simple first-order upwind scheme in primitive variables was introduced. Based of the sign of a characteristic quantity (usually, this is a velocity normal to the cell boundary), any variable on the boundary was established to have either the value from the left or the right side. In the context of the present approach to conservative methods and high-speed compressible flow, there is no simple scheme of this type. This becomes obvious from the hyperbolicity of the Jacobian $\partial f / \partial u$ and its eigenvalues.

In general, the real part of the eigenvalues can have any sign and a simple one-sided differencing scheme will be appropriate only if the real parts of all eigenvalues have the same sign. The general system will however have some eigenvalues with a positive real part, and one side will be upwind for them, while the others have a negative sign on the real part and consequently the upwind side will be opposite for them. A typical way to resolve this problem is to split such a system into one with a positive real part of the eigenvalues and one with a negative real part and to treat them separately. These are the flux vector splitting methods discussed in this section.

The flux vector splitting approach is also called Boltzmann approach and works as follows [174]. As before, the Jacobian of the system of nonlinear hyperbolic conservation laws (74) is of interest.

$$A(u) = \frac{\partial f(u)}{\partial u}$$

Due to hyperbolicity, it may be expressed as

$$A = K\Lambda K^{-1} \tag{80}$$

Here, Λ is the diagonal matrix of eigenvalues λ_i of A. The matrix K is the matrix of right column eigenvectors K_i. The flux vector splitting methods aim at splitting the flux $f(u)$ into components $f^+(u)$ and $f^-(u)$ based on the following equality.

$$f(u) = f^+(u) + f^-(u)$$

Following the introduction of this section, the split fluxes are established such that the eigenvalues $\hat{\lambda}_i^+$, $\hat{\lambda}_i^-$ of the Jacobian

$$\hat{A}^+ = \frac{\partial f^+(u)}{\partial u},$$
$$\hat{A}^- = \frac{\partial f^-(u)}{\partial u}$$

fulfill $Re\left(\hat{\lambda}_i^+\right) \geq 0$ and $Re\left(\hat{\lambda}_i^-\right) \leq 0$.

The Steger-Warming Monotone Flux: In order to establish such a splitting, the homogeneity property of (74) may be exploited. If the system of hyperbolic conservation laws fulfills this property, then

$$f(u) = A(u)\, u \tag{81}$$

like in the linear constant coefficient case. The unsteady Euler equations fulfill this property and consequently the splitting may utilize the structure exposed in (80), that is, the splitting may be applied to the diagonal matrix Λ. Steger and Warming [166] proposed a splitting of the eigenvalues λ_i,

$$\lambda_i = \lambda_i^+ + \lambda_i^- \tag{82}$$

Here, $\lambda_i^+ \geq 0$ and $\lambda_i^- \leq 0$. Consequently, Λ is split as

$$\Lambda = \Lambda^+ + \Lambda^- \tag{83}$$

$\Lambda^\pm$ are the diagonal matrices of the split eigenvalues $\lambda_i^\pm$. This leads directly to the splitting of A.

$$A = A^+ + A^- \tag{84}$$

where $A^\pm = K\Lambda^\pm K^{-1}$. If the property (81) is fulfilled, one arrives at an expression for the flux splitting.

$$f(u) = f^+(u) + f^-(u) \tag{85}$$

Here, $f^\pm(u) = A^\pm u$.

The crucial question is how to choose $\lambda_i^\pm$ in (82). Steger and Warming [166] suggested to use to following equations.

$$\lambda_i^+ = \frac{1}{2}\left(\lambda_i + |\lambda_i|\right) = \max(\lambda_i, 0) \tag{86}$$

$$\lambda_i^- = \frac{1}{2}\left(\lambda_i - |\lambda_i|\right) = \min(\lambda_i, 0) \tag{87}$$

When exercising this approach, the following Steger-Warming monotone flux is established.

$$g_{SW}(u) = f^+(u) + f^-(u)$$

with

$$f^{\pm}(u) = \frac{\rho}{2\kappa}\left[\begin{array}{c} \lambda_1^{\pm} + 2(\kappa-1)\lambda_2^{\pm} + \lambda_3^{\pm} \\ (v-c)\lambda_1^{\pm} + 2(\kappa-1)v\lambda_2^{\pm} + (v+c)\lambda_3^{\pm} \\ (h-vc)\lambda_1^{\pm} + (\kappa-1)v^2\lambda_2^{\pm} + (h+vc)\lambda_3^{\pm} \end{array}\right]$$

The eigenvalues are given by (86) and (87). The remaining variables have to be evaluated according to the definition of the flux, i.e., for $f^+(u)$ the values from the left such as ρ_l, u_l and for $f^-(u)$ the values from the right such as ρ_r, u_r.

Note that for this specific splitting $\hat{A}^+ \neq A^+$ and $\hat{\lambda}_i^+ \neq \lambda_i^+$. There are however more advanced splittings, which avoid this pathology. See [174] for more information on this subject.

12.2.1.3 Centered Methods

Schemes, whose support does not depend on the sign of the characteristic speeds, are called centered schemes.

The Rusanov Monotone Flux, a local Lax-Friedrichs Flux: The Lax-Friedrichs flux is one of the simplest and most approximate methods considered herein. It was originally developed in the context of finite-difference methods and later applied to the finite-volume context.

Similarly to the HLLE method, only an expansion and a compression wave are considered. In the original Lax-Friedrichs flux, the speed of each wave was assumed to be such that it reached the cell boundaries exactly within a time step Δt. For uniform grids, each wave of the global problem therefore had the same speed, which is an even more approximate solution than in the HLLE method. As, in the present context, no fully explicit scheme is employed but the method of lines, no time step Δt is defined. For this reason and to slightly improve accuracy, a local form of the Lax-Friedrichs monotone flux, the Rusanov monotone flux [153], is considered. In the Lax-Friedrichs flux,

$$g_{LF}(u_l, u_r) = \frac{1}{2}(f(u_r) + f(u_l)) - \frac{1}{2}\frac{\Delta x}{\Delta t}(u_r - u_l)$$

the signal speed $\Delta x/\Delta t$ is replaced by $\lambda_{max} = \max\left((|v|+c)_l, (|v|+c)_r\right)$. Then, the Rusanov monotone flux is defined as follows.

$$g_{Rus}(u_l, u_r) = \frac{1}{2}(f(u_r) + f(u_l)) - \frac{1}{2}\lambda_{max}(u_r - u_l) \qquad (88)$$

First-Order Centered Monotone Flux: The First-Order Centered Monotone flux (FORCE scheme) [173] is obtained when replacing the random sampling of Riemann problems in Random Choice Methods with deterministic integral averages.

According to Toro [174], for fully explicit schemes, the result is the arithmetic mean of the Lax-Friedrichs and Richtmyer [148] fluxes. The Richtmyer flux is a second-order scheme with constant coefficients and is thus, according to Godunov's classic theorem [71], not monotone and results in spurious oscillations.

For the fully explicit version of the Richtmyer flux, an intermediate state is first defined,

$$u_{Ri} = \frac{1}{2}(u_l + u_r) + \frac{1}{2}\frac{\Delta t}{\Delta x}(f(u_l) + f(u_r))$$

and then the flux is evaluated at it.

$$g_{Ri}(u_l, u_r) = f(u_{Ri})$$

Then, the FORCE flux is the arithmetic mean of the Lax-Friedrichs and Richtmyer fluxes [174]

$$g_{Force}(u_l, u_r) = \frac{1}{2}(g_{LF}(u_l, u_r) + g_{Ri}(u_l, u_r))$$

Again, the local version of the Lax-Friedrichs flux (the Rusanov flux presented in previous section) and a local version of the Richtmyer flux are used, is again obtained by replacing $\Delta x/\Delta t$ with λ_{max}.

After introducing some monotone numerical fluxes, methods to obtain higher-order approximations of the solution to (74) are considered.

12.2.2 Total Variation Diminishing schemes

Godunov's theorem [71] was mentioned already. It provides the theoretical foundation to the observation that *linear* second-order schemes are more accurate in smooth regions of a problem solution to (74) than first-order schemes. Near strong gradients and shocks, these methods produce spurious oscillations however. Monotone methods however do not exhibit such spurious oscillations. In case of linear schemes, their limited first-order accuracy is disadvantageous however.

One option to eliminate or reduce spurious oscillations for higher-order methods is to introduce artificial viscosity. This can be tuned such that it is large enough to suppress oscillations in the neighborhood of discontinuities and small elsewhere to maintain accuracy. A disadvantage of this approach is however, that the quantity of artificial viscosity is problem dependent and therefore requires fine-tuning by the user. This approach is not followed here and instead a less empirical approach to introduce viscosity is adopted.

Therefore, in order to circumvent the limitations formulated by Godunov's theorem, schemes with variable coefficients, i.e., nonlinear schemes, are considered. Such schemes can adapt themselves to the local nature of the solution.

Harten [79] defined High-Resolution Methods as numerical methods with the following properties.

1. Second or higher-order of accuracy in smooth parts of the solution
2. The solution is free of spurious oscillations.
3. The resolution of discontinuities in the solution is high, i.e., the number of cells containing the numerical reproduction of the discontinuity is smaller in comparison with that of first-order monotone schemes.

A class of methods fulfilling these properties is that of Total Variation Diminishing methods [79]. See this reference for a definition of the total variation. For brevity, only the case of a smooth function $u(t)$, for which the total variation is

$$TV\left(u\right) = \int_{-\infty}^{\infty} \left|u'\left(x\right)\right| dx$$

and the case of a mesh function $u^n = \{u_i^n\}$ are mentioned. For the latter, the total variation is defined as

$$TV\left(u^n\right) = \sum_{i=-\infty}^{\infty} \left|u_{i+1}^n - u_i^n\right|$$

Fundamental properties of the exact solution of the conservation law (74) such as no creation of new local extrema lead to the conclusion that the total variation $TV(u(t))$ is a decreasing function of time [79]. Consequently, Total Variation Diminishing methods mimic a property of the exact solution.

For a general scalar conservation law, Harten [79] provided a theorem on a sufficient condition for a particular class of nonlinear schemes with two coefficients to be Total Variation Diminishing (TVD). These conditions are essentially four inequalities on these two coefficients. As the coefficients may in general be data dependent, Harten's theorem provides a tool for the construction of nonlinear schemes that circumvent Godunov's theorem stated above.

The classic TVD approach to adaptively switch between the characteristics of a monotone first-order numerical flux g^{LO} and those of a higher-order constant coefficient flux g^{HI} is to make the following assumption [168].

$$g^{TVD} = g^{LO} + \varphi\left[g^{HI} - g^{LO}\right]$$

Here, φ is a flux limiter function that implements the adaptive algorithm. Analysis of Harten's theorem led to the identification of the Sweby TVD region [168]. In this region, various flux limiters have been defined such as the well-known limiters Superbee, Minbee, and Ultrabee.

In the following sections, this approach is not followed directly. Instead of flux limiters, slope limiters are used, which are analogous to the flux limiters.

For the reasons described in section 12.2.1.3, both an upwind TVD and a central TVD method are considered. Before the details of these methods are described, it is highlighted that most of the theory of the TVD methods was developed for case of a *scalar* conservation law. In many cases, the extension to systems of hyperbolic conservation laws is empirical, but has been found to work robustly and efficiently.

12.2.2.1 A MUSCL Upstream TVD Scheme

Van Leer [183, 184, 185] introduced a higher-order method along the concept of reconstruction mentioned in the introduction of this chapter. MUSCL stands for Monotone Upstream-Centered Scheme for Conservation Laws.

The first-order schemes discussed so far use monotone fluxes directly by assuming piecewise constant data over the cells I_i, i.e., $u^-_{i+1/2} \approx u_i$ and $u^+_{i+1/2} \approx u_{i+1}$. In the simplest MUSCL scheme, piecewise linear local reconstructions are used. The reconstruction has to maintain the integral average, which is trivially fulfilled for piecewise linear local reconstructions.

First, slope vectors $\Delta_{i\pm1/2}$ are defined as follows.

$$\Delta_{i-1/2} = u_i - u_{i-1} \tag{89}$$

$$\Delta_{i+1/2} = u_{i+1} - u_i \tag{90}$$

Strictly speaking, these slopes are not slopes but differences of the vector of conserved quantities in adjacent cells. The terminology used in literature is adopted however and therefore $\Delta_{i\pm1/2}$ are called slope vectors. In order to implement a TVD scheme, the approach of limited slopes described by Quirk [145] is used.

$$\hat{\Delta}_i = \begin{cases} \max\left[0, \min\left(\beta\Delta_{i-1/2}, \Delta_{i+1/2}\right), \min\left(\Delta_{i-1/2}, \beta\Delta_{i+1/2}\right)\right] & \Delta_{i+1/2} > 0 \\ \min\left[0, \max\left(\beta\Delta_{i-1/2}, \Delta_{i+1/2}\right), \max\left(\Delta_{i-1/2}, \beta\Delta_{i+1/2}\right)\right] & \Delta_{i+1/2} < 0 \end{cases}$$

The value $\beta = 1$ does, in the scalar case, reproduce the Minbee flux limiter, and $\beta = 2$ the Superbee flux limiter.

Based on the piecewise linear local reconstruction,

$$u_i(x,t) = \overline{u_i}(t) + \frac{x - x_i}{\Delta x_i}\hat{\Delta}_i$$

The values at the extreme points of the cell I_i are established.

$$u^+_{i-1/2} = \overline{u_i} - \frac{1}{2}\widehat{\Delta}_i \tag{91}$$

$$u^-_{i+1/2} = \overline{u_i} + \frac{1}{2}\widehat{\Delta}_i \tag{92}$$

In order to finally obtain the second-order accurate upstream flux, some first-order monotone upstream flux is employed with the reconstructed values $u^-_{i+1/2}$, $u^+_{i+1/2}$.

$$g^{TVDu}_{i+1/2} = g^{mu}_{i+1/2}\left(u^-_{i+1/2}, u^+_{i+1/2}\right)$$

Note that $u^-_{i+1/2}$ is obtained from a reconstruction in cell I_i, and $u^+_{i+1/2}$ from a reconstruction in cell I_{i+1}.

12.2.2.2 A MUSCL Centered TVD Scheme

As mentioned before, also a second-order TVD centered scheme is introduced. It also follows the concept of the MUSCL scheme but uses a first-order monotone *centered* flux.

This approach is base on a slope limiter ξ_i, for which the following equation holds.

$$\hat{\Delta}_i = \xi_i \Delta_i$$

Here, the slope vector of the cells, Δ_i, is used.

$$\Delta_i = \frac{1}{2}(1+\omega)\Delta_{i-1/2} + \frac{1}{2}(1-\omega)\Delta_{i+1/2}$$

This is a weighted average of the side slope vectors $\Delta_{i\pm1/2}$, see (89) and (90). The weighting parameter has to fulfill $\omega \in [-1, 1]$. In computations conducted for this thesis, the value of $\omega = 0$ was used. Additionally, the ratio r_i of the cell side slope vectors is introduced.

$$r_i = \frac{\Delta_{i-1/2}}{\Delta_{i+1/2}}$$

Then, a slope limiter analogous to the Superbee flux limiter is [174]

$$\xi_{sb}(r) = \begin{cases} 0 & r \leqslant 0 \\ 2r & 0 \leqslant r \leqslant \frac{1}{2} \\ 1 & \frac{1}{2} \leqslant r \leqslant 1 \\ \min\left(r, \xi_r(r), 2\right) & r \geqslant 1 \end{cases}$$

A van Leer-type slope limiter is [174]

$$\xi_{vl}(r) = \begin{cases} 0 & r \leqslant 0 \\ \min\left(\frac{2r}{1+r}, \xi_r(r)\right) & r \geqslant 0 \end{cases}$$

A Minbee-type slope limiter is [174]

$$\xi_{mb}(r) = \begin{cases} 0 & r \leqslant 0 \\ r & 0 \leqslant r \leqslant 1 \\ \min\left(1, \xi_r(r)\right) & r \geqslant 1 \end{cases}$$

Above, $\xi_r(r)$, a TVD region limit that is defined as follows, was used.

$$\xi_r(r) = \frac{2}{1-\omega+(1+\omega)r}$$

As before, the conservative variable vector is approximated via the limited slope $\hat{\Delta}_i$ and equations (91) and (92). Then, the second-order accurate centered flux is obtained via a first-order monotone centered flux with the reconstructed values $u^-_{i+1/2}$, $u^+_{i+1/2}$. For this purpose, the FORCE flux can be used.

$$g^{TVDc}_{i+1/2} = g^{Force}_{i+1/2}\left(u^-_{i+1/2}, u^+_{i+1/2}\right)$$

Note again that $u^-_{i+1/2}$ is obtained from a reconstruction in cell I_i, and $u^+_{i+1/2}$ from a reconstruction in cell I_{i+1}.

12.2.3 Weighted Essentially Non-Oscillatory schemes

One disadvantage of TVD schemes is that the accuracy near discontinuities is reduced. In the schemes presented above, this was directly visible in the slope for example. Also, the accuracy necessarily is reduced to first-order near *smooth* extrema.

In this section, both Essentially Non-Oscillatory and Weighted Essentially Non-Oscillatory schemes are presented, which are self-similar (i.e., there is no mesh size dependent parameter), uniformly high-order accurate, yet essentially non-oscillatory for piecewise smooth functions (i.e., the magnitude of the oscillations decays with order of accuracy of the scheme). Piecewise smooth functions are smooth except at finitely many isolated points. At these points, the function and its derivatives are assumed to have finite left and right limits.

The key element of these schemes is the reconstruction. This is a specific interpolation technique, which was developed for piecewise smooth functions. It works by automatically choosing the locally smoothest stencil, and by that avoiding crossing discontinuities in the interpolation procedure as much as possible.

The Essentially Non-Oscillatory reconstruction algorithm starts with a stencil containing one or two cells only. It then adds either the cell to the right or the one to the left of the stencil, depending on which results in the less oscillatory interpolant.

Instead of choosing one of the candidate stencils and discarding the others, Weighted Essentially Non-Oscillatory reconstruction uses a convex combination of the interpolant through all candidate stencils.

First, the given two reconstructions are presented and then it is described how to establish a numerical flux from the corresponding reconstructions. This section is based on Shu [162].

12.2.3.1 Essentially Non-Oscillatory Reconstruction

Before describing the Essentially Non-Oscillatory (ENO) reconstruction, an important detail of interpolation methods used for reconstruction has to be addressed. In section 12.2.2 it was mentioned that linear interpolation in the MUSCL scheme was uncritical with respect to maintaining the proper cell average of the interpolant. In the context of the present methods, higher-order interpolation is considered and therefore the interpolant must be established in a way that maintains the cell average.

Assume that some function, say, velocity, is considered. The cell averages $\overline{v_i}$ of that function $v(x)$ are given on a grid. One is interested in a polynomial $p_i(x)$ of degree $k-1$ for each cell I_i. This then forms a k-th order approximation to $v(x)$ in the cell I_i. The polynomial shall be constructed such that its cell average shall agree with that of the original function $\overline{v_i}$.

Assume that, additionally to the cell I_i and the order of accuracy k, one is given a stencil $S(i)$ of k consecutive cells. The stencil is given via the left shift r, i.e., the stencil includes r cells to the left and s cells to the right of I_i, with $r+s+1=k$.

$$S\left(i\right)=\left\{I_{i-r},\ldots,I_{i+s}\right\} \tag{93}$$

In order to preserve the cell average, the interpolant over the stencil is established via the primitive function of $v(x)$.

$$V\left(x\right)=\int_{-\infty}^{x} v\left(\xi\right)d\xi$$

Then, the interpolant can be established. In computational implementations, this interpolation step is usually accelerated via the computation of so-called reconstruction coefficients. This is possible, because one is usually not interested in the complete interpolant but only in values of it at specific stations such as $x_{i+1/2}$. Due to the linearity of the mapping from the cell averages $\overline{v_i}$ to the interpolated values, these reconstruction coefficients depend on the left shift of the stencil r, the order k, and the mesh spacing Δx_i, but not on the function v itself.

The actual ENO approximation is addressed next. Here, an adaptive stencil is used. This means that the left shift r is not constant. A left shift r that is constant over the cells I_i would lead to a fixed stencil approximation (e.g., a central stencil) for which it was shown that it leads to spurious oscillations if of order two or higher with constant coefficients. In ENO approximation, the left shift is thus established for each cell I_i in a way that avoids including a cell with a discontinuous change in the stencil.

Harten et al. [80] showed that a robust criterion to identify the stencil with the “smoother” interpolant is to choose the one with the smaller absolute value of the Newton divided difference.

Recall the definition of the Newton divided differences. For the primitive function $V(x)$ the 0-th degree divided difference is

$$V\left[x_{i-1/2}\right] = V\left(x_{i-1/2}\right)$$

and the general j-th degree divided difference with $j \geq 1$ is defined as

$$V\left[x_{i-1/2}, \ldots, x_{i+j-1/2}\right] = \frac{V\left[x_{i+1/2}, \ldots, x_{i+j-1/2}\right] - V\left[x_{i-1/2}, \ldots, x_{i+j-3/2}\right]}{x_{i+j-1/2} - x_{i-1/2}}$$

Similarly, the divided differences of the cell averages are

$$\overline{v}\left[x_i\right] = \overline{v_i}$$

and in general

$$\overline{v}\left[x_i, \ldots, x_{i+j}\right] = \frac{\overline{v}\left[x_{i+1}, \ldots, x_{i+j}\right] - \overline{v}\left[x_i, \ldots, x_{i+j-1}\right]}{x_{i+j} - x_i} \tag{94}$$

Note that the zeroth degree divided difference of $\overline{v_i}$ is identical to the first degree divided difference of $V(x)$ due to the definition of the primitive function.

$$V\left[x_{i-1/2}, x_{i+1/2}\right] = \frac{V\left(x_{i+1/2}\right) - V\left(x_{i-1/2}\right)}{x_{i+1/2} - x_{i-1/2}} = \overline{v_i} \tag{95}$$

This equality allows to express the divided differences of $V(x)$ of degree $j \geq 1$ by those of $\overline{v_i}$ of degree $j \geq 0$. Taking the derivative of the k-th degree interpolation polynomial $P(x)$ to approximate $V(x)$, one finds that only divided difference of $\overline{v_i}$ of degree $j \geq 1$ are required to express $p(x)$.

The ENO approximation thus identifies the "smoothest" stencil in $\overline{v_i}$ via a stencil of $V(x)$, which is labeled $\hat{S}(i)$. Notice that from the latter the corresponding stencil in $\overline{v_i}$ can be identified via (95). First, the divided differences of the primitive function $V(x)$ are computed using (95) and, for degrees $j \geq 2$, using (94). Then, the algorithm starts with a two point stencil in $V(x)$,

$$\hat{S}_2(i) = \left\{x_{i-1/2}, x_{i+1/2}\right\}$$

This stencil is then consecutively enlarged for $l = 2, \ldots, k$. From the preceding step the following stencil is known

$$\hat{S}_l(i) = \left\{x_{i+1/2}, \ldots, x_{j+l-1/2}\right\}$$

and one of the neighboring points $x_{j-1/2}$ and $x_{j+l+1/2}$ is added to the stencil. If

$$\left|V\left[x_{j-1/2}, \ldots, x_{j+l-1/2}\right]\right| < \left|V\left[x_{j+1/2}, \ldots, x_{j+l+1/2}\right]\right|$$

then $x_{j-1/2}$ is added to $\hat{S}_l(i)$ to obtain $\hat{S}_{l+1}(i)$. If the inequality is not fulfilled, then $x_{j+l+1/2}$ is added to the stencil.

As soon as the stencil is completely established, Lagrange or Newton interpolation can be used to find the interpolants. In computational implementations the reconstruction coefficients mentioned at the beginning of this section are usually used instead. By the choice of the stencil the left shift r is established. Then, the proper reconstruction coefficients can be used to instantly establish the interpolated values at the interface locations.

Figure 81 illustrates the interpolants chosen by Essentially Non-Oscillatory schemes. For the example $\overline{v} = \{10, 10.4, 10.25, 10, 3, 2.5, 2.25, 2\}$ and $x = \{1, 2, 3, 4, 5, 6, 7, 8\}$ were assumed. First, consider the resulting interpolant for cell 3. The scheme described above starts the stencil with this cell and extends it twice (i.e., *order* $- 1$ times) to the left or right. As described, the schemes includes either neighbor point that results in a smoother interpolant according to the criterion of divided differences. For cell 3, the scheme once selects a cell to the left and once a cell to the right for inclusion in the stencil. For cell 4 in turn, including the right cell (cell 5) would lead to rather large gradients in the interpolant each time. Therefore, the stencil is extended twice to the left. The interpolant for cell 4 is therefore identical to that of cell 3. For cells 5 and 6, the stencil is only extended to points to the right for similar reasons.

The left limit of $v_{4+1/2}$ is established based on the interpolant of cell 4, i.e., $v^{-}_{4+1/2} = 9.84$. The right limit is $v^{+}_{4+1/2} = 3.33$.

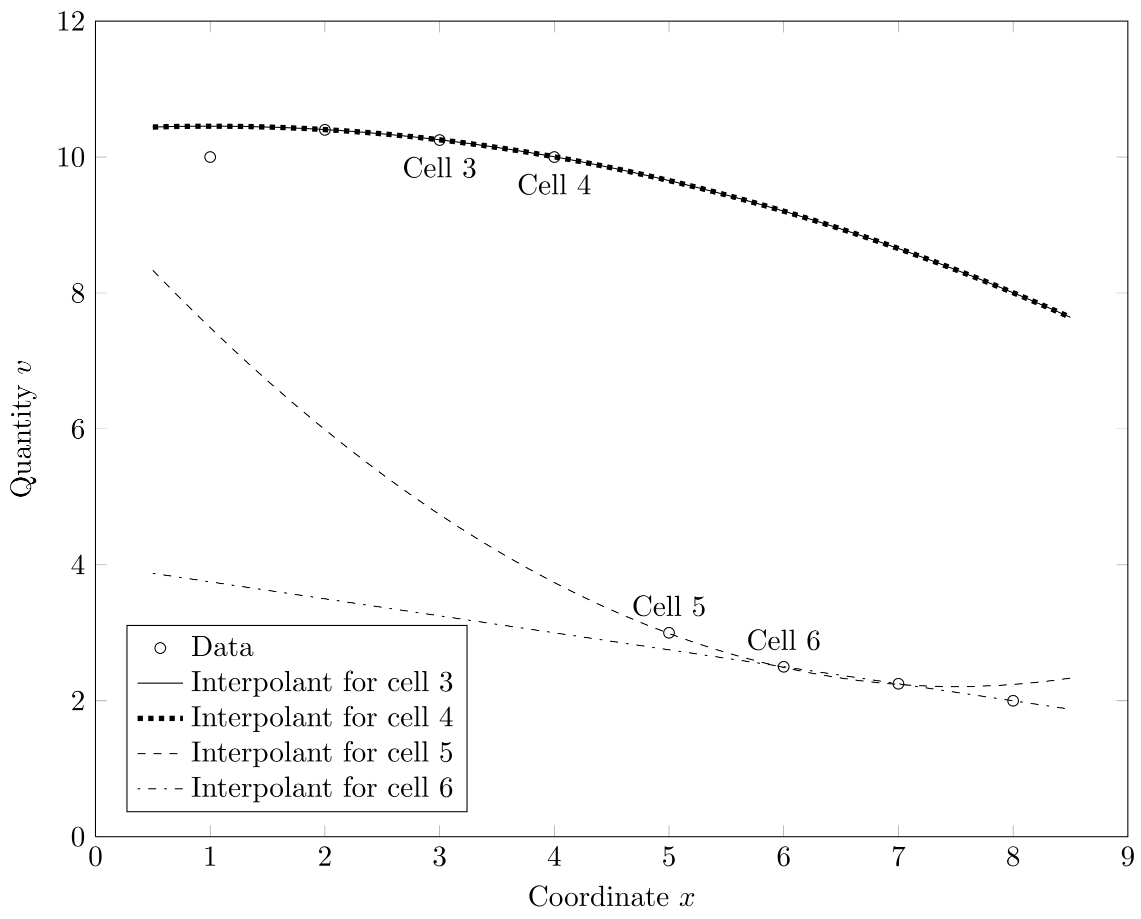


Figure 81: Third-order ENO reconstruction

12.2.3.2 Weighted Essentially Non-Oscillatory Reconstruction

ENO schemes are uniformly high-order accurate right up to the discontinuity, which is achieved by adaptively switching the stencil used for interpolation. However, certain properties leave room for improvements [162]:

- The stencil may change near zeros of the solution even by a round-off error perturbation.
- As the left shift of the stencil may change at neighboring points, the resulting numerical flux is not smooth.
- To the reconstruction scheme, $2k - 1$ cells are available. In the end, only k cells are used. This results in k-th order accuracy when $2k-1$-th order accuracy is theoretically possible in smooth regions of the solution.

The idea of Weighted Essentially Non-Oscillatory (WENO) reconstruction is to use a convex combination of the interpolants through several stencils. If, however, a candidate stencil contains a discontinuity, its weight shall be close to zero to mimic the successful properties of ENO schemes.

For each cell I_i k candidate stencils are consequently available.

$$S_r\left(i\right) = \{x_{i-r}, \ldots, x_{i-r+k-1}\}$$

with $r = 0, \ldots, k-1$. Using the reconstruction coefficients, each stencil produces a different reconstruction of $v_{i+1/2}$, which is labeled $v^{(r)}_{i+1/2}$. A convex combination of these values is used to define the reconstruction using the WENO method.

$$v_{i+1/2} = \sum_{r=0}^{k-1} \omega_r v^{(r)}_{i+1/2}$$

For stability and consistency, $\omega_r \geq 0$ and $\sum\limits_{r=0}^{k-1} \omega_r = 1$ need to be imposed. In smooth regions, these weights should approximate optimal high-order weights to $k-1$-th order, which would imply $(2k-1)$-th order of the complete reconstruction scheme. The question is now what these optimal weights are. In the general case, this leads to an overdetermined system of equations, which can be solved, e.g., by using a least-squares algorithm. In the case of a uniform mesh, the equation system becomes square and an explicit solution can be computed. Jiang and Shu [91] gave optimal weights d_r for uniform grids and $1 \leq k \leq 3$. Herein, $k = 3$ is considered. For this value of k, the following optimal weights have been established.

$$d_0 = \frac{3}{10}, \qquad d_1 = \frac{3}{5}, \qquad d_2 = \frac{1}{10}$$

Furthermore, Jiang and Shu [91] suggested the following form of the weights

$$\omega_r = \frac{\alpha_r}{\sum\limits_{s=0}^{k-1} \alpha_s}$$

for $r = 0, \ldots, k-1$. Coefficients α_r in turn are defined as follows

$$\alpha_r = \frac{d_r}{(\varepsilon + \beta_r)^2}$$

Here, $\varepsilon > 0$ is introduced to avoid division by zero. Following Jiang and Shu [91], $\varepsilon = 10^{-6}$ was used in computations. β_r are called smooth indicators in the given reference and have been defined as follows

$$\beta_r = \sum_{l=1}^{k-1} \int_{x_{i-1/2}}^{x_{i+1/2}} \Delta x^{2l-1} \left(\frac{\partial^l p_r(x)}{\partial x^l} \right)^2 dx$$

This is the sum of the squares of the scaled L^2 norms for all derivatives of the interpolation polynomial $p_r(x)$ over the interval $\left(x_{i-1/2}, x_{i+1/2}\right)$. For $k = 3$, the result is a $2k - 1 = 5$-th order accurate reconstruction.

Figure 82 illustrates Weighted Essentially Non-Oscillatory reconstruction on the same example as figure 81. The reconstruction of the left limit of $v_{4+1/2}$ is considered, i.e., $v^-_{4+1/2}$. For this, the scheme uses three stencils $S_r(4)$ with increasing left-shift r. The interpolants based on these stencils are illustrated in the figure. Note the strong gradients in the interpolants using $S_0(4)$ and $S_1(4)$. This is also an illustration that the stencil selection of the ENO scheme shown in figure 81 for cell 4 was reasonable.

The WENO scheme proceeds with the different reconstruction values $v^{(0)}_{4+1/2}$ to $v^{(2)}_{4+1/2}$, which are each marked with a filled circle in figure 82. For this particular example, the scheme results in weights $\omega_0 = 1.3 \cdot 10^{-6}$, $\omega_1 = 15.6 \cdot 10^{-6}$, $\omega_2 = 0.999983$. This means, that the interpolant with left-shift $r = 2$ dominates and $v^-_{4+1/2} \approx v^{(2)}_{4+1/2}$.

12.2.3.3 ENO and WENO numerical fluxes

So far, two different algorithms for the reconstruction of piecewise smooth functions were introduced. The question is now how to construct corresponding higher-order numerical fluxes for the system of hyperbolic conservation laws (74) from these reconstructions.

Probably, the easiest way to do this is to apply the reconstruction to each component of the vector of conserved variables u separately and thus reconstruct the left and right limit $u^{\pm}_{i+1/2}$ at the location $x_{i+1/2}$. Then, a monotone first-order flux can be used to establish an essentially non-oscillating higher-order numerical flux.

Shu [162] remarks that only low-order schemes are highly sensitive to the choice of first-order monotone flux. This sensitivity decreases with increasing order of accuracy and therefore a simple Lax-Friedrichs monotone flux is used in the given reference to construct higher-order WENO numerical fluxes.

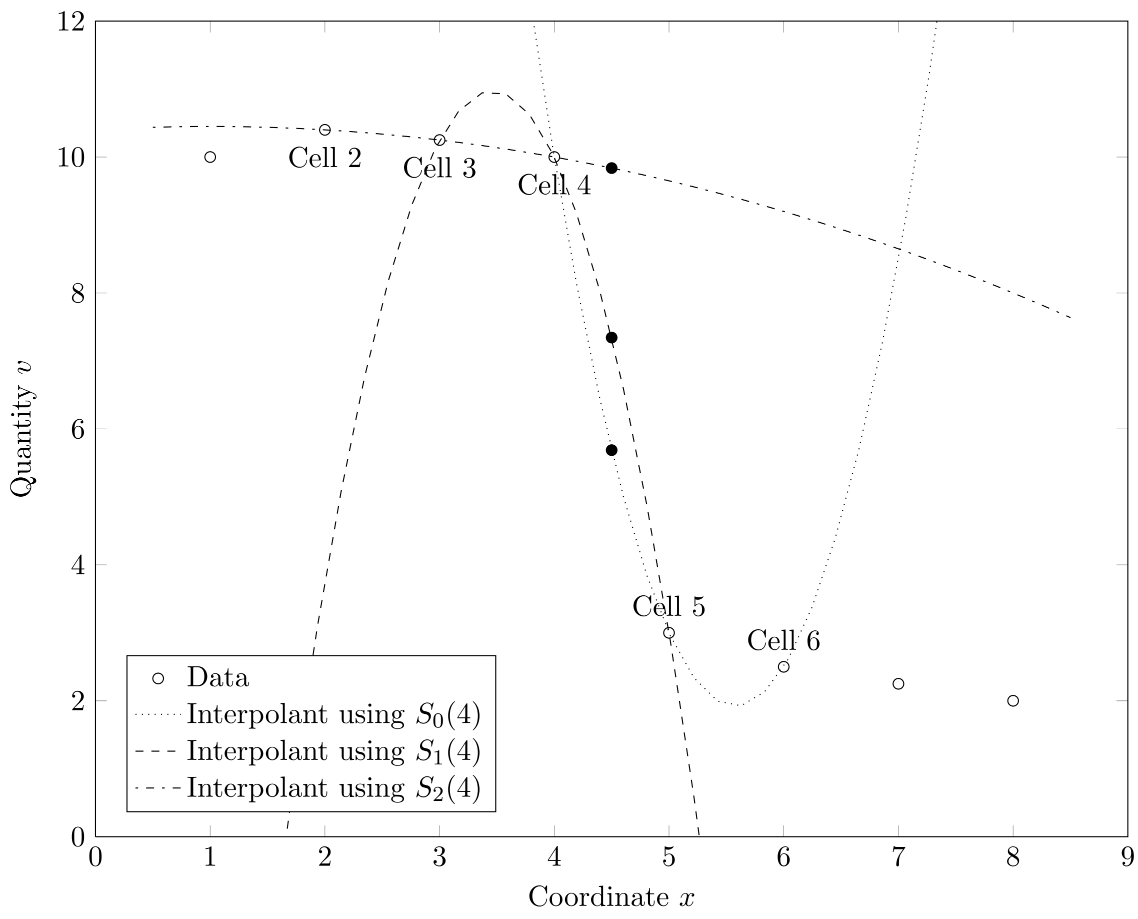


Figure 82: Fifth-order WENO reconstruction

The given component-wise approach to construct a numerical flux based on ENO and WENO reconstructions is simple to implement. Also, the resulting schemes work reasonably well for many applications, in particular if the order of the scheme is not high. Shu [162] mentions "second or sometimes third-order".

If the order of the scheme is high or a more demanding test problem shall be solved, the following characteristic decomposition is much more robust and should be implemented instead.

Recall the diagonal decomposition of the Jacobian of the flux in section 12.2.1.2 on flux vector splitting, (80). A change of variables $v = K^{-1}u$ leads to a decoupling of the system of conservation laws (74). Then, the component-wise application of the ENO or WENO reconstruction is fundamentally justified. The reconstructed values $v^{\pm}_{i+1/2}$ are then transformed back into the physical space of conserved variables,

$$u^{\pm}_{i+1/2} = Kv^{\pm}_{i+1/2}$$

A remaining question is the choice of K, which depends on u, $K = K(u)$. For this purpose, the Roe averages introduced in section 12.2.1.1 were used, as this leads to advantageous properties such as the satisfaction of the mean value theorem.

Based on the reconstructed left and right limit $u^{\pm}_{i+1/2}$ at the location $x_{i+1/2}$, a monotone first-order flux is used again to establish an essentially non-oscillating higher-order numerical flux.

12.3 Object-oriented conservative methods

Two libraries for object-oriented modeling and simulation of gas dynamics were developed within this thesis. Both were written in the equation-based, object-oriented modeling language Modelica. The first one is a library specific to ideal gases, which allows several simplifications and results in little computational overhead. The second one is a gas dynamics library for generic thermodynamic property models. These thermodynamic property models are implemented according to the object-oriented interface Modelica.Media [58]. This interface had to be extended with two additional methods to be suitable for applications in gas dynamics. These and other implementation aspects are discussed in this section.

12.3.1 Ideal gas and generic thermodynamic property models

A large fraction of the literature on discretization methods using conservative methods considers ideal gas equations of state only. Discretizations using real gas[1] equations of state in turn consider non-ideal media, too. Several articles make assumptions on the structure of the real gas equations of state however (e.g,. Liou et al. [104] assume a "general pressure function" but require that is be explicit in density, specific internal energy, and mass fractions, and Gallouët et al. [66] explicitly assume Tammann and van der Waals equations of state).

In equation-based, object-oriented modeling and simulation, one aims to encapsulate the equations of state in separate classes and implement discretization methods independently using a generic interface. As the given real gas schemes require structural assumptions on the equations of state, too, a generic interface had to be extended with several methods specific to these structural assumptions. A clean separation between discretization scheme and equation of state appears to be difficult in this case.

A large fraction of the methods described in the previous section 12.2 are specific to ideal gases with constant specific heat capacity c_p. Specialized Riemann solvers can be constructed easily for some of these methods (such as the HLLE method described in section 12.2.1.1). In the context of equation-based, object-oriented modeling languages, such approximate Riemann solvers had to be exchanged *concurrently* with the equations of state. A more practical solution is the use of centered schemes. These schemes are independent of any Riemann solver and can thus be used with any thermodynamic property model. As described in section 12.2.1.3, the support of these schemes does not depend on the sign of the characteristic speeds. While the upwind schemes as discussed in sections 12.2.1.1 and 12.2.1.2 are more accurate in several cases than their centered counterparts, they are usually more complex and computationally expensive [174]. Therefore, in the libraries described herein, monotone and TVD centered schemes as well as schemes using higher-order reconstruction with a centered scheme are implemented for general thermodynamic property models and upwind methods are restricted to ideal gases.

12.3.2 Generic interface to thermodynamic property computations

As described above, the object-oriented interface of Modelica.Media [58] is used for thermodynamic property computations. In order to be applicable to gas dynamics, this interface has to be extended with two additional methods.

The first extension is required for the conversion of conserved variables to primitive variables. In the gas dynamics library for generic equations of state the primitive variables are velocity v and the thermodynamic state record of the medium[2]. For the conversion of the vector u as defined in equation (75) to the primitive variables an additional `setState` function is thus required. From u, density and specific internal energy can be established. Therefore, a function `setState_duX` is used.

The second extension is required for the conversion of the classic primitive variables $\{\rho, v, p\}$ to the ones used in the object-oriented implementation for generic thermodynamic property computations, the thermodynamic state record and velocity. This is necessary in case of a characteristic decomposition such as the one discussed in section 12.2.3.3. For this purpose, a function `setState_pdX` is required. Note that this is only required if a gas dynamics library for generic thermodynamic property models shall also be used with ideal gases.

[1]In this thesis, a real gas is one that is not both thermally and calorically ideal.

[2]In place of the velocity the mass flow rate could have been used, too. This selection is ambiguous and was eventually made for similarity with conventional implementations of gas dynamics.

12.3.3 Conservative and non-conservative formulations

In order to obtain valid simulation results, the conserved quantities in the governing equations and the conservation statements they imply have to make physical sense [174]. Formulations that are conservative purely in a mathematical sense (i.e., formally, they can be expressed as (74), but there is no corresponding conservation law in physics) will, in case of shock waves, result in wrong shock speeds and therefore wrong solutions [174].

In the context of equation-based, object-oriented modeling languages, a simple solution is to explicitly select the conserved variables themselves as state variables, i.e., $u(x,t)$. This is done in the gas dynamics library specific to ideal gases. For ideal gases that are both thermally and calorically ideal (in particular, c_p is not a function of temperature), all intensive quantities can be established in closed form based on any two thermodynamic potentials. Therefore, no distinction between independent and dependent variables is required for such media.

For generic thermodynamic property models this is different. In general, such models are explicit in a number of thermodynamic potentials only (e.g., pressure and specific enthalpy). As long as the physical flux is not changed, it is then possible to use the independent variables of a thermodynamic property model as state variables instead. This is the approach followed in the gas dynamics library for generic thermodynamic property models.

12.3.4 Inhomogeneous problems

In several references on computational methods for gas dynamics, fully explicit conservative methods are considered in contrast to (78). In the context of equation-based, object-oriented modeling, it is natural however to use a semi-discretized formulation. Furthermore, this has advantages for inhomogeneous problems. No source term splitting schemes [167] are required for the present approach. With the semi-discretization (also called method of lines) both the numeric fluxes and the source term are algebraic expressions and no further complications arise for inhomogeneous problems.

12.3.5 Library design

In this section, the design of the two gas dynamics libraries is sketched. The one considering generic thermodynamic property models is emphasized and some remarks are made on the one specific to ideal gases. For readability, the code illustrates single-substance media only. Mass fractions of multiple-substance media can be covered analogously to the other primitive variables, because they are similarly dominated by convection.

The connector has to implement the stencil defined in equation (12). Its length depends on the stencil length required by the discretization scheme. If the stencil for a flux computation has to include n cells, then at least $n/2$ of these cells are inside the domain modeled by the respective component and need not be accessed via the connector. This implies that at most $n/2$ cells of the stencil have to be provided by the connector. Therefore, the following connector definition is used[3].

```
connector Stencil_a
  "Interface for quasi one-dimensional high-speed flow"

  replaceable package Medium =
    Modelica.Media.Interfaces.PartialMedium "Medium model";
```

[3]Obviously, the `stream` prefix introduced in chapter 5 could be used for the present stencil definition. However, high-speed compressible flow applications would use a part of the functionality of stream connectors only (a type of "flip" functionality to meet requirement 10 introduced in section 3.4). The equations for mixing in non one-to-one connections are not meaningful for the present application. Therefore, a connector definition with `input` and `output` prefixes was used instead. A possible alternative for Modelica language design was the introduction of a `flip()`-operator for the functionality of interest in cases like the present one. Then, the additional functionality provided by the `inStream()`-operator could be placed in explicit junction models leading to a leaner language definition.

```

  replaceable package Discretization =
      GasDynamics.Discretizations.Partial.PartialDiscretization
    "Discretization";

  output Medium.ThermodynamicState
    state_a[Discretization.halfStencilLength]
    "Thermodynamic state stencil";
  output SI.Velocity v_a[Discretization.halfStencilLength]
    "Velocity stencil";
  output SI.Length x_side_a[Discretization.halfStencilLength]
    "Cell side coordinate";

  input Medium.ThermodynamicState
    state_b[Discretization.halfStencilLength]
    "Thermodynamic state stencil";
  input SI.Velocity v_b[Discretization.halfStencilLength]
    "Velocity stencil";
  input SI.Length x_side_b[Discretization.halfStencilLength]
    "Cell side coordinate";
end Stencil_a;
```

Listing 39: Connector for high-speed compressible flow

Note the replaceable discretization package ("`Discretization`") in the connector definition in addition to the replaceable package containing the thermodynamic property model ("`Medium`"). A vector of thermodynamic states and one of velocities of the given length are defined twice. Different causal prefixes are used to handle how one component "prescribes" and "reads" which variables[4]. The library considering ideal gases only uses density and pressure vectors in place of the thermodynamic state.

Additionally, information about the computational mesh has to be included in the connector. In the proposed connector definition, the coordinates of the sides of the cells are used. They are defined in a local coordinate system, whose origin is set to the side shared by two components connected together. The coordinate of this shared side can thus be omitted and the same number of side coordinates and cell center variables on the thermodynamic state and velocity is included. The side coordinates for `Stencil_a` are defined strictly positive; those for `Stencil_b` strictly negative.

Analogous to the `Stencil_a` connector definition in listing 39, a connector `Stencil_b` is defined. It differs only in inverted causality prefixes (`input` instead of `output` and vice versa).

The discretization package contains structural parameters including the stencil length, conversion functions, an exchangeable thermodynamic properties model, and flux functions. Its interface is defined as follows.

```
partial package PartialDiscretization
  "Interface for discretization in compact flux form"

  // Description
  constant String discretizationName =
    "unusablePartialDiscretization"
    "Name of the discretization";

  // Type of discretization
  constant Boolean idealGasOnly = false
```

[4]The causal prefixes are used in the acausal modeling language just to define a nominal causality, not an actual one.

```
    " = true, if contains specifics of ideal gases";
  constant Integer order(min=1) = 1
    "Order of discretization method";

  // Stencil definition
  constant Integer halfStencilLength = 1
    "Half of length of stencil for flux f_(i+1/2)";
  final constant Integer stencilLength = 2*halfStencilLength
    "Length of stencil for flux f_(i+1/2)";

  // ...

end PartialDiscretization;
```

Listing 40: Discretization interface, structural parameters

The structural parameters of a Discretization are its name, whether it uses equations applicable to ideal gases, its order of accuracy, and the stencil length.

```
partial package PartialDiscretization
  "Interface for discretization in compact flux form"

  // ...

  function primitiveToConserved
    "Convert primitive variables to conserved variables"
    input Medium.ThermodynamicState state "Thermodynamic state";
    input SI.Velocity v "Velocity";
    output Real u[3] "Vector of conserved variables";
  algorithm
    u := {Medium.density(state), Medium.density(state)*v,
      Medium.density(state)*
       (Medium.specificInternalEnergy(state) + 1/2*v*v)};
  end primitiveToConserved;

  function conservedToPrimitive
    "Convert conserved variables to primitive variables"
    input Real u[3] "Vector of conserved variables";
    output Medium.ThermodynamicState state "Thermodynamic state";
    output SI.Velocity v "Velocity";
  algorithm
    v := u[2]/u[1];
    state := Medium.setState_duX(u[1], u[3]/u[1]-1/2*v*v,
                Medium.X_default);
  end conservedToPrimitive;

  // ...

end PartialDiscretization;
```

Listing 41: Discretization interface, conversion functions

The conversion functions of a Discretization convert the set of primitive variables (thermodynamic state record and velocity) to the vector of conserved variables as defined in equation (75) and vice versa. Note that these functions need not be replaceable, because the implementations are generally valid. Note that in the second conversion function in listing 41 one of the additional functions mentioned in section 12.3.2 is used (setState_duX()).

```
partial package PartialDiscretization
  "Interface for discretization in compact flux form"

  // ...

  replaceable partial function monotoneFlux
    "First-order flux approximation"
    input Medium.ThermodynamicState state_l
      "Stencil of thermodynamic states on left (i)";
    input Medium.ThermodynamicState state_r
      "Stencil of thermodynamic states on right (i+1)";
    input SI.Velocity v_l "Velocity in x-dir on left, v_(i)";
    input SI.Velocity v_r "Velocity in x-dir on right, v_(i+1)";
    output Real flux[3] "Fluxes f_(i+1/2)";
  end monotoneFlux;

  replaceable partial function flux "Numeric flux approximation"
    input Medium.ThermodynamicState state[stencilLength]
      "Thermodynamic state stencil";
    input SI.Velocity v[stencilLength] "Velocity stencil";
    input Real x_side[stencilLength + 1]
      "Coordinates of cell sides (i-1/2), (i+1/2) etc.";
    output Real flux[3] "Fluxes f_(i+1/2)";
  end flux;

  // ...

end PartialDiscretization;
```

Listing 42: Discretization interface, flux functions

The key elements of a Discretization are the flux functions. Their interfaces are described in listing 42. For readability, interfaces are defined for both a monotone first-order flux and the arbitrary-order numerical flux. This allows to clearly separate the reconstruction and the Riemann solver for instance. In models, only the arbitrary-order numerical flux is used and therefore the use of the monotone flux function is optional. The monotone flux arguments are the left and right thermodynamic state and the flow velocities. It returns the flux vector. The arbitrary-order flux function has a stencil of thermodynamic states and of velocity as well as the cell side coordinates as inputs and also returns the flux vector. The Discretization package also contains a replaceable package implementing thermodynamic properties. This is not shown in listings 40 to 42. Discretization packages were implemented using the Local Lax-Friedrichs flux, Roe's Riemann solver, the HLLE Riemann solver, the Steger-Warming flux vector splitting, the First-Order Centered flux, the Muscl-Hancock TVD scheme with several limiters and monotone fluxes both in upstream and in centered versions, third- to ninth-order ENO schemes and several fifth-order WENO schemes with and without characteristic decomposition.

The implementation of a Discretization is illustrated for a second-order Muscl-Hancock scheme with a Superbee limiter and a Local Lax-Friedrichs flux. The listing is given in two parts for readability.

```
package MusclHancockSuperbeeLF
  "TVD: Upwind Muscl-Hancock (Superbee, Lax-Friedrichs)"
  extends Partial.PartialDiscretization(
    discretizationName="Upwind Muscl-Hancock (SB, LF)",
    order=2,
    idealGasOnly=false,
    halfStencilLength=2);
  redeclare function extends monotoneFlux
    "Local Lax Friedrichs monotone flux"
  protected
    Real lambda_max "Local wave speed";
  algorithm
    // Local wave speed
    lambda_max := max(abs(v_l) +
      Medium.velocityOfSound(state_l), abs(v_r) +
      Medium.velocityOfSound(state_r));
    // Mass flux
    flux[1] := 0.5*(Medium.density(state_r)*v_r +
        Medium.density(state_l)*v_l) -
      0.5*lambda_max*(Medium.density(state_r) -
        Medium.density(state_l));
    // Momentum flux
    flux[2] := 0.5*((Medium.density(state_r)*v_r*v_r +
        Medium.pressure(state_r)) +
      (Medium.density(state_l)*v_l*v_l +
        Medium.pressure(state_l))) -
      0.5*lambda_max*(Medium.density(state_r)*v_r -
        Medium.density(state_l)*v_l);
    // Energy flux
    flux[3] := 0.5*(
      v_r*(Medium.density(state_r)*
        (Medium.specificInternalEnergy(state_r) + 1/2*v_r^2) +
        Medium.pressure(state_r)) +
      v_l*(Medium.density(state_l)*
        (Medium.specificInternalEnergy(state_l) + 1/2*v_l^2) +
        Medium.pressure(state_l))) - 0.5*lambda_max*(
      Medium.density(state_r)*Medium.specificInternalEnergy(state_r) -
      Medium.density(state_l)*Medium.specificInternalEnergy(state_l));
  end monotoneFlux;

  // ...

end MusclHancockSuperbeeLF;
```

Listing 43: Listing of a Muscl-Hancock discretization, part 1

First, structural parameters such at the order of approximation and the length of the stencil are defined. The monotone flux implements equation (88).

```
package MusclHancockSuperbeeLF
  "TVD: Upwind Muscl-Hancock (Superbee, Lax-Friedrichs)"

  // ...

  redeclare function extends flux
  protected
    Real u_i[3] = primitiveToConserved(state[2], v[2])
      "Vector of conserved variables u_(i)";
    Real u_ip1[3] = primitiveToConserved(state[3], v[3])
```

```
      "Vector of conserved variables u_(i+1)";
    Real Delta_imh[3] =
      u_i - primitiveToConserved(state[1], v[1])
      "Delta_(i-1/2)";
    Real Delta_iph[3] =
      u_ip1 - u_i "Delta_(i+1/2)=Delta_(i+1-1/2)";
    Real Delta_ip3h[3]=
      primitiveToConserved(state[4], v[4]) - u_ip1
      "Delta_(i+3/2)";

    Real Delta_i_limited[3]={if Delta_iph[i] > 0 then
      max({0,min(2*Delta_imh[i],Delta_iph[i]),
        min(Delta_imh[i], 2*Delta_iph[i])}) else
      min({0,max(2*Delta_imh[i], Delta_iph[i]),
        max(Delta_imh[i], 2*Delta_iph[i])}) for i in 1:3}
      "Delta_(i)_limited";
    Real Delta_ip1_limited[3]={if Delta_ip3h[i] > 0 then
      max({0,min(2*Delta_iph[i],Delta_ip3h[i]),
        min(Delta_iph[i], 2*Delta_ip3h[i])}) else
      min({0,max(2*Delta_iph[i], Delta_ip3h[i]),
        max(Delta_iph[i], 2*Delta_ip3h[i])}) for i in 1:3}
      "Delta_(i+1)_limited";
    Real u_i_r[3] = u_i + 0.5*Delta_i_limited
      "Vector of conserved variables u_(i)_r";
    Real u_ip1_l[3] = u_ip1 - 0.5*Delta_ip1_limited
      "Vector of conserved variables u_(i+1)_l";
    Medium.ThermodynamicState state_i_r "At u_(i)_r";
    Medium.ThermodynamicState state_ip1_l "At u_(i+1)_l";
    SI.Velocity v_i_r "Velocity u_(i)_r";
    SI.Velocity v_ip1_l "Velocity u_(i+1)_l";
  algorithm
    (state_i_r, v_i_r) := conservedToPrimitive(u_i_r);
    (state_ip1_l, v_ip1_l) := conservedToPrimitive(u_ip1_l);
    flux := monotoneFlux(state_i_r, state_ip1_l, v_i_r, v_ip1_l);
  end flux;
end MusclHancockSuperbeeLF;
```

Listing 44: Listing of a Muscl-Hancock discretization, part 2

The reconstruction implements the scheme described in section 12.2.2.1 for a Superbee limiter.

12.3.6 Applications

Results of a Sod-type problem are shown in figure 83. Here, the results of computations using the Local Lax-Friedrichs scheme (a first-order monotone centered method) are compared to those using a fifth-order WENO scheme (using Roe's first-order monotone flux and a characteristic decomposition). The figure illustrates the generally accepted result that proper higher-order reconstructions lead to higher resolution of shock waves, expansion fans, and contact discontinuities [174]. That is, such phenomena are smeared over fewer computational cells.

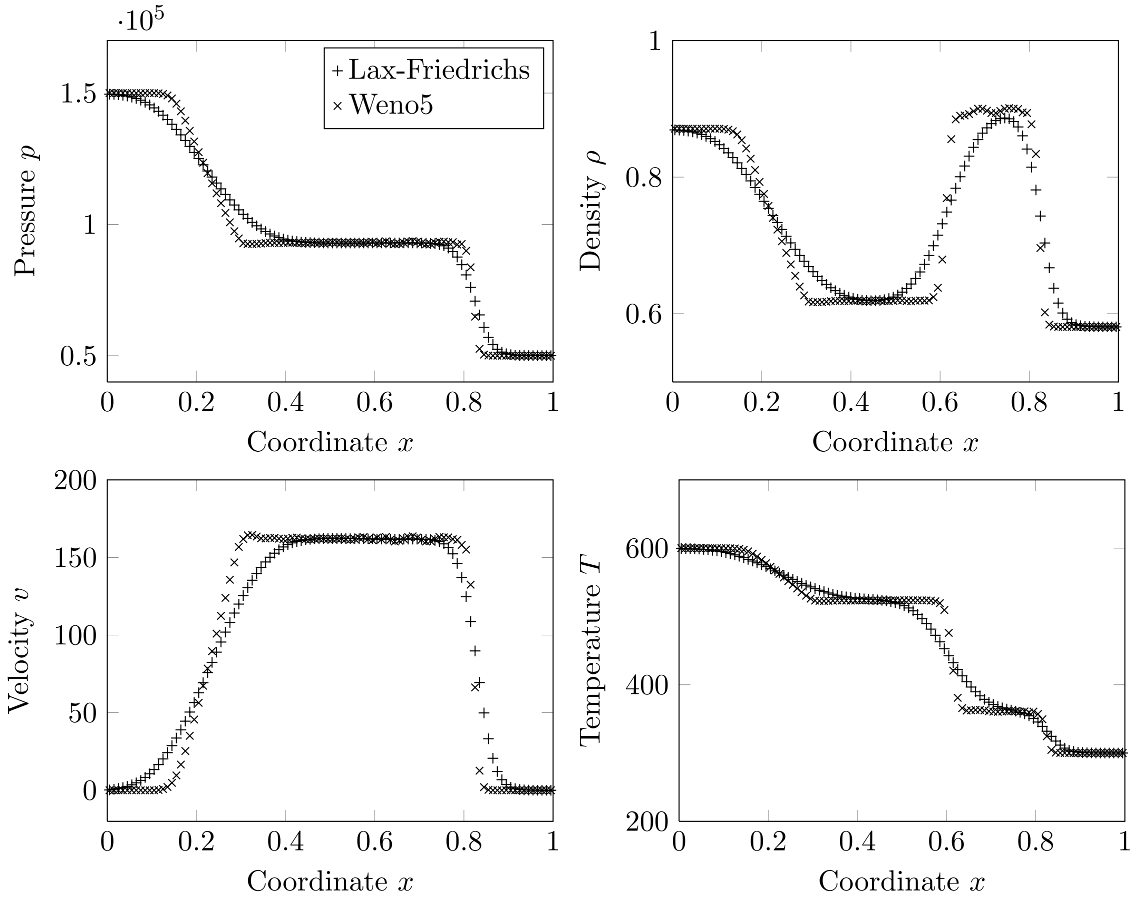


Figure 83: Comparison of Local Lax-Friedrichs and fifth-order WENO schemes on a Sod-type problem

CHAPTER 13

DESIGN OF ENVIRONMENTAL CONTROL SYSTEMS

The objective of this chapter is to analyze the general aircraft systems design problem and to propose a suitable design methodology. The design methodology shall be applied to implement a design environment for ECS. Such a design environment has to build upon a modeling and simulation capability, which was addressed in chapter 11, and provides optimization functionality. In order to be useful for optimization, all components in the modeling and simulation environment have to be implemented as parametric elements. Then, parameters relating component performance and weight can be varied within an optimization platform to converge toward a set of optimal designs. Ideally, such an optimization platform additionally provides instruments to capture the contradicting requirements that constitute each design problem of airborne systems. After all, when striving for both energy efficiency and weight reduction on system-level, some trade-off has to be made. Furthermore, an optimization platform for unconventional system architectures has to address the definition of control logic. This is necessary because a plant model by itself is often not a well-defined (i.e., square) problem, unless control variables are defined. Whenever unconventional architectures are investigated, such control logic is not known a-priori and a sequential approach of first establishing and implementing appropriate control logic and then optimizing the architecture is considered impractical.

The design of general aircraft systems (and ECS) is termed a local optimization problem. It is local, because it considers a single aircraft system architecture at a time and depends on global aircraft data reflecting aircraft constraints and dependencies (such as trade factors for specific fuel consumption, SFC, and mission block fuel, see section 1.3.2).

The data on aircraft constraints and mission block fuel have to be provided by some global aircraft systems architecture optimization process. For the design of distribution systems and energy management systems like electrical or thermal management systems peak power and power profiles from consumer systems need to be provided. Eventually, such local design platforms need to be integrated in an global aircraft systems architecture design effort.

In this section, some formal characteristics of the local optimization problem and associated challenges are discussed. Then, a local design platform is proposed and its implementation is presented. Finally, illustrative results are provided.

13.1 Implicit sizing

A challenge in optimization of ECS architectures lies in the sizing process of the components, which is not explicit. That is, a given set of functional requirements does not lead directly to a sizing of the system components. In some domains, this is different. For Wing Ice Protection Systems (WIPS) for example, explicit sizing is possible via the Extend of Protection and technology parameters. For architectural models of Electric Power Generation and Distribution Systems, performance metrics such as power or current to be provided by a given component lead to an explicit sizing of said components.

For ECS design such explicit sizing rules are only available for a set of conventional architectures (e.g., the one illustrated in figure 79). The methods discussed in this thesis need to scale up to unconventional architectures however, for which such explicit sizing processes are not established. Furthermore, an additional aspect has to be considered.

In reality, the sizing process of an airborne system involves a trade-off between sub-system efficiencies and sub-system weights. As illustrated in figure 84, maximum subsystem efficiency (e.g., ECS energy efficiency) does usually not correspond to maximum system efficiency (i.e., aircraft fuel burn). For the systems, which are sized explicitly, this trade-off can be factored out of the design process. This may be the case, because efficiency is more a technology parameter than a question of the size of a component. For ECS architecture, this is different. A heat

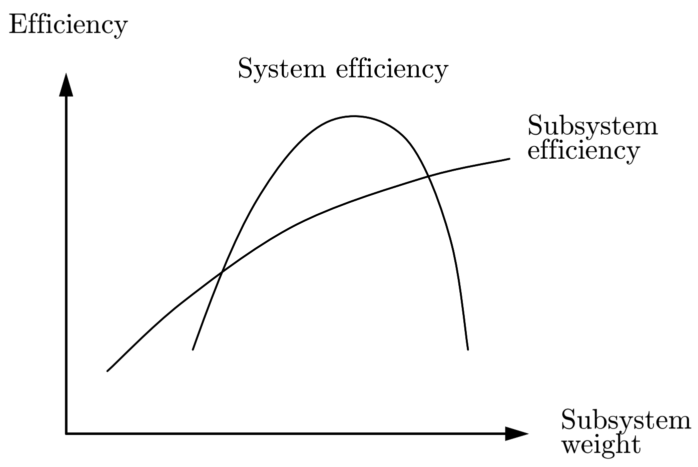


Figure 84: System efficiency has a maximum at optimum trade-off between subsystem efficiency and weight

exchanger becomes, independently of the particular technology employed, more efficient if the heat transfer surface and thus its size increases.

For the applications considered here, the conclusion is obvious. The trade-off between subsystem weight and its efficiency for maximum system efficiency cannot be found a-priori but has to be investigated for each design separately. This leads to an implicit sizing process and the requirement for an elaborate optimization process to design ECS architectures.

13.2 Control variables

Before providing a somewhat formal definition of the problem to be solved, the scope of the optimization problem shall be investigated. For this purpose, it has to be noted that a simulation problem of an ECS plant model by itself is not a well-defined (i.e., square) problem, because control variables need to be defined (e.g., the valve position of the Temperature Control Valve of the Three Wheel Bootstrap ECS shown in figure 79). The question arises of how to choose these control variables during optimization of ECS architecture. It is noted that if some control logic for a specific architecture is known, it is obviously possible to establish the control variables based on this logic. Whenever unconventional architectures are investigated, such control logic is not known. Then, a sequential approach of first establishing and implementing appropriate control logic and then optimizing the architecture is considered impractical. Therefore, an obvious choice is to do so in parallel. Consequently, the optimization problem has to cover both the actual architecture optimization and the definition of control logic.

13.3 Optimization problem

Formally, an optimization problem is described by a set of design variables (also called tuners), inequality or equality constraints, and objective functions. These are discussed step by step in order to define the problem to be solved herein.

Based on the remarks made in the preceding section, the set of design variables consists of variables describing the architecture and ones defining the operation of the system on a given evaluation or sizing mission. Architecture design variables can be continuous variables (e.g., heat exchanger dimensions) or discrete (e.g., Engine Bleed Air System stages, the number of the compressor stage at which intermediate or high pressure air is bled from the engine). The set of potential architecture design variables is defined by the architecture itself and by the modeling hypotheses of the components used in the simulation model.

The set of design variables defining the operation of the system on the evaluation mission similarly depends on the architecture. For the Three Wheel Bootstrap ECS shown in figure 79, these are the position of the Temperature Control Valve and of the Ram Air Channel (RAC) actuators. Additionally, the set of control variables depends on whether a direct model (calculating in performance direction) or an inverse model is used. For the former, usually the whole set of control variables has to be established, for the latter only a subset. In this thesis, only direct and semi-inverse models were used for optimization, because they can be evaluated for architectures, which do not fulfill all performance requirements. As the optimizer reduces the excess performance available in a design in order to reduce weight, such infeasible designs are routinely suggested by the optimizer. If, however, an inverse model is evaluated with, e.g., a pack discharge temperature below the minimum discharge temperature, the evaluation will fail and the optimizer cannot be returned a value quantifying how infeasible the architecture is. A direct model in turn will return the minimum discharge temperature and its deviation from the demanded discharge temperature is an exact means of quantifying how infeasible the architecture is.

Constraints are typically present with respect to the operation of the ECS system. When establishing the control strategy, inequality constraints have typically to be respected (e.g., on maximum temperatures due to the usage of aluminum alloys or on maximum water content of the pack discharge). Additionally, equality constraints may be present in optimization problems utilizing direct models (e.g., on pack discharge temperature).

The top-level objective of the optimization problem is, for example, aircraft-level energy efficiency expressed by mission block fuel. Additionally, ECS system weight is usually of interest. As illustrated in figure 84, ECS system weight is not an aircraft-level optimization metric. Nevertheless, system weight is important during system development. Therefore, the given objectives can be understood as a trade-off between aircraft and system-level criteria.

Formally, this is a Mixed Integer Nonlinear Programming problem (MINLP). Furthermore, it is typically a multi-objective problem, for which the entire Pareto Front [146] has to be established. Then, in industry, management has to agree to a compromise between the top-level objectives (i.e., a point on the Pareto Front). Finally, the problem is expected to exhibit multi-modal behavior (i.e., be non-convex). The given three properties of the problem, together with the high number of design variables (approximately ten to 20 architecture variables plus two to five control variables for ten to 15 mission segments, that is, 30 to 95 design variables) yield an optimization problem, which is prohibitively expensive to evaluate using state of the art algorithms.

13.4 Implementation

Analysis reveals that some of the properties of the problem that make it difficult to solve are either related to the architecture design variables or the control design variables. For example, the set of discrete design variables contains only architecture design variables. Furthermore, the criteria that are exclusively a function of the control variables are mostly convex (even though local extrema were observed in some cases). Due to these distinct properties and the complexity of the problem no direct solution of the problem was attempted. Instead, it is suggested to re-factor the problem into two parts: First, the architecture optimization problem itself, and second, an optimal open-loop control problem of the architecture for different operating conditions. It is possible to do so, because the trade-off between ECS energy efficiency and weight, which shall be balanced by the architecture optimizer for best possible aircraft energy efficiency, is not affected by the particular choice of the operating point of the ECS in the open-loop control problem. This is directly visible in figure 84. The architecture design variables cover all factors sizing the components of the ECS (e.g., heat exchanger dimensions). Therefore, as soon as a vector of design variables has been suggested by the architecture optimizer, the value on the abscissa is already fixed (ECS weight). The open-loop control problem does not need to know about the multi-objective nature of the optimization problem. It may simply search for operating point with maximum aircraft-level energy efficiency (ordinate value). As metric for aircraft-level energy efficiency in a specific mission segment the author used specific fuel consumption (a typical aircraft-level energy efficiency objective is mission block fuel, which is minimized when minimizing the specific fuel consumption of the mission segments).

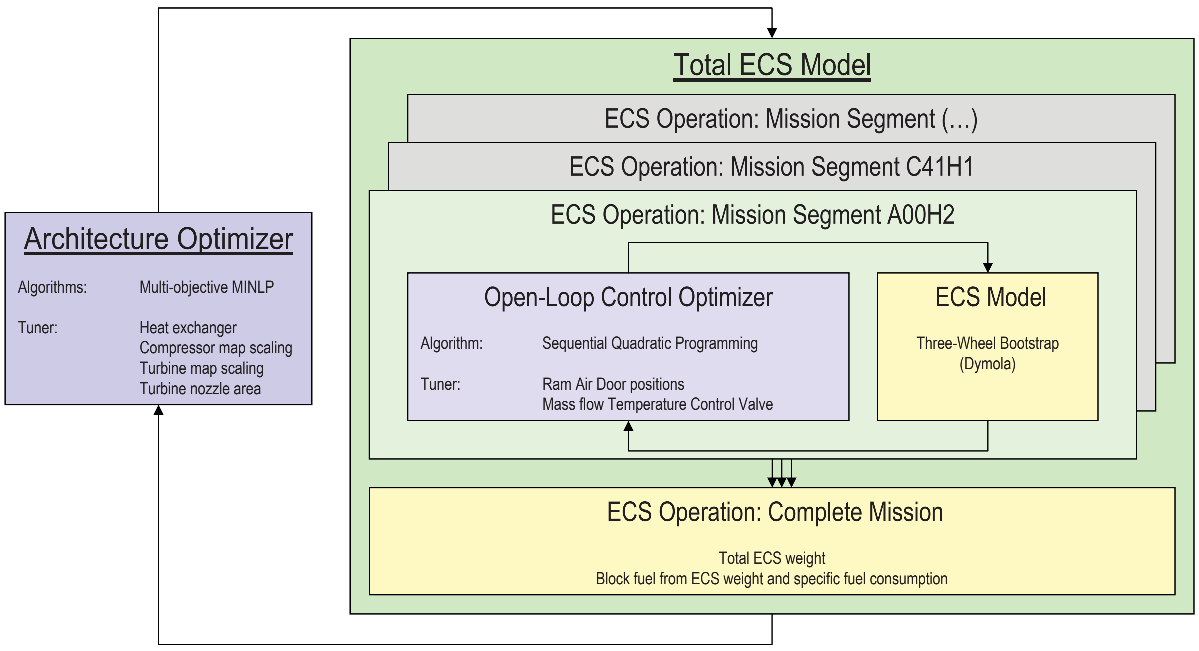

Total ECS Model
ECS Operation: Mission Segment (...)
ECS Operation: Mission Segment C41H1
ECS Operation: Mission Segment A00H2
Open-Loop Control Optimizer
Algorithm: Sequential Quadratic Programming
Tuner: Ram Air Door positions
Mass flow Temperature Control Valve
ECS Model
Three-Wheel Bootstrap (Dymola)
ECS Operation: Complete Mission
Total ECS weight
Block fuel from ECS weight and specific fuel consumption
Architecture Optimizer
Algorithms: Multi-objective MINLP
Tuner: Heat exchanger
Compressor map scaling
Turbine map scaling
Turbine nozzle area

Figure 85: Bi-level approach to ECS optimization

The suggested procedure is as follows: The optimal open-loop control aims at finding the lowest specific fuel consumption for each segment of a predefined flight mission by varying the actuators of the ECS (in figure 85, this is the inner loop of the open-loop control optimizer and the ECS model). This problem has to be solved for all segments of the mission. Afterward, the results are used to establish the mission block fuel based on the ECS weight and the specific fuel consumptions (in figure 85, this is the mission evaluation model on the lower right). Then, the complete ECS architecture model ("Total ECS Model" in the figure) is run by the architecture optimizer. This algorithm varies the architecture design variables to search for the Pareto Front of the given objectives in general and the global minimum of block fuel consumption in particular. This process is repeated until convergence.

The problem was implemented utilizing the Multi-Objective Parameter Synthesis tool [93] (MOPS). The software provides several key features such as state of the art optimization algorithms and transparent parallelization for High Performance Computing clusters. The open-loop control problem is implemented as a normal multi-case model in MOPS. The mission assessment and computation of mission block fuel is realized as a final model. Finally, the architecture optimization is implemented as a normal model.

13.4.1 Optimal open-loop control problem

In order to discuss the implementation in some detail, the conventional Three Wheel Bootstrap ECS shown in figure 79 is taken as example for the remainder of this section. The plant model is called semi-inverse, as the pack discharge temperature is calculated as in a direct model and the pack discharge pressure is prescribed as in an inverse model.

Following the reasoning outlined in the last section, the generic open-loop control problem solves for minimum specific fuel consumption (SFC). It is solved for all regular mission segments. For the Three Wheel Bootstrap ECS, the architecture parameters (e.g., the geometry, the compressor stages of the HP and IP bleed ports) have to be known. These data are furnished by the architecture optimization discussed below. Additionally, several parameters describing the ambient conditions have to be provided. Furthermore, the aircraft altitude and the demanded pack discharge pressure have to be provided (the latter for example may be established from the models of the mix manifold, air distribution network, and cabin or be prescribed via tables calculated separately). Therefore, the optimization problem is as follows.

Minimize	Specific fuel consumption
Given	Architecture parameters
	EBAS stages
	Ambient conditions
	$T_{dis,demanded}$
Subject To	$T_{cmp2} \leq T_{cmp2,max}$
	$x_{dis} \leq x_{dis,max}$
	$T_{WE} \geq T_{WE,min}$
And	$T_{dis} = T_{dis,demanded}$
By Varying	Ram air channel actuator
	Temperature Control Valve (TCV)

Here, $T_{dis,demanded}$ is the demanded pack discharge temperature. T_{cmp2} and $T_{cmp2,max}$ are the actual and the maximum compressor outlet temperatures (these constraints are due to material properties of aluminum alloys). Similarly, x_{dis} and $x_{dis,max}$ are the actual and the maximum water content at pack discharge. Variables T_{WE} and $T_{WE,min}$ are the actual and the minimum permissible temperature in the water extractor. Finally, T_{dis} is the actual pack discharge temperature.

During the architecture optimization of the Three Wheel Bootstrap architecture, this generic open-loop control problem is solved for most regular cases. Alternatively, two other set-ups are used to establish control parameters. One of them is solved for all failure cases. These are the cases with only one engine operational. It is considered important that the ECS is able to provide the required performance in these cases, but the specific fuel consumption is not relevant. Consequently, in this open-loop control problem, the objective function is not the specific fuel consumption but the deviation of the actual discharge temperature from the demanded pack discharge temperature. The other set-up is used for top of descent to ensure minimum pressure for cabin pressurization.

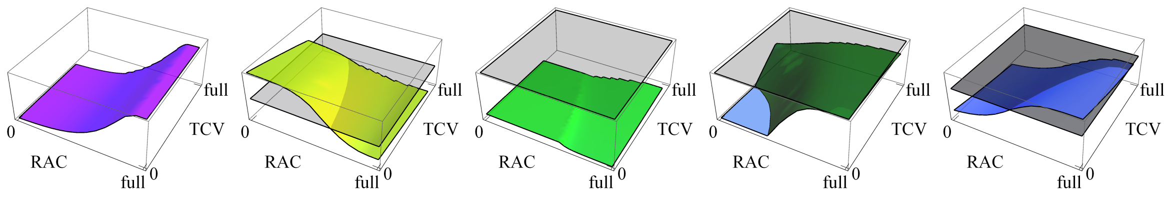


Figure 86: Exemplary topology for C35H2: Criteria on SFC, compressor outlet temperature, water content at pack discharge, water extractor inlet temperature, pack discharge temperature

In order to accelerate convergence, and due to the mostly well-behaved topologies observed for the open-loop control problems, a sequential quadratic programming (SQP) algorithm of [93] is used. In order to deal with the mildly non-convex topologies, a multiple starting point approach is used, which seems to offer a suitable compromise between computational expense and robustness.

In figure 86, an exemplary topology is shown for the optimal open-loop control problem. It refers to a cruise case at 35,000 ft altitude, hot ambient conditions and normal operation. The four graphs to the left show the objective to minimize SFC, the inequality constraint on compressor outlet temperature, water content at pack discharge, and temperature in the water separator are shown. The graph to the right shows the equality constraint on pack discharge temperature. In either case, the objective or, respectively, the normalized constraint value (colored) is shown together with the reference value (gray). The constraint has to maintain a value below of (for inequality constraints) or equal to (for equality constraints) the reference value.

The equality constraint on pack discharge temperature shown to the right of figure 86 effectively restricts the subspace to be considered by the optimizer to a one-dimensional manifold. The inequalities on water content at pack discharge and water separator temperature are not relevant for flight cases due to low ambient humidity in high altitudes. However, the compressor outlet temperature may not grow beyond a specific value, which renders the lower left of the one-dimensional manifold fulfilling the equality constraint infeasible. Comparing the objective to be minimized to the latter, one can infer that the optimum for this particular optimal open-loop control problem is at precisely this location, namely, at medium RAC mass flow and nearly zero mass flow rate through the TCV (upon a further reduction of the TCV flow, the compressor outlet temperature would then grow beyond the prescribed limit).

This behavior exactly mimics the established Three Wheel Bootstrap ECS control logic. In a normal cruise case, the ram air channel doors are closed as much as possible while maintaining the maximum allowed compressor outlet temperature.

13.4.2 Architecture problem

When solving the optimal open-loop control problem for each of the mission segments involved and establishing mission block fuel based on the results, all top-level optimization objectives are available for a given architecture. The outer loop shown in figure 85 contains the complete ECS architecture model, which is driven by the architecture optimization algorithm. This is typically a global, gradient-free algorithm suitable for MINLP optimization problems.

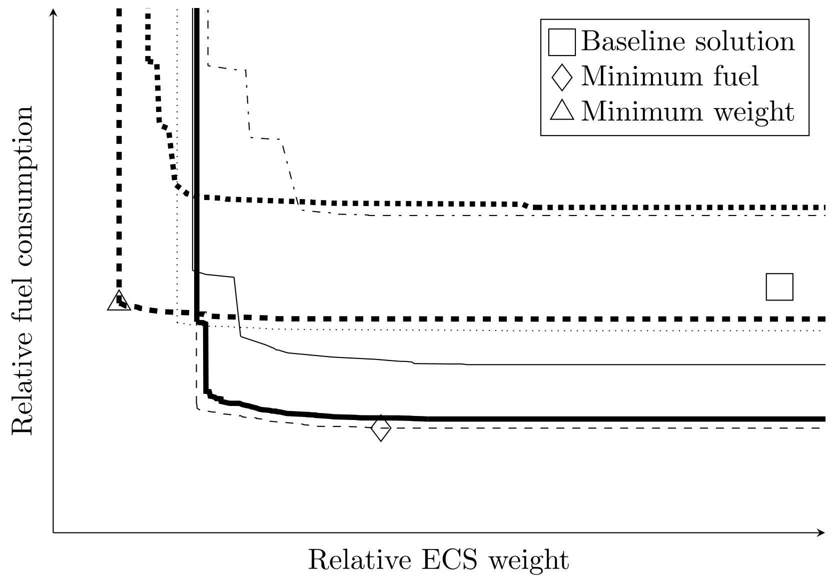


Figure 87: ECS architecture optimization Pareto Fronts with baseline and single-objective optimal solutions

An exemplary result of this ECS architecture optimization is shown in figure 87. This figure illustrates the trade-off between the aircraft-level energy efficiency metric mission block fuel and the system-level metric ECS weight in terms of a Pareto Front. The optimizer iterates until convergence are not shown for clarity of the illustration. Between the two extremes of minimum block fuel consumption (cyan) and minimum ECS weight (magenta), many other compromises are possible. However, in order to improve one metric of the ECS, one has to deteriorate the other (Pareto Optimality).

The typical wall time on a 32 node HPC cluster at DLR Institute of Robotics and Mechatronics is several weeks for a solution to a problem like the one illustrated in figure 87.

With the availability of these results, the local system optimization is complete. Optionally, a global aircraft power system architecture optimization platform could now be used to conduct a re-sizing of the aircraft and thus to provide updated exchange rates.

13.5 Conclusions

Based on physical plant models of ECS, a powerful method for optimization of ECS architectures has been developed. Plant models used during performance assessment of a given ECS architecture are used for optimization of such architecture. The resulting optimization method provides a flexible and comprehensive way of improving ECS architectures based on new technologies. In addition to the optimum parameter sets found by optimization and the resulting Pareto Fronts for the multi-objective optimization problem of low fuel consumption and weight, the tool provides a method for detailed exploration of the design space. This supports better understanding of the sensitivities of design parameters, particularly important for new architectures.

The presented bi-level local system optimization approach includes design variables determining system operation, which ensures comparison of different ECS architectures in their respective operational optimum. Furthermore this supports development of an optimal open-loop control strategy for new systems in early development phases.

CHAPTER 14

CONCLUSIONS

Two substantial contributions were made to mature system-level simulation using equation-based, object-oriented modeling languages further. First, robustness issues were traced to the mathematical formulation of convective transport phenomena, which is posed via the non-causal interface definitions for thermo-fluid dynamics. To establish a rigorous analysis, established interface definitions were presented in a consistent manner and assessed based on formal requirements. Due to deficiencies, which were identified with respect to varying requirements in all interfaces, a more robust and user-friendly interface was proposed and tested. Though superior according to the formal requirements, the interface uses relatively complex semantics and built-in operators, which are specific to thermo-fluid applications.

Second, robustness issues in the initialization of large DAE systems were explained via theoretical convergence properties of typical classes of algorithms and the robustness on selected test cases. For the latter, a quantitative metric was proposed and applied. This metric, the robustness profile, allowed to exhibit the robustness problem in a rigorous manner. However, each resulting profile is always problem-specific and thus a single "snap shot" *for a given test case*, not a general result *of the algorithm only.* The robustness deficiencies were solved on a number of test cases using probability-one homotopy, a method from topology. While the results with respect to robustness are convincing, they required substantial upfront investments. For instance, application-specific coercivity proofs were required, which in turn demand a profound understanding of the underlying physics and probability-one homotopy theory.

In addition, high-speed compressible flow problems were addressed in the framework of equation-based, object-oriented modeling languages using state of the art numerical methods with up to fifth order. These results are not only relevant for pneumatic aircraft energy systems, but for a wide class of applications (including automotive internal combustion engines for instance). Due to the required spatial resolution, computational meshes for complete architecture models tend to become large. Therefore, computation may be slow. However, sparse matrix algorithms in equation-based, object-oriented modeling language compilers allow to mitigate some of the computational expense.

Finally, a substantial contribution to the physics-based design of aircraft energy systems was made. The proposed design environment enables physics-based design of unconventional system architectures and is suitable for industrial problems, both in terms of expressiveness and computational efficiency. It can serve as a building block of a future physics-based aircraft energy systems architecture design methodology and platform. As described in chapter 1, until today, only *simulation* environments were described in literature for such applications.

The design environment and underlying methodology are generally applicable both for explicitly and implicitly sized systems. They address multi-objective problems involving not only one optimization objective but several of them (e.g., fuel burn and weight). They are applicable for design problems requiring the definition of both optimal system design and energy-optimal open-loop control. And they naturally address industrial problems, in which a system has to be optimized for a large number of different operating conditions alike.

However, the tool requires large investments into mathematical plant models, both in terms of engineering hours and skill. It claims substantial computational resources, which limits manageable problem complexity. Additionally, the tool does not address all aspects, which may be relevant for general aircraft energy systems, for instance, the reliability of electric power generation and distribution systems. Finally, aircraft-level assessment metrics are linked manually to external code via exchange rates. For problems, which call for a closer coupling, an automatic interface to a global aircraft optimization platform could be required.

REFERENCES

[1] Air Transport Association, "Specification for manufacturers' technical data," 1999. 1.1

[2] Airbus SAS, "Global market forecast 2010–2029." www.airbus.com/company/market/gmf2010/, accessed August 2011. 1.1

[3] Alexander, J. and Yorke, J., "The homotopy continuation method: numerically implementable topological procedures," *Transactions Of The American Mathematical Society*, vol. 242, pp. 271–284, 1978. 8.1

[4] Allgower, E. and Georg, K., "Continuation and path following," *Acta Numerica*, vol. 2, pp. 1–64, 1992. 8, 8.1, 9.1

[5] Allgower, E. L. and Georg, K., *Introduction to numerical continuation methods.* SIAM Classics in Applied Mathematics, 2003. 7.3, 8, 9.1

[6] Anderson, J., *Computational fluid dynamics: the basics with applications.* McGraw-Hill, 1995. 2.1

[7] Arbeitskreis Masseanalyse, *Luftfahrttechnisches Handbuch.* Industrieanlagen Betriebsgesellschaft, 1990. 1.3.1

[8] Armstrong, M., Garcia, E., and Mavris, D., "Aircraft mission and system failure considerations for functional induction based conceptual architecture design," in *26th International Congress of the Aeronautical sciences (ICAS), Proceedings of the*, (Nice, France), September 2010. 1.3.2

[9] Åström, K., Elmqvist, H., and Mattson, S., "Evolution of continuous-time modeling and simulation," in *Proceedings of the 12th European Simulation Multiconference on Simulation-Past, Present and Future*, pp. 9–18, 1998. 1.5, 1.5.1, 1.5.2

[10] Aubry, M., Baues, H., Halperin, S., and Lemaire, J., *Homotopy theory and models.* Birkhauser, 1995. 8

[11] Baharev, A. and Rév, E., "Reliable computation of equilibrium cascades with affine arithmetic," *AIChE Journal*, vol. 54, pp. 1782–1797, 2008. 7.3.2

[12] Baharev, A. and Rév, E., "A complete nonlinear system solver using affine arithmetic," in *Interval Analysis and Constraint Propagation for Applications (IntCP 2009), Workshop held in conjunction with the 15th International Conference on Principles and Practice of Constraint Programming*, (Lisbon, Portugal), 2009. 7.3.2

[13] Bals, J., Hofer, G., Pfeiffer, A., and Schallert, C., "Object-oriented inverse modelling of multi-domain aircraft equipment systems with modelica," in *Proceedings of the Third International Modelica Conference* (Fritzson, P., ed.), (Linköping, Sweden), 2003. 1.3.2

[14] Bals, J., Hofer, G., Pfeiffer, A., and Schallert, C., "Virtual iron bird–a multidisciplinary modelling and simulation platform for new aircraft system architectures," in *Proceedings of the DGLR Jahrestagung 2005*, (Friedrichshafen, Germany), September 2005. 1.3.2

[15] Bell, B. M. and Burke, J. V., "Algorithmic differentiation of implicit functions and optimal values," in *Advances in Automatic Differentiation* (Bischof, C. H., Bücker, H. M., Hovland, P. D., Naumann, U., and Utke, J., eds.), pp. 67–77, Springer, 2008. 7.3.1

[16] Beltramo, M. N., Trapp, D. L., Kimoto, B. W., and Marsh, D. P., "Parametric study of transport aircraft systems cost and weight," Tech. Rep. NASA-CR-151970, NASA Center for Aerospace Information (CASI), April 1977. 1.3.1

[17] Billups, S. and Watson, L., "A probability-one homotopy algorithm for nonsmooth equations and mixed complementarity problems," *SIAM Journal On Optimization*, vol. 12, no. 3, pp. 606–626, 2002. 10.1.1

[18] Bock, H. G., Eich, E., and Schlöder, J. P., "Numerical solution of constrained least squares boundary value problems in differential-algebraic equations," in *Numerical Treatment of Differential and Integral Equations*, Teubner, 1988. 7.1

[19] Boeing Commercial Airplanes, "Current market outlook 2010–2029." www.boeing.com/commercial/cmo/pdf/Boeing_Current_Market_Outlook_2010 _to_2029.pdf, accessed August 2011. 1.1

[20] Bouaricha, A. and Schnabel, R. B., "Tensolve: a software package for solving systems of nonlinear equations and nonlinear least-squares problems using tensor methods (algorithm 768)," *ACM Transactions On Mathematical Software*, vol. 23, no. 2, pp. 174–195, 1997. 7.3.1.3

[21] Branin, F., "Widely convergent method for finding multiple solutions of simultaneous nonlinear equations," *IBM Journal Of Research And Development*, vol. 16, no. 5, pp. 504–522, 1972. 8.1

[22] Brent, R., *Algorithms for minimization without derivatives.* Prentice-Hall Series in Automatic Computation, 1973. 5.1.2.7

[23] Broman, D., Fritzson, P., and Furic, S., "Types in the Modelica language," in *Proceedings of the Fifth International Modelica Conference*, pp. 303–315, 2006. 4.1

[24] Carl, U., "Aircraft systems III." Lecture Notes, Technical University Hamburg-Harburg, 1995. 77

[25] Carpanzano, E., "Order reduction of general nonlinear DAE systems by automatic tearing," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 6, no. 2, pp. 145–168, 2000. 4.2.1.5, 5.1.2

[26] Casas, W. and Schmitz, G., "Experiences with a gas driven, desiccant assisted air conditioning system with geothermal energy for an office building," *Energy And Buildings*, vol. 37, no. 5, pp. 493–501, 2005. 1.4.1

[27] Casas, W., *Untersuchung und Optimierung sorptionsgestützter Klimatisierungsprozesse.* PhD thesis, Technical University of Hamburg-Harburg, Institute of Thermo-Fluid Dynamics, 2006. 1.4.1

[28] Casella, F., Otter, M., Proelss, K., Richter, C., and Tummescheit, H., "The modelica fluid and media library for modeling of incompressible and compressible thermo-fluid pipe networks," in *Proceedings of the Fifth International Modelica Conference*, pp. 631–640, 2006. 2.6

[29] Casella, F. and Leva, A., "Modelica open library for power plant simulation: design and experimental validation," in *Proceedings of the Third International Modelica Conference* (Fritzson, P., ed.), (Linköping, Sweden), pp. 41–50, 2003. 1.4.1, 2.4, 2.4, 5.1.1, 5.1.2, 1, 5.1.2, 5.1.3, 5.3

[30] Casella, F. and Leva, A., "Modelling of thermo-hydraulic power generation processes using Modelica," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 12, no. 1, pp. 19–33, 2006. 1.4.1, 2.4, 2.4, 5.1.1, 5.1.2, 1, 5.1.2, 5.1.3, 5.3

[31] Cayley, A., "The Newton-Fourier imaginary problem," *American Journal Of Mathematics*, vol. 2, p. 97, 1879. 7.2.1

[32] Cellier, F., *Continuous system modeling*. Springer-Verlag, 1991. 1.4.1

[33] Cellier, F. E. and Kofman, E., *Continuous system simulation*. Springer, 2006. 4.2.1.5, 5.1.2

[34] Chang, Y. and Wang, C., "A generalized heat transfer correlation for louver fin geometry," *International Journal Of Heat And Mass Transfer*, vol. 40, no. 3, pp. 533–544, 1997. 4.1.2

[35] Choi, S. H. and Book, N. L., "Unreachable roots for global homotopy continuation methods," *AIChE Journal*, vol. 37, pp. 1093–1095, July 1991. 8.1

[36] Chow, S.-N., Mallet-Paret, J., and Yorke, J. A., "Finding zeroes of maps: Homotopy methods that are constructive with probability one," *Mathematics Of Computation*, vol. 32, pp. 887–899, July 1978. 8.1, 8.1, 9, 9.1, 9.1.1, 9.1.1, 9.1.2, 9.1.2

[37] Cronin, M. J., "The prospects and potential of all electric aircraft," in *American Institute of Aeronautics and Astronautics, Aircraft Design, Systems and Technology Meeting, Fort Worth, TX*, October 1983. 1.1

[38] Davies, M., *The standard handbook for aeronautical and astronautical engineers*. McGraw-Hill Professional, 2003. 1.1

[39] de Tenorio, C., Mavris, D., Garcia, E., and Armstrong, M., "Methodology for aircraft system architecture sizing," in *26th International Congress of the Aeronautical sciences (ICAS), Proceedings of the*, (Anchorage, AK, USA), September 2008. 1.3.2

[40] Dempsey, M. Personal communication, 2009. 7.3.2

[41] Dennis, J. E. and Schnabel, R. B., *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM Classics in Applied Mathematics, 1996. 7.3

[42] Dent, D., Paprzycki, M., and Kucaba-Pietal, A., "Comparing solvers for large systems of nonlinear algebraic equations," in *Proceedings of the Southern Conference on Computing*, 2000. 7.2.1

[43] Dent, D., Paprzycki, M., and Kucaba-Pietal, A., "Solvers for systems of nonlinear algebraic equations - Their sensitivity to starting vectors," in *Numerical Analysis and Its Applications* (Vulkov, L., Yalamov, P., and Wasniewski, J., eds.), vol. 1988 of *Lecture Notes in Computer Science*, pp. 230–237, Springer Verlag, 2001. 7.2.1

[44] Deuflhard, P., Fiedler, B., and Kunkel, P., "Efficient numerical path following beyond critical points," *SIAM Journal On Numerical Analysis*, vol. 24, no. 4, pp. 912–927, 1987. 8.2

[45] Deuflhard, P., *Newton Methods for Nonlinear Problems. Affine Invariance and Adaptive Algorithms*. Springer Verlag, 2004. 7.3, 7.3.1.2

[46] Dimo, P., *Nodal analysis of power systems*. Taylor & Francis, 1975. 10.2.1, 10.2.1, 10.2.2

[47] Ding, G., "Recent developments in simulation techniques for vapour-compression refrigeration systems," *International Journal of Refrigeration*, vol. 30, no. 7, pp. 1119–1133, 2007. 1.4.1

[48] Dolan, E. D. and Moré, J. J., "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, pp. 201–213, 2002. 7.2.1

[49] Dollmayer, J. and Carl, U., "Consideration of fuel consumption caused by aircraft systems in aircraft design," in *25th International Congress of the Aeronautical sciences (ICAS), Proceedings of the*, (Hamburg, Germany), September 2006. 1.3.2

[50] Dollmayer, J. and Carl, U., "Einfluss des Leistungsbedarfs von Flugzeugsystemen auf den Kraftstoffverbrauch," in *Deutscher Luft- und Raumfahrtkongress 2004, Proceedings of the*, (Dresden, Germany), September 2004. 1.3.2

[51] Duffin, R., "Nonlinear networks IIa," *Bulletin Of The American Mathematical Society*, vol. 53, pp. 963–971, 1947. 10.2.2

[52] Eaves, B., "Homotopies for computation of fixed points," *Mathematical Programming*, vol. 3, no. 1, pp. 1–22, 1972. 9.1

[53] Eborn, J., *On Model Libraries for Thermo-hydraulic Applications*. PhD thesis, Lund University, Department of Automatic Control, 2001. 2.5

[54] Elmqvist, H., Brück, D., and Otter, M., "Dymola users' manual." Dynasim AB, Research Park Ideon, Lund, Sweden, 1995. 7.3.1, 7.3.1.4, 7.3.2

[55] Elmqvist, H., *A Structured Model Language for Large Continuous Systems*. PhD thesis, Lund University, Department of Automatic Control, Sweden, May 1978. 10.1.2

[56] Elmqvist, H., Casella, F., Otter, M., and Mattsson, S. E., "New Modelica_Fluid principles." Modelica White Paper, January 9th 2008. 4.1.1.2

[57] Elmqvist, H. and Otter, M., "Methods for tearing systems of equations in object-oriented modeling," in *Proceedings of the European Simulation Multiconference*, (Barcelona, Spain), pp. 326–332, June 1994. 4.2.1.5, 5.1.2, 7.3.1, 7.3.2

[58] Elmqvist, H., Tummescheit, H., and Otter, M., "Object-oriented modeling of thermo-fluid systems," in *Proceedings of the Third International Modelica Conference* (Fritzson, P., ed.), (Linköping, Sweden), pp. 269–286, 2003. 2.5, 2.6, 5.3, 12.3, 12.3.2

[59] Federal Aviation Administration, "Federal aviation regulations, part 25, airworthiness standards: Transport category airplanes," 2002. 1.3.2

[60] Ferziger, J. and Perić, M., *Computational methods for fluid dynamics*, vol. 2. Springer Berlin, 1999. 2, 2.2.2

[61] Figliola, R., Tipton, R., and Li, H., "Exergy approach to decision-based design of integrated aircraft thermal systems," *AIAA Journal Of Aircraft*, vol. 40, no. 1, pp. 49–55, 2003. 1.3.2

[62] Fisher, M., Gould, F., and Tolle, J., "A simplicial approximation algorithm for solving systems of nonlinear equations," in *Proceedings of the conference on mathematical programming and its applications, Symposia Mathematica*, vol. 19, pp. 73–90, Academic Press, New York, 1976. 8.1

[63] Franke, R., Casella, F., Otter, M., Sielemann, M., Mattson, S.-E., Olsson, H., and Elmqvist, H., "Stream connectors—an extension of Modelica for device-oriented modeling of convective transport phenomena," in *Proceedings of the seventh International Modelica conference* (Casella, F., ed.), (Como), pp. 108–121, September 2009. 5.1.1, 2, 5.1.2.1, 5.1.3, 5.1.3, 5.3

[64] Franke, R., Tummescheit, H., Casella, F., and Sielemann, M., "Proposal for extension of Modelica for fluid modeling." Modelica White Paper, April 9th 2008. 5.1.1, 5.1.3, 5.3

[65] Fung, E., "Homotopy methods for DC operating point and periodic steady-state analysis: Spice3 implementation," Master's thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 1998. 8.2

[66] Galllouët, T., Hérard, J., and Seguin, N., "Some recent finite volume schemes to compute euler equations using real gas eos," *International Journal For Numerical Methods In Fluids*, vol. 39, no. 12, pp. 1073–1138, 2002. 12.3.1

[67] Garcia, C. and Gould, F., "Relations between several path following algorithms and local and global newton methods," *SIAM Review*, vol. 22, no. 3, pp. 263–274, 1980. 8.1

[68] Garcia, C. and Zangwill, W., "An approach to homotopy and degree theory," *Mathematics Of Operations Research*, vol. 4, no. 4, pp. 390–405, 1979. 8.1

[69] Ge, Y., Watson, L., Collins Jr, E., and Bernstein, D., "Probability-one homotopy algorithms for full-and reduced-order h_2/h_∞ controller synthesis," *Optimal Control Applications and Methods*, vol. 17, no. 3, pp. 187–208, 1996. 8.1, 10.1.1

[70] Getreu, I. E., *Modeling the Bipolar Transistor*. Elsevier Science, 1978. 8.1

[71] Godunov, S. K., "A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics," *Matematicheskii Sbornik*, vol. 47, pp. 357–393, 1959. 12.2.1.3, 12.2.2

[72] Grace, A., "SIMULAB, an integrated environment for simulation and control," in *International Conference on Control, Proceedings of the*, pp. 466–470, IEEE, March 1991. 1.5.1

[73] Green, M. and Melville, R., "Sufficient conditions for finding multiple operating points of dc circuits using continuation methods," in *IEEE International Symposium on Circuits and Systems*, (Seattle), pp. 117–120, 1995. 9, 10.2

[74] Griffin, J. D. and Kolda, T. G., "Nonlinearly-constrained optimization using asynchronous parallel generating set search," Tech. Rep. SAND2007-3257, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, May 2007. Submitted for publication. 7.2.1

[75] Gungor, K. and Winterton, R., "Simplified general correlation for saturated flow boiling and comparisons of correlations with data," *Chemical Engineering Research and Design*, vol. 65, pp. 148–156, 1987. 4.1.2

[76] Gustafson, J. and Beris, A., "Evaluating all real roots of nonlinear equations using a global fixed-point homotopy method," *AIChE Journal*, vol. 37, no. 11, pp. 1749–1752, 1991. 8.1

[77] Güttinger, T. E., Dorn, C., and Morari, M., "Experimental study of multiple steady states in homogeneous azeotropic distillation," *Industrial & Engineering Chemistry Research*, vol. 36, pp. 794–802, 1997. 7.3.2

[78] Harlow, F. and Welch, J., "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface," *Physics of Fluids*, vol. 8, pp. 2182–2189, 1965. 2.2.3

[79] Harten, A., "High resolution schemes for hyperbolic conservation laws," *Journal Of Computational Physics*, vol. 49, pp. 357–393, 1983. 12.2.2, 12.2.2

[80] Harten, A., Engquist, B., Osher, S., and Chakravarthy, S., "Uniformly high order essentially non-oscillatory schemes, III," *Journal Of Computational Physics*, vol. 71, pp. 231–303, 1987. 12.2.3.1

[81] Harten, A., Lax, P. D., and van Leer, B., "On upstream differencing and Godunov-type schemes for hyperbolic conservation law," *SIAM Review*, vol. 25, no. 1, pp. 35–61, 1983. 12.2.1.1

[82] Heroux, M. A., Bartlett, R. A., Howle, V. E., Hoekstra, R. J., Hu, J. J., Kolda, T. G., Lehoucq, R. B., Long, K. R., Pawlowski, R. P., Phipps, E. T., Salinger, A. G., Thornquist, H. K., Tuminaro, R. S., Willenbring, J. M., Williams, A., and Stanley, K. S., "An overview of the Trilinos project," *ACM Transactions On Mathematical Software*, vol. 31, no. 3, pp. 397–423, 2005. 10.1.3

[83] Honkala, M., Roos, J., and Karanko, V., "On nonlinear iteration methods for DC analysis of industrial circuits," in *Progress in Industrial Mathematics at ECMI 2004* (Bock, H.-G., Hoog, F., Friedman, A., Gupta, A., Neunzert, H., Pulleyblank, W. R., Rusten, T., Santosa, F., Tornberg, A.-K., Capasso, V., Mattheij, R., Neunzert, H., Scherzer, O., Bucchianico, A., Mattheij, R., and Peletier, M., eds.), vol. 8 of *Mathematics in Industry*, pp. 144–148, Springer Berlin Heidelberg, 2006. 7.2.1

[84] Hou, T. Y. and LeFloch, P., "Why non-conservative schemes converge to the wrong solutions: Error analysis.," *Mathematics Of Computation*, vol. 62, pp. 497–530, 1994. 12.2

[85] IEEE Computer Society, "IEEE standard VHDL analog and mixed-signal extensions." IEEE 1076.1-1, 1999. 1.5.2

[86] Inoue, Y., "A practical algorithm for DC operating-point analysis of large-scale circuits," *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, vol. 77, no. 10, pp. 49–62, 1994. 9, 10.1.1, 10.2

[87] Issa, R. I., "Solution of the implicitly discretised fluid flow equations by operator-splitting," *Journal Of Computational Physics*, vol. 62, no. 1, pp. 40–65, 1986. 7.1

[88] Jackson, S., *Systems engineering for commercial aircraft.* Ashgate, 1997. 1.1

[89] Jensen, J. M., *Dynamic Modeling of Thermo-Fluid Systems with focus on evaporators for refrigeration.* PhD thesis, Technical University of Denmark, Department of Mechanical Engineering, 2003. 1.4.1

[90] Jensen, J., Jensen, J., and Tummescheit, H., "Moving boundary models for dynamic simulations of two-phase flows," in *Proceedings of the Second International Modelica Conference*, 2002. 1.4.1

[91] Jiang, G. and Shu, C.-W., "Effcient implementation of weighted ENO schemes," *Journal Of Computational Physics*, vol. 126, pp. 202–228, 1996. 12.2.3.2

[92] Joint Aviation Authorities, "Joint aviation regulations, part 25, airworthiness standards: Transport category airplanes," 2011. 1.3.2

[93] Joos, H.-D., Bals, J., Looye, G., Schnepper, K., and Varga, A., "A multi-objective optimisation-based software environment for control systems design," in *Proceedings of the IEEE CADCS Symposium*, (Glasgow), pp. 7–14, 2002. 13.4, 13.4.1

[94] Kaslusky, S., Sabatino, D., and Zeidner, L., "ITAPS: A process and toolset to support aircraft level system integration studies," in *45th AIAA Aerospace Science Meeting and Exhibit, Proceedings of the*, (Reno, NV, USA), January 2007. 1.3.2

[95] Keller, H., "Global homotopies and Newton methods," in *Recent Advances in Numerical Analysis* (de Boor, C. and Golub, G., eds.), (New York), pp. 73–94, Academic Press, 1978. 8.1

[96] Kelley, C. T., *Solving nonlinear equations with Newton's method.* SIAM Classics in Applied Mathematics, 2003. 7.3

[97] Kirby, M., *A methodology for technology identification, evaluation, and selection in conceptual and preliminary aircraft design.* PhD thesis, Georgia Institute of Technology, Aerospace Systems Design Laboratory, 2001. 1.3.1

[98] Kloas, M., Friesen, V., and Simons, M., "Smile — A simulation environment for energy systems," *Systems Analysis Modelling Simulation*, vol. 18, pp. 503–506, 1995. 1.5.2

[99] Koeppen, C., *Methodik zur modellbasierten Prognose von Flugzeugsystemparametern im Vorentwurf von Verkehrsflugzeugen.* PhD thesis, Technical University of Hamburg-Harburg, Institute for Aircraft Systems Engineering, 2005. 1.3.1

[100] Kuno, M. and Seader, J., "Computing all real solutions to systems of nonlinear equations with a global fixed-point homotopy," *Industrial & Engineering Chemistry Research*, vol. 27, no. 7, pp. 1320–1329, 1988. 8.1

[101] Lax, P. D. and Wendroff, B., "Systems of conservation laws," *Communications on Pure and Applied Mathematics*, vol. 13, pp. 217–237, 1960. 12.2

[102] Lehle, W., "Konzeption und Entwicklung von Flugzeugklimaanlagen." Vortrag, DGLR-Bezirksgruppe Hamburg, Germany, June 2006. 1.1

[103] Linnett, K. and Crabtree, R., "What's next in commercial aircraft environmental control systems?," in *Proceedings of the International Conference On Environmental Systems*, SAE technical paper 932057, July 1993. 78

[104] Liou, M., Leer, B., and Shuen, J., "Splitting of inviscid fluxes for real gases," *Journal Of Computational Physics*, vol. 87, no. 1, pp. 1–24, 1990. 12.3.1

[105] Liscouët-Hanke, S., Pufe, S., and Maré, J., "A simulation framework for aircraft power management," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 222, no. 6, pp. 749–756, 2008. 1.3.2

[106] Liscouët-Hanke, S., *A model-based methodology for integrated preliminary sizing and analysis of aircraft power system architectures.* PhD thesis, University of Toulouse (INSA), Mechanical Engineering Laboratory of Toulouse, 2008. 1.3.2

[107] Lohn, C., "Entwicklung eines Verfahrens zur Abschätzung der Systemmassen für das Passagierflugzeug A3XX." Diplomarbeit, Lehrstuhl und Institut für Flugzeugbau und Leichtbau der Technischen Universität Braunschweig, 1998. 1.3.1

[108] Lucia, A. and Feng, Y., "Multivariable terrain methods," *AIChE Journal*, vol. 49, no. 10, pp. 2553–2563, 2003. 8

[109] Luksan, L. and Vleck, J., "Sparse and partially separable test problems for unconstrained and equality constrained optimization.," Tech. Rep. 767, Institute of Computer Science, Academy of Sciences of the Czech Republic, 1999. 7.3.2

[110] Malinen, I. and Tanskanen, J., "Modified bounded homotopies to enable a narrow bounding zone," *Chemical Engineering Science*, vol. 63, no. 13, pp. 3419–3430, 2008. 8.1, 8.2

[111] Mathis, W., Trajkovic, L., Koch, M., and Feldmann, U., "Parameter embedding methods for finding DC operating points of transistor circuits," in *Third international specialist workshop on Nonlinear Dynamics of Electronic Systems, NDES 1995*, (Dublin, Ireland), pp. 147–150, July 1995. 9, 10.1.1, 10.2

[112] Mattsson, S. E. and Andersson, M., "Omola — An object-oriented modelling language.," in *Recent Advances in Computer-Aided Control Systems Engineering, Studies in Automation and Control* (Jamshidi, M. and Herget, C. J., eds.), pp. 291–310, Elsevier Science Publishers, 1992. 1.5.2

[113] Mattsson, S., Elmqvist, H., and Otter, M., "Physical system modeling with Modelica," *Control Engineering Practice*, vol. 6, no. 4, pp. 501–510, 1998. 1.5.2

[114] Mavris, D., de Tenorio, C., and Armstrong, M., "Methodology for aircraft system architecture definition," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics, 2008. 1.3.2

[115] Melville, R., Moinian, S., Feldmann, P., and Watson, L., "Sframe: An efficient system for detailed DC simulation of bipolar analog integrated circuits using continuation methods," *Analog Integrated Circuits And Signal Processing*, vol. 3, no. 3, pp. 163–180, 1993. 9, 10.2

[116] Melville, R. C., Trajkovic, L., Fang, S.-C., and Watson, L. T., "Globally convergent homotopy methods for the DC operating point problem," Tech. Rep. 90-61, Virginia Polytechnic Institute and State University, 1990. 55

[117] Melville, R. C., Trajkovic, L., Fang, S.-C., and Watson, L. T., "Artificial parameter homotopy methods for the DC operating point problem," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, pp. 861–877, June 1993. 7.3.2, 8.1, 8.1, 8.2, 9, 9.2.1, 9.2.1.2, 9.2.1.2, 9.2.1.2, 9.2.1.2, 9.2.1.3, 9.2.1.3, 9.2.1.3, 9.2.1.3, 9.2.3, 9.2.3.1, 9.2.3.3, 10, 10.1.1, 10.2, 10.2.1, 10.2.1, 10.2.1, 10.2.2, 10.2.2, 10.2.2, 10.2.3

[118] Merlet, J.-P., "The Coprin examples page." http://www-sop.inria.fr/coprin/logiciels/ALIAS/Benches/benches.html, 2006. 7.3.2, 7.3.2

[119] Metropolis, N. and Ulam, S., "The Monte Carlo method," *Journal of The American Statistical Association*, vol. 44, pp. 335–341, 1949. 7.2.2

[120] Mitchell, J. C., *Concepts in Programming Languages.* Cambridge University Press, 2003. 4.1

[121] Modelica Association, "Modelica and the Modelica Association." www.modelica.org, accessed March 2010. 1.5.2

[122] Modelica Association, "Minutes of the 55th design meeting, fluid group," January 2008. 5.1.1, 5.3

[123] Modelica Association, "Modelica — a unified object-oriented language for physical systems modeling, language specification 3.2," March 2010. 2.5

[124] Modelica Fluid Task Force, "Design concepts for the Modelica.Fluid library." Modelica White Paper, March 18th 2008. 5.3

[125] Moir, I. and Seabridge, A., *Aircraft Systems: Mechanical, electrical, and avionics subsystems integration.* John Wiley & Sons Inc, 2008. 1.1

[126] Moré, J. J., Garbow, B. S., and Hillstrom, K. E., "User guide for MINPACK-1," Tech. Rep. ANL-80-74, Argonne National Laboratory, 1980. 7.3.1.1

[127] Moré, J. J. and Wild, S. M., "Benchmarking derivative-free optimization algorithms," *SIAM Journal On Optimization*, vol. 20, no. 1, pp. 172–191, 2009. 7.2.1

[128] Moretti, G. and Bleich, G., "A time-dependent computational method for blunt-body flows," *AIAA Journal*, vol. 4, pp. 2136–2141, 1966. 12.2

[129] MSC Software, "EASY5 2008 user guide," March 2008. 1.5.1

[130] Nagel, L., "Spice2: A computer program to simulate semiconductor circuits," tech. rep., University of California, Berkeley, 1975. 8.1

[131] Neumaier, A., *Interval methods for systems of equations.* Cambridge University Press, 1990. 7.3, 8

[132] Nowak, U. and Weimann, L., "A family of Newton codes for systems of highly nonlinear equations," Tech. Rep. TR-91-10, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1991. 7.3.1.2

[133] Oh, M. and Pantelides, C. C., "A modelling and simulation language for combined lumped and distributed parameter systems," *Computers & Chemical Engineering*, vol. 20, no. 6-7, pp. 611–633, 1996. 1.5.2

[134] Ordonez, J. and Bejan, A., "Minimum power requirement for environmental control of aircraft," *Energy*, vol. 28, no. 12, pp. 1183–1202, 2003. 1.3.2

[135] Ortega, J. M. and Rheinboldt, W. C., *Iterative Solution of Nonlinear Equations in Several Variables.* SIAM Classics in Applied Mathematics, 1970. 7.3

[136] Osborne Jr., B. P., "Technical proposal for subsystems integrated design assessment technology (SIDAT)," Tech. Rep. SOL PRDA 95-05-FIK, Lockheed Aeronautical Systems Company, March 1995. 1.3.1

[137] Paloschi, J., "Bounded homotopies to solve systems of algebraic nonlinear equations," *Computers & Chemical Engineering*, vol. 19, no. 12, pp. 1243–1254, 1995. 8.1, 8.2

[138] Paloschi, J., "Bounded homotopies to solve systems of sparse algebraic nonlinear equations," *Computers & Chemical Engineering*, vol. 21, no. 5, pp. 531–541, 1996. 8.1, 8.2

[139] Patankar, S. and Spalding, D., "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows," *International Journal Of Heat And Mass Transfer*, vol. 15, pp. 1787–1806, 1972. 2, 2.2.2, 7.1

[140] Percell, P., "Note on a global homotopy," *Numerical Functional Analysis And Optimization*, vol. 2, no. 1, pp. 99–106, 1980. 8.1

[141] Pfafferott, T., *Dynamische Simulation von CO2-Kälteprozessen für mobile Anwendungen.* PhD thesis, Technical University of Hamburg-Harburg, Institute of Thermo-Fluid Dynamics, 2005. 1.4.1, 4.1.2

[142] Prölss, K. and Schmitz, G., "Modeling of frost growth on heat exchanger surfaces," in *Proceedings of the Fifth International Modelica Conference*, 2006. 1.4.1

[143] Prölss, K., *Untersuchung von Energie- und Massespeicherungsvorgängen in Pkw-Kälteanlagen.* PhD thesis, Technical University of Hamburg-Harburg, Institute of Thermo-Fluid Dynamics, 2009. 1.4.1

[144] Provost, M., "The more electric aero-engine: a general overview from an engine manufacturer," in *Power Electronics, Machines and Drives, 2002. International Conference on*, no. 487, pp. 246–251, IEEE, 2002. 1.1

[145] Quirk, J. J., "An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two dimensional bodies," *Computers and Fluids*, vol. 23, no. 1, pp. 125–142, 1994. 12.2.2.1

[146] Rao, S., *Engineering optimization: theory and practice.* Wiley, 2009. 13.3

[147] Richter, C. C., *Proposal of New Object-Oriented Equation-Based Model Libraries for Thermodynamic Systems.* PhD thesis, Technical University Braunschweig, Institute for Thermodynamics, 2008. 1.4.1

[148] Richtmyer, R. D. and Morton, K. W., *Difference Methods for Initial Value Problems.* Interscience-Wiley, New York, 1967. 12.2.1.3

[149] Roe, P. L., "Approximate Riemann solvers, parameter vectors, and difference schemes," *Journal Of Computational Physics*, vol. 43, pp. 357–372, 1981. 12.2.1.1, 12.2.1.1

[150] Roskam, J., *Airplane Design: Component Weight Estimation.* Roskam Aviation and Engineering Corp., 1985. 1.3.1

[151] Roychowdhury, J. S. and Melville, R. C., "Homotopy techniques for obtaining a DC solution of large-scale mos circuits," in *Proceedings of the 33rd Design Automation Conference*, pp. 286–291, 1996. 9, 10.2, 10.2.4, 10.2.4, 10.2.4

[152] Roychowdhury, J. and Melville, R., "Delivering global DC convergence for large mixed-signal circuits via homotopy/continuation methods," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 66–78, January 2006. 8.2, 9, 9.2.3.3, 10.1.1, 10.1.1, 10.2, 10.2.4, 10.2.4, 10.2.4, 10.2.4

[153] Rusanov, V. V., "Calculation of interaction of non-steady shock waves with obstacles," *USSR Computational Mathematics and Mathematical Physics*, vol. 1, pp. 267–279, 1961. 12.2.1.3

[154] Saigal, R., "On the convergence rate of algorithms for solving equations that are based on methods of complementary pivoting," *Mathematics Of Operations Research*, pp. 108–124, 1977. 9.1

[155] Scarf, H., "The approximation of fixed points of a continuous mapping," *SIAM Journal On Applied Mathematics*, vol. 15, no. 5, pp. 1328–1343, 1967. 9.1

[156] Schallert, C., Pfeiffer, A., and Bals, J., "Generator power optimisation for a more-electric aircraft by use of a virtual iron bird," in *25th International Congress of the Aeronautical sciences (ICAS), Proceedings of the*, (Hamburg, Germany), September 2006. 1.3.2

[157] Schallert, C., "A novel tool for the conceptual design of aircraft on-board power systems," in *AeroTech Congress and Exhibition, Proceedings of the*, (Los Angeles, CA, USA), September 2007. 1.3.2

[158] Schallert, C., "Incorporation of reliability analysis methods with Modelica," in *Sixth International Modelica conference, Proceedings of the* (Bachmann, B., ed.), (Bielefeld, Germany), March 2008. 1.3.2

[159] Scholz, D., "Computer aided engineering for the design of flight control and hydraulic systems," *SAE Transactions*, vol. 105, no. 1, pp. 203–212, 1996. 1.3.2

[160] Shacham, M., Brauner, N., and Pozin, M., "Comparing software for interactive solution of systems of nonlinear algebraic equations," *Computers & Chemical Engineering*, vol. 22, 1998. 7.2.1

[161] Shah, M., "A general correlation for heat transfer during film condensation inside pipes," *International Journal Of Heat and Mass Transfer*, vol. 22, pp. 547–556, 1979. 4.1.2

[162] Shu, C.-W., "Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws," *Advanced numerical approximation of nonlinear hyperbolic equations*, vol. 1697, pp. 325–432, 1998. 1.4.2, 12.2.1, 12.2.3, 12.2.3.2, 12.2.3.3

[163] Sielemann, M., Casella, F., Otter, M., Clauss, C., Eborn, J., Mattsson, S., and Olsson, H., "Robust initialization of differential-algebraic equations using homotopy," in *Proceedings of Eighth International Modelica Conference*, (Dresden, Germany), March 2011. 1

[164] Smale, S., "A convergent process of price adjustment and global newton methods," *Journal Of Mathematical Economics*, vol. 3, no. 2, pp. 107–120, 1976. 8.1

[165] Sommese, A., Wampler, C., and Wampler, C., *The Numerical solution of systems of polynomials arising in engineering and science.* World Scientific Pub Co Inc, 2005. 9.1.4

[166] Steger, J. L. and Warming, R. F., "Flux vector splitting of the inviscid gasdynamic equations with applications to finite difference methods," *Journal Of Computational Physics*, vol. 40, pp. 263–293, 1981. 12.2.1.2, 12.2.1.2

[167] Strang, G., "On the construction and comparison of difference schemes," *SIAM Journal On Numerical Analysis*, vol. 5, no. 3, pp. 506–517, 1968. 12.3.4

[168] Sweby, P. K., "High resolution schemes using flux limiters for hyperbolic conservation laws," *SIAM Journal On Numerical Analysis*, vol. 21, pp. 995–1011, 1984. 12.2.2

[169] Tarjan, R., "Depth-first search and linear graph algorithms," *SIAM Journal On Computing*, vol. 1, pp. 146–160, 1972. 10.1.2

[170] Tillner-Roth, R. and Baehr, H., "An international standard formulation for the thermodynamic properties of 1, 1, 1, 2-tetrafluoroethane (HFC-134a) for temperatures from 170 K to 455 K and pressures up to 70 MPa," *Journal of Physical and Chemical Reference Data*, vol. 23, p. 657, 1994. 4.1.2

[171] Tipton, R., Figliola, R., and Ochterbeck, J., "Thermal optimization of the ECS on an advanced aircraft with an emphasis on system efficiency and design methodology," in *SAE Aerospace Systems Conference, Proceedings of the*, May 1997. 1.3.2

[172] Torenbeek, E., *Synthesis of subsonic airplane design.* Delft University Press, 1976. 1.1, 1.3.1

[173] Toro, E. F., "On two Glimm-related schemes for hyperbolic conservation laws," in *Proceedings of the Fifth Annual Conference of the CFD Society of Canada*, pp. 3.49–3.54, University of Victoria, Canada, 1997. 12.2.1.3

[174] Toro, E. F., *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction.* Springer, 1997. 1.4.2, 12.1, 12.1, 12.2, 12.2.1, 12.2.1.1, 12.2.1.1, 12.2.1.2, 12.2.1.2, 12.2.1.3, 12.2.2.2, 12.3.1, 12.3.3, 12.3.6

[175] Trajkovic, L., Melville, R. C., and Fang, S.-C., "Finding DC operating points of transistor circuits using homotopy methods," in *Proc. IEEE Int Circuits and Systems Sympoisum*, pp. 758–761, 1991. 9, 10.2

[176] Trajkovic, L., Melville, R. C., and Fang, S.-C., "Improving DC convergence in a circuit simulator using a homotopy method," in *Proc. Custom Integrated Circuits Conf. the IEEE 1991*, 1991. 9, 10.2

[177] Trajkovic, L., Melville, R., and Fang, S.-C., "Passivity and no-gain properties establish global convergence of a homotopy method for DC operating points," in *IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 914–917, May 1990. 9, 9.2.1, 9.2.1.2, 9.2.1.2, 9.2.1.2, 9.2.1.2, 9.2.1.3, 9.2.1.3, 9.2.1.3, 9.2.1.3, 9.2.3, 9.2.3.1, 10.2

[178] Trajkovic, L., *The Electrical Engineering Handbook*, ch. Nonlinear Circuits, pp. 75–82. Academic Press, 2005. 9

[179] Trajkovic, L. and Mathis, W., "Parameter embedding methods for finding DC operating points: formulation and implementation," in *1995 International Symposium on Nonlinear Theory and its Applications, NOLTA 1995*, (Las Vegas NE, USA), pp. 1159–1164, December 1995. 9, 10.1.1, 10.2

[180] Tummescheit, H., Eborn, J., and Prölss, K., "Airconditioning–a Modelica library for dynamic simulation of AC systems," in *Proceedings of the Fourth International Modelica Conference* (Schmitz, G., ed.), (Hamburg, Germany), 2005. 1.4.1

[181] Tummescheit, H., *Design and Implementation of Object-Oriented Model Libraries using Modelica.* PhD thesis, Lund University, Department of Automatic Control, 2002. 1.4.1, 2, 2.5

[182] Tummescheit, H., Eborn, J., and Wagner, F. J., "Development of a Modelica base library for modeling of thermo-hydraulic systems," in *Modelica Workshop 2000 Proceedings*, pp. 41–51, 2000. 2.5

[183] van Leer, B., "Towards the ultimate conservative difference scheme I. the quest for monotonicity," *Lecture Notes In Physics*, vol. 18, pp. 163–168, 1973. 12.2.2.1

[184] van Leer, B., "Towards the ultimate conservative difference scheme II. monotonicity and conservation combined in a second order scheme," *Journal Of Computational Physics*, vol. 14, pp. 361–370, 1974. 12.2.2.1

[185] van Leer, B., "Towards the ultimate conservative difference scheme III. upstream-centered finite difference schemes for ideal compressible flow," *Journal Of Computational Physics*, vol. 23, pp. 263–275, 1977. 12.2.2.1

[186] Vargas, J. and Bejan, A., "Integrative thermodynamic optimization of the environmental control system of an aircraft," *International Journal Of Heat And Mass Transfer*, vol. 44, no. 20, pp. 3907–3917, 2001. 1.3.2

[187] Vasel, J. and Schmitz, G., "Transient simulation of a direct-evaporating CO2 cooling system for an aircraft," in *25th International Congress of the Aeronautical sciences (ICAS), Proceedings of the*, (Hamburg, Germany), September 2006. 1.4.1

[188] Wagner, W. and Kretzschmar, H.-J., *International Steam Tables - Properties of Water and Steam based on the Industrial Formulation IAPWS-IF97.* Springer, 1997. 7.3.1

[189] Watson, L. T., "Probability-one homotopies in computational science," *Journal Of Computational And Applied Mathematics*, vol. 140, pp. 785–807, 2002. 9.1

[190] Watson, L. T., Billups, S. C., and Morgan, A. P., "Algorithm 652: Hompack, a suite of codes for globally convergent homotopy algorithms," *ACM Transactions On Mathematical Software*, vol. 13, pp. 281–310, 1987. 8.1

[191] Watson, L. T., "Globally convergent homotopy methods: A tutorial," *Applied Mathematics And Computation*, vol. 31, pp. 369–396, May 1989. 9.1, 9.1

[192] Wayburn, T. and Seader, J., "Homotopy continuation methods for computer-aided process design," *Computers & Chemical Engineering*, vol. 11, no. 1, pp. 7–25, 1987. 8.1, 8.1, 9

[193] Willson Jr., A. N., "The no-gain property for networks containing three-terminal elements," *IEEE Transactions on Circuits and Systems*, vol. 22, pp. 678–687, August 1975. 9, 9.2.3.1, 9.2.3.2, 10.2.2

[194] Wise, S. M., Sommese, A. J., and Watson, L. T., "Algorithm 801: POLSYS_PLP, a partitioned linear product homotopy code for solving polynomial systems of equations," *ACM Transactions On Mathematical Software*, vol. 26, pp. 176–200, March 2000. 9.1.4

[195] Wolf, D. and Sanders, S., "Multiparameter homotopy methods for finding DC operating points of nonlinear circuits," *IEEE Transactions on Circuits and Systems*, vol. 43, no. 10, pp. 824–838, 1996. 8.1

[196] Wyatt Jr., J. L., Chua, L. O., Gannett, J. W., Gknar, I. C., and Green, D. N., "Energy concepts in the state-space theory of nonlinear n-ports: Part I—passivity," *IEEE Transactions on Circuits and Systems*, vol. CAS-28, no. 1, pp. 48–61, 1981. 9

[197] Yamamura, K., Sekiguchi, T., and Inoue, Y., "A fixed-point homotopy method for solving modified nodal equations," *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 46, no. 6, pp. 654–665, 1999. 8.1, 9, 9.1.5, 10.1.1, 10.2

[198] young Lee, T. and kyung Shim, J., "Elimination-based solution method for the forward kinematics of the general Stewart-Gough platform," in *Proceedings of the Computational Kinematics Conference*, (Seoul), 2001. 7.3.2

[199] Zangwill, W. and Garcia, C., *Pathways to solutions, fixed points, and equilibria.* Prentice-Hall, 1975. 9.1.5