Fortschritt-Berichte VDI

VDI

Reihe 6

Energietechnik

Dipl.-Ing. (FH) Matthias Kunick,
Görlitz

Nr. 618

Fast Calculation of Thermophysical Properties in Extensive Process Simulations with the Spline-Based Table Look-Up Method (SBTL)

Fast Calculation of Thermophysical Properties in Extensive Process Simulations with the Spline-Based Table Look-Up Method (SBTL)

Submitted to the
Faculty of Mechanical Science and Engineering of the
Technische Universität Dresden
in partial fulfillment of the requirements for the degree of
Doktor-Ingenieur (Dr.-Ing.)

Dipl.-Ing. (FH) Matthias Kunick
born December 26, 1978 in Altdöbern, Germany

Day of submittal: June 9, 2017
Day of defense: October 23, 2017

Fortschritt-Berichte VDI

Reihe 6

Energietechnik

Dipl.-Ing. (FH) Matthias Kunick,
Görlitz

Nr. 618

Fast Calculation of Thermophysical Properties in Extensive Process Simulations with the Spline-Based Table Look-Up Method (SBTL)

VDI verlag

Kunick, Matthias
Fast Calculation of Thermophysical Properties in Extensive Process Simulations with the Spline-Based Table Look-Up Method (SBTL)
Fortschr.-Ber. VDI Reihe 6 Nr. 618. Düsseldorf: VDI Verlag 2018.
172 Seiten, 80 Bilder, 58 Tabellen.
ISBN 978-3-18-361806-4, ISSN 0178-9414
€ 62,00/VDI-Mitgliederpreis € 55,80.

Keywords: Spline-Based Table Look-up Method – Thermophysical Properties – Spline-Interpolation – Table Look-Up Method – Equation of State – Process Simulations – Computational Fluid Dynamics – Heat-Cycle Simulations – Transient Processes

The presented Spline-Based Table Look-Up Method (SBTL) is intended to be used for fast and accurate property calculations in computationally extensive process simulations, such as Computational Fluid Dynamics (CFD), heat-cycle calculations, simulations of non-stationary processes, and real-time process optimizations, where conventional multiparameter equations of state may be unsuitable because of their computing time consumption. Through the use of the SBTL method, the results of existing property formulations are accurately reproduced at high computational speed. SBTL property functions, their first derivatives, and inverse functions are continuous and numerically consistent with each other. The developed algorithms are successfully applied in commercial and non-commercial software products for CFD, heat-cycle calculations, and simulations of non-stationary processes. The International Association for the Properties of Water and Steam has adopted the "Guideline on the Fast Calculation of Steam and Water Properties with the Spline-Based Table Look-Up Method (SBTL)".

Bibliographische Information der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Daten sind im Internet unter www.dnb.de abrufbar.

Bibliographic information published by the Deutsche Bibliothek
(German National Library)
The Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliographie (German National Bibliography); detailed bibliographic data is available via Internet at www.dnb.de.

Dissertation

Technische Universität Dresden / Fakultät Maschinenwesen

Datum der Verteidigung: 23. Oktober 2017

ISSN 0178-9414

ISBN 978-3-18-361806-4

Acknowledgements

This work was carried out within the scope of a co-operative doctoral procedure between the Faculty of Mechanical Science and Engineering at the Technische Universität Dresden and the Faculty of Mechanical Engineering at the Zittau/Görlitz University of Applied Sciences under the supervision of Prof. Dr.-Ing. Uwe Gampe and Prof. Dr.-Ing. Hans-Joachim Kretzschmar. In this acknowledgement, I want to thank all the people who have, in various ways, supported me and my work over the past several years.

I am particularly grateful to Prof. Dr.-Ing. Hans-Joachim Kretzschmar for encouraging me in this work and for his supervision. His ongoing support was crucial for the completion of this work and its success. I am indebted to Prof. Dr.-Ing. Uwe Gampe for enabling the realization of this work and his active interest in it. I am very grateful to Prof. Dr.-Ing. Wolfgang Wagner for his willingness to be a co-referee and for the interesting conversations we had on various occasions.

Furthermore, I am grateful to the members of the International Association for the Properties of Water and Steam (IAPWS), in particular to the members of the IAPWS Task Group “CFD Steam Property Formulation” and to Dr. Allan H. Harvey, Dr. Daniel G. Friend, Dr. Tobias Löw, Dr. Reiner Pawellek, and Ingo Weber for the fruitful collaboration. I thank Prof. Dr. Francesca di Mare and Pascal Post for the pleasant and productive collaboration. The opportunity to test the property functions developed in this work in an advanced Computational Fluid Dynamics code has been of extraordinary importance.

For numerous good reasons, I am extremely grateful to Dr. Eric W. Lemmon. His invitation brought me to the National Institute of Standards and Technology in Boulder (CO/USA) where he taught me many important lessons on equations of state.

I want to thank my friends and colleagues, in particular but not exclusively, Mareike Weidner, Jutta Pfitzner, Bert Salomo, Christoph Ebermann, Dr.-Ing. Sebastian Herrmann, Roman Schneider, and Felix Rothe from the Faculty of Mechanical Engineering at the Zittau/Görlitz University of Applied Sciences for providing a good working atmosphere. I am also grateful to Prof. Dr.-Ing. Roland Span, Dr.-Ing. Monika Thol (“Alles wird gut!”), Prof. Dr.-Ing. Markus Richter, Dr.-Ing. Andreas Jäger, Dr.-Ing. Robin Wegge, Theresa Eckermann, Stefan Herrig, Sebastian Hielscher, Nico Schneider, and many other present and former staff members of the Chair of Thermodynamics at the Ruhr-Universität Bochum for their friendship and support.

My warmest thanks go to my family, in particular to my wife Kerstin. Their understanding and support rendered this work possible.

Finally, I thank Dr. Eric W. Lemmon, Dr. Ian Bell, JinHyuk Hong, and Jeremy Cope for proofreading this thesis and for their numerous changes to the use of the English language and recommendations for improving this thesis.

The financial support from the Saxon State Ministry for Science and Art is gratefully acknowledged.

Görlitz, October 2017

Contents

List of Symbols and Nomenclature

Symbols	
a	Spline polynomial coefficient
c_p	Specific isobaric heat capacity
c_v	Specific isochoric heat capacity
CTR	Computing-Time Ratio
f	Specific Helmholtz free energy
f	Function
floor()	Round down to nearest integer
$\boldsymbol{F}$	Vector of functions
g	Specific Gibbs free energy
h	Specific enthalpy
i	Interval index in x_1 direction
(i)	Left node of the interval $\{i\}$ (for knots equal to the nodes)
(i)	Node within the interval $\{i\}$ (for knots between the nodes)
$\{i\}$	Interval $\overline{x}_{1,i} \le \overline{x}_1 < \overline{x}_{1,i+1}$ (for knots equal to the nodes)
$\{i\}$	Interval $\overline{x}^{\mathrm{K}}_{1,i} \le \overline{x}_1 < \overline{x}^{\mathrm{K}}_{1,i+1}$ (for knots between the nodes)
(i,j)	Lower left node of the cell $\{i,j\}$ (for knots equal to the nodes)
(i,j)	Node within the cell $\{i,j\}$ (for knots between the nodes)
$\{i,j\}$	Cell defined by the intervals $\{i\}$ and $\{j\}$
I	Number of nodes along x_1
j	Interval index in x_2 direction
$\{j\}$	Interval $\overline{x}_{2,j} \le \overline{x}_2 < \overline{x}_{2,j+1}$ (for knots equal to the nodes)
$\{j\}$	Interval $\overline{x}^{\mathrm{K}}_{2,j} \le \overline{x}_2 < \overline{x}^{\mathrm{K}}_{2,j+1}$ (for knots between the nodes)
J	Number of nodes along x_2
$\boldsymbol{J}$	Jacobian matrix
p	Pressure
R	Specific gas constant
s	Specific entropy
T	Absolute temperature
TOL	Tolerance for iterative procedures (typically less than or equal to 10^{-8})
u	Specific internal energy
v	Specific volume
w	Speed of sound
x	Vapor fraction
x_1	Independent variable
$\overline{x}_1$	Transformed independent variable
x_2	Independent variable
$\overline{x}_2$	Transformed independent variable

X	Vector of unknowns
z	Dependent variable
$\overline{z}$	Transformed dependent variable

Greek Symbols

δ	Reduced density
Φ	Reduced Helmholtz free energy
γ	Reduced Gibbs free energy
η	Reduced enthalpy
η	Dynamic viscosity
θ	Reduced temperature
λ	Thermal conductivity
π	Reduced pressure
ρ	Mass density
τ	Inverse reduced temperature

Subscripts

B	At region boundary
c	At the critical point
i	Interval index in x_1 direction
j	Interval index in x_2 direction
liq_spin	At liquid spinodal
min	Minimum value
max	Maximum value
perm	Permissible value
RMS	Root-mean-square value of a quantity, see below
s	At saturation
0	Reference state
*	Reducing quantity

Superscripts

AUX	Auxiliary spline function
G	Spline function for the gas region
HT	Spline function for the high-temperature region
INV	Inverse spline function
K	Knot
L	Spline function for the liquid region
MG	Spline function for the metastable-vapor and the gas region
SPL	Spline function

T	Transposed
-T	Inverted and transposed
o	Ideal-gas state; ideal-gas part
r	Residual part
′	State of saturated liquid
″	State of saturated vapor

The root-mean-square value is

$$\Delta x_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(\Delta x_n)^2} \,,$$

where Δx_n can be either the absolute or percentage difference between the corresponding quantities x; N is the number of Δx_n values (depending on the property, between 10 million and 100 million points are uniformly distributed over the respective range of validity).

Definitions	
Backward function	Inverse function for $x_1(z)$, $x_1(z,x_2)$, or $x_2(x_1,z)$
Forward function	Explicit function for $z(x_1)$ or $z(x_1,x_2)$
Knot	Connection point of neighboring spline polynomials
Node	Point to be intersected by a spline polynomial
Spline function	Continuous, piecewise-defined function consisting of several spline polynomials
Spline polynomial	Polynomial whose coefficients are determined with a spline algorithm

Abstract

Numerical process simulations, such as flow analysis with Computational Fluid Dynamics (CFD), power-plant design with heat cycle calculations, and real-time process optimizations, are widely used in power engineering. These simulations are computationally expensive, especially when transient processes are considered. During the computation, the thermophysical properties of the utilized working fluids need to be calculated extremely often. Therefore, fast and accurate property functions are required. Furthermore, numerical process simulations require these property functions to be continuously differentiable once and numerically consistent with each other. Because of their computing-time consumption, accurate multiparameter equations of state are unsuitable for some extensive process simulations and faster, but often less accurate, property calculation algorithms are applied.

In order to provide fast and accurate property calculation algorithms for computationally expensive process simulations, the International Association for the Properties of Water and Steam (IAPWS) has established the task group "CFD Steam Property Formulation". Within this task group, the Spline-Based Table Look-up Method (SBTL) has been developed in this work. The SBTL method combines polynomial spline interpolation techniques and specialized coordinate transformations to reproduce the results of an underlying property formulation, *e.g.*, the industrial formulation IAPWS-IF97 or the scientific formulation IAPWS-95 for water and steam, with high accuracy and low computing time. Depending on the order of the applied spline polynomials, SBTL property functions are at least one time continuously differentiable. Furthermore, the so-called inverse spline functions are numerically consistent with their corresponding forward spline functions, *e.g.*, $u(p,v)$ and $p(v,u)$.

In this work, the development of the SBTL method, as well as its practical application for property calculations in numerical process simulations, is described. To begin, currently applied property calculation methods are discussed regarding their accuracy and their computing-time consumption. From the obtained findings, conclusions for the development of a new property calculation method are drawn. Then the developed SBTL method is described in detail. The SBTL method is exemplified by its application to the industrial formulation IAPWS-IF97 and the scientific formulation IAPWS-95 along with the current transport property formulations for water and steam. For these formulations, SBTL property functions of specific volume and specific internal energy (v,u), as required in CFD, are presented. From these SBTL property functions, numerically consistent inverse functions for calculations from (p,v) and (u,s) are derived. Analogously, SBTL property functions of pressure and specific enthalpy (p,h), as required in heat cycle calculations, are described. With these functions thermodynamic and transport properties, their derivatives, and inverse functions are calculable in the single-phase, two-phase, and metastable regions. The properties calculated from the SBTL property functions represent those of the underlying IAPWS standards with very high accuracy. Typically, the maximum relative deviations amount to between 1 to 100 ppm depending on the property and the range of state. Computations from the (v,u) spline functions are more than 100 times faster than calculations with IAPWS-IF97 and are more than 200 times faster than calculations with IAPWS-95.

The applicability of the SBTL method is verified in the CFD code TRACE, developed at the German Aerospace Center (DLR), as well as in two different heat cycle calculation software

tools, namely in KRAWAL-modular, developed by SIEMENS PG, and in EBSILON® Professional, developed by STEAG Energy Services. Additionally, the use of the SBTL method is verified in RELAP-7, the thermalhydraulic program for the simulation of transient processes in nuclear reactors and plants, developed by the Idaho National Laboratory (INL). The numerical results of the process simulations with the SBTL method show negligible differences from those obtained through the direct application of the underlying property formulations, but the overall computing times are reduced significantly.

In order to apply the SBTL method to property functions for any fluid, the software FluidSplines has been developed.

Based on the results outlined above, the "IAPWS Guideline on the Fast Calculation of Steam and Water Properties with the Spline-Based Table Look-Up Method (SBTL)" has been developed, which was adopted by IAPWS in 2015.

Kurzfassung

Numerische Prozesssimulationen, wie beispielsweise rechnergestützte Analysen strömungsmechanischer Vorgänge (englisch: Computational Fluid Dynamics, CFD), Kreisprozessberechnungen zur Auslegung kraftwerkstechnischer Anlagen und Betriebsoptimierungen in Echtzeit, werden in der Energietechnik vielfältig eingesetzt. Diese Simulationen sind rechentechnisch sehr aufwändig, insbesondere wenn instationäre Vorgänge betrachtet werden müssen. Während der Prozessberechnung müssen die thermophysikalischen Eigenschaften der verwendeten Arbeitsfluide extrem häufig ermittelt werden. Hierfür werden schnelle und genaue Stoffwertfunktionen benötigt. Die verwendeten Stoffwert-Berechnungsalgorithmen müssen einmal stetig differenzierbar und numerisch konsistent zueinander sein. Aufgrund ihrer langen Rechenzeiten sind genaue empirische Zustandsgleichungen für den Einsatz in aufwändigen numerischen Prozesssimulationen nicht geeignet, weshalb auf einfachere, jedoch häufig auch ungenauere Stoffwert-Berechnungsalgorithmen zurückgegriffen wird.

Um schnelle und gleichzeitig sehr genaue Stoffwert-Berechnungsalgorithmen zur Verfügung zu stellen, hat die International Association for the Properties of Water and Steam (IAPWS) die Task Group “CFD Steam Property Formulation“ gebildet. Innerhalb dieser Task Group wurde das Spline-basierte Table Look-up Verfahren (SBTL) im Rahmen dieser Arbeit entwickelt. Das SBTL Verfahren kombiniert Spline-Interpolationsalgorithmen mit speziellen Koordinatentransformationen um die zugrunde gelegte Stoffwertgleichung, beispielsweise die Industrieformulation IAPWS-IF97 oder die wissenschaftliche Formulation IAPWS-95 für Wasser und Wasserdampf, mit hoher Genauigkeit und geringer Rechenzeit wiederzugeben. Abhängig vom Grad der verwendeten Spline-Polynome sind SBTL Stoffwertfunktionen mindestens einmal stetig differenzierbar. Zudem ermöglicht das SBTL Verfahren die Berechnung numerisch konsistenter Umkehrfunktionen.

In der vorliegenden Arbeit wird die Entwicklung des SBTL Verfahrens sowie dessen praktische Anwendung zur Stoffwertberechnung in numerischen Prozesssimulationen beschrieben. Dazu werden zunächst die derzeit verwendeten Stoffwert-Berechnungsalgorithmen hinsichtlich ihrer Genauigkeit und ihres Rechenzeitbedarfs diskutiert. Ausgehend von den gewonnenen Erkenntnissen werden Zielstellungen und Ansätze für die Entwicklung eines neuen Stoffwert-Berechnungsverfahrens formuliert. Anschließend wird das entwickelte SBTL Verfahren im Detail erläutert. Das SBTL Verfahren wird beispielhaft auf die Industrieformulation IAPWS-IF97, die wissenschaftliche Formulation IAPWS-95 sowie die aktuellen IAPWS Formulationen für die Transporteigenschaften für Wasser und Wasserdampf angewendet. Für diese zugrunde liegenden Gleichungen werden SBTL Stoffwertfunktionen von spezifischem Volumen und spezifischer innerer Energie (v,u), wie sie beispielsweise in CFD Simulationen zum Einsatz kommen, vorgestellt. Zudem werden aus diesen SBTL Stoffwertfunktionen numerisch konsistente Umkehrfunktionen von (p,v) und (u,s) entwickelt. Analog werden SBTL Stoffwertfunktionen für die in Kreisprozessberechnungen häufig auftretende Variablenkombination von Druck und spezifischer Enthalpie (p,h) sowie entsprechende Umkehrfunktionen von (p,T), (p,s) und (h,s) beschrieben. Mit diesen Funktionen können die thermophysikalischen Eigenschaften sowie deren Ableitungen und Umkehrfunktionen im Ein- und Zweiphasengebiet berechnet werden. Die aus den SBTL Funktionen berechneten Stoffwerte stimmen mit den zugrundeliegenden Gleichungen mit sehr hoher

Genauigkeit überein, beanspruchen aber wesentlich geringere Rechenzeiten. Typische maximale Abweichungen betragen je nach Stoffwertfunktion und Gültigkeitsbereich 1 bis 100 ppm. Im Vergleich mit dem Industriestandard IAPWS-IF97 sind die SBTL Funktionen von (v,u) mehr als 100-mal schneller. Gegenüber dem wissenschaftlichen Standard IAPWS-95 sind diese Funktionen mehr als 200-mal schneller.

Die Anwendbarkeit des SBTL Verfahrens wird im CFD-Code TRACE, entwickelt am Deutschen Zentrum für Luft- und Raumfahrt (DLR), sowie in den Kreisprozessberechnungsprogrammen KRAWAL-modular, entwickelt von SIEMENS PG, und EBSILON® Professional, entwickelt von STEAG Energy Services, nachgewiesen. Weiterhin wird der Nutzen des SBTL Verfahrens in RELAP-7, der vom Idaho National Laboratory (INL) entwickelten Software zur Simulation instationärer Prozesse in Kernreaktoren, aufgezeigt. Die Ergebnisse der Prozessberechnungen mit dem SBTL Verfahren weisen gegenüber der direkten Verwendung der zugrunde liegenden Gleichungen vernachlässigbare Differenzen auf. Die Gesamtrechenzeiten der Prozessberechnungen werden jedoch signifikant reduziert.

Für die Anwendung des SBTL Verfahrens auf weitere Stoffwertfunktionen und beliebige Fluide ist in dieser Arbeit die Software FluidSplines entwickelt worden.

Auf Grundlage der Ergebnisse dieser Arbeit ist die neue “IAPWS Guideline on the Fast Calculation of Steam and Water Properties with the Spline-Based Table Look-Up Method (SBTL)” erarbeitet worden, welche von der IAPWS im Jahr 2015 als internationale Richtlinie verabschiedet wurde.

1 Introduction

Extensive numerical process simulations, such as Computational Fluid Dynamics (CFD), heat cycle calculations, and real-time process optimizations are indispensable tools for power engineering. In the development of advanced processes and plants, accurate process simulations replace costly prototypes and enable detailed optimizations regarding efficiency, lifetime, flexibility, costs, etc. Moreover, numerical process simulations are used to optimize process parameters during plant operation.

In order to obtain accurate simulation results, the process to be analyzed needs to be described with an appropriate mathematical model that requires a certain amount of computing time. Driven by the pursuit of advancements in engineering, the complexity of numerical process simulations is growing, leading to increased computing times. Detailed numerical process simulations are computationally expensive, especially when transient processes are considered. A large proportion of the computing time is spent on the calculation of thermophysical properties of the working fluids used. Therefore, very fast property calculation algorithms need to be applied. Moreover, the numerical algorithms employed in process simulations require the property functions to be continuously differentiable once and numerically consistent with each other; otherwise the computation may not reach convergence.

For water and steam, as the most important working fluid in power engineering, the International Association for the Properties of Water and Steam (IAPWS) provides internationally accepted formulations for thermodynamic and transport properties. The IAPWS Formulation 1995 for General and Scientific Use (IAPWS-95) [1, 2] is the most accurate representation of the thermodynamic properties of the fluid phases of water substance over a wide range of conditions currently available. As for other conventional Helmholtz equations of state, property calculations from IAPWS-95 are computationally expensive. In the steam power industry, property functions of pressure and temperature (p,T), pressure and specific enthalpy (p,h), pressure and specific entropy (p,s), and specific enthalpy and specific entropy (h,s) are called millions of times when designing steam turbines, steam generators, and heat cycles. These functions need to be calculated by iteration from IAPWS-95, which is very time-consuming. This often leads to inacceptable computing times in extensive numerical process simulations.

To meet the requirements of the steam power industry, the IAPWS Industrial Formulation 1997 (IAPWS-IF97) [3, 4] and its supplementary releases [5, 6, 7, 8] are available. This formulation is sufficiently accurate for industrial applications and enables fast property calculations from (p,T), (p,h), (p,s), and (h,s). This is achieved by the combination of computationally efficient fundamental equations, each of which represents a different region in the range of validity of IAPWS-IF97, and so-called “backward equations” for calculating inverse property functions without time-consuming iterations. Due to the imperfect numerical consistency with the basic equations of IAPWS-IF97, the application of backward equations for simulating non-stationary processes can lead to convergence problems. In these situations, inverse property functions should be calculated by iteration from the basic equations with starting values determined from the available backward equations.

For density based CFD solvers, property functions of specific volume and specific internal energy (v,u), as well as of pressure and specific volume (p,v), are required. Due to the absence of backward equations for these variable combinations, these functions need to be calculated by iteration from the corresponding fundamental equation. This is computationally expensive and therefore inappropriate for CFD. In order to reduce the computing times, property calculations are often simplified, for example, through the use of the ideal-gas equation or a cubic equation of state. Depending on the range of state, these simplifications cause inaccuracies in the results of the process simulation.

As an alternative, table look-up methods are frequently applied for fast and accurate property calculations. For these methods, discrete values of the required properties are calculated from accurate equations of state and are stored in look-up tables. During the process simulation, properties are determined from these look-up tables with simple interpolation or approximation algorithms. A prime example for table look-up methods is the Tabular Taylor Series Expansion method (TTSE), which was adopted as an IAPWS Guideline [9, 10, 11] in 2003. The desired properties are calculated from second order Taylor series expansions obtained from the tabulated derivatives at the midpoint of the corresponding cell in the look-up table. The TTSE method is very fast, but adjacent Taylor series are not connected continuously. This characteristic leads to numerical problems in CFD and non-stationary simulations with very small spatial and time discretization.

In order to provide property calculation algorithms that fulfill the requirements of extensive numerical process simulations regarding accuracy, computing speed, differentiability, and numerical consistency, the task group “CFD Steam Property Formulation” was established by the IAPWS working group “Industrial Requirements and Solutions” in 2008. Resulting from the activities in this task group, the Spline-Based Table Look-Up method (SBTL) is proposed in this work. The SBTL method is intended to be a supplement to existing property formulations, not only for water and steam, but also for other fluids. The SBTL method aims to represent the underlying property formulation with very good agreement, but with significantly reduced computing time. Additionally, the SBTL method is intended to provide numerically consistent forward and backward functions, *e.g.*, of $u(p,v)$ and $p(v,u)$.

Section 2 gives an overview of thermodynamic property formulations that are relevant for computationally expensive numerical simulations. From the discussion of these formulations with regard to their accuracy, computing speed, and numerical consistency, conclusions for the development of alternative property calculation algorithms are drawn. In Section 3, the fundamentals of the newly developed SBTL method are explained in detail.

For generating SBTL property functions, the software FluidSplines has been developed. This software enables the application of the SBTL method to one- or two-dimensional property functions of any fluid. The basic structure and key features of FluidSplines are described in Section 4.

The application of the SBTL method to the industrial formulation IAPWS-IF97 is described in Section 5. SBTL property functions of (v,u) as well as numerically consistent inverse functions of (p,v) and (u,s) are presented in Section 5.1. Analogously, SBTL property functions of (p,h), and numerically consistent inverse functions of (p,T), (p,s), and (h,s) are presented in

Section 5.2. Through the use of the provided SBTL functions, thermodynamic and transport properties, their derivatives, and inverse functions are calculable in the single-phase region and in the two-phase region. SBTL property functions of (v,u) and (p,h) covering the stable and the metastable vapor region are given in Section 5.3. The SBTL method has also been applied to the IAPWS Formulation 1995 for General and Scientific Use as shown in Section 6. For every SBTL property function presented in Sections 5 and 6, the deviations from the underlying property formulation and the results of the computing-time comparisons are given. The application of spline functions to thermodynamic potentials is discussed with an example for s(v,u) in Section 7.

The applicability of the SBTL method in CFD is demonstrated in Section 8.1. In a joint project with the German Aerospace Center (DLR), SBTL property functions based on IAPWS-IF97 have been implemented into the advanced CFD software TRACE. The numerical results and computing times of test calculations with the SBTL property functions have been compared to those obtained through the use of the direct application of IAPWS-IF97. In Section 8.2 the use of the SBTL method in heat cycle calculation software is demonstrated. The SBTL functions of (p,h) and the corresponding inverse functions of (p,T), (p,s), and (h,s) based on IAPWS-IF97 (see Sec. 5.2) have been implemented in KRAWAL-modular (developed by SIEMENS PG) and EBSILON® Professional (developed by STEAG Energy Services). The application of the SBTL method in RELAP-7, the nuclear reactor system safety analysis code developed at the Idaho National Laboratory (INL), is explained in Section 8.3. For this purpose, SBTL functions of (v,u) and inverse functions of (p,T), (p,v), (p,h), (p,s), and (h,s) were developed based on IAPWS-95. The range of validity of these property functions includes the metastable-liquid and metastable-vapor regions at the vapor-liquid phase transition. This enables the application of a novel 7-equation non-equilibrium two-phase model, developed at INL.

2 Property Calculation Algorithms for Numerical Process Simulations

A prerequisite of many scientific and industrial calculations is the knowledge of the thermophysical properties of the working fluids used. A basic theory to describe these properties over the entire fluid range of state within the uncertainties of available measurements is not known yet. Therefore, thermophysical properties are approximated either under some theoretical assumptions or by empirical equations of state. In this section, selected thermodynamic property formulations for water and their applicability in computationally expensive numerical process simulations are discussed. The accuracy of these formulations is described in the respective sections and computing-time comparisons are given in Sec. 2.5.3.

2.1 Simple Thermal Equations of State

Thermal equations of state describe the relationships between pressure, temperature, and specific volume, *i.e.*, $p(T,v)$ or $v(p,T)$. In conjunction with an equation for the isobaric heat capacity of the ideal gas $c_p^{\mathrm{o}}(T)$ the remaining caloric properties can be derived from fundamental thermodynamic relations. The isobaric heat capacity of the ideal gas $c_p^{\mathrm{o}}(T)$ is either obtained from theoretical models or from measurements extrapolated to zero density.

2.1.1 Ideal-Gas Model

Under the theoretical assumption that the molecules of a gas do not possess a volume and do not interact with each other except for elastic collisions, the ideal-gas equation

$$p = \frac{RT}{v} \tag{2.1}$$

is derived as a thermal equation of state from statistical mechanics. The energy content of the ideal gas is stored in the vibrational, rotational, and translational movement of its molecules and is temperature dependent only. For water vapor, a simple fourth-order polynomial for the isobaric heat capacity of the ideal gas $c_p^{\mathrm{o}}(T)$ is given by Poling et al. [12] and reads

$$c_p^{\mathrm{o}}(T) = R\left(a_0 + a_1 T + a_2 T^2 + a_3 T^3 + a_4 T^4\right). \tag{2.2}$$

Through the use of fundamental thermodynamic relations all remaining thermodynamic properties can be derived from Eq. (2.1) and $c_p^{\mathrm{o}}(T)$ as shown in Table 1.

The deviations in specific volume and isobaric heat capacity of the ideal-gas model from the real fluid behavior of water, *i.e.*, from the reference equation of state IAPWS-95 [1], are shown in Figs. 1 and 2. The pressure and temperature ranges covered by these diagrams include the range of state of modern steam power cycles. The ideal-gas model can be applied to gases at low densities and temperatures above the critical temperature. At higher densities and lower temperatures, the volumes of the molecules and the interactions between them become more significant and the real fluid behavior must be taken into account.

Due to its simplicity, the ideal-gas model is well established in CFD and other extensive numerical process simulations. In many situations, the ideal-gas model is utilized for property calculations in the non-ideal range of state, where a more complex formulation for the real fluid would lead to inacceptable computing times. Depending on the range of state, this implies inaccuracies in the results of the process simulation.

Table 1: Relationships between thermodynamic properties and the ideal-gas model

Property	Relationship
Specific internal energy u°	$u^{\circ}(T) = h_0^{\circ} + \int_{T_0}^{T} c_p^{\circ}(T)\,\mathrm{d}T - RT$
Specific enthalpy h°	$h^{\circ}(T) = h_0^{\circ} + \int_{T_0}^{T} c_p^{\circ}(T)\,\mathrm{d}T$
Specific entropy s°	$s^{\circ}(T) = s_0^{\circ} + \int_{T_0}^{T} \frac{c_p^{\circ}(T)}{T}\,\mathrm{d}T - R\ln\left(\frac{p}{p_0}\right)$
Specific isochoric heat capacity c_v°	$c_v^{\circ}(T) = c_p^{\circ}(T) - R$
Speed of sound w°	$w^{\circ}(T) = \sqrt{\frac{c_p^{\circ}(T)}{c_p^{\circ}(T) - R}RT}$

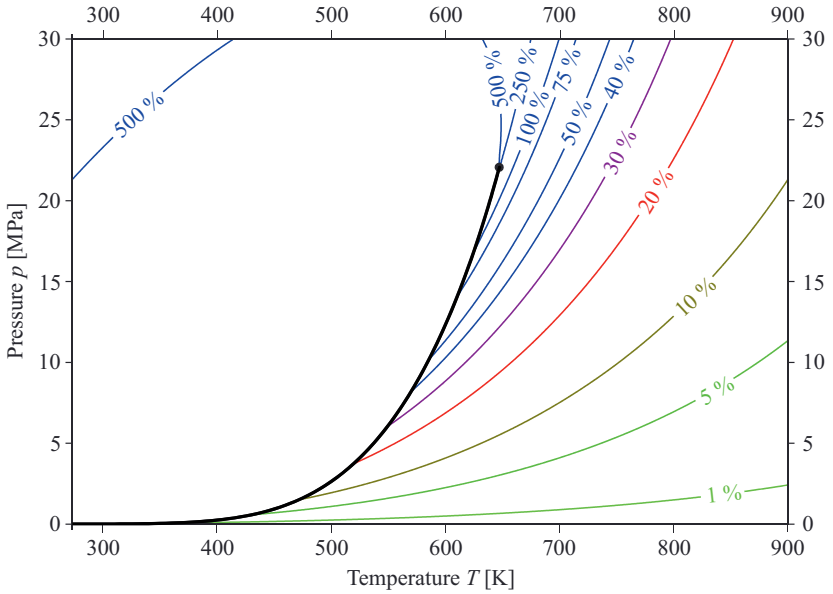

Figure 1: Relative deviation in specific volume v of the ideal-gas model from the reference equation of state IAPWS-95 [1].

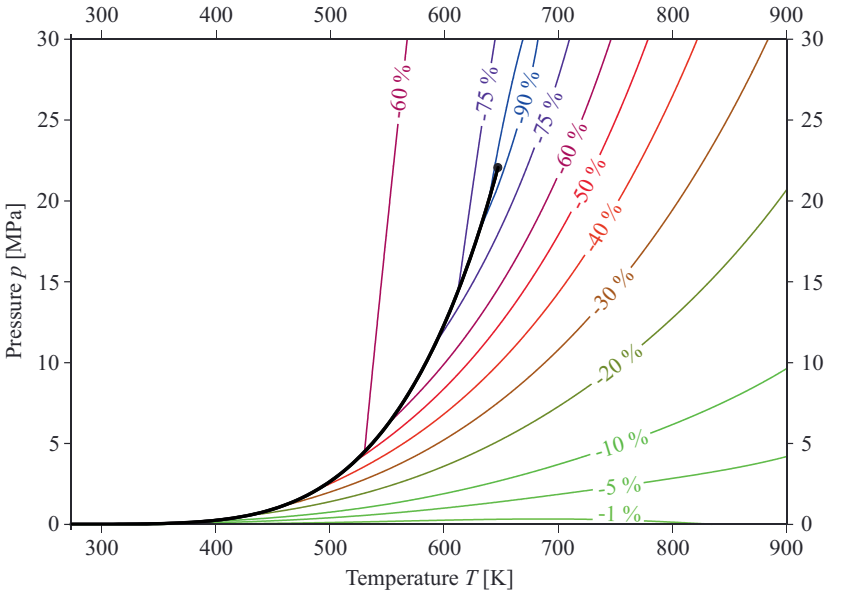

Figure 2: Relative deviation in specific isobaric heat capacity c_p° of the ideal gas from the reference equation of state IAPWS-95 [1].

2.1.2 Cubic Equations of State

For process simulations with phase transitions from liquid to vapor, or vice versa, a consistent property formulation for both phases is required. The first qualitatively correct representation of thermal properties across the entire fluid range of state was given by van der Waals [13] as

$$p = \frac{RT}{v-b} - \frac{a}{v^2}. \tag{2.3}$$

The parameter b represents the physical volume of the molecules and therefore the term $(v-b)$ in Eq. (2.3) can be considered as the remaining volume for the movement of the molecules. The parameter a takes the intermolecular attraction forces into account, so that $\left(a/v^2\right)$ can be considered as a pressure drop due to these forces.

When rearranged as a function of specific volume v, the van der Waals equation of state, Eq. (2.3), becomes a cubic polynomial

$$0 = v^3 - \left(b + \frac{RT}{p}\right)v^2 + \frac{a}{p}v - \frac{ab}{p}$$

and is therefore named a cubic equation of state. The constants a, b, and R are obtained from the parameters p_c, T_c, and v_c at the critical point, where $\left(\partial p/\partial v\right)_T$ and $\left(\partial^2 p/\partial v^2\right)_T$ are zero. The resulting constants are

$$a = 3 p_c v_c^2, \tag{2.4}$$

$$b = \frac{v_c}{3}, \tag{2.5}$$

$$R = \frac{8}{3}\frac{p_c v_c}{T_c}. \tag{2.6}$$

The specific gas constant R obtained from Eq. (2.6) does not match the value of $R = \bar{R}/M$. Therefore, the van der Waals equation, Eq. (2.3), does not match the ideal-gas equation, Eq. (2.1), for $v >> b$ and $v >> a$.

In order to improve the accuracy in the liquid region and the critical region, and to predict the vapor-liquid equilibrium more accurately, several cubic equations of state have been developed. The Peng-Robinson equation of state [14] can be expressed as

$$p = \frac{RT}{v-b} - \frac{a(T)}{v(v+2b)-b^2}. \tag{2.7}$$

The parameters $a(T)$ and b are calculated from

$$a(T) = a_0\left(1 + n\left(1 - \sqrt{\frac{T}{T_c}}\right)\right)^2. \tag{2.8}$$

and

$$b = 0.0778\frac{RT_c}{p_c}.$$

In Eq. (2.8), the parameters a_0 and n are calculated from

$$a_0 = 0.45724\frac{(RT_c)^2}{p_c} \qquad \text{and} \qquad n = 0.37464 + 1.5422\omega - 0.26993\omega^2$$

where ω is the acentric factor that takes into account the non-centricity of the molecules. The acentric factor was introduced by Pitzer [15] and is defined as

$$\omega = -\log_{10}\left(\frac{p_s(0.7T_c)}{p_c}\right) - 1.$$

Rearranging Eq. (2.7) as a function of specific volume v yields

$$0 = v^3 + \left(b - \frac{RT}{p}\right)v^2 + \left(\frac{a - RT2b}{p} - 3b^2\right)v - \left(\frac{RTb^2 - ab}{p} + b^3\right).$$

Cubic equations of state can be solved in terms of specific volume analytically, and calculations from (p,T) are therefore possible without any iteration. However, the analytical solution for v(p,T) requires the computation of transcendental functions and is therefore comparatively time consuming. As discussed by Deiters and Macías-Salinas [16], the calculation of v(p,T) with an appropriate iteration scheme is computationally more efficient in most cases.

Table 2: Relationships between thermodynamic properties and thermal equations of state $p(T,v)$ in conjunction with the isobaric heat capacity of the ideal gas $c_p^{\circ}(T)$

Property	Relationship
Specific internal energy u	$u(T,v) = h_0^{\circ} + \int_{T_0}^{T} c_p^{\circ}(T)\,\mathrm{d}T - RT - \int_{v_0}^{v} \left\{ p(T,v) - T\left(\frac{\partial p}{\partial T}\right)_v \right\}_T \mathrm{d}v$
Specific enthalpy h	$h(T,v) = h_0^{\circ} + \int_{T_0}^{T} c_p^{\circ}(T)\,\mathrm{d}T + p(T,v)\,v - RT - \int_{v_0}^{v} \left\{ p(T,v) - T\left(\frac{\partial p}{\partial T}\right)_v \right\}_T \mathrm{d}v$
Specific entropy s	$s(T,v) = s_0^{\circ} + \int_{T_0}^{T} \frac{c_p^{\circ}(T)}{T}\,\mathrm{d}T - R\ln\left(\frac{T}{T_0}\right) + R\ln\left(\frac{v}{v_0}\right) - \int_{v_0}^{v} \left\{ \frac{R}{v} - \left(\frac{\partial p}{\partial T}\right)_v \right\}_T \mathrm{d}v$
Specific isochoric heat capacity c_v	$c_v(T,v) = c_p^{\circ}(T) - R + \int_{\infty}^{v} \left\{ T\left(\frac{\partial^2 p}{\partial T^2}\right)_v \right\}_T \mathrm{d}v$
Specific isobaric heat capacity c_p	$c_p(T,v) = c_v(T,v) - T\frac{\left(\frac{\partial p}{\partial T}\right)_v^2}{\left(\frac{\partial p}{\partial v}\right)_T}$
Speed of sound w	$w(T,v) = -v\sqrt{\frac{c_p(T,v)}{c_v(T,v)}\left(\frac{\partial p}{\partial v}\right)_T}$

The deviations in the specific volume calculated from the Peng-Robinson equation of state from the real fluid behavior of water, *i.e.*, IAPWS-95 [1], are shown in Fig. 3. The deviations in the isobaric heat capacity, calculated from the Peng-Robinson equation in conjunction with the isobaric heat capacity of the ideal gas (see Table 2) and from IAPWS-95, are shown in Fig. 4. In the range of state of modern steam power cycles, these deviations are considerable and lead to incorrect simulation results.

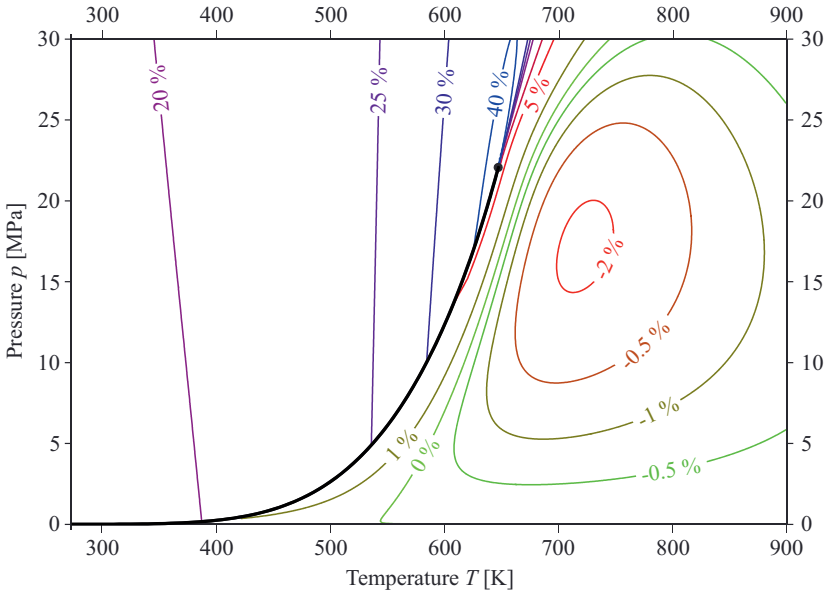

Figure 3: Relative deviation in specific volume v of the Peng-Robinson equation of state from the reference equation of state IAPWS-95 [1].

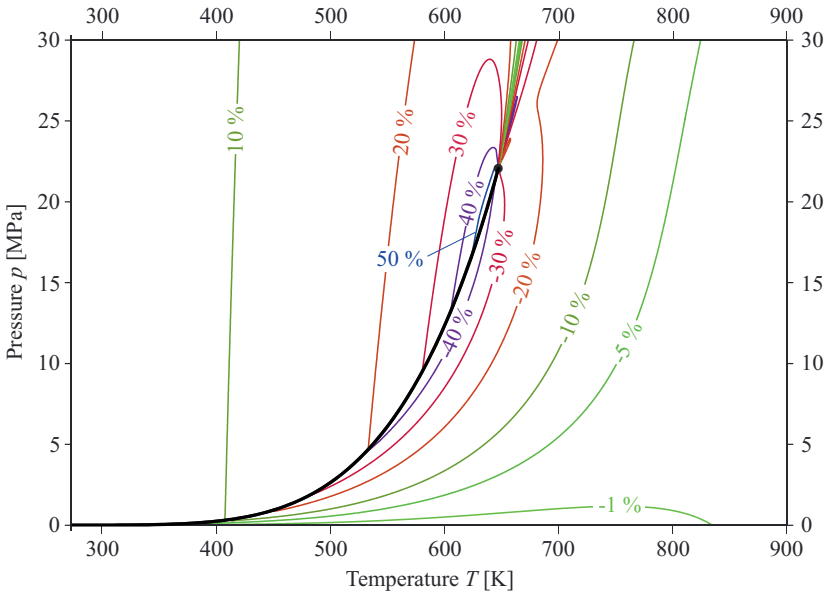

Figure 4: Relative deviation in specific isobaric heat capacity c_p of the Peng-Robinson equation of state from the reference equation of state IAPWS-95 [1].

Since cubic equations of state only require knowledge of the critical parameters and for some versions the acentric factor, they are available for many fluids. Therefore, cubic equations of state are frequently applied in chemical engineering. Due to their comparatively short computing times, these equations of state are often applied in CFD simulations. A comparative study of the computing times is given in Sec. 2.5.3.

2.2 Fundamental Equations of State

For quasi-static reversible processes, the first law of thermodynamics describes the relation of infinitesimal heat flux and infinitesimal volumetric work supplied to a closed system with its infinitesimal change in internal energy as

$$\delta q - p\mathrm{d}v = \mathrm{d}u \,. \tag{2.9}$$

From the second law of thermodynamics, the infinitesimal heat flux of the reversible process at the temperature T yields

$$\delta q = T\mathrm{d}s \,. \tag{2.10}$$

Inserting Eq. (2.10) into Eq. (2.9) yields

$$\mathrm{d}u = T\mathrm{d}s - p\mathrm{d}v \,. \tag{2.11}$$

Comparing Eq. (2.11) to the total differential of $u(v,s)$, namely

$$\mathrm{d}u = \left(\frac{\partial u}{\partial v}\right)_s \mathrm{d}v + \left(\frac{\partial u}{\partial s}\right)_v \mathrm{d}s \,, \tag{2.12}$$

shows that

$$\left(\frac{\partial u}{\partial v}\right)_s = -p \quad \text{and} \quad \left(\frac{\partial u}{\partial s}\right)_v = T \,. \tag{2.13, 2.14}$$

Since the partial derivatives in the total differential Eq. (2.12) are equal to full thermodynamic properties, the full information to describe the thermodynamic state of the system is contained in $u(v,s)$. Therefore, $u(v,s)$ is a thermodynamic potential from which all other thermodynamic properties can be derived.

For the conjugate pairs of variables (p,v) or (T,s) in Eq. (2.11), the Legendre transformation can be applied to obtain other thermodynamic potentials. For instance, the product rule for $\mathrm{d}(pv)$ reads

$$\mathrm{d}(pv) = p\mathrm{d}v + v\mathrm{d}p \,. \tag{2.15}$$

Adding both sides of the Eqs. (2.11) – (2.15) yields

$$\mathrm{d}u + \mathrm{d}(pv) = T\mathrm{d}s + v\mathrm{d}p \,, \tag{2.16}$$

and finally, with applying the addition rule and defining the specific enthalpy as $h = u + pv$, to

$$\mathrm{d}h = T\mathrm{d}s + v\mathrm{d}p \,. \tag{2.17}$$

By applying the Legendre transformation analogously to Eq. (2.11),

$$\mathrm{d}f = -p\mathrm{d}v - s\mathrm{d}T \tag{2.18}$$

can be derived, where the specific Helmholtz free energy is defined as $f = u - Ts$. Applying the Legendre transformation to Eq. (2.18) to switch the conjugate pair of variables (p,v), the following differential equation is obtained

$$\mathrm{d}g = v\mathrm{d}p - s\mathrm{d}T\,, \tag{2.19}$$

where the specific Gibbs free energy is given as $g = h - Ts$.

The differential equations, Eqs. (2.11), (2.17), (2.18), and (2.19), show that the functions

$u(v,s)$, $h(p,s)$, $f(T,v)$, and $g(p,T)$,

describe thermodynamic potentials. These functions are called fundamental equations of state. In contrast to thermal equations of state (see Sec. 2.1), the calculation of thermodynamic properties from a fundamental equation of state only requires its derivatives, but not integrals. Among the independent variables of the equations above, only p, T, and v are accessible through direct measurements. Therefore, fundamental equations of state are established for either $f(T,v)$ or $g(p,T)$. At the liquid-vapor phase boundary, the derivatives of $g(p,T)$, *i.e.*, $\left(\partial g/\partial p\right)_T = v$ and $\left(\partial g/\partial T\right)_p = -s$, are discontinuous. Therefore, it is not possible to cover the entire fluid region with a single equation for $g(p,T)$. Nevertheless, fundamental equations explicit in the Gibbs free energy g have been established in some cases to describe the liquid phase and the vapor phase independently.

In process simulations, the solution of mass, energy, and entropy balance equations often leads to (v,u), (p,h), (p,s), or (h,s) inputs for fluid property calculations. Therefore, the development of fundamental equations of state with these pairs of independent variables from a preliminary Helmholtz equation $f(T,v)$ has been pursued, for instance by P. G. Hill for $s(p,h)$ (unpublished). The functional form of such an equation would be comparable to that of state of the art Helmholtz equations. For independent variables other than those of the chosen fundamental equation of state, property calculations would still require iterative procedures.

2.2.1 Reference Equations of State

Reference equations of state represent the measurements they are based on to within their uncertainties. For water and steam, the scientific formulation IAPWS-95 [1, 2] is available. Its fundamental equation is expressed as the reduced Helmholtz free energy $\Phi = f/(RT)$ as a function of reduced density $\delta = \rho/\rho_\mathrm{c}$ and inverse reduced temperature $\tau = T_\mathrm{c}/T$ and reads

$$\Phi(\delta,\tau) = \Phi^\mathrm{o}(\delta,\tau) + \Phi^\mathrm{r}(\delta,\tau)\,, \tag{2.20}$$

where the ideal part is

$$\Phi^\mathrm{o}(\delta,\tau) = \ln(\delta) + n_1^\mathrm{o} + n_2^\mathrm{o}\tau + n_3^\mathrm{o}\ln(\tau) + \sum_{i=4}^{8} n_i^\mathrm{o}\ln\left[1-\exp(-\gamma_i^\mathrm{o}\tau)\right] \tag{2.21}$$

and the residual part is

$$\begin{aligned}\Phi^\mathrm{r}(\delta,\tau) = {} & \sum_{i=1}^{7} n_i\delta^{d_i}\tau^{t_i} + \sum_{i=8}^{51} n_i\delta^{d_i}\tau^{t_i}\exp\left(-\delta^{c_i}\right) \\ & + \sum_{i=52}^{54} n_i\delta^{d_i}\tau^{t_i}\exp\left(-\alpha_i(\delta-\varepsilon_i)^2 - \beta_i(\tau-\gamma_i)^2\right) + \sum_{i=55}^{56} n_i\Delta^{b_i}\delta\psi\end{aligned}\,, \tag{2.22}$$

with

$$\Delta = \theta^2 + B_i\left[(\delta - 1)^2\right]^{a_i}, \qquad \theta = (1-\tau) + A_i\left[(\delta-1)^2\right]^{\frac{1}{2\beta_i}},$$

$$\psi = \exp\left(-C_i(\delta-1)^2 - D_i(\tau-1)^2\right).$$

The parameters for Eqs. (2.21) and (2.22) are given in [1, 2]. The relationships of thermodynamic properties to the reduced Helmholtz free energy are given in Table 3.

Table 3: Relationships of thermodynamic properties to the reduced Helmholtz free energy Φ and its derivatives

Property	Relationship
Pressure p $p = \rho^2 \left(\partial f/\partial\rho\right)_T$	$\frac{p(\delta,\tau)}{\rho RT} = 1 + \delta\Phi^{\mathrm{r}}_{\delta}$
Specific internal energy u $u = f - T\left(\partial f/\partial T\right)_\rho$	$\frac{u(\delta,\tau)}{RT} = \tau\left(\Phi^{\mathrm{o}}_{\tau} + \Phi^{\mathrm{r}}_{\tau}\right)$
Specific enthalpy h $h = f - T\left(\partial f/\partial T\right)_\rho + \rho\left(\partial f/\partial\rho\right)_T$	$\frac{h(\delta,\tau)}{RT} = 1 + \delta\Phi^{\mathrm{r}}_{\delta} + \tau\left(\Phi^{\mathrm{o}}_{\tau} + \Phi^{\mathrm{r}}_{\tau}\right)$
Specific entropy s $s = -\left(\partial f/\partial T\right)_\rho$	$\frac{s(\delta,\tau)}{R} = \tau\left(\Phi^{\mathrm{o}}_{\tau} + \Phi^{\mathrm{r}}_{\tau}\right) - \Phi^{\mathrm{o}} - \Phi^{\mathrm{r}}$
Specific isochoric heat capacity c_v $c_v = \left(\partial u/\partial T\right)_\rho$	$\frac{c_v(\delta,\tau)}{R} = -\tau^2\left(\Phi^{\mathrm{o}}_{\tau\tau} + \Phi^{\mathrm{r}}_{\tau\tau}\right)$
Specific isobaric heat capacity c_p $c_p = \left(\partial h/\partial T\right)_p$	$\frac{c_p(\delta,\tau)}{R} = -\tau^2\left(\Phi^{\mathrm{o}}_{\tau\tau} + \Phi^{\mathrm{r}}_{\tau\tau}\right) + \frac{\left(1 + \delta\Phi^{\mathrm{r}}_{\delta} - \delta\tau\Phi^{\mathrm{r}}_{\delta\tau}\right)^2}{1 + 2\delta\Phi^{\mathrm{r}}_{\delta} + \delta^2\Phi^{\mathrm{r}}_{\delta\delta}}$
Speed of sound w $w = \sqrt{\left(\partial p/\partial\rho\right)_s}$	$\frac{w^2(\delta,\tau)}{RT} = 1 + 2\delta\Phi^{\mathrm{r}}_{\delta} + \delta^2\Phi^{r}_{\delta\delta} - \frac{\left(1 + \delta\Phi^{\mathrm{r}}_{\delta} - \delta\tau\Phi^{\mathrm{r}}_{\delta\tau}\right)^2}{\tau^2\left(\Phi^{\mathrm{o}}_{\tau\tau} + \Phi^{\mathrm{r}}_{\tau\tau}\right)}$

The uncertainties of IAPWS-95 in specific volume v and specific isobaric heat capacity c_p for the region considered in Secs. 2.1.1 and 2.1.2 are given in Figs. 5 and 6.

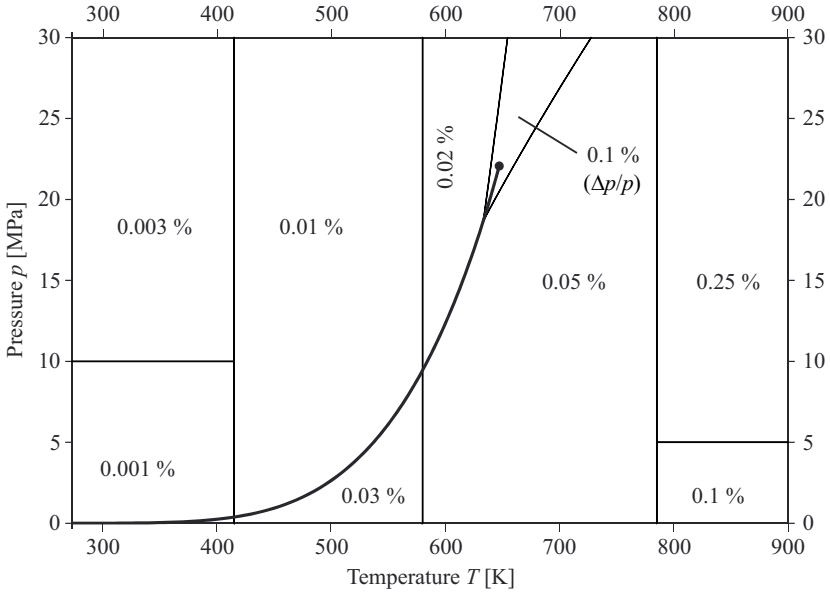

Figure 5: Uncertainties in specific volume v as estimated for IAPWS-95 [1].

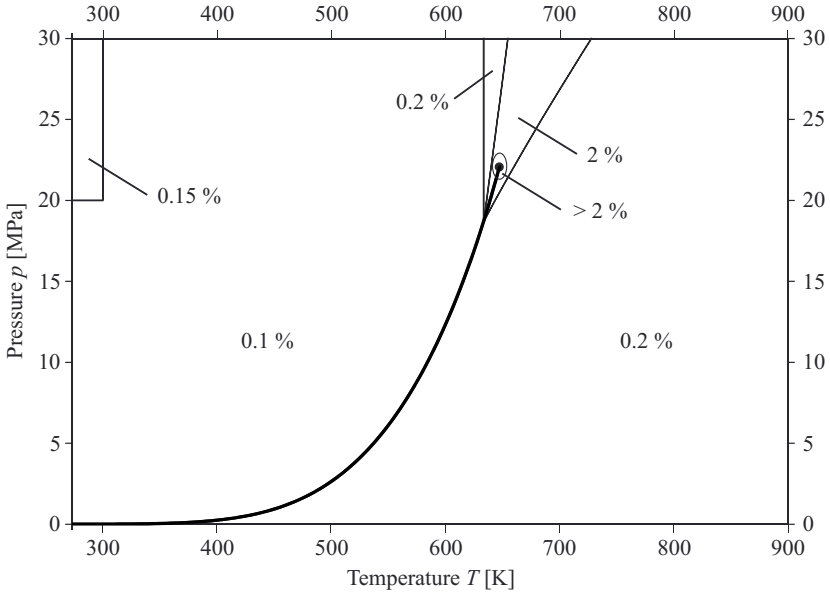

Figure 6: Uncertainties in specific isobaric heat capacity c_p as estimated for IAPWS-95 [1].

The residual part, Eq. (2.22), contains 7 polynomial terms, 44 exponential terms, 3 Gaussian bell shaped terms, and 2 non-analytical terms. The large number of transcendental functions, such as exponential functions and logarithms, not only in Eq. (2.22) but also in Eq. (2.21), causes long computing times as discussed in Sec. 2.5. Therefore, the computing speed of such complex reference equations of state is often much too slow for extensive process simulations.

2.2.2 Short Fundamental Equations of State for Industrial Applications

For various fluids, short fundamental equation of state have been published by Span and Wagner [17, 18, 19], Lemmon and Span [20], and Kunz et al. [21]. In accordance with Eq. (2.20), the short fundamental equation of state for water of Kunz et al. [21] consists of the ideal part

$$\Phi^{\mathrm{o}}(\delta,\tau) = \ln(\delta) + n_1^{\mathrm{o}} + n_2^{\mathrm{o}}\tau + n_3^{\mathrm{o}}\ln(\tau) + \sum_{k=4,6} n_k^{\mathrm{o}} \ln\left|\sinh\left(\vartheta_k^{\mathrm{o}}\tau\right)\right| - \sum_{k=5,7} n_k^{\mathrm{o}} \ln\left|\cosh\left(\vartheta_k^{\mathrm{o}}\tau\right)\right| \tag{2.23}$$

and the residual part

$$\Phi^{\mathrm{r}}(\delta,\tau) = \sum_{i=1}^{7} n_i \delta^{d_i}\tau^{t_i} + \sum_{i=8}^{16} n_i \delta^{d_i}\tau^{t_i} \exp\left(-\delta^{c_i}\right). \tag{2.24}$$

The ideal part, Eq. (2.23), was derived from the isobaric heat capacity equation given by Jaeschke and Schley [22]. The parameters of the residual part, Eq. (2.24), were fitted to data calculated from IAPWS-95. The parameters of both equations are given in [21].

In contrast to the reference fundamental equations of state, the residual parts of these short equations contain 12-16 terms only. Therefore, their accuracies are not as high as those of reference equations of state (see Sec. 2.2.1), but sufficient for many industrial applications. According to Lemmon and Span [20], computations from the short fundamental equations of state in comparison to reference equations of state are 2-10 times faster. A comparative study of the computing times is given in Sec. 2.5.3.

2.2.3 Fast Fundamental Equations for Separate Regions

The industrial formulation for water and steam IAPWS-IF97 [3, 4] consists of separate fundamental equations of state for the liquid region 1, the vapor region 2, the critical and supercritical region 3, and the high-temperature region 5 as depicted in Fig. 7. These so-called basic equations were fitted to data calculated from IAPWS-95. For example, the basic equation for region 2 represents the reduced Gibbs free energy $\gamma = g/(RT)$ and reads

$$\gamma(\pi,\tau) = \gamma^{\mathrm{o}}(\pi,\tau) + \gamma^{\mathrm{r}}(\pi,\tau), \tag{2.25}$$

where the ideal-gas part is given by

$$\gamma^{\mathrm{o}}(\pi,\tau) = \ln(\pi) + \sum_{i=1}^{9} n_i^{\mathrm{o}} \tau^{J_i^{\mathrm{o}}}, \tag{2.26}$$

and the residual part is

$$\gamma^{\mathrm{r}}(\pi,\tau)=\sum_{i=1}^{43} n_i \pi^{I_i}(\tau-0.5)^{J_i}. \tag{2.27}$$

In Eqs. (2.25) – (2.27), the reduced variables are defined as $\pi = p/1\mathrm{MPa}$ and $\tau = 540\,\mathrm{K}/T$. The parameters of Eqs. (2.26) and (2.27) are given in [3, 4]. The relationships of thermodynamic properties to the reduced Gibbs free energy are given in Table 4.

The uncertainties of IAPWS-IF97 in specific volume v and specific isobaric heat capacity c_p for the region considered in Secs. 2.1.1, 2.1.2, and 2.2.1 are given in Figs. 8 and 9.

Among the terms in Eqs. (2.26) and (2.27), the natural logarithm in Eq. (2.26) is the only transcendental function. This term vanishes for each derivative with respect to π or becomes algebraic for each derivative with respect to τ. Thus, IAPWS-IF97 allows for a computationally efficient implementation as discussed in Sec. 2.5.3.

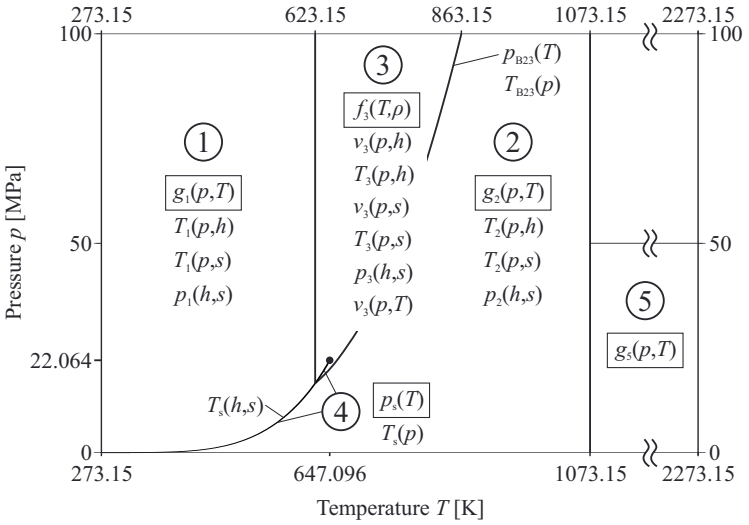

Figure 7: Regions, basic equations, and backward equations of IAPWS-IF97.

Table 4: Relationships of thermodynamic properties to the reduced Gibbs free energy γ and its derivatives

Property	Relationship
Specific volume v $v = \left(\partial g/\partial p\right)_T$	$v(\pi,\tau)\dfrac{p}{RT} = \pi\left(\gamma_\pi^{\mathrm{o}} + \gamma_\pi^{\mathrm{r}}\right)$
Specific internal energy u $u = g - T\left(\partial g/\partial T\right)_p - p\left(\partial g/\partial p\right)_T$	$\dfrac{u(\pi,\tau)}{RT} = \tau\left(\gamma_\tau^{\mathrm{o}} + \gamma_\tau^{\mathrm{r}}\right) - \pi\left(\gamma_\pi^{\mathrm{o}} + \gamma_\pi^{\mathrm{r}}\right)$
Specific enthalpy h $h = g - T\left(\partial g/\partial T\right)_p$	$\dfrac{h(\pi,\tau)}{RT} = \tau\left(\gamma_\tau^{\mathrm{o}} + \gamma_\tau^{\mathrm{r}}\right)$
Specific entropy s $s = -\left(\partial g/\partial T\right)_p$	$\dfrac{u(\pi,\tau)}{R} = \tau\left(\gamma_\tau^{\mathrm{o}} + \gamma_\tau^{\mathrm{r}}\right) - \left(\gamma^{\mathrm{o}} + \gamma^{\mathrm{r}}\right)$
Specific isochoric heat capacity c_v $c_v = \left(\partial u/\partial T\right)_v$	$\dfrac{c_v(\pi,\tau)}{R} = -\tau^2\left(\gamma_{\tau\tau}^{\mathrm{o}} + \gamma_{\tau\tau}^{\mathrm{r}}\right) - \dfrac{\left(1 + \pi\gamma_\pi^{\mathrm{r}} - \pi\tau\gamma_{\pi\tau}^{\mathrm{r}}\right)^2}{1 - \pi^2\gamma_{\pi\pi}^{\mathrm{r}}}$
Specific isobaric heat capacity c_p $c_p = \left(\partial h/\partial T\right)_p$	$\dfrac{c_p(\pi,\tau)}{R} = -\tau^2\left(\gamma_{\tau\tau}^{\mathrm{o}} + \gamma_{\tau\tau}^{\mathrm{r}}\right)$
Speed of sound w $w = v\sqrt{-\left(\partial p/\partial v\right)_s}$	$\dfrac{w^2(\pi,\tau)}{RT} = \dfrac{1 + 2\pi\gamma_\pi^{\mathrm{r}} + \left(\pi\gamma_\pi^{\mathrm{r}}\right)^2}{\left(1 - \pi^2\gamma_{\pi\pi}^{\mathrm{r}}\right) + \dfrac{\left(1 + \pi\gamma_\pi^{\mathrm{r}} - \pi\tau\gamma_{\pi\tau}^{\mathrm{r}}\right)^2}{\tau^2\left(\gamma_{\tau\tau}^{\mathrm{o}} + \gamma_{\tau\tau}^{\mathrm{r}}\right)}}$

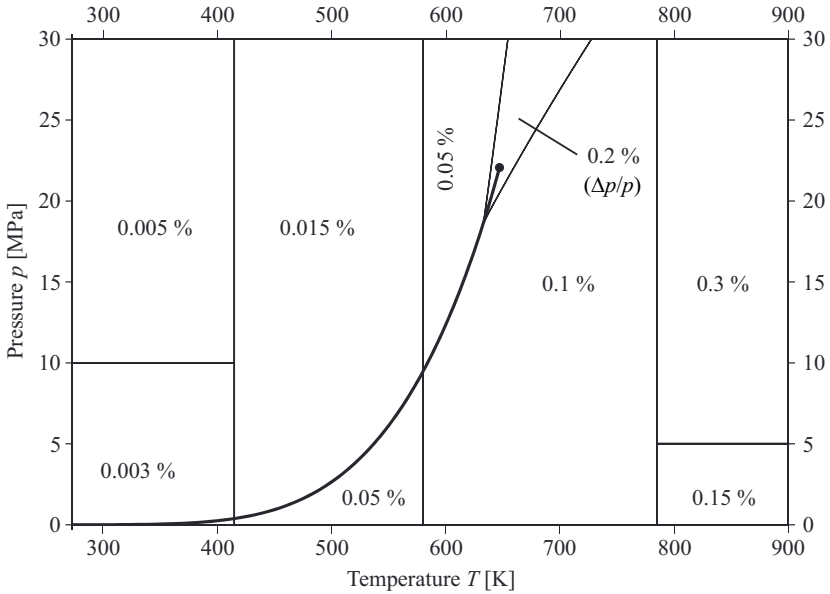

Figure 8: Uncertainties in specific volume v as estimated for IAPWS-IF97 [3].

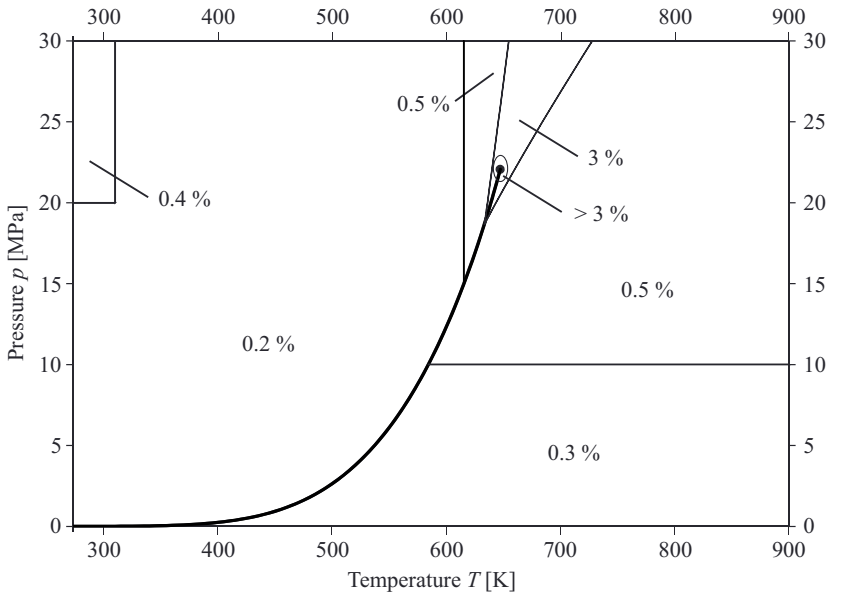

Figure 9: Uncertainties in specific isobaric heat capacity c_p as estimated for IAPWS-IF97 [3].

2.3 Backward Equations

In the steam power industry, the calculation of heat cycles and their components, such as steam generators, turbines, condensers, and pumps, frequently requires property functions of (p,h), (p,s), and (h,s). To calculate property functions from these pairs of variables without time-consuming iterations, IAPWS-IF97 and its supplementary releases [3, 4, 5, 6, 7] provide so-called backward equations. Additionally, a set of backward and auxiliary equations for $v(p,T)$ [8] is provided for the critical and supercritical region 3. The structure of IAPWS-IF97 along with its backward equations is depicted in Fig. 7. For the calculation of $T(p,h)$ in region 2 of IAPWS-IF97, this region is divided into the three subregions 2a, 2b, and 2c, as depicted in Fig. 10. The corresponding backward equations are given by

$$\frac{T_{2a}(p,h)}{T^*} = \theta(\pi,\eta) = \sum_{i=1}^{34} n_i \pi^{I_i} (\eta - 2.1)^{J_i}, \tag{2.28}$$

$$\frac{T_{2b}(p,h)}{T^*} = \theta(\pi,\eta) = \sum_{i=1}^{38} n_i (\pi - 2)^{I_i} (\eta - 2.6)^{J_i}, \tag{2.29}$$

and

$$\frac{T_{2c}(p,h)}{T^*} = \theta(\pi,\eta) = \sum_{i=1}^{23} n_i (\pi + 25)^{I_i} (\eta - 1.8)^{J_i}. \tag{2.30}$$

The boundary between the subregions 2b and 2c is given by

$$\frac{p_{2bc}(h)}{p^*} = \pi(\eta) = n_1 + n_2\eta + n_3\eta^2. \tag{2.31}$$

The parameters of Eqs. (2.28) – (2.31) are given in [3, 4]. The maximum (max) and root-mean-square (RMS) inconsistencies between Eqs. (2.28) – (2.30) and the basic equation $g_2(p,T)$ of IAPWS-IF97 region 2, Eq. (2.25), along with the permissible values (perm), are given in Table 5.

Table 5: Maximum (max) and root-mean-square (RMS) inconsistencies between the backward equations $T_{2a}(p,h)$, $T_{2b}(p,h)$, and $T_{2c}(p,h)$, Eqs. (2.28) – (2.30), and the basic equation $g_2(p,T)$ of IAPWS-IF97 region 2, Eq. (2.25), along with the permissible values (perm) according to [3]

Backward Eq.	$\lvert\Delta T\rvert_{perm}$ [mK]	$\lvert\Delta T\rvert_{max}$ [mK]	$\lvert\Delta T\rvert_{RMS}$ [mK]
$T_{2a}(p,h)$	10	9.3	2.9
$T_{2b}(p,h)$	10	9.6	3.9
$T_{2c}(p,h)$	25	23.7	10.4

Region determinations for (h,s) inputs would normally require iterative calculations from the basic equations along the region boundaries. Similarly, the region boundary between region 3 and region 4 would require iterative calculations for (p,h) and (p,s) inputs. To avoid these time consuming iterations, so-called region-boundary equations are provided in [6] and [7].

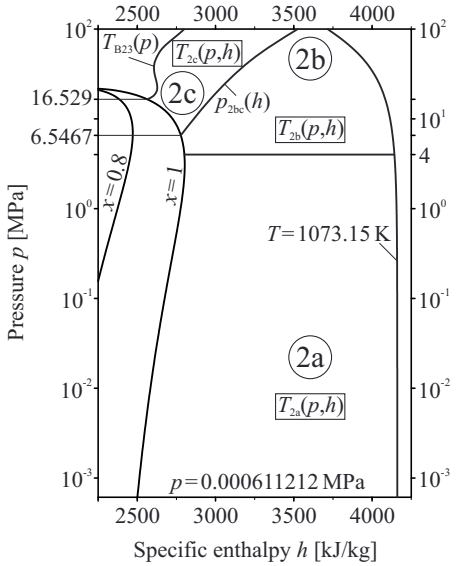

Figure 10: Subregions 2a, 2b, and 2c with their corresponding backward equations $T(p,h)$.

For numerical simulations of transient processes, for instance in CFD, the requirements regarding numerical consistency of the applied property functions are extremely high. Investigations revealed that the numerical consistency of the existing backward equations with their corresponding fundamental equations is insufficient for numerical simulations with very small spatial and time discretizations and can lead to convergence problems. In these situations, inverse property functions must be calculated by iteration from the corresponding fundamental equation with starting values determined from the available backward equations. Where applicable, the region boundaries must be calculated by iteration from their original definitions in the (p,T) plane. The starting values for these calculations are obtained from the available region-boundary equations.

2.4 Table Look-Up Methods

To calculate fluid properties with reasonable accuracy but with short computing times, so-called table look-up methods are frequently applied. For these methods, the desired fluid properties are determined from previously tabulated values. If the tabulated data originates from an accurate equation and no additional smoothing of these data is required, then interpolation methods can be applied. For tabulated measurement values that have not been pre-correlated, approximation methods can be used to obtain a smooth representation of these values. This is discussed for spline approximation methods in Secs. 2.4.1.3 and 2.4.2.3. To describe the thermophysical properties of pure fluids and mixtures at constant composition, functions of one and two independent variables are required. Phase boundaries, such as $T_s(p)$, are described with functions of the form $z(x_1)$, whereas the properties in the single-phase region, such as

T(*p*,*h*), are represented with functions of the form $z(x_1, x_2)$. The basic principles of several table look-up methods for one- and two-dimensional functions are explained in Secs. 2.4.1 and 2.4.2.

2.4.1 One-Dimensional Functions

If a table look-up method is applied to reproduce the results of an underlying function $z(x_1)$, then a series of discrete data points, the so-called nodes, can be defined arbitrarily along x_1. In order to enhance the accuracy of any interpolation or approximation method, it is advantageous to linearize the function to be interpolated first by means of suitable variable transformations. This is illustrated in Figs. 11 and 12. Both the independent variable x_1 and the dependent variable *z* can be transformed into $\bar{x}_1$ and $\bar{z}$. The number of nodes *I* and their $\bar{x}_{1,i}$ locations are chosen to ensure the desired accuracy. For some methods, the nodes are clustered in regions with strong curvature to achieve the required accuracy and to minimize the number of nodes over the domain of definition. For other methods, the nodes are distributed equidistantly along $\bar{x}_1$ as shown in Fig. 12. The $\bar{z}_i$ values of the nodes (*i*) are calculated from the transformed underlying equation $\bar{z}(\bar{x}_1)$. The values between the nodes are interpolated by means of polynomials $\bar{z}_{\{i\}}(\bar{x}_1)$, which are locally defined in each interval $\{i\}$. For the methods described below, an interval $\{i\}$ usually ranges between two nodes. In order to interpolate or approximate values from a given series of nodes, the interval $\{i\}$ must be determined first. For non-equidistant nodes, a search algorithm is required to identify the interval index *i*. For equidistant nodes along $\bar{x}_1$, an extensive search algorithm can be avoided and the interval index *i* is simply calculated from

$$i = \text{floor}\left(\frac{\bar{x}_1 - \bar{x}_{1,1}}{\Delta\bar{x}_1}\right). \tag{2.32}$$

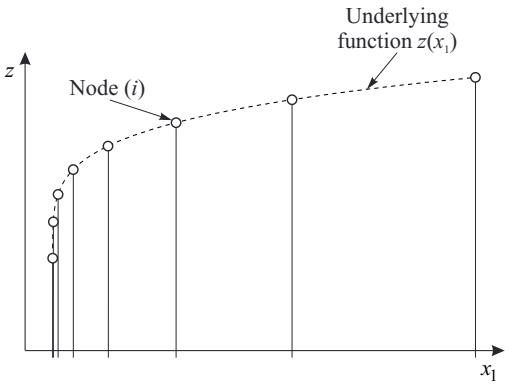

Figure 11: Untransformed underlying function $z(x_1)$ with nodes equidistant in $\bar{x}_1$ rather than in x_1.

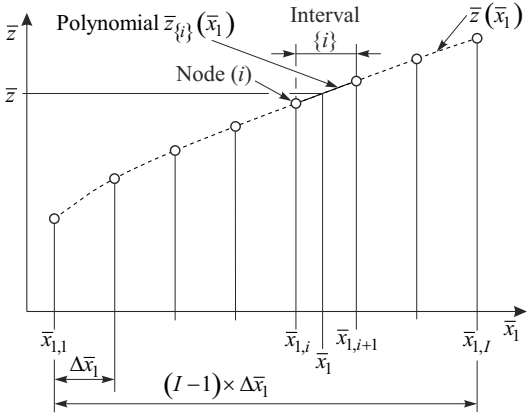

Figure 12: Transformed underlying function $\bar{z}(\bar{x}_1)$ with nodes equidistant in $\bar{x}_1$, and interpolating polynomial $\bar{z}_{\{i\}}(\bar{x}_1)$ in the interval $\{i\}$.

2.4.1.1 Local Polynomial Interpolation

Local polynomial interpolation provides simple ways for calculating values from look-up tables. The term "local" means, in this context, that the polynomials are piecewise defined for each interval $\{i\}$, whereas "global" means that a single polynomial interpolates values across the entire domain. For local methods, lower order polynomials can be used. For global methods, the polynomial degrees depend on the number of nodes. Thus, a large number of nodes leads to higher order polynomials that tend to oscillate. This problem can become very dominant if the nodes are distributed equidistantly (Runge's Phenomenon). Therefore global polynomial interpolation methods are not considered here.

In contrast to spline interpolation methods (see Sec. 2.4.1.3), the local polynomials are defined without any additional constraints to the adjacent intervals. Therefore, the continuity of derivatives is limited as discussed later in this section for cubic polynomials. For local polynomial interpolation methods, a polynomial $\bar{z}_{\{i\}}(\bar{x}_1)$ is defined in each interval $\{i\}$ between the nodes (i) and $(i+1)$.

A polynomial of N-th order is given by

$$\bar{z}_{\{i\}}(\bar{x}_1) = \sum_{k=1}^{N+1} a_{ik}\left(\bar{x}_1 - \bar{x}_{1,i}\right)^{k-1}. \tag{2.33}$$

For the linear case, *i.e.*, $N=1$, Eq. (2.33) has two coefficients a_{ik} that can be obtained from two constraints. These constraints can be defined at the boundaries of the interval $\{i\}$ at which Eq. (2.33) intersects the nodes (i) and $(i+1)$. The resulting piecewise linear function is continuous across the interval boundaries, but not continuously differentiable. A mathematical description of linear interpolation methods can be found in [23].

In order to provide a property function with a continuous first derivative, cubic interpolation is often recommended. The four coefficients of the cubic polynomial, Eq. (2.33) with $N=3$, can be determined from given values of $\bar{z}$ and $(\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1)$ at the two bounding nodes of each interval.

In contrast to this local method, a cubic spline function also provides a continuous second derivative as described in Sec. 2.4.1.3.

2.4.1.2 Tabular Taylor Series Expansion Method (TTSE)

The Tabular Taylor Series Expansion method (TTSE) was adopted by IAPWS as a guideline [9, 10] in 2003. For this method, the nodes are typically located at the center of each interval $\{i\}$. From each of these nodes a Taylor series $\overline{z}_{\{i\}}\left(\overline{x}_1\right)$ is expanded. The applied second order Taylor series expansion is given by

$$\overline{z}_{\{i\}}\left(\overline{x}_1\right) = \overline{z}_i + \left(\overline{x}_1 - \overline{x}_{1,i}\right)\left(\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\right)\left(\overline{x}_{1,i}\right) + \frac{1}{2}\left(\overline{x}_1 - \overline{x}_{1,i}\right)^2\left(\frac{\mathrm{d}^2\overline{z}}{\mathrm{d}\overline{x}_1{}^2}\right)\left(\overline{x}_{1,i}\right). \tag{2.34}$$

Eq. (2.34) approximates the underlying function independently for each interval $\{i\}$ without any constraint to the neighboring intervals. The resulting inconsistencies at the boundaries between two adjacent intervals are illustrated in Fig. 13.

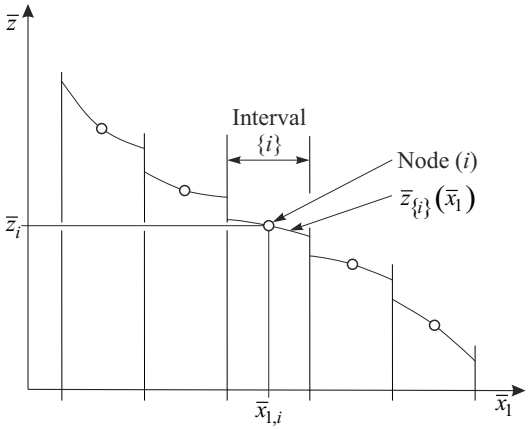

Figure 13: Series of nodes with an interval $\{i\}$ where the Taylor series $\overline{z}_{\{i\}}\left(\overline{x}_1\right)$ is valid.

In [9], the TTSE method is applied to IAPWS-95. The TTSE property function $T_s(p)$ and its numerically consistent inverse function $p_s(T)$ are created on the same series of 174 nodes. The functions are valid in the following pressure and temperature ranges:

612.8 Pa $\leq p \leq$ 22.064 MPa 275 K $\leq T \leq$ 647.096 K.

The maximum deviation of the TTSE property function $T_s(p)$ from IAPWS-95 is 1 mK, except in the vicinity of the critical point, where the maximum deviation is 6 mK. The maximum inconsistency at the interval boundaries is 2 mK. The nodes along with the corresponding derivatives for each property are stored in a look-up table. The nodes are not equidistant along p or T and the desired interval index i is determined by means of the bisection method. The interval index i is stored for subsequent function calls to minimize the effort for the search algorithm for contiguous state points. For random inputs, the search algorithm for the interval index is the dominant factor for the computing time of the TTSE method.

2.4.1.3 Spline Interpolation and Approximation Algorithms

Spline interpolation has its roots in engineering mechanics. The term “spline” refers to a flexible rod that serves as a drafting tool for smooth curves and was introduced in 1946 by Schoenberg [24]. The rod is fixed to a number of points and takes the shape with minimum bending energy. Thus, this shape is considered to be the smoothest possible. The bending line of the thin rod is described by the Euler-Bernoulli bending theory, a second order differential equation, which reads

$$EIz''(x_1) = M(x_1), \tag{2.35}$$

where E is the modulus of elasticity, I is the moment of inertia, z'' is the curvature, and M is the bending moment. A simple example is shown in Figs. 14 and 15.

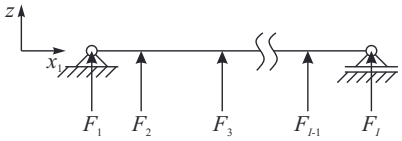

Figure 14: Forces at a thin rod between two supports (example).

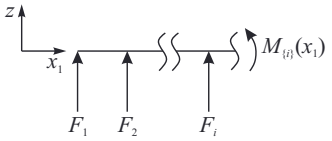

Figure 15: Free body diagram for the example given in Fig. 14.

The bending moment at the location x_1 in the interval $\{i\}$, where $x_{1,i} \le x_1 < x_{1,i+1}$, is derived from the moment equilibrium, see Fig. 15, and reads

$$M_{\{i\}}(x_1) = \sum_{k=1}^{i} F_k (x_1 - x_{1,k}). \tag{2.36}$$

The integration of Eq. (2.35) with Eq. (2.36) for each interval $\{i\}$ yields

$$EIz_{\{i\}}(x_1) = \sum_{k=1}^{i} \frac{F_k}{6}(x_1 - x_{1,k})^3 + C_{\{i\},1}(x_1 - x_{1,k}) + C_{\{i\},2}. \tag{2.37}$$

Substituting $(x_1 - x_{1,k})$ in Eq. (2.37) with $(x_1 - x_{1,i}) + (x_{1,i} - x_{1,k})$ and rearranging for $z_{\{i\}}$ leads to a piecewise-defined function of cubic polynomials for the bending line, which reads

$$z_{\{i\}}(x_1) = \sum_{k=1}^{i} \frac{F_k}{6EI}(x_1 - x_{1,i})^3 + \overline{C}_{\{i\},1}(x_1 - x_{1,i}) + \overline{C}_{\{i\},2}. \tag{2.38}$$

For known forces F_i, $i = 2 \dots I-1$, the support forces F_1 and F_I can be obtained from the force and moment equilibria. The $2(I-1)$ constants, $\overline{C}_{\{i\},1}$ and $\overline{C}_{\{i\},2}$, can then be determined from the conditions

$$z_{\{i=1\}}(x_1 = 0) = 0, \tag{2.39}$$

$$z_{\{i=I-1\}}(x_1 = x_{1,I}) = 0, \tag{2.40}$$

$$z_{\{i\}}(x_{1,i+1}) = z_{\{i+1\}}(x_{1,i+1}) \qquad i = 1, \dots, I-2, \tag{2.41}$$

$$\left.\frac{\mathrm{d}z}{\mathrm{d}x_1}\right|_{\{i\}}\left(x_{1,i+1}\right)=\left.\frac{\mathrm{d}z}{\mathrm{d}x_1}\right|_{\{i+1\}}\left(x_{1,i+1}\right) \qquad i=1,\ldots,I-2\,. \tag{2.42}$$

Because of the continuity of Eq. (2.36) and the chosen conditions Eq. (2.41) and Eq. (2.42), the piecewise-defined function for the bending line is continuously differentiable twice.

Based on the considerations described above, spline interpolation algorithms were developed in the middle of the 20th century. A spline function is a piecewise-defined function, consisting of several spline polynomials that interpolate values between a series of discrete data points, the so-called nodes. The spline polynomials are connected at knots, which can be either equal or unequal to the nodes. In analogy to the piecewise-defined bending line discussed above, cubic spline functions with the knots equal to the nodes were developed first. In between the I nodes, $I-1$ intervals $\{i\}$ are defined as $\bar{x}_{1,i} \le \bar{x}_1 < \bar{x}_{1,i+1}$. In each interval $\{i\}$, a cubic spline polynomial is defined as depicted in Fig. 12. The cubic spline polynomial reads

$$\bar{z}_{\{i\}}\left(\bar{x}_1\right)=\sum_{k=1}^{4} a_{ik}\left(\bar{x}_1-\bar{x}_{1,i}\right)^{k-1}. \tag{2.43}$$

Each of the $I-1$ polynomials $\bar{z}_{\{i\}}\left(\bar{x}_1\right)$ must intersect the adjacent nodes (i) and ($i+1$), so that

$$\bar{z}_{\{i\}}\left(\bar{x}_{1,i}\right)=\bar{z}_i\left(\bar{x}_{1,i}\right) \qquad i=1,\ldots,I-1, \tag{2.44}$$

$$\bar{z}_{\{i\}}\left(\bar{x}_{1,i+1}\right)=\bar{z}_{i+1}\left(\bar{x}_{1,i+1}\right) \qquad i=1,\ldots,I-1, \tag{2.45}$$

At each of the $I-2$ knots between two adjacent polynomials, the first two derivatives with respect to $\bar{x}_1$, $\left(\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1\right)$ and $\left(\mathrm{d}^2\bar{z}/\mathrm{d}\bar{x}_1{}^2\right)$, must also be equal

$$\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i\}}\left(\bar{x}_{1,i+1}\right)=\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i+1\}}\left(\bar{x}_{1,i+1}\right) \qquad i=1,\ldots,I-2\,, \tag{2.46}$$

$$\left.\frac{\mathrm{d}^2\bar{z}}{\mathrm{d}\bar{x}_1{}^2}\right|_{\{i\}}\left(\bar{x}_{1,i+1}\right)=\left.\frac{\mathrm{d}^2\bar{z}}{\mathrm{d}\bar{x}_1{}^2}\right|_{\{i+1\}}\left(\bar{x}_{1,i+1}\right) \qquad i=1,\ldots,I-2\,. \tag{2.47}$$

Two additional conditions are required to determine the $4(I-1)$ unknown coefficients a_{ik} of the $I-1$ spline polynomials. For this purpose, the first derivatives $\left(\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1\right)$ at the endpoints ($i=1$) and ($i=I$) are frequently provided, so that

$$\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i=1\}}\left(\bar{x}_{1,1}\right)=\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\left(\bar{x}_{1,1}\right), \tag{2.48}$$

$$\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i=I-1\}}\left(\bar{x}_{1,I}\right)=\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\left(\bar{x}_{1,I}\right). \tag{2.49}$$

Alternatively, these derivatives can be approximated. The $4(I-1)$ unknown coefficients a_{ik} of the $I-1$ spline polynomials are obtained by solving the linear system of equations Eqs. (2.44) – (2.49). A comprehensive discussion of cubic spline interpolation algorithms with various constraints is given by Späth [23]. Computationally efficient implementations of these methods are also given in [23].

In 1962, Landis and Nilson published an article [28] on cubic spline functions and their application for determining thermodynamic derivatives from tabulated values. They established cubic spline functions for the reduced internal energy u/RT as a function of temperature T along isochores. From these spline functions, values of c_v/R were calculated by analytical differentiation. In addition, a least-squares spline-approximation technique is discussed that can be employed to fit smooth spline curves to experimental data. Details of such a spline-approximation technique were published by Klaus and van Ness [29] in 1967. The underlying cubic spline interpolation method for this approximation technique is similar to the algorithm given by Eqs. (2.43) – (2.49). Therefore, the actual equations provided in [29] are not repeated here, but the similarities and differences to the cubic spline interpolation algorithm given above are discussed.

The $I-1$ polynomials approximate $N>I$ fixed data points $\hat{\bar{z}}_n\left(\hat{\bar{x}}_{1,n}\right)$ as shown in Fig. 16. The number of intervals $I-1$ is an arbitrary choice and the $\bar{x}_1$ location of each inner knot i coincides with the $\bar{x}_1$ coordinate of a data point $n(i)$. Hence, in addition to the $4(I-1)$ coefficients a_{ik}, the $\bar{z}$ positions of the I knots are unknown.

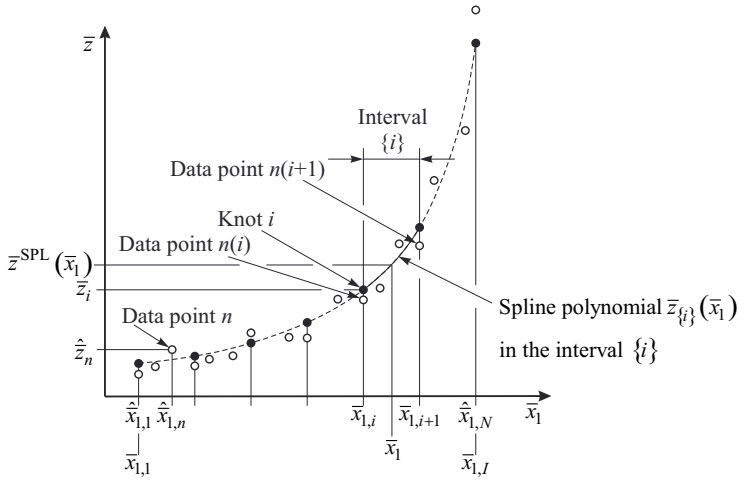

Figure 16: Data points to be approximated along with the series of knots and the interval $\{i\}$, where the spline polynomial $\bar{z}_{\{i\}}\left(\bar{x}_1\right)$ is valid.

The adjacent polynomials must fulfill the $3(I-2)$ conditions given by Eqs. (2.46), (2.47), and

$$\bar{z}_{\{i\}}\left(\bar{x}_{1,i+1}\right)=\bar{z}_{\{i+1\}}\left(\bar{x}_{1,i+1}\right) \qquad i=1,\ \ldots\ ,I-2\,. \tag{2.50}$$

Furthermore, the sum of squares

$$\sigma^2=\sum_{n=1}^{N}\left(\hat{\bar{z}}_n\left(\hat{\bar{x}}_{1,n}\right)-\bar{z}^{\mathrm{SPL}}\left(\hat{\bar{x}}_{1,n}\right)\right)^2 \tag{2.51}$$

is to be minimized. The method of Lagrange's underdetermined multipliers [30] is used to solve this problem. For the $M = 4(I-1) + I$ unknowns and the $J = 3(I-2)$ constraints, the M additional equations

$$\frac{\partial F}{\partial X_m} + \sum_{j=1}^{J} \lambda_j \frac{\partial \phi_j}{\partial X_m} = 0 \qquad m = 1, \ldots, M \tag{2.52}$$

are formulated. In the M Eqs. (2.52), F is the function to be minimized, *i.e.*, Eq. (2.51), and X_m represents one of the $4(I-1)$ unknown coefficients a_{ik} or one of the unknown values of z_i at the I knots. The constraints ϕ_j correspond to the Eqs. (2.46), (2.47), and (2.50), which have to be given in the form $\phi_j = 0$. The J unknown Lagrange multipliers λ_j increase the total number of unknowns to $M+J$. The system of equations to be solved simultaneously is then given by the J constraints themselves, *i.e.*, Eqs. (2.46), (2.47), and (2.50), and the M Eqs. (2.52). In [29], the additional assumptions

$$\mathrm{d}^2\overline{z}_{\{i=1\}} \Big/ \mathrm{d}\overline{x}_1^{\,2}\left(\overline{x}_{1,1}\right) = \mathrm{d}^2\overline{z}_{\{i=1\}} \Big/ \mathrm{d}\overline{x}_1^{\,2}\left(\overline{x}_{1,2}\right), \tag{2.53}$$

and

$$\mathrm{d}^2\overline{z}_{\{i=I-1\}} \Big/ \mathrm{d}\overline{x}_1^{\,2}\left(\overline{x}_{1,I}\right) = \mathrm{d}^2\overline{z}_{\{i=I-1\}} \Big/ \mathrm{d}\overline{x}_1^{\,2}\left(\overline{x}_{1,I-1}\right) \tag{2.54}$$

were made.

Another spline-approximation approach with cubic polynomials was presented by Schot [31] in 1968. Again, the underlying cubic spline interpolation method is similar to the algorithm given by Eqs. (2.43) – (2.49). Therefore, a mathematically equivalent description of the method can be provided with Eqs. (2.43), (2.46), (2.47), (2.50), and (2.52). For this method, the number and the $\overline{x}_1$ locations of the knots is equal to those of the given data points, *i.e.*, $I = N$ and $\overline{x}_{1,i} = \hat{\overline{x}}_{1,i}$. The sum of squares to be minimized is given by

$$\sigma^2 = \sum_{i=1}^{I} \left(\hat{\overline{z}}_i\left(\hat{\overline{x}}_{1,i}\right) - \overline{z}^{\mathrm{SPL}}\left(\hat{\overline{x}}_{1,i}\right)\right)^2 + w \sum_{i=2}^{I-1} \left(\frac{\Delta^2 \hat{\overline{z}}_i}{\Delta \hat{\overline{x}}_{1,i}^{\,2}} - \frac{\mathrm{d}^2\overline{z}^{\mathrm{SPL}}}{\mathrm{d}\hat{\overline{x}}_1^{\,2}}\left(\hat{\overline{x}}_{1,i}\right)\right)^2, \tag{2.55}$$

where

$$\frac{\Delta^2 \hat{\overline{z}}_i}{\Delta \hat{\overline{x}}_{1,i}^{\,2}} = \frac{2}{\hat{\overline{x}}_{1,i+1} - \hat{\overline{x}}_{1,i-1}} \left(\frac{\hat{\overline{z}}_{i+1} - \hat{\overline{z}}_i}{\hat{\overline{x}}_{1,i+1} - \hat{\overline{x}}_{1,i}} - \frac{\hat{\overline{z}}_i - \hat{\overline{z}}_{i-1}}{\hat{\overline{x}}_{1,i} - \hat{\overline{x}}_{1,i-1}} \right). \tag{2.56}$$

For $w = 0$, the spline function is fitted to the given data points only. For $w > 0$, the differences in the second derivatives from the second difference quotients are additionally minimized, which reduces possible fluctuations in the resulting spline curve.

The spline approximation methods discussed above were developed to provide smooth and computationally efficient representations of tabulated values from rather inexact property formulations or measurement values with experimental errors. The development of new measurement methods, for instance by Wagner and Kleinrahm [32], and enhanced fitting techniques for empirical equations, such as those of Setzmann and Wagner [33], as well as of Lemmon and Jacobsen [34], led to very accurate fundamental equations of state (see Sec. 2.2). In order to provide fast property calculation algorithms based on such accurate equations, spline

interpolations are preferred to spline approximations since there is no additional smoothing required. Since the middle of the 19th century, the theory of splines has evolved quickly into a new field of research in mathematics. Meanwhile, many spline interpolation and approximation algorithms have been developed, not only for one-dimensional curves, but also for multidimensional problems. A comprehensive overview of spline algorithms is given by Schumaker [35]. The evaluation of polynomial splines, such as the cubic spline function described earlier in this section, is simple and allows for short computing times. Furthermore, polynomial splines are capable of fulfilling the requirements regarding continuity and differentiability. Polynomials of a degree lower than five can be solved analytically, which enables numerically consistent inverse functions. For these reasons, simple polynomial splines are preferred for property calculations rather than more complex approaches such as non-uniform B-splines, etc.

Cubic spline interpolation was applied by Müller [25] for the property functions $T_s(p)$, $v'(p)$, $v''(p)$, $h'(p)$, and $h''(p)$ of water and steam. The tabulated data were calculated from IAPS-84 [26, 27]. The derivatives with respect to p are calculable from the generated spline functions. A simple algorithm was applied to optimize the locations of the nodes along p. The resulting series of nodes is not equidistant, and a search algorithm is required to determine the interval that corresponds to the given value of p.

2.4.2 Two-Dimensional Functions

For functions of the form $z(x_1, x_2)$, the nodes could be arbitrarily distributed in the (x_1, x_2) plane. The local definition of polynomials would require a triangulation algorithm and an extensive search to identify the adjacent points for interpolating z at (x_1, x_2). Therefore, the following discussion is restricted to methods where the nodes are ordered in structured grids. Normally, rectangular grids with rectangular cells are used, as shown in Figs. 17 and 18. Analogously to the description of one-dimensional functions (Sec. 2.4.1), it is assumed that a table look-up method is applied to reproduce the results of an underlying equation, and a grid of nodes can be defined arbitrarily. Again, all variables z, x_1, and x_2 can be transformed into $\overline{z}$, $\overline{x}_1$, and $\overline{x}_2$ to enhance the accuracy of the interpolation. For some methods, the nodes are clustered in regions with strong curvature to achieve the required accuracy and to minimize the number of nodes over the domain of definition. For other methods, the nodes are distributed equidistantly along $\overline{x}_1$ and along $\overline{x}_2$. The number of nodes IJ and their $(\overline{x}_{1,i}, \overline{x}_{2,j})$ locations are chosen to ensure the desired accuracy. The $\overline{z}_{ij}(\overline{x}_{1,i}, \overline{x}_{2,j})$ values of the nodes are calculated from the transformed underlying function $\overline{z}(\overline{x}_1, \overline{x}_2)$. The values between the nodes are interpolated by means of polynomials $\overline{z}_{\{ij\}}(\overline{x}_1, \overline{x}_2)$, which are locally defined in each cell $\{i,j\}$. For the methods described below, a cell $\{i,j\}$ usually ranges between the four neighboring nodes. In order to interpolate values from a given grid of nodes, the cell $\{i,j\}$ must be determined first. For non-equidistant nodes, a search algorithm is required to identify the interval indices i and j. For equidistant nodes along $\overline{x}_1$ and $\overline{x}_2$, an extensive search algorithm can be avoided and the interval indices i and j are simply calculated from

$$i = \text{floor}\left(\frac{\overline{x}_1 - \overline{x}_{1,1}}{\Delta\overline{x}_1}\right) \text{ and } \qquad j = \text{floor}\left(\frac{\overline{x}_2 - \overline{x}_{2,1}}{\Delta\overline{x}_2}\right). \tag{2.57, 2.58}$$

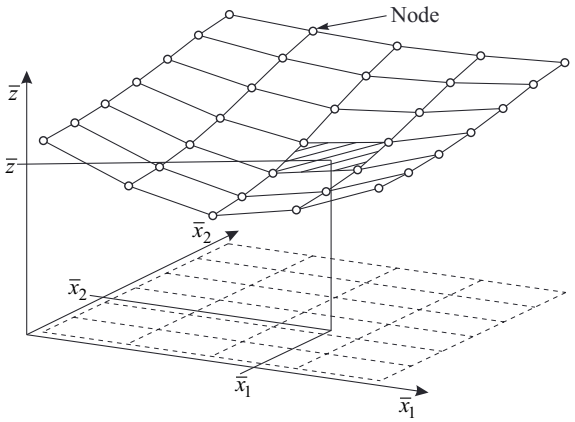

Figure 17: Nodes aligned in a rectangular grid in the $(\bar{x}_1, \bar{x}_2)$ plane, depicted in a $\bar{x}_1 - \bar{x}_2 - \bar{z}$ diagram.

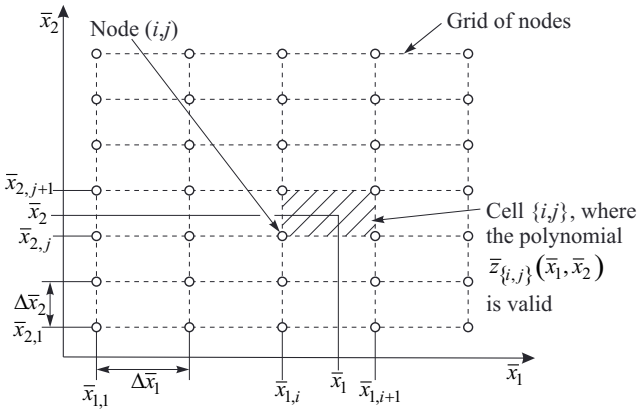

Figure 18: Projection of the grid of nodes into the $(\bar{x}_1, \bar{x}_2)$ plane and the definition of node (i,j) and cell $\{i,j\}$.

2.4.2.1 Local Polynomial Interpolation

Following the reasoning given in Sec. 2.4.1.1, only local polynomial interpolation methods are discussed in this work. Again, in contrast to spline interpolation methods (see Sec. 2.4.2.3), the local polynomials are defined without additional constraints to the adjacent cells. Therefore, the continuity of derivatives is limited as discussed later in this section for bicubic polynomials. For local polynomial interpolation methods, a polynomial $\bar{z}_{\{i,j\}}(\bar{x}_1, \bar{x}_2)$ is defined in each cell $\{i,j\}$ between the nodes (i,j), $(i+1,j)$, $(i,j+1)$, and $(i+1,j+1)$.

A polynomial of *N*-th order in $\overline{x}_1$ and $\overline{x}_2$ dimension is given by

$$\overline{z}_{\{i,j\}}\left(\overline{x}_1,\overline{x}_2\right)=\sum_{k=1}^{N+1}\sum_{l=1}^{N+1}a_{ijkl}\left(\overline{x}_1-\overline{x}_{1,i}\right)^{k-1}\left(\overline{x}_2-\overline{x}_{2,j}\right)^{l-1}. \tag{2.59}$$

For the bilinear case, *i.e.*, $N=1$, Eq. (2.59) has four coefficients a_{ijkl} that can be obtained from four constraints. For a rectangular cell $\{i,j\}$, these constraints are defined by the four corner points at which Eq. (2.59) intersects the nodes (i,j), $(i+1,j)$, $(i,j+1)$, and $(i+1,j+1)$. The resulting surface is not planar, but linear along $\overline{x}_1$ for constant $\overline{x}_2$ and vice versa. If the bilinear function is defined on a triangle, then the resulting surface is planar. Piecewise bilinear functions are continuous across the cell boundaries, but not continuously differentiable. A mathematical description of bilinear interpolation methods can be found in [36]. Bilinear interpolation on rectangular grids has been applied for property calculations for instance by Pini et al. [37]. Property functions based on bilinear interpolation on triangulated grids are discussed by Schulze [38]. In order to simplify the search algorithm, the nodes of the triangles are aligned along constant $\overline{x}_1$ or constant $\overline{x}_2$.

In order to provide a property function with continuous first derivatives, bicubic interpolation on a rectangular grid is often recommended. The 16 coefficients of the bicubic polynomial, Eq. (2.59) with $N=3$, can be determined from given values of $\overline{z}$, $\left(\partial\overline{z}/\partial\overline{x}_1\right)_{\overline{x}_2}$, $\left(\partial\overline{z}/\partial\overline{x}_2\right)_{\overline{x}_1}$, and $\left(\partial^2\overline{z}/\partial\overline{x}_1\partial\overline{x}_2\right)$ at the four corner nodes for each cell. In contrast to this local method, a bicubic spline function also provides a continuous second derivative (see Sec. 2.4.2.3). The local bicubic interpolation method is available in CoolProp [39] and was also applied by Müller [25] and Schulze [38]. Laughman et al. [40] applied local bicubic interpolation to describe the fundamental relation $\Phi\left(T,v\right)$. Similarly, Pini et al. [37] applied local bicubic interpolation to describe the thermodynamic potential $u(v,s)$. Instead of providing the derivatives $\left(\partial\overline{z}/\partial\overline{x}_1\right)_{\overline{x}_2}$, $\left(\partial\overline{z}/\partial\overline{x}_2\right)_{\overline{x}_1}$, and $\left(\partial^2\overline{z}/\partial\overline{x}_1\partial\overline{x}_2\right)$ at the four corner nodes, the bicubic polynomials are calculated from the $\overline{z}$ values of the four corner nodes and the additional twelve adjacent nodes in [37]. As described in [36], the resulting function does not provide continuity across the cell boundaries. In order to provide a thermodynamically consistent table look-up approach, Swesty [41] suggests to apply biquintic Hermite interpolation to reproduce Helmholtz free energy equations with two continuous derivatives.

2.4.2.2 Tabular Taylor Series Expansion Method (TTSE)

For two-dimensional TTSE functions [9, 10], the range of validity is divided in *IJ* rectangular cells as shown in Fig. 19. Typically, in the center of each cell $\{i,j\}$ a node is located from which a Taylor series $\overline{z}_{\{i,j\}}\left(\overline{x}_1,\overline{x}_2\right)$ is expanded. Exceptions are made for cases where the midpoint of the cell is out of range. In these cases, the node is shifted onto the corresponding region boundary. This eliminates the need for extrapolations far beyond the desired range of validity, as discussed in Sec. 2.4.2.3. The applied second order Taylor series expansion is given by

$$\begin{aligned}\bar{z}_{\{i,j\}}(\bar{x}_1,\bar{x}_2) = \bar{z}_{i,j} &+ (\bar{x}_1-\bar{x}_{1,i})\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}(\bar{x}_{1,i},\bar{x}_{2,j}) + \frac{1}{2}(\bar{x}_1-\bar{x}_{1,i})^2\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1{}^2}\right)_{\bar{x}_2}(\bar{x}_{1,i},\bar{x}_{2,j}) \\ &+ (\bar{x}_2-\bar{x}_{2,j})\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}(\bar{x}_{1,i},\bar{x}_{2,j}) + \frac{1}{2}(\bar{x}_2-\bar{x}_{2,j})^2\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_2{}^2}\right)_{\bar{x}_1}(\bar{x}_{1,i},\bar{x}_{2,j}) \quad . \quad (2.60) \\ &+ (\bar{x}_1-\bar{x}_{1,i})(\bar{x}_2-\bar{x}_{2,j})\left(\frac{\partial \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)(\bar{x}_{1,i},\bar{x}_{2,j})\end{aligned}$$

Equation (2.60) approximates the fluid property surface independently for each cell without any constraint to neighboring cells. The resulting inconsistencies at the grid lines are illustrated on the right hand side in Fig. 19.

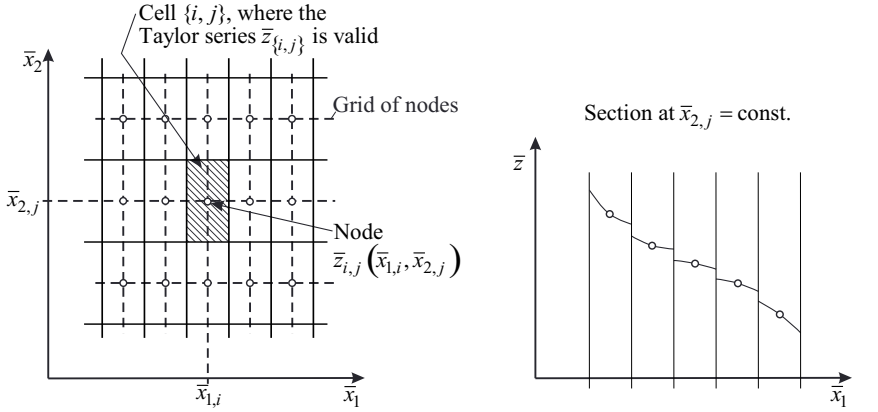

Figure 19: Grid of nodes in the $(\bar{x}_1,\bar{x}_2)$ projection with cell $\{i,j\}$, where the Taylor series $\bar{z}_{\{i,j\}}(\bar{x}_1,\bar{x}_2)$ is valid (left hand side) and a section at constant $\bar{x}_{2,j}$ illustrating the inconsistencies at the grid lines (right hand side).

In [9], the TTSE method is applied to IAPWS-95, and TTSE property functions for $T(p,h)$, $v(p,h)$, and $s(p,h)$ are created on a common grid with 200x200 nodes. The range of validity covers the single-phase region in the following temperature and pressure ranges:

$$275\ \text{K} \le T \le 1000\ \text{K} \qquad 612.8\ \text{Pa} \le p \le 100\ \text{MPa}\ .$$

The maximum (max) and root-mean-square (RMS) deviations of the TTSE property function for $T(p,h)$ from that of IAPWS-95 along with the permissible values (perm) are given in Table 6 for the vapor region. The maximum inconsistency at the interval boundaries is 20 mK.

The nodes along with the corresponding values and derivatives for each property are stored in a look-up table. For the given example, the nodes are aligned equidistantly along the enthalpy axis, which allows for a fast determination of the cell index, which corresponds to the given enthalpy. However, the pressure coordinates are not equidistant and the desired pressure cell index is determined by means of the bisection method. Inverse functions of (p,T), (p,s), and (h,s) are consistently calculated from the tabulated values in the (p,h) plane. In order to

determine the cell indices, which correspond to the given pair of variables, the bisection method is also applied.

Table 6: Maximum (max) and root-mean-square (RMS) deviations of the TTSE property function for $T(p,h)$ from that of IAPWS-95 along with the permissible values (perm) for the vapor region according to [9]

	$\lvert\Delta T\rvert_{\text{perm}}$ [mK]	$\lvert\Delta T\rvert_{\text{max}}$ [mK]	$\lvert\Delta T\rvert_{\text{RMS}}$ [mK]
$T(p,h)$	10 (25) [a,b]	9.21 [a]	0.25 [a]

[a] The vapor region corresponds to IAPWS-IF97 region 2 except for the pressure and temperature limits as described above.

[b] The permissible deviations differ for the subregions 2a, 2b, and 2c of IAPWS-IF97 region 2 as shown in Fig. 10 and given in Table 5.

The TTSE method provides fast property functions and is implemented for instance in the CoolProp [39] software package. The inconsistencies between adjacent Taylor series are small for a sufficient number of nodes. However, these inconsistencies may lead to problems in simulations with small spatial and time discretization.

2.4.2.3 Spline Interpolation and Approximation Algorithms

In Sec. 2.4.1.3, the one-dimensional cubic spline function and its analogy to the bending line of a flexible rod are discussed. The spline concept can be generalized to higher dimensions. For two-dimensional functions, a bicubic spline function can be established as described in [36]. In between the IJ nodes, $(I-1)(J-1)$ cells $\{i,j\}$ are defined. Each cell ranges over $\overline{x}_{1,i} \le \overline{x}_1 < \overline{x}_{1,i+1}$ and $\overline{x}_{2,j} \le \overline{x}_2 < \overline{x}_{2,j+1}$. In each cell $\{i,j\}$, a bicubic spline polynomial is defined as depicted in Fig. 18. The bicubic spline polynomial reads

$$\overline{z}_{\{i,j\}}\left(\overline{x}_1,\overline{x}_2\right) = \sum_{k=1}^{4}\sum_{l=1}^{4} a_{ijkl}\left(\overline{x}_1-\overline{x}_{1,i}\right)^{k-1}\left(\overline{x}_2-\overline{x}_{2,j}\right)^{l-1}. \tag{2.61}$$

The $16(I-1)(J-1)$ coefficients a_{ijkl} are determined by defining the following constraints. For all cells $\{i,j\}$, with $i=1,\ldots,I-1$ and $j=1,\ldots,J-1$, each polynomial $\overline{z}_{\{i,j\}}\left(\overline{x}_1,\overline{x}_2\right)$ intersects the four nodes at its corners

$$\overline{z}_{\{i,j\}}\left(\overline{x}_{1,i},\overline{x}_{2,j}\right) = \overline{z}_{i,j}\left(\overline{x}_{1,i},\overline{x}_{2,j}\right), \tag{2.62}$$

$$\overline{z}_{\{i,j\}}\left(\overline{x}_{1,i+1},\overline{x}_{2,j}\right) = \overline{z}_{i,j}\left(\overline{x}_{1,i+1},\overline{x}_{2,j}\right), \tag{2.63}$$

$$\overline{z}_{\{i,j\}}\left(\overline{x}_{1,i},\overline{x}_{2,j+1}\right) = \overline{z}_{i,j}\left(\overline{x}_{1,i},\overline{x}_{2,j+1}\right), \tag{2.64}$$

$$\overline{z}_{\{i,j\}}\left(\overline{x}_{1,i},\overline{x}_{2,j}\right) = \overline{z}_{i,j}\left(\overline{x}_{1,i},\overline{x}_{2,j}\right). \tag{2.65}$$

For given values of $\left(\partial\overline{z}/\partial\overline{x}_1\right)_{\overline{x}_2}$ at the nodes $(i=1,j)$ and $(i=I,j)$, with $j=1,\ldots,J$,

$$\left(\frac{\partial\overline{z}}{\partial\overline{x}_1}\right)_{\overline{x}_2}\Bigg|_{1,j}\left(\overline{x}_{1,1},\overline{x}_{2,j}\right) = \left(\frac{\partial\overline{z}}{\partial\overline{x}_1}\right)_{\overline{x}_2}\left(\overline{x}_{1,1},\overline{x}_{2,j}\right), \tag{2.66}$$

$$\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{I,j}\left(\bar{x}_{1,I},\bar{x}_{2,j}\right)=\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\left(\bar{x}_{1,I},\bar{x}_{2,j}\right), \tag{2.67}$$

the remaining $\left(\partial\bar{z}/\partial\bar{x}_1\right)_{\bar{x}_2}$ values at the nodes (i,j), with $i=2,\ldots,I-1$ and $j=1,\ldots,J$, are obtained from one-dimensional cubic spline interpolations. For this purpose, cubic spline functions, see Sec. 2.4.1.3, are established to interpolate between the nodes (i), with $i=1,\ldots,I$, along each grid line j. Then, the $\bar{z}_i$ and $\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1$ values in Eqs. (2.44), (2.45), (2.48), and (2.49) correspond to the given values of $\bar{z}_{i,j}$ and $\left(\partial\bar{z}/\partial\bar{x}_1\right)_{\bar{x}_2}$ along the considered grid line j. At each node (i,j), with $i=2,\ldots,I-1$ and $j=1,\ldots,J$, the required derivative is then obtained from the polynomial $\bar{z}_{\{i\},j}\left(\bar{x}_1\right)$ of the corresponding spline function for grid line j

$$\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i,j}=\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i\},j}\left(\bar{x}_{1,i}\right)=a_{i2,j}\,. \tag{2.68}$$

Analogously, for given values of $\left(\partial\bar{z}/\partial\bar{x}_2\right)_{\bar{x}_1}$ at the nodes $(i,j=1)$ and $(i,j=J)$, with $i=1,\ldots,I$,

$$\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,1}\left(\bar{x}_{1,i},\bar{x}_{2,1}\right)=\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\left(\bar{x}_{1,i},\bar{x}_{2,1}\right), \tag{2.69}$$

$$\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,J}\left(\bar{x}_{1,i},\bar{x}_{2,J}\right)=\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\left(\bar{x}_{1,i},\bar{x}_{2,J}\right), \tag{2.70}$$

the remaining $\left(\partial\bar{z}/\partial\bar{x}_2\right)_{\bar{x}_1}$ values at the nodes (i,j), with $i=1,\ldots,I$ and $j=2,\ldots,J-1$, are obtained from cubic spline functions along each grid line i. At each of these nodes (i,j), the required derivative is determined from the polynomial $\bar{z}_{\{j\},i}\left(\bar{x}_2\right)$ of the corresponding spline function for grid line i

$$\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j}=\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_2}\right|_{\{j\},i}\left(\bar{x}_{2,j}\right)=a_{j2,i}\,. \tag{2.71}$$

Similarly, cubic spline functions are used to determine $\left(\partial^2\bar{z}/(\partial\bar{x}_1\partial\bar{x}_2)\right)$ at all nodes except for the four corner nodes (1,1), $(I,1)$, $(1,J)$, and (I,J). Since $\left(\partial\bar{z}/\partial\bar{x}_2\right)_{\bar{x}_1}$ is cubic along $\bar{x}_1$, cubic spline functions can be established for the grid line $j=1$ and $j=J$. Then, the $\bar{z}_i$ and $\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1$ values in Eqs. (2.44), (2.45), (2.48), and (2.49) correspond to the previously determined values of $\left(\partial\bar{z}/\partial\bar{x}_2\right)_{\bar{x}_1}$ along the considered grid line j and the given values of $\left(\partial^2\bar{z}/(\partial\bar{x}_1\partial\bar{x}_2)\right)$ at the endpoints $(1,j)$ and (I,j). At each node (i,j), with $i=2,\ldots,I-1$ and $j=1,J$, the required derivative is then obtained from the polynomial $\left(\partial\bar{z}/\partial\bar{x}_2\right)_{\bar{x}_1}\Big|_{\{i\},j}\left(\bar{x}_1\right)$ of the corresponding spline function for grid line j

$$\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1\partial \bar{x}_2}\right)\Bigg|_{i,j}=\left.\frac{\mathrm{d}\left(\partial\bar{z}/\partial\bar{x}_2\right)_{\bar{x}_1}}{\mathrm{d}\bar{x}_1}\right|_{\{i\},j}\left(\bar{x}_{1,i}\right)=a_{i2,j}\,. \tag{2.72}$$

At each of the remaining nodes (i,j), with $i=1,\ldots,I$ and $j=2,\ldots,J-1$, the values of $\left(\partial^2\bar{z}/(\partial\bar{x}_1\partial\bar{x}_2)\right)$ are then determined from the polynomial $\left(\partial\bar{z}/\partial\bar{x}_1\right)_{\bar{x}_2}\Big|_{\{j\},i}\left(\bar{x}_2\right)$ of the corresponding spline function for grid line i

$$\left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{i,j} = \left.\frac{\mathrm{d}\left(\partial \bar{z}/\partial \bar{x}_1\right)_{\bar{x}_2}}{\mathrm{d}\bar{x}_2}\right|_{\{j\},i} \quad \left(\bar{x}_{2,j}\right) = a_{j2,i}\,. \tag{2.73}$$

Once the values of $\left(\partial \bar{z}/\partial \bar{x}_1\right)_{\bar{x}_2}$, $\left(\partial \bar{z}/\partial \bar{x}_2\right)_{\bar{x}_1}$, and $\left(\partial^2 \bar{z}/\left(\partial \bar{x}_1 \partial \bar{x}_2\right)\right)$ are known at each node (i,j), the 16 coefficients of each polynomial $\bar{z}_{\{i,j\}}\left(\bar{x}_1, \bar{x}_2\right)$, Eq. (2.61), can be obtained from Eqs. (2.62) – (2.65) and

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{\{i,j\}} \left(\bar{x}_{1,i}, \bar{x}_{2,j}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{i,j}, \tag{2.74}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{\{i,j\}} \left(\bar{x}_{1,i+1}, \bar{x}_{2,j}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{i+1,j}, \tag{2.75}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{\{i,j\}} \left(\bar{x}_{1,i}, \bar{x}_{2,j+1}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{i,j+1}, \tag{2.76}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{\{i,j\}} \left(\bar{x}_{1,i+1}, \bar{x}_{2,j+1}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_1}\right)_{\bar{x}_2}\right|_{i+1,j+1}, \tag{2.77}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{\{i,j\}} \left(\bar{x}_{1,i}, \bar{x}_{2,j}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{i,j}, \tag{2.78}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{\{i,j\}} \left(\bar{x}_{1,i+1}, \bar{x}_{2,j}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{i+1,j}, \tag{2.79}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{\{i,j\}} \left(\bar{x}_{1,i}, \bar{x}_{2,j+1}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{i,j+1}, \tag{2.80}$$

$$\left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{\{i,j\}} \left(\bar{x}_{1,i+1}, \bar{x}_{2,j+1}\right) = \left.\left(\frac{\partial \bar{z}}{\partial \bar{x}_2}\right)_{\bar{x}_1}\right|_{i+1,j+1}, \tag{2.81}$$

$$\left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{\{i,j\}} \left(\bar{x}_{1,i}, \bar{x}_{2,j}\right) = \left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{i,j}, \tag{2.82}$$

$$\left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{\{i,j\}} \left(\bar{x}_{1,i+1}, \bar{x}_{2,j}\right) = \left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{i+1,j}, \tag{2.83}$$

$$\left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{\{i,j\}} \left(\bar{x}_{1,i}, \bar{x}_{2,j+1}\right) = \left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{i,j+1}, \tag{2.84}$$

$$\left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{\{i,j\}} \left(\bar{x}_{1,i+1}, \bar{x}_{2,j+1}\right) = \left.\left(\frac{\partial^2 \bar{z}}{\partial \bar{x}_1 \partial \bar{x}_2}\right)\right|_{i+1,j+1}. \tag{2.85}$$

A comprehensive description of the method and a computationally efficient implementation is given by Späth [36].

The spline approximation method for one-dimensional functions developed by Schot [31] (see Sec. 2.4.1.3) was also extended to two-dimensional functions. This was achieved by fitting cubic spline functions for $\bar{z}\left(\bar{x}_1\right)_{\bar{x}_2}$ for several values of $\bar{x}_2$, and then fitting cubic spline functions for the obtained coefficients $a_{ik,\bar{x}_2}\left(\bar{x}_2\right)$. In addition to the one-dimensional functions $p_\mathrm{s}\left(T\right)$, $v'\left(T\right)$, and $v''\left(T\right)$, a two-dimensional function $v(p,T)$ was approximated from the 1963 International Skeleton Tables for water and steam [42]. The applied two-dimensional cubic spline algorithm for $v(p,T)$ requires a rectangular domain in the (p,T) plane. Therefore, the domain has been divided into several subdomains, which leads to inconsistencies at the boundaries between these subdomains. From the spline functions for $v(p,T)$ and an equation for $c_p^\mathrm{o}\left(T\right)$ [43], the Gibbs free energy g, enthalpy h, and entropy s were derived according to Table 2. The method was programmed in FORTRAN and was intended to be used for fast and accurate property calculations in a computer-aided design process for ships.

Herbst [44] has extended the spline-approximation technique by Klaus and van Ness [29], see Sec. 2.4.1.3, to functions of two independent variables. In his work, a function for the Gibbs free energy $g(p,T)$ is established for liquid water at pressures up to 100 MPa and temperatures up to 550 K. This was achieved by creating spline functions for $v(p,T)$ and $s\left(T\right)_{p_0}$ first, and applying the equation

$$g\left(p,T\right) = g_0 + \int_{p_0}^{p} v\left(p,T\right)\mathrm{d}p - \int_{T_0}^{T} s\left(T\right)_{p_0}\,\mathrm{d}T\,, \tag{2.86}$$

which can be obtained by integrating Eq. (2.19) from the chosen reference state, denoted by the subscript "0", to the actual state point (p,T). Furthermore, an inverse function for the specific enthalpy $h(p,s)$ has been derived analytically from the function for $g(p,T)$. The bicubic spline function $v(p,T)$ was created in a similar fashion as done by Schot [31].

In parallel to the development of cubic spline approximation methods a least-squares fitting method with biquadratic polynomials has been proposed by White [45]. The resulting property surface is continuously differentiable once. Over the past several decades numerous new spline interpolation and approximation algorithms have been developed. A comprehensive overview and a detailed description of some of these methods is available for instance in [23], [35], and [36]. As discussed in Sec. 2.4.1.3, spline interpolation algorithms are preferred to spline approximations when accurate underlying equations are available.

For many two-dimensional spline interpolation methods a rectangular grid of nodes needs to be defined over a rectangular domain. In many cases, this would require an appropriate extrapolation for nodes beyond the range of validity, as illustrated in Fig. 20. As for example in [31], this is often avoided by dividing the considered domain into several subdomains, which leads to inconsistencies at the boundaries between those subdomains. Kretzschmar et al. [46, 47] aimed to develop interpolation methods with optimized grid structures. Aligning the nodes in the grid in a way that prevents potential problems at the region boundaries was suggested. This is achieved by distributing grid lines for constant values of one independent variable, *e.g.*, for constant values of $\bar{x}_2$, across the domain. For each grid line, the number and the locations of nodes along $\bar{x}_1$ are defined independently. This is depicted in Fig. 21, where the number of

nodes at $\overline{x}_{2,j}$ is different from the number of nodes at $\overline{x}_{2,j+1}$. The interpolation of values for $\overline{z}(\overline{x}_1, \overline{x}_2)$ is realized as illustrated in Fig. 22. First, the interval $\{j\}$ which corresponds to the transformed given value $\overline{x}_2$ is determined. Then the auxiliary values $\overline{z}_j$ and $\overline{z}_{j+1}$ are determined by cubic spline interpolation along the corresponding grid lines j and $j+1$ for the transformed given value $\overline{x}_1$. Afterwards, the value for $\overline{z}(\overline{x}_1, \overline{x}_2)$ is calculated by linear interpolation between $\overline{z}_j$ and $\overline{z}_{j+1}$ for the transformed given value $\overline{x}_2$. In the vicinity of a region boundary, the $\overline{z}_B$ value at the region boundary itself is used for the interpolation as shown in Fig. 23. The grid optimization method is summarized in Appendix A1.

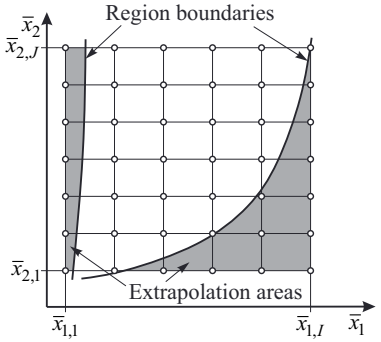

Figure 20: Rectangular grid of nodes with areas beyond the region boundaries (grey), where nodes need to be extrapolated.

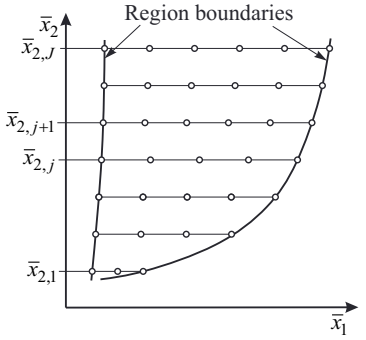

Figure 21: Distribution of nodes along constant values of $\overline{x}_2$ with independent numbers and locations along $\overline{x}_1$.

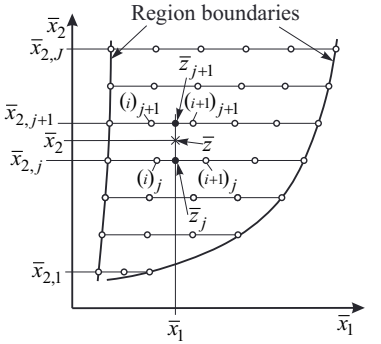

Figure 22: Interpolation of $\overline{z}(\overline{x}_1, \overline{x}_2)$ from the auxiliary values $\overline{z}_j$ and $\overline{z}_{j+1}$.

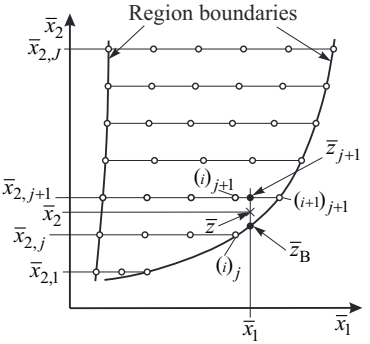

Figure 23: Interpolation of $\overline{z}(\overline{x}_1, \overline{x}_2)$ from the auxiliary values $\overline{z}_B$ and $\overline{z}_{j+1}$.

Variable transformations of the form $\bar{x}_1 = x_1 - x_{1,\mathrm{B}}(\bar{x}_2)$ or $\bar{x}_2 = x_2 - x_{2,\mathrm{B}}(\bar{x}_1)$ have been proposed by Müller [25] to shift the nodes onto the region boundary B as illustrated in Figs. 24 and 25. The same approach was also employed by Gräber et al. [48] for property calculations from bicubic splines functions. For the proposed variable transformations, the values at the region boundary need to be computed first. In previous studies, for instance in [49], it was proposed to determine the nodes beyond the considered range of validity by extrapolation. In some cases this can simply be done with the underlying function. In other cases a suitable extrapolation algorithm has to be applied. In [48] the following statement can be found: "Many thermodynamic properties exhibit discontinuities of the first derivative or even jumps at the saturation curve. With the approach of Kunick et al. (2008), the saturation curve runs across the interpolation grid. This leads to either very inaccurate data or a comparable high number of grid nodes." This is not right. The concerning article [49] clearly describes that all nodes were calculated from the underlying equation of state and extrapolations were made where required. Moreover, the grid of nodes and the resulting deviations from the underlying equation of state are given. The variable transformation described above has neither an influence on the local node density nor does it linearize the property function. But it extends the computing time. Therefore, the $\bar{z}$ values at the nodes outside the range of validity should be obtained from appropriate extrapolations, for instance from the underlying equation of state.

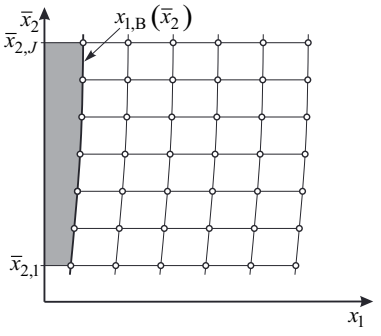

Figure 24: Projection of the grid of nodes in the $(x_1, \bar{x}_2)$ plane.

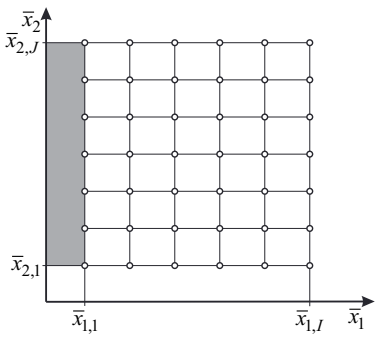

Figure 25: Projection of the grid of nodes in the $(\bar{x}_1, \bar{x}_2)$ plane.

2.4.3 Thermodynamic Consistency of Table Look-up Methods

As stressed by Swesty [41], a table look-up method must ensure thermodynamic consistency for the provided property functions. Otherwise, thermodynamic inconsistencies may lead to unphysical behavior in process simulations. If, for instance, the thermodynamic properties pressure p and entropy s are calculated from specific volume and internal energy (v,u), then thermodynamic consistency requires that

$$p(v,u) = -\left(\frac{\partial u}{\partial v}\right)_s = \frac{\left(\frac{\partial s}{\partial v}\right)_u}{\left(\frac{\partial s}{\partial u}\right)_v}. \tag{2.87}$$

If such equations are not satisfied, then the evaluation of the balance equations for mass, energy, and entropy for finite differences may lead to unphysical production or loss of these quantities.

By definition, thermodynamic consistency can only be realized if all thermodynamic properties are derived from the same description of a thermodynamic potential (see Sec. 2.2). To ensure continuity for the required property functions, this description must be at least two times continuously differentiable. The equations employed in a table look-up method are frequently simple polynomials, which require very short computing times. A polynomial equation of at least third degree would be required to fulfill the aforementioned differentiability. The computational effort to calculate an inverse function from a polynomial increases with its degree. Therefore, fluid properties are often interpolated from separate look-up tables for each property with lower order polynomials, rather than from a tabulated thermodynamic potential with higher order polynomials. Such an approach does not provide full thermodynamic consistency. If all properties including their derivatives and inverse functions are uniquely defined, then the property formulation is said to be numerically consistent. If the tabulated properties are derived from the same thermodynamic potential and the applied interpolation method describes these properties with very high accuracy, then the resulting property formulation is quasi thermodynamically consistent.

The importance of thermodynamic consistency is also discussed in the documentation of the CFD software ANSYS CFX [50]. Nevertheless, for fast property calculations, look-up tables based on cubic equations of state or IAPWS-IF97 are generated in this software [51]. The fluid properties are tabulated as functions of pressure and temperature. The default tolerances for the bilinearly interpolated enthalpy and entropy values are set to 1% and 3%, respectively. According to [51], these tolerances are adequate for property calculations in CFD. A table look-up method that is quasi thermodynamically consistent should therefore be applicable in process simulations without limitations.

2.5 Computing Time

The computing time an algorithm requires for its execution in a specific environment (hardware and operating system) depends on its computational costs regarding the applied mathematical operations, conditional constructs, loops, memory access, etc., as described in Sec. 2.5.1. With regard to these computational costs, the considered algorithm must be implemented efficiently to achieve short computing times. In Sec. 2.5.2 several possible optimizations for computationally efficient property calculations are discussed. The computing times of the several property calculation algorithms are compared to each other in Sec. 2.5.3.

The test procedures for the computing times of various operations and property calculation algorithms, see Secs. 2.5.1 and 2.5.3, have been compiled into single-threaded 32-bit programs with the Intel Composer 2011 with default options. The tests were carried out on a Windows 8 computer equipped with an Intel Core i7-3840QM CPU with 2.8 GHz and 16 GB RAM.

2.5.1 Computing Times of Various Operations

In order to understand and to optimize the computing speed of an algorithm, some important aspects about the execution of various operations on a modern computer are summarized here. The information provided below applies to standard personal computers (PCs) with Windows, Linux, or Mac operating systems (32-bit or 64-bit). It is assumed that the algorithms are compiled into executable binary code directly, which is usually the case for FORTRAN and C/C++ compilers.

The computing times of various mathematical operations for double precision floating point operands are described in Table 7. Additions, subtractions, multiplications, divisions, and square roots are performed as single instructions on the CPU, while power functions, exponential functions, and logarithms are calculated using math libraries provided with the compilers. As described in [52], modern CPUs have two or more execution units for integer operations, one execution unit for floating point multiplications, and one execution unit for floating point additions. This enables the processor to perform certain operations, for instance a floating point addition $(a+b)$ and a floating point multiplication $(c\times d)$, at the same time. Moreover, an execution unit can start an operation before the preceding operation is completed. This is beneficial if the operations are independent from each other, such as the multiplications in $(a\times b+c\times d)$. In this context, the latency of an instruction is defined as the number of clock cycles it takes to have the resulting data available for use by another instruction. The throughput is defined as the number of clock cycles it takes for the instruction unit to accept the next instruction of the same kind. Therefore, in expressions where the operations are dependent on each other, such as in $\big((x+a)\times x+b\big)\times x$, the latency of the operations must be considered. For this example, the latency is 16 clock cycles for two multiplications and the two additions, which is $5.3\overline{3}$ times longer than the latency of an addition. The expression $a\times b+c\times d+e$ requires the exact same operations, but the multiplications $(a\times b)$ and $(c\times d)$ are independent of each other and can therefore be calculated in $5+1=6$ clock cycles. The additions take additional $3+3=6$ clock cycles, so that the entire expression is computed in 12 clock cycles, which is 4 times longer than the latency of an addition.

Table 7: Latencies and throughputs of mathematical operations

Operation	Instruction or Intel Intrinsic Function	Reference values from [53, 54, 55] for: Intel Core i7-3840QM, Ivy Bridge (CPUID: 06_3AH, 06_3EH)				Measured values
		Latency	Throughput	Rel. latency [a)]	Rel. throughput [a)]	Rel. latency [a)]
		[clock cycles]		[-]		[-]
$(a+b)$	ADDSD	3	1	1	1	1
$(a-b)$	SUBSD	3	1	1	1	1
$(a\times b)$	MULSD	5	1	$1.6\overline{6}$	1	1.67
(a/b)	DIVSD (__mm_div_sd)	20	14	$6.6\overline{6}$	14	6.67
$\sqrt{a}$	SQRTSD (__mm_sqrt_sd)	20	14	$6.6\overline{6}$	14	6.7
$\sqrt[4]{a}=\sqrt{\sqrt{a}}$	-	-	-	$13.3\overline{3}$ [b)]	34	13.3
a^b	___libm_sse2_pow	-	-	-	-	~35 [c)]
e^a	___libm_sse2_exp	-	-	-	-	~17.7 [c)]
$\ln(a)$	___libm_sse2_log	-	-	-	-	~20 [c)]
$(x+a)\times x$	-	-	-	$2.6\overline{6}$ [b)]	-	2.67
$((x+a)\times x+b)\times x$	-	-	-	$5.3\overline{3}$ [b)]	-	5.33
$a\times b+c\times d+e$	-	-	-	4 [b)]	-	4

[a] With regard to latency or throughput of $(a+b)$.

[b] Theoretical values calculated from latencies and throughputs of the involved operations.

[c] Average latencies (operand dependent).

For the mathematical operations that are performed by compiler specific math libraries, such as power functions, exponential functions, and logarithms, the latencies are operand dependent and no definite values are provided in the literature. In order to provide average values for these functions, the computing times of their implementations as Intel Intrinsic Functions [55] have been measured. Profilers, as provided with many compilers, are unsuitable for such measurements. This is because the computing times of the expressions considered here are often smaller than the time resolution a profiler permits. Therefore, a program has been developed in this work to determine the latencies of mathematical operations. For verification, the reference values for the latencies given in Table 7 have been reproduced and the theoretical value for the latency of $\sqrt[4]{a}$ has been confirmed. The average latencies for a^b, e^a, and $\ln(a)$ with randomly distributed values for the operands a and b in the ranges described in Table 8 have been determined.

Table 8: Ranges for operands a and b in a^b, e^a, and $\ln(a)$ for latency measurements

Operation	$a_{\min} \le a \le a_{\max}$	$b_{\min} \le b \le b_{\max}$
a^b	$0.1 \le a \le 10$	$0.001 \le a \le 10$
e^a	$0.001 \le a \le 100$	-
$\ln(a)$	$0.001 \le a \le 1000$	-

The computing time required for branch predictions of conditional constructs, such as “if-then-else” or “switch” statements, depends on their predictability. Whereas the correct prediction of a branch adds 0-2 additional clock cycles, a branch misprediction adds 12-25 clock cycles [52]. Loops with a low and fixed number of repetitions can be predicted perfectly. It is less efficient if the number of repetitions depends on calculations in the loop, but for some tasks, such as iteration methods, this cannot be avoided. More detailed information on branch prediction is provided in [56].

The computational speed of memory access operations depends on the location of the affected variable. If the variable is stored in the CPU’s register already, the computing time of a read operation is ≤ 1 clock cycle. If the variable has to be fetched from the random access memory (RAM) this may require more than 100 clock cycles. To accelerate memory access operations, modern CPUs are equipped with caches. Fetching data from the cache memory is much faster than from the RAM. Since the sizes of the different cache levels are rather small, modern CPUs automatically prefetch data to the cache. If a variable cannot be fetched from the cache, this is called a “cache miss”. To minimize cache misses, data that is used together should be stored together. Since memory access latencies depend on core frequency, RAM speed, etc., no definite values are provided in the literature. Table 9 provides rough approximations of memory access latencies for the Intel® Xeon® Processor 5500 Series, as reported in [57].

Table 9: Memory access latencies for the Intel® Xeon® Processor 5500 Series as given in [57]

Data source	Latency [clock cycles] (rough approximation)
Level 1 cache	4
Level 2 cache	10
Level 3 cache	40
RAM	100-150

The computational costs of mathematical operations dominate for thermodynamic and transport property equations. For complex cell-search algorithms in table look-up methods, branch mispredictions can have a considerable influence on computing times. While the computing-time consumption of memory access can be neglected for the computation of thermodynamic and transport property equations, it can become noticeable when using table look-up methods with a large amount of data. The actual effect of cache misses must be tested in the process simulation where the property calculation algorithms are applied.

2.5.2 Computationally Efficient Implementation of Property Formulations

The computing time of an algorithm in a specific hardware and software environment depends on the way it is implemented. In many situations, several different implementations of the same algorithm are possible, leading to very different computing times. Some examples for computationally efficient implementations are discussed in this section.

As described in Table 7 in the previous section, the power function a^b is comparatively slow and should therefore be avoided if possible. For integer exponents b, a^b should therefore be calculated as

$$a^b = \prod_{i=1}^{b} a \,. \tag{2.88}$$

This optimization will be performed automatically by some compilers if b is known at compile time. Similarly, the implementation of polynomial equations can take advantage of Horner's method, which can be written as

$$\sum_{i=0}^{I} a_i x^i = a_0 + x\left(a_1 + x\left(a_2 + ... + x a_I\right)...\right)\right). \tag{2.89}$$

The factorized expression on the right hand side is computationally more efficient than the calculation with power functions shown on the left hand side. Thus, Horner's method can be advantageously used for calculating the polynomial for the isobaric heat capacity of the ideal-gas given by Eq. (2.2) for instance. In simple cases, the factorization will be performed by the compiler if the exponents are known at compile time. Trübenbach [58] has developed an algorithm to efficiently factorize more complex polynomial equations such as the basic equation for region 2 of IAPWS-IF97, Eq. (2.27). Analogously, Horner's method can also be applied to

the residual part of the short fundamental equation of state for water, Eq. (2.24). In this equation, all exponents of the inverse reduced temperature τ are multiples of 1/8. Therefore, if $\overline{\tau} = \sqrt[8]{\tau}$ is calculated once, the equation can be rewritten as a polynomial function in terms of $\overline{\tau}$ and δ and Horner's method can be applied. However, the natural exponential function in the terms $i = 8,\ldots,16$ in Eq. (2.24) would have to be calculated in advance.

For the more general case with rational exponents of τ and δ, a combination of logarithm rules is applied in REFPROP [59] to calculate the residual terms of the Helmholtz free energy of the form

$$\Phi_i^{\mathrm{r}}(\delta,\tau) = n_i \delta^{d_i} \tau^{t_i} \exp\left(f_i(\delta,\tau)\right). \tag{2.90}$$

The natural logarithms of τ and δ are calculated once and each term is then computed from

$$\Phi_i^{\mathrm{r}}(\delta,\tau) = n_i \exp\left(d_i \ln(\delta) + t_i \ln(\tau) + f_i(\delta,\tau)\right). \tag{2.91}$$

The non-analytical terms $i = 55,56$ in Eq. (2.22) are computed in a similar manner. It is often useful to store the values of the terms Φ_i^{r} along with some other expressions dependent on τ and δ for later use.

For many thermodynamic properties, several derivatives of the fundamental equation of state are required. It is advantageous to express the derivatives of each term Φ_i^{r} as a product of Φ_i^{r} itself and a factor $F_i^{\delta(n)}$ or $F_i^{\tau(m)}$ for the considered derivative, where the superscripts $\delta(n)$ and $\tau(m)$ denote the n-th derivative with respect to δ and the m-th derivative with respect to τ. As shown in Appendix A2, the factors $F_i^{\delta(n)}$ and $F_i^{\tau(m)}$ can be determined recursively from

$$F_i^{\delta(n)} = \delta F_{i,\delta}^{\delta(n-1)} + F_i^{\delta(n-1)}\left[F_i^{\delta} - (n-1)\right], \tag{2.92}$$

$$F_i^{\tau(m)} = \tau F_{i,\tau}^{\tau(m-1)} + F_i^{\tau(m-1)}\left[F_i^{\tau} - (m-1)\right], \tag{2.93}$$

with

$$F_i^{\delta(0)} = 1, \tag{2.94}$$

$$F_i^{\tau(0)} = 1, \tag{2.95}$$

$$F_{i,\delta}^{\delta(n)} = \delta F_{i,\delta\delta}^{\delta(n-1)} + F_{i,\delta}^{\delta(n-1)}\left[F_i^{\delta} - n + 2\right] + F_i^{\delta(n-1)} F_{i,\delta}^{\delta}, \tag{2.96}$$

and

$$F_{i,\tau}^{\tau(m)} = \tau F_{i,\tau\tau}^{\tau(m-1)} + F_{i,\tau}^{\tau(m-1)}\left[F_i^{\tau} - m + 2\right] + F_i^{\tau(m-1)} F_{i,\tau}^{\tau}. \tag{2.97}$$

The derivatives of the terms are then calculated from

$$\delta^n \Phi_{i,\delta(n)}^{\mathrm{r}} = \Phi_i^{\mathrm{r}} F_i^{\delta(n)} \tag{2.98}$$

and

$$\tau^m \Phi_{i,\tau(m)}^{\mathrm{r}} = \Phi_i^{\mathrm{r}} F_i^{\tau(m)}. \tag{2.99}$$

If the factor $F_i^{\delta(1)}$ is independent from τ and $F_i^{\tau(1)}$ is independent from δ, then the crossed derivatives can simply be calculated from

$$\delta^n \tau^m \Phi^{\mathrm{r}}_{i,\delta(n),\tau(m)} = \Phi^{\mathrm{r}}_i F_i^{\delta(n)} F_i^{\tau(m)} . \tag{2.100}$$

Eq. (2.100) can be used for all term forms appearing in Eq. (2.22) except for the non-analytical terms $i = 55{,}56$, where the product rule for higher order derivatives must be employed.

The calculation scheme presented above can also be used for other types of equations of state or transport property equations.

To calculate fluid properties from input variables other than temperature and density, efficient iteration procedures need to be applied. If analytical derivatives are available, the required property function can be calculated by iteration using Newton's or Halley's method. Using Newton's method, the root of a one-dimensional function $f(x)$ is determined by the iteration procedure

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} . \tag{2.101}$$

This procedure begins with a starting value x_0 and is repeated until $\left| f(x_k) \right| \le TOL$. The iteration procedure of Halley's second order method reads

$$x_{k+1} = x_k - \frac{2 f(x_k) f'(x_k)}{2\left[f'(x_k)\right]^2 - f(x_k) f''(x_k)} . \tag{2.102}$$

Both methods are applicable to multi-dimensional problems as well. For example, the iteration procedure of Newton's method for a system of non-linear equations $\boldsymbol{F}(\boldsymbol{X})$ is written as

$$\boldsymbol{J}(\boldsymbol{X}_k)\, \Delta \boldsymbol{X}_k = \boldsymbol{F}(\boldsymbol{X}_k) \tag{2.103}$$

and

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \Delta \boldsymbol{X}_k , \tag{2.104}$$

where $\boldsymbol{F}$ is the vector of functions f_i and $\boldsymbol{X}$ is vector of unknowns x_j. In each iteration step k the Jacobian matrix $\boldsymbol{J}(\boldsymbol{X}_k)$ must be determined, before the system of linear equations Eq. (2.103) is solved. The element of the Jacobian matrix in the i-th row and the j-th column reads

$$\boldsymbol{J}_{i,j}(\boldsymbol{X}_k) = \frac{\partial f_i}{\partial x_j} .$$

In the Appendices A3 and A4, computationally efficient formulations of Newton's method for two and three non-linear equations are given.

The linearization of the iterated function through the use of variable transformations can drastically reduce the required number of iterations. Neither Newton's nor Halley's method guarantees convergence. Therefore, sufficiently accurate starting values need to be provided. As an alternative, the Dekker-Brent algorithm [60] is recommended for one-dimensional iterations. For this method, convergence is guaranteed if the root is located in the given interval and the function is continuous. The Dekker-Brent algorithm does not require analytical derivatives of the function to be iterated and provides an at least linear rate of convergence.

2.5.3 Computing-Time Comparisons

In this section, the computing times of the ideal-gas model, the Peng-Robinson equation of state [14] (PR-EOS), IAPWS-95 [2], the short fundamental equation of state (ShortFEOS) for water by Kunz et al. [21], IAPWS-IF97 [4], IAPWS-IF97 backward equations [4] (IF97-BWE), and the TTSE method [10] (see Secs. 2.1 – 2.4) are compared with each other. In order to do so, the computing times of the property functions for $p(v,u)$, $T(v,u)$, $T(p,h)$, and $v(p,h)$ have been determined and compared with those of the iterative calculations from the Peng-Robinson equation of state. For this purpose, the so-called Computing-Time Ratio (*CTR*) has been determined, which is defined as

$$CTR = \frac{\text{Computing time for the iterative calculation from the PR-EOS}}{\text{Computing time for the calculation from the considered algorithm}}.$$

The considered property calculation algorithms have been optimized for computing speed as described in Sec. 2.5.2. For the isobaric heat capacity of the ideal gas c_p^o, Eq. (2.2), as well as its integrals used in conjunction with the ideal-gas equation, Eq. (2.1), and the Peng-Robinson equation, Eq. (2.7), Horner's method is applied. The IAPWS-IF97 basic functions and their derivatives are also implemented using this method. The derivatives of the Helmholtz free energy equation of IAPWS-95 [2] and the short fundamental equation of state of Kunz et al. [21] are computed with the well optimized internal routines of REFPROP [59]. REFPROP's internal iteration procedures for functions of (v,u) or (p,h) are implemented to ensure convergence for all calculable fluids. The determination of starting values has a more general nature and the iteration procedure relies on an algorithm with bisection. For the test calculations presented here, Newton's method for either one or two dimensions (see Sec. 2.5.2) with analytical derivatives is applied instead. The stopping criteria for these iterative calculations with regard to the independent variables are as follows:

v: $\left|\frac{\Delta v}{v}\right| \le 10^{-6}$,

$u < 1$ kJ/kg: $|\Delta u| \le 10^{-6}$ kJ/kg, $u \ge 1$ kJ/kg: $\left|\frac{\Delta u}{u}\right| \le 10^{-6}$,

p: $\left|\frac{\Delta p}{p}\right| \le 10^{-6}$,

$h < 1$ kJ/kg: $|\Delta h| \le 10^{-6}$ kJ/kg, $h \ge 1$ kJ/kg: $\left|\frac{\Delta h}{h}\right| \le 10^{-6}$.

The starting values for $T(p,h)$ and $v(p,h)$ of IAPWS-IF97 are determined from the corresponding backward equations [4]. For all other cases, the starting values are determined from simple approximations, for instance from the ideal-gas model with constant heat capacities. Since the independent variables of the various equations of state are either (T,v) or (p,T), the determination of the starting values differs among the tested property calculation algorithms. To consider the influence of the different algorithms for the starting values, the average number of iterations (*ANI*) has been determined in addition to the *CTR* values. It is assumed that the phase that corresponds to the given inputs is known for all calculations. Otherwise, the

saturation properties at vapor-liquid equilibrium would have to be determined first. If these properties are calculated from the Maxwell criterion, the computing time of the phase test is considerably higher than that of the actual property calculation.

The computing times were measured by means of a software similar to NIFBENCH [4] with 100,000 randomly distributed state points in the corresponding region of IAPWS-IF97. The test calculations were carried out for the liquid region 1 and the vapor region 2 (see Fig. 7). The employed compiler to build the test programs and the computer to run the test calculations are described at the beginning of Sec. 2.5. The results of the computing-time comparisons are summarized in Tables 10, 11, 12, and 13. These tables also show the average number of iterations it takes for each algorithm to converge for the given starting values and the defined stopping criteria. The tested property calculation algorithms have been conscientiously implemented. However, it must be noted that the results of computing-time comparisons are always implementation dependent and should be considered as such.

The computing times of the Peng-Robinson equation of state may be considered as just acceptable for extensive process simulations, such as CFD. In the vapor phase, the (v,u) property functions calculated from IAPWS-95 are at least 100 times slower than those calculated from the Peng-Robinson equation of state. With regard to the property functions of (p,h), IAPWS-95 is at least 45 times slower than the Peng-Robinson equation. Although calculations from the short fundamental equation of state for water are roughly 4 times faster than those from IAPWS-95, this equation is still much too slow for extensive process simulations. The example of $T(p,h)$ shows that the IAPWS-IF97 backward equations are at least 7 times faster than the iterative calculations from the Peng-Robinson equation. However, the numerical inconsistencies between the backward equations and the basic equations of IAPWS-IF97 may lead to convergence problems in simulations with very small spatial or time discretization. In these situations, inverse property functions must be calculated by iteration from the basic equations of IAPWS-IF97 with starting values determined from the available backward equations. The computing speeds of the resulting property functions for $T(p,h)$ are similar to those of the Peng-Robinson equation of state. In order to provide high accuracy, short computing times, and numerically consistent inverse functions, table look-up methods can be applied to accurate fundamental equations of state. In some process simulation codes, such as in ANSYS CFX [51], a simple table look-up method is applied even to the Peng-Robinson equation to reduce the computing times. The computing-time comparisons show that the IAPWS-95 based TTSE property functions for $T(p,h)$ are at least 4 times faster than those of the Peng-Robinson equation. Due to the discontinuous behavior of the TTSE property functions, the TTSE method cannot be used for simulations with small spatial or time discretizations.

Table 10: Computing-time ratios (*CTR*) with regard to the Peng-Robinson equation of state along with the average number of iterations (*ANI*) for $p(v,u)$ computed from various property calculation algorithms in the liquid region 1 and the vapor region 2 of IAPWS-IF97

$p(v,u)$				
	IAPWS-IF97 Region 1 (liquid)		IAPWS-IF97 Region 2 (vapor)	
Algorithm	*ANI*	*CTR*	*ANI*	*CTR*
Ideal gas	-	-	3.83	1.48
Peng-Robinson	3	1.00	3.08	1.00
IAPWS-IF97	2.94	1/6.08	3.64	1/11.1
Short FEOS	2.99	1/20.1	3.63	1/23.3
IAPWS-95	2.94	1/76.9	3.68	1/93.0

Table 11: Computing-time ratios (*CTR*) with regard to the Peng-Robinson equation of state along with the average number of iterations (*ANI*) for $T(v,u)$ computed from various property calculation algorithms in the liquid region 1 and the vapor region 2 of IAPWS-IF97

$T(v,u)$				
	IAPWS-IF97 Region 1 (liquid)		IAPWS-IF97 Region 2 (vapor)	
Algorithm	*ANI*	*CTR*	*ANI*	*CTR*
Ideal gas	-	-	3.83	1.48
Peng-Robinson	3	1.00	3.08	1.00
IAPWS-IF97	2.94	1/6.18	3.64	1/11.3
Short FEOS	2.99	1/20.1	3.63	1/23.3
IAPWS-95	2.94	1/76.9	3.68	1/93.0

Table 12: Computing-time ratios (*CTR*) with regard to the Peng-Robinson equation of state along with the average number of iterations (*ANI*) for $T(p,h)$ computed from various property calculation algorithms in the liquid region 1 and the vapor region 2 of IAPWS-IF97

$T(p,h)$				
	IAPWS-IF97 region 1 (liquid)		IAPWS-IF97 region 2 (vapor)	
Algorithm	*ANI*	*CTR*	*ANI*	*CTR*
Ideal gas	-	-	3.68	3.66
Peng-Robinson	5.72	1.00	3.47	1.00
IF97-BWE	-	31.0	-	7.83
IAPWS-IF97	1.99 [a]	2.51 [a]	2.02 [a]	1.01 [a]
Short FEOS	4.56	1/10.0	3.63	1/13.2
IAPWS-95	4.2	1/33.9	3.62	1/47.4
TTSE	-	7.33	-	4.61

[a] The starting values for the iteration from the IAPWS-IF97 basic equations are obtained from the corresponding backward equations for $T(p,h)$.

Table 13: Computing-time ratios (*CTR*) with regard to the Peng-Robinson equation of state along with the average number of iterations (*ANI*) for $v(p,h)$ computed from various property calculation algorithms in the liquid region 1 and the vapor region 2 of IAPWS-IF97

$v(p,h)$				
	IAPWS-IF97 region 1 (liquid)		IAPWS-IF97 region 2 (vapor)	
Algorithm	*ANI*	*CTR*	*ANI*	*CTR*
Ideal gas	-	-	3.68	3.64
Peng-Robinson	5.72	1.00	3.47	1.00
IF97-BWE	-	7.05 [a]	-	2.53 [a]
IAPWS-IF97	1.99 [a]	1.86 [a]	2.02 [a]	1/1.28 [a]
Short FEOS	4.56	1/10.2	3.63	1/13.3
IAPWS-95	4.2	1/33.8	3.62	1/47.8
TTSE	-	7.46	-	4.48

[a] The starting values for the iteration from the IAPWS-IF97 basic equations are obtained from the corresponding backward equations for $T(p,h)$.

2.6 Conclusions for the Development of a Fast and Accurate Property Calculation Method for Extensive Process Simulations

The fluid property functions applied in extensive process simulations, such as transient CFD or heat cycle calculations, need to meet the contradictory demands regarding high accuracy and very short computing times. Moreover, these property functions need to be thermodynamically and numerically consistent with each other.

The discussion of the available property calculation algorithms in the previous sections shows that the calculation of fluid properties from equations of state is always a trade-off between accuracy and computing speed. While reference equations of state, *e.g.*, IAPWS-95 for water and steam, provide the highest accuracy and reasonable extrapolation behavior, their computing speed is very slow. Short fundamental equations of state are sufficiently accurate for industrial calculations and certainly faster than reference equations of state. However, they are still too slow for a direct application in extensive process simulations such as CFD. The fastest thermodynamically consistent approach is given by fundamental equations of state for separate regions in conjunction with backward equations. The development of such equations is a very time consuming task. Therefore, this was realized for the industrial formulation IAPWS-IF97 for water and steam only. To meet the high demands regarding the numerical consistency in numerical process simulations with small spatial and time discretizations, inverse functions of (v,u), (p,h), (p,s), and (h,s) must be calculated by iteration. For calculations from (p,h), (p,s), and (h,s) inputs, the available backward equations should be used to compute the starting values. Backward equations for other pairs of variables, such as (v,u) or (p,v), are not available. Due to the necessity of iterative procedures for property functions of these pairs of variables, even IAPWS-IF97 is too slow for a direct application in CFD.

Table look-up methods provide an alternative for fast and accurate property calculations. These methods can be flexibly applied to any existing equation of state or transport property equation. Therefore, the development of an advanced table look-up method for property calculations in CFD and other extensive process simulations is pursued in this work. The capabilities and limitations of different table look-up methods have been discussed in Sec. 2.4. Generally, the accuracy of these methods is determined by the number and the distribution of the tabulated data points, the so called nodes, and the applied algorithm to interpolate or approximate the values between them. In order to enhance the accuracy of a table look-up method, the function to be interpolated should be linearized by means of suitable variable transformations. If the fluid properties are calculated from separate look-up tables rather than from a tabulated thermodynamic potential, then the accuracy of the resulting fluid property functions must not only meet the requirements of the process simulation, but also needs to ensure a certain thermodynamic consistency (see Sec. 2.4.3).

The computing time of a table look-up method largely depends on the desired search algorithm to identify the interval or cell, which corresponds to the given variables. To simplify these algorithms for two-dimensional functions, the nodes are often ordered in rectangular grids with rectangular cells. For some of the table look-up methods currently applied, for instance in [10] and in [25], the nodes in these grids are clustered in regions with strong curvature to achieve the required accuracy. For some other methods, for instance in [37] and in [50], the look-up tables are not prepared for the independent variables that are actually used most often

for property calculations. For all these methods iterative cell search algorithms are required, which suffer from branch mispredictions (see Sec. 2.5.1) and prolong the computing times. To further simplify the search algorithms, equidistant nodes are preferable. To meet the requirements of extensive process simulations, the computational speed of the newly developed table look-up method should be comparable to or less than that of the IAPWS-IF97 backward equations.

Some of the discussed table look-up methods do not fulfill the requirements regarding the differentiability of the provided property functions and are therefore not recommended for extensive process simulations. As discussed in Secs. 2.4.1.3 and 2.4.2.3, spline interpolation algorithms are capable of representing property functions continuously. A polynomial spline function of N-th degree can be $(N-1)$ times continuously differentiable. The spline polynomials can be computed with very low computational effort and are therefore very fast. Moreover, spline polynomials of a degree lower than five can be solved analytically for their independent variables. This enables numerically consistent inverse functions. However, the analytical solution of third or higher order polynomials involves the evaluation of some transcendental functions, which are computationally expensive. Second order polynomials can be solved very efficiently and fulfill the differentiability requirements of most process simulations. Therefore, the newly developed table look-up method should employ spline interpolation methods, preferably with second order polynomials.

Many two-dimensional spline interpolation methods require the definition of a rectangular grid of nodes over a rectangular domain. This rectangular domain must include the range of validity, which is irregularly shaped in the general case. In some cases it is sufficient to extrapolate the nodes beyond the range of validity appropriately. For other cases, efficient algorithms are required to transform the range of validity into a rectangular shape and to control the local node density.

In order to make the new table look-up method applicable to any one- or two-dimensional property function, a suitable software tool needs to be developed.

3 The Spline-Based Table Look-Up Method (SBTL)

The Spline-Based Table Look-Up method (SBTL) applies polynomial spline interpolation techniques to approximate the results of existing equations of state, with high accuracy and low computing time. The accuracy, computing time, and memory storage advantages are enabled with specialized coordinate transformations and simplified search algorithms as described below. The properties in the single-phase regions, such as $T(p,h)$, are represented by two-dimensional spline functions in the common form $z^{\mathrm{SPL}}(x_1, x_2)$, whereas the phase boundaries, such as $T_\mathrm{s}(p)$, are represented by one-dimensional spline functions $z^{\mathrm{SPL}}(x_1)$. Algorithms for calculating properties in the two-phase region that are consistent with those of the single-phase regions are also provided. The explanations given in this section are similar to those already published in [61], but are extended for the sake of completeness.

3.1 One-Dimensional Spline Functions

3.1.1 Spline Functions

A one-dimensional polynomial spline function $z^{\mathrm{SPL}}(x_1)$ is a continuous, piecewise-defined function consisting of several spline polynomials. The spline function interpolates values between a series of discrete data points, the so-called nodes (see Fig. 26). The number I and the location $x_{1,i}$ of the nodes are chosen to ensure the desired accuracy. The $z_i(x_{1,i})$ values of the nodes are calculated from the underlying function $z(x_1)$. The spline polynomials are connected at knots, which can either be equal or unequal to the nodes. For the SBTL method, the knots are located at the midpoint between the nodes along x_1, which results in symmetric boundary conditions leading to superior accuracy [23]. A spline polynomial ranges over the interval $\{i\}$ between two knots and intersects the node (i) within. The z positions of the knots result from the spline algorithm as explained below.

In most numerical process simulations, fluid property functions need to be continuously differentiable once. The quadratic spline function is the simplest approach to continuously represent a one-dimensional function and its first derivative. Furthermore, the quadratic spline polynomial can easily be inverted. This enables the calculation of numerically consistent backward functions, which are the so-called inverse spline functions. Therefore, in this document the calculation of properties with the SBTL method is carried out through the use of quadratic spline polynomials, as opposed to higher order polynomials, to create a spline function $z^{\mathrm{SPL}}(x_1)$ from the underlying function $z(x_1)$.

In order to increase the accuracy of the spline function, both the independent variable x_1 and the dependent variable z are transformed into $\overline{x}_1$ and $\overline{z}$, respectively, so that the transformed spline function yields $\overline{z}^{\mathrm{SPL}}(\overline{x}_1)$. A description of the transformations for one-dimensional spline functions can be found in Sec. 3.1.2.

The spline function is created in transformed coordinates through the use of quadratic spline polynomials

$$\overline{z}_{\{i\}}\left(\overline{x}_1\right) = \sum_{k=1}^{3} a_{ik}\left(\overline{x}_1 - \overline{x}_{1,i}\right)^{k-1}, \tag{3.1a}$$

where $\bar{x}_1$ is the transformed independent variable and $\bar{z}_{\{i\}}$ is the transformed dependent variable in the interval $\{i\}$. In Eq. (3.1a), $\bar{x}_{1,i}$ is the transformed value of the independent variable at the node (i), and a_{ik} are the three coefficients of the quadratic spline polynomial valid in the interval $\{i\}$. Eq. (3.1a) can also be written as

$$\bar{z}_{\{i\}}\left(\bar{x}_1\right) = a_{i1} + a_{i2}\left(\bar{x}_1 - \bar{x}_{1,i}\right) + a_{i3}\left(\bar{x}_1 - \bar{x}_{1,i}\right)^2 . \tag{3.1b}$$

The I polynomials are connected at knots aligned as shown in Fig. 26, where I denotes the number of nodes along $\bar{x}_1$. Each polynomial $\bar{z}_{\{i\}}\left(\bar{x}_1\right)$ is used in an interval $\{i\}$ and intersects the node (i) at $\bar{z}_i(\bar{x}_{1,i})$.

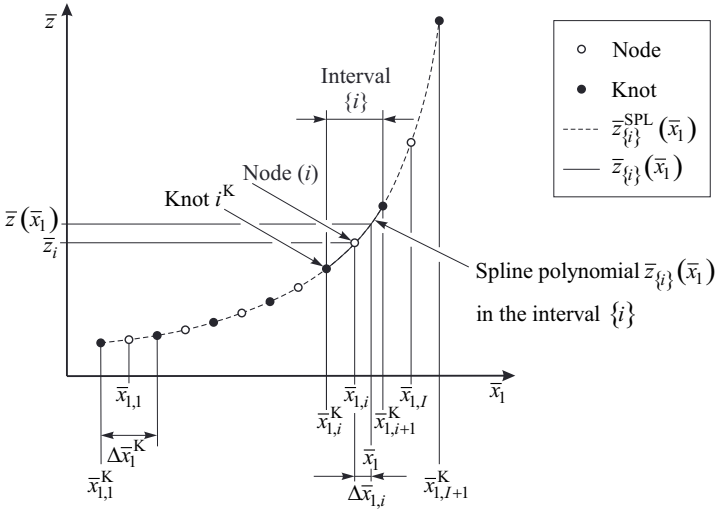

Figure 26: Series of nodes and series of knots with interval $\{i\}$ and spline polynomial $\bar{z}_{\{i\}}\left(\bar{x}_1\right)$.

The $\bar{x}^{\mathrm{K}}_{1,i}$ values of the $I+1$ knots are located at the midpoint between the nodes along $\bar{x}_1$, so that

$$\bar{x}^{\mathrm{K}}_{1,i+1} = \frac{1}{2}\left(\bar{x}_{1,i} + \bar{x}_{1,i+1}\right), \qquad i = 1, \ldots, I-1 \tag{3.2}$$

$$\bar{x}^{\mathrm{K}}_{1,1} = \bar{x}_{1,1} - \frac{1}{2}\left(\bar{x}_{1,2} - \bar{x}_{1,1}\right), \text{ and} \qquad \bar{x}^{\mathrm{K}}_{1,I+1} = \bar{x}_{1,I} + \frac{1}{2}\left(\bar{x}_{1,I} - \bar{x}_{1,I-1}\right). \tag{3.3, 3.4}$$

The number of nodes I is chosen to ensure the required accuracy of the spline function over its full domain of definition $\left[\bar{x}_{1,1} = \bar{x}_1\left(x_{1,\min}\right), \bar{x}_{1,I} = \bar{x}_1\left(x_{1,\max}\right)\right]$. The nodes are distributed equidistantly along $\bar{x}_1$ so that a simple search algorithm can be used to determine the interval $\{i\}$ in the series of knots that fulfills $\bar{x}^{\mathrm{K}}_{1,i} \le \bar{x}_1 < \bar{x}^{\mathrm{K}}_{1,i+1}$ for a given transformed variable $\bar{x}_1$. For equidistant nodes, and therefore equidistant knots, i can easily be calculated from

$$i = \mathrm{floor}\left(\frac{\bar{x}_1 - \bar{x}_{1,1}^{\mathrm{K}}}{\Delta\bar{x}_1^{\mathrm{K}}} \right). \tag{3.5}$$

The distribution of nodes and knots can also be manipulated by piecewise equidistant nodes, in ranges for which $\Delta\bar{x}_1 = \bar{x}_{1,i+1} - \bar{x}_{1,i}$ is constant. Furthermore, the node spacing along x_1 depends on the transformation $\bar{x}_1(x_1)$. Basic principles of transformation techniques are outlined in Sec. 3.1.2.

The $3I$ coefficients a_{ik} of the I spline polynomials are obtained from the following conditions. Each of the I polynomials $\bar{z}_{\{i\}}(\bar{x}_1)$ must intersect the node (i)

$$\bar{z}_{\{i\}}(\bar{x}_{1,i}) = \bar{z}_i(\bar{x}_{1,i}) \qquad i = 1, \ldots, I. \tag{3.6}$$

Furthermore, the $\bar{z}$ values at the inner $I-1$ knots have to be equal for the adjacent polynomials

$$\bar{z}_{\{i\}}(\bar{x}_{1,i+1}^{\mathrm{K}}) = \bar{z}_{\{i+1\}}(\bar{x}_{1,i+1}^{\mathrm{K}}) \qquad i = 1, \ldots, I-1. \tag{3.7}$$

The derivative $(\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1)$ at each of these knots must also be equal

$$\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i\}}(\bar{x}_{1,i+1}^{\mathrm{K}}) = \left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i+1\}}(\bar{x}_{1,i+1}^{\mathrm{K}}) \qquad i = 1, \ldots, I-1. \tag{3.8}$$

At the outer knots, these derivatives are to be calculated from the underlying function $z(x_1)$ with

$$\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i=1\}}(\bar{x}_{1,1}^{\mathrm{K}}) = \frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}(\bar{x}_{1,1}^{\mathrm{K}}) \quad \text{and} \quad \left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{\{i=I\}}(\bar{x}_{1,I+1}^{\mathrm{K}}) = \frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}(\bar{x}_{1,I+1}^{\mathrm{K}}), \tag{3.9, 3.10}$$

where

$$\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1} = \frac{\mathrm{d}\bar{z}}{\mathrm{d}z}\frac{\mathrm{d}z}{\mathrm{d}x_1}\frac{\mathrm{d}x_1}{\mathrm{d}\bar{x}_1}.$$

A comprehensive description of the method along with a computationally efficient implementation is given by Späth [23], where the function values at the outer knots are given instead of the derivatives, Eqs. (3.9) and (3.10). The linear system of $3I$ equations, Eqs. (3.6) – (3.10), is reduced to the $I-1$ equations

$$\begin{aligned} &\frac{1}{\Delta\bar{x}_{1,1}}\left(\frac{5}{2} + \frac{\Delta\bar{x}_{1,2}}{\Delta\bar{x}_{1,1}+\Delta\bar{x}_{1,2}}\right)\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{i^{\mathrm{K}}=2} + \frac{1}{\Delta\bar{x}_{1,1}+\Delta\bar{x}_{1,2}}\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{i^{\mathrm{K}}=3}, \\ &\qquad = \frac{4}{\Delta\bar{x}_{1,1}^2}(\bar{z}_{i=2} - \bar{z}_{i=1}) - \frac{1}{2\Delta\bar{x}_{1,1}}\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}(\bar{x}_{1,1}^{\mathrm{K}}) \end{aligned} \tag{3.11}$$

$$\begin{aligned} &\frac{1}{\Delta\bar{x}_{1,i-1}+\Delta\bar{x}_{1,i}}\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{i^{\mathrm{K}}} \\ &+\frac{1}{\Delta\bar{x}_{1,i}}\left(2 + \frac{\Delta\bar{x}_{1,i-1}}{\Delta\bar{x}_{1,i-1}+\Delta\bar{x}_{1,i}} + \frac{\Delta\bar{x}_{1,i+1}}{\Delta\bar{x}_{1,i}+\Delta\bar{x}_{1,i+1}}\right)\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{i^{\mathrm{K}}+1} \qquad i = 2, \ldots, I-2, \\ &+\frac{1}{\Delta\bar{x}_{1,i}+\Delta\bar{x}_{1,i+1}}\left.\frac{\mathrm{d}\bar{z}}{\mathrm{d}\bar{x}_1}\right|_{i^{\mathrm{K}}+2} = \frac{4}{\Delta\bar{x}_{1,i}^2}(\bar{z}_{i+1} - \bar{z}_i) \end{aligned} \tag{3.12}$$

and

$$\frac{1}{\Delta\overline{x}_{1,I-2}+\Delta\overline{x}_{1,I-1}}\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\bigg|_{i^{\mathrm{K}}=I-1}+\frac{1}{\Delta\overline{x}_{1,I-1}}\left(\frac{5}{2}+\frac{\Delta\overline{x}_{1,I-2}}{\Delta\overline{x}_{1,I-2}+\Delta\overline{x}_{1,I-1}}\right)\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\bigg|_{i^{\mathrm{K}}=I} = \frac{4}{\Delta\overline{x}_{1,I-1}^2}\left(\overline{z}_{i=I}-\overline{z}_{i=I-1}\right)-\frac{1}{2\Delta\overline{x}_{1,I-1}}\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\left(\overline{x}_{1,I+1}^{\mathrm{K}}\right) \tag{3.13}$$

The variables $\Delta\overline{x}_{1,i}$ in the equations above are defined as

$$\Delta\overline{x}_{1,i}=\overline{x}_{1,i+1}-\overline{x}_{1,i}\,. \tag{3.14}$$

The Eqs. (3.11) – (3.13) are solved for the unknown derivatives $\mathrm{d}\overline{z}/\mathrm{d}\overline{x}_1\big|_{i^{\mathrm{K}}}$ at the knots $i^{\mathrm{K}}=2,\ldots,I$. The coefficients a_{ik} are then obtained from

$$a_{i1}=\overline{z}_i\left(\overline{x}_{1,i}\right) \qquad i=1,\ldots,I\,, \tag{3.15}$$

$$a_{i2}=\frac{1}{\Delta\overline{x}_{1,i-1}+\Delta\overline{x}_{1,i}}\left(\Delta\overline{x}_{1,i-1}\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\bigg|_{i^{\mathrm{K}}+1}+\Delta\overline{x}_{1,i}\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\bigg|_{i^{\mathrm{K}}}\right) \qquad i=1,\ldots,I\,, \tag{3.16}$$

and

$$a_{i3}=\frac{1}{\Delta\overline{x}_{1,i-1}+\Delta\overline{x}_{1,i}}\left(\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\bigg|_{i^{\mathrm{K}}+1}-\frac{\mathrm{d}\overline{z}}{\mathrm{d}\overline{x}_1}\bigg|_{i^{\mathrm{K}}}\right) \qquad i=1,\ldots,I\,. \tag{3.17}$$

In Eqs. (3.16) and (3.17), the values of $\Delta\overline{x}_{1,0}$ and $\Delta\overline{x}_{1,I}$ are

$$\Delta\overline{x}_{1,0}=\Delta\overline{x}_{1,1} \qquad \text{and} \qquad \Delta\overline{x}_{1,I}=\Delta\overline{x}_{1,I-1}\,. \tag{3.18, 3.19}$$

Once all the coefficients a_{ik} are determined, they are stored together with the values of the nodes and knots in a look-up table. In order to calculate $z^{\mathrm{SPL}}(x_1)$, the variable x_1 is first transformed into $\overline{x}_1$ with the transformation function $\overline{x}_1(x_1)$. From Eq. (3.5), the index i of the interval is then determined. Finally, the transformed variable $\overline{z}$ is calculated from the spline polynomial $\overline{z}_{\{i\}}(\overline{x}_1)$, Eq. (3.1), and converted to z with the inverse transformation function $z(\overline{z})$.

3.1.2 Transformations

In order to increase the accuracy of a quadratic spline function, the coordinates are transformed in such a way that the third derivative, *i.e.*, the change in curvature, is reduced. Both the independent variable x_1 and the dependent variable z can be transformed with functions of the form $\overline{x}_1(x_1)$ and $\overline{z}(z)$. If $\overline{z}(z)$ is nearly proportional to $\overline{x}_1(x_1)$, then the change in curvature of the transformed function $\overline{z}(\overline{x}_1)$ is smaller than that of $z(x_1)$.

The transformation functions are continuous and monotonic. An analytic solution for the inverse transformation function $z(\overline{z})$ is provided. For the inverse spline function $x_1^{\mathrm{INV}}(z)$, the inverse transformation function $x_1(\overline{x}_1)$ should also be analytical.

The effect of variable transformations is illustrated in Figs. 27 and 28. The untransformed function, see Fig. 27, exhibits a non-zero third derivative, which cannot be described with a quadratic function. If, for instance, z is nearly proportional to $\overline{x}_1(x_1)$, see Fig. 28, the accuracy of the interpolation between the nodes increases because the spline polynomial can better reproduce the transformed function.

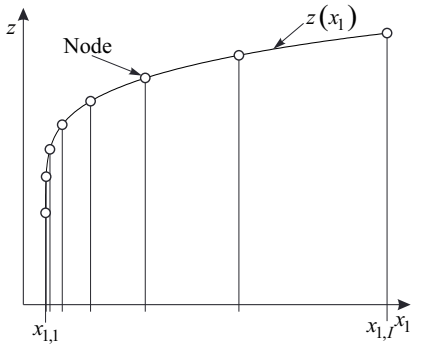

Figure 27: Untransformed function $z(x_1)$ with nodes equidistant in $\bar{x}_1$, rather than in x_1.

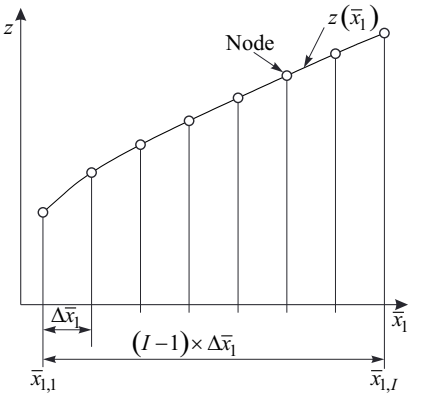

Figure 28: Transformed function $z(\bar{x}_1)$ with nodes equidistant in $\bar{x}_1$.

In many cases, several alternatives of analogous transformations of z and x_1 are feasible. Due to more suitable node distributions, the transformation of x_1 into $\bar{x}_1$ is usually superior to the transformation of z. Another useful approach to efficiently reduce the change in curvature is a transformation of the form $\bar{z}(z, x_1)$. If required, the accuracy and computing time of the spline function itself, and its inverse spline function, must be assessed for the different transformation approaches to determine the trade-off between these criteria.

The concepts explained above offer several alternatives to create a spline function, and can be combined. Considering the requirements for accuracy, computing speed, range of validity, and memory consumption, different transformation techniques must be assessed and the most suitable variant must be chosen.

3.1.3 Inverse Spline Functions

From the spline function $z^{\mathrm{SPL}}(x_1)$, the inverse spline function $x_1^{\mathrm{INV}}(z)$ can be calculated with complete numerical consistency. The transformed variable $\bar{x}_1$ is obtained by inverting the polynomial $\bar{z}_{\{i\}}(\bar{x}_1)$, Eq. (3.1), in the interval $\{i\}$, which results in

$$\bar{x}_{1,\{i\}}^{\mathrm{INV}}(\bar{z}) = \frac{\left(-B_i \pm \sqrt{B_i^2 - 4A_iC_i(\bar{z})}\right)}{2A_i} + \bar{x}_{1,i} \tag{3.20}$$

with

$A_i = a_{i3}$,

$B_i = a_{i2}$, and

$C_i(\bar{z}) = a_{i1} - \bar{z}$.

For a monotonic spline polynomial $\bar{z}_{\{i\}}(\bar{x}_1)$ in the interval $\{i\}$, the sign (±) in Eq. (3.20) is negative if $\mathrm{sgn}(A_i)(\mathrm{d}\bar{z}/\mathrm{d}\bar{x}_1)(\mathrm{d}^2\bar{z}/\mathrm{d}\bar{x}_1^2) < 0$, otherwise it is positive. The inequality yields

$B_i < 0$. Therefore, the sign (±) in Eq. (3.20) equals $\mathrm{sgn}(B_i)$ if the spline polynomial is monotonic in the interval $\{i\}$.

In order to determine the interval index *i* from Eq. (3.5) along $\overline{x}_1$ for a given $\overline{z}$, an auxiliary spline function $\overline{x}_1^{\mathrm{AUX}}(\overline{z})$ is used to calculate an estimate for $\overline{x}_1$.

The procedure for calculating $x_1^{\mathrm{INV}}(z)$ is as follows. First, the variable *z* is transformed into $\overline{z}$. The index *i* of the interval that belongs to $\overline{z}$ is determined with the auxiliary spline function $\overline{x}_1^{\mathrm{AUX}}(\overline{z})$ and Eq. (3.5). The inverse spline polynomial $\overline{x}_{1,\{i\}}^{\mathrm{INV}}(\overline{z})$, Eq. (3.20), is then evaluated. The result must fulfill the condition $\overline{x}_{1,i}^{\mathrm{K}} \le \overline{x}_1 \le \overline{x}_{1,i+1}^{\mathrm{K}}$; otherwise, the index *i* needs to be incremented or decremented, and the calculation repeated. Eventually, $\overline{x}_1$ is converted to x_1 with the inverse transformation function $x_1(\overline{x}_1)$.

Non-monotonic functions have two valid solutions in the interval $\{i\}$ where the extremum of $\overline{z}^{\mathrm{SPL}}(\overline{x}_1)$ is located. This extremum is calculated from

$$\hat{\overline{x}}_{1,\{i\}} = -\frac{B_i}{2A_i} + \overline{x}_{1,i} \quad \text{and} \quad \hat{\overline{z}}_{\{i\}} = a_{i3}\left(\hat{\overline{x}}_{1,\{i\}} - \overline{x}_{1,i}\right)^2 + a_{i2}\left(\hat{\overline{x}}_{1,\{i\}} - \overline{x}_{1,i}\right) + a_{i1}. \qquad (3.21, 3.22)$$

The coefficients of the $\overline{x}_1^{\mathrm{AUX}}(\overline{z})$ and $\overline{z}^{\mathrm{SPL}}(\overline{x}_1)$ spline polynomials along with values of nodes and knots are stored together in a look-up table. This table, and the associated algorithm for calculating the inverse spline function, is written to a source code file for application in computer programs (see Sec. 4).

3.1.4 Derivatives

The first derivative of the spline function $z^{\mathrm{SPL}}(x_1)$ with respect to the independent variable x_1 is calculated analytically from

$$\left(\frac{\mathrm{d}\,z_{\{i\}}}{\mathrm{d}\,x_1}\right) = \left(\frac{\mathrm{d}\,\overline{z}_{\{i\}}}{\mathrm{d}\,\overline{x}_1}\right)\left(\frac{\partial z}{\partial \overline{z}}\right)_{\overline{x}_1}\left(\frac{\mathrm{d}\,\overline{x}_1}{\mathrm{d}\,x_1}\right), \qquad (3.23)$$

where the derivative of the spline function with the transformed variables, Eq. (3.1), within interval $\{i\}$ is calculated from

$$\left(\frac{\mathrm{d}\,\overline{z}_{\{i\}}}{\mathrm{d}\,\overline{x}_1}\right) = a_{i2} + 2a_{i3}\left(\overline{x}_1 - \overline{x}_{1,i}\right). \qquad (3.24)$$

The derivative of the general transformation function $\overline{z}(z, \overline{x}_1)$ is simplified to

$$\left(\frac{\partial z}{\partial \overline{z}}\right)_{\overline{x}_1} = \left(\frac{\mathrm{d}\,z}{\mathrm{d}\,\overline{z}}\right) \qquad (3.25)$$

if the transformation of $\overline{z}$ is independent of $\overline{x}_1$, *i.e.*, $\overline{z} = \overline{z}(z)$.

3.2 Two-Dimensional Spline Functions

3.2.1 Spline Functions

A two-dimensional polynomial spline function $z^{\mathrm{SPL}}(x_1,x_2)$ is a continuous, piecewise-defined function consisting of several spline polynomials. The spline function interpolates values between a set of discrete data points, the so-called grid of nodes (see Fig. 29). The number of nodes *IJ* and their $\left(x_{1,i},x_{2,j}\right)$ locations are chosen to ensure the desired accuracy. The $z_{ij}(x_{1,i},x_{2,j})$ values of the nodes are calculated from the underlying function $z(x_1,x_2)$. The spline polynomials are connected at knots, which can either be equal or unequal to the nodes. For the SBTL method, the knots are located at the midpoint between the nodes along x_1 and x_2 respectively, which results in symmetric boundary conditions leading to superior accuracy [36]. A spline polynomial ranges over a rectangular cell $\{i,j\}$ between four knots and intersects the node within. The *z* positions of the knots result from the spline algorithm as explained below.

In most numerical process simulations, fluid property functions need to be continuously differentiable once. The biquadratic spline polynomial is the simplest approach that is capable of fulfilling this requirement. Furthermore, the biquadratic spline polynomial can easily be inverted. This enables the calculation of numerically consistent backward functions, the so-called inverse spline functions. Therefore, in this document the SBTL method is carried out through the use of biquadratic spline polynomials as opposed to higher order polynomials to create a spline function $z^{\mathrm{SPL}}(x_1,x_2)$ from the underlying function $z(x_1,x_2)$.

In order to increase the accuracy of the spline function, both the independent variables x_1 and x_2, as well as the dependent variable *z*, are transformed into $\overline{x}_1$, $\overline{x}_2$, and $\overline{z}$ so that the transformed spline function yields $\overline{z}^{\mathrm{SPL}}(\overline{x}_1,\overline{x}_2)$. The biquadratic spline interpolation across rectangular cells with continuous first derivatives requires a rectangular grid of nodes in the $(\overline{x}_1,\overline{x}_2)$ projection. Through the use of transformations, the irregularly shaped domain of validity of a function can be transformed into a rectangle, and the distribution of nodes can be controlled more effectively. Alternatively, the function $z(x_1,x_2)$ must be extrapolated. A description of the transformations for two-dimensional spline functions can be found in Sec. 3.2.2.

The spline function is created in transformed coordinates through the use of biquadratic spline polynomials

$$\overline{z}_{\{i,j\}}\left(\overline{x}_1,\overline{x}_2\right)=\sum_{k=1}^{3}\sum_{l=1}^{3}a_{ijkl}\left(\overline{x}_1-\overline{x}_{1,i}\right)^{k-1}\left(\overline{x}_2-\overline{x}_{2,j}\right)^{l-1}, \tag{3.26a}$$

where $\overline{x}_1$ and $\overline{x}_2$ represent the transformed independent variables, $\overline{z}_{\{i,j\}}$ is the transformed dependent variable in the cell $\{i,j\}$, $\overline{x}_{1,i}$ and $\overline{x}_{2,j}$ are the transformed values of the independent variables at the node (*i*,*j*), and a_{ijkl} are the nine coefficients of the spline polynomial valid in the cell $\{i,j\}$. Equation (3.26a) can also be written as

$$\begin{aligned}\bar{z}_{\{i,j\}}\left(\bar{x}_1,\bar{x}_2\right) = {} & a_{ij11} + a_{ij21}\left(\bar{x}_1-\bar{x}_{1,i}\right) + a_{ij31}\left(\bar{x}_1-\bar{x}_{1,i}\right)^2 \\ & + \left[a_{ij12} + a_{ij22}\left(\bar{x}_1-\bar{x}_{1,i}\right) + a_{ij32}\left(\bar{x}_1-\bar{x}_{1,i}\right)^2\right]\left(\bar{x}_2-\bar{x}_{2,j}\right) \\ & + \left[a_{ij13} + a_{ij23}\left(\bar{x}_1-\bar{x}_{1,i}\right) + a_{ij33}\left(\bar{x}_1-\bar{x}_{1,i}\right)^2\right]\left(\bar{x}_2-\bar{x}_{2,j}\right)^2\end{aligned} \quad . \tag{3.26b}$$

It is preferable to connect *IJ* polynomials at knots aligned as shown in the $(\bar{x}_1,\bar{x}_2)$ projection of Fig. 29, where *I* and *J* denote the number of grid lines along $\bar{x}_1$ and $\bar{x}_2$ in the grid of nodes. Each polynomial is used in a cell $\{i,j\}$ and intersects the node $\bar{z}_{\{i,j\}}(\bar{x}_{1,i},\bar{x}_{2,j})$ therein. The $\bar{x}^{\mathrm{K}}_{1,i}$ and $\bar{x}^{\mathrm{K}}_{2,j}$ values of the $(I+1)(J+1)$ knots are located at the midpoint between the nodes along $\bar{x}_1$ and $\bar{x}_2$, so that

$$\bar{x}^{\mathrm{K}}_{1,i+1} = \frac{1}{2}\left(\bar{x}_{1,i} + \bar{x}_{1,i+1}\right), \qquad i = 1, \ldots, I-1 \tag{3.27}$$

$$\bar{x}^{\mathrm{K}}_{2,j+1} = \frac{1}{2}\left(\bar{x}_{2,j} + \bar{x}_{2,j+1}\right), \qquad j = 1, \ldots, J-1 \tag{3.28}$$

$$\bar{x}^{\mathrm{K}}_{1,1} = \bar{x}_{1,1} - \frac{1}{2}\left(\bar{x}_{1,2} - \bar{x}_{1,1}\right), \qquad \bar{x}^{\mathrm{K}}_{1,I+1} = \bar{x}_{1,I} + \frac{1}{2}\left(\bar{x}_{1,I} - \bar{x}_{1,I-1}\right), \tag{3.29, 3.30}$$

$$\bar{x}^{\mathrm{K}}_{2,1} = \bar{x}_{2,1} - \frac{1}{2}\left(\bar{x}_{2,2} - \bar{x}_{2,1}\right), \text{ and } \quad \bar{x}^{\mathrm{K}}_{2,J+1} = \bar{x}_{2,J} + \frac{1}{2}\left(\bar{x}_{2,J} - \bar{x}_{2,J-1}\right). \tag{3.31, 3.32}$$

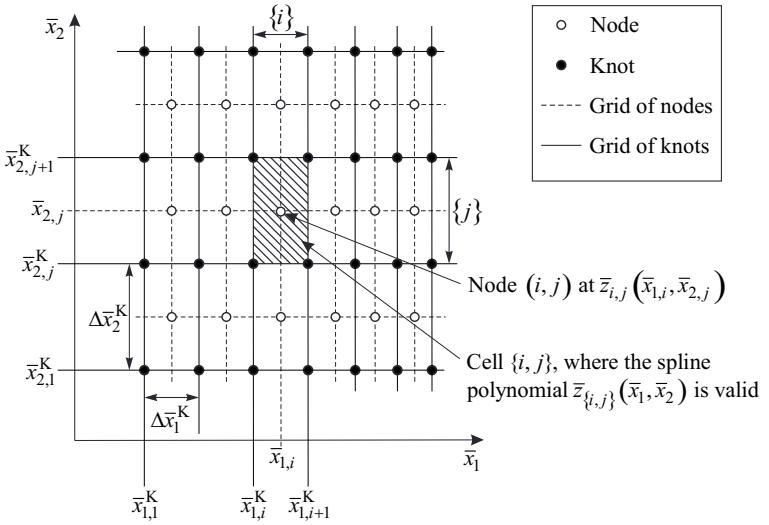

Figure 29: Grid of nodes and grid of knots in the $(\bar{x}_1,\bar{x}_2)$ projection with cell $\{i,j\}$, where the spline polynomial $\bar{z}_{\{i,j\}}\left(\bar{x}_1,\bar{x}_2\right)$ is valid.

The number of nodes *IJ* is chosen to ensure the required accuracy of the spline function over its full domain $\left[\bar{x}_{1,1} = \bar{x}_1\left(x_{1,\min}\right), \bar{x}_{1,I} = \bar{x}_1\left(x_{1,\max}\right)\right]$ and $\left[\bar{x}_{2,1} = \bar{x}_2\left(x_{2,\min}\right), \bar{x}_{2,J} = \bar{x}_2\left(x_{2,\max}\right)\right]$.

The nodes are distributed equidistantly along $\bar{x}_1$ and $\bar{x}_2$, so that a simple search algorithm can be used to determine the cell $\{i,j\}$ in the rectangular grid of knots that fulfills $\bar{x}_{1,i}^{\mathrm{K}} \le \bar{x}_1 < \bar{x}_{1,i+1}^{\mathrm{K}}$ and $\bar{x}_{2,j}^{\mathrm{K}} \le \bar{x}_2 < \bar{x}_{2,j+1}^{\mathrm{K}}$ for a given pair of transformed variables $(\bar{x}_1, \bar{x}_2)$. For equidistant nodes, and therefore equidistant knots, the indices *i* and *j* can easily be calculated from

$$i = \mathrm{floor}\left(\frac{\bar{x}_1 - \bar{x}_{1,1}^{\mathrm{K}}}{\Delta\bar{x}_1^{\mathrm{K}}}\right) \qquad \text{and} \qquad j = \mathrm{floor}\left(\frac{\bar{x}_2 - \bar{x}_{2,1}^{\mathrm{K}}}{\Delta\bar{x}_2^{\mathrm{K}}}\right). \tag{3.33, 3.34}$$

The distribution of nodes and knots can also be manipulated by piecewise equidistant nodes, in ranges for which $\Delta\bar{x}_1 = \bar{x}_{1,i+1} - \bar{x}_{1,i}$ and $\Delta\bar{x}_2 = \bar{x}_{2,j+1} - \bar{x}_{2,j}$, respectively, are constant. Furthermore, the node spacing along x_1 and x_2 depends on the transformations $\bar{x}_1(x_1)$ and $\bar{x}_2(x_2)$. Basic principles of these transformations are outlined in Sec. 3.2.2.

The 9*IJ* coefficients a_{ijkl} of all spline polynomials are determined as explained by Späth [36]. Figure 30 illustrates the boundary conditions at a cell, where the superscript K denotes the grid of knots. Each of the *IJ* polynomials $\bar{z}_{\{i,j\}}(\bar{x}_1, \bar{x}_2)$ intersects the node (*i*,*j*). The $\bar{z}$ values at the midpoints of the cell boundaries (i^{K}, j), $(i^{\mathrm{K}}+1, j)$, (i, j^{K}), and $(i, j^{\mathrm{K}}+1)$, marked with gray circles in Fig. 30, are equal to the corresponding values of the adjacent cells. Furthermore, the derivatives $(\partial\bar{z}/\partial\bar{x}_1)_{\bar{x}_2}$ at (i^{K}, j) and $(i^{\mathrm{K}}+1, j)$, as well as $(\partial\bar{z}/\partial\bar{x}_2)_{\bar{x}_1}$ at (i, j^{K}) and $(i, j^{\mathrm{K}}+1)$, are equal to the corresponding derivatives of the adjacent cells. In addition, the $\bar{z}$ values and the crossed derivatives $\left(\partial^2\bar{z}/(\partial\bar{x}_1\partial\bar{x}_2)\right)$ at the four knots at the corners $(i^{\mathrm{K}}, j^{\mathrm{K}})$, $(i^{\mathrm{K}}, j^{\mathrm{K}}+1)$, $(i^{\mathrm{K}}+1, j^{\mathrm{K}})$, and $(i^{\mathrm{K}}+1, j^{\mathrm{K}}+1)$ are equal to the corresponding values of the neighboring cells. For these conditions, the required derivatives at the cell boundaries are calculated considering the requirement that the resulting biquadratic spline function is continuously differentiable once.

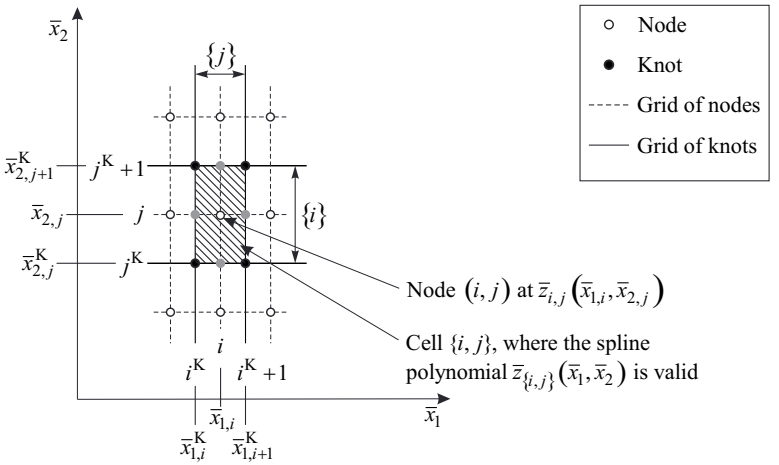

Figure 30: Locations of points where boundary conditions are defined for a cell.

The variables $\Delta\bar{x}_{1,i}$ and $\Delta\bar{x}_{2,j}$ in the equations below are defined as

$$\Delta\bar{x}_{1,i} = \bar{x}_{1,i+1} - \bar{x}_{1,i} \qquad \text{and} \qquad \Delta\bar{x}_{2,j} = \bar{x}_{2,j+1} - \bar{x}_{2,j} \,. \tag{3.35}$$

For given derivatives $\left(\partial\bar{z}/\partial\bar{x}_1\right)_{x_2}$ at the left and right boundaries of the grid of knots

$$\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j} = \left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\left(\bar{x}^{\mathrm{K}}_{1,i},\bar{x}_{2,j}\right) \qquad i^{\mathrm{K}} = 1, I+1\,, \qquad j = 1,\ldots,J\,, \tag{3.36}$$

the derivatives at the remaining knots $\left(i^{\mathrm{K}}, j\right)$, with $i^{\mathrm{K}} = 2,\ldots,I$ and $j = 1,\ldots,J$, are obtained by solving for each grid line j, with $j = 1,\ldots,J$,

$$\begin{aligned}\frac{1}{\Delta\bar{x}_{1,1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{1,2}}{\Delta\bar{x}_{1,1}+\Delta\bar{x}_{1,2}}\right)\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}=2,j} + \frac{1}{\Delta\bar{x}_{1,1}+\Delta\bar{x}_{1,2}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}=3,j} \\ = \frac{4}{\Delta\bar{x}^2_{1,1}}\left(\bar{z}_{i=2,j}-\bar{z}_{i=1,j}\right) - \frac{1}{2\Delta\bar{x}_{1,1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}=1,j}\end{aligned}\,, \tag{3.37}$$

$$\begin{aligned}&\frac{1}{\Delta\bar{x}_{1,i-1}+\Delta\bar{x}_{1,i}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j} \\ &+\frac{1}{\Delta\bar{x}_{1,i}}\left(2+\frac{\Delta\bar{x}_{1,i-1}}{\Delta\bar{x}_{1,i-1}+\Delta\bar{x}_{1,i}}+\frac{\Delta\bar{x}_{1,i+1}}{\Delta\bar{x}_{1,i}+\Delta\bar{x}_{1,i+1}}\right)\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}+1,j} \qquad i = 2,\ldots,I-2\,, \\ &+\frac{1}{\Delta\bar{x}_{1,i}+\Delta\bar{x}_{1,i+1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}+2,j} = \frac{4}{\Delta\bar{x}^2_{1,i}}\left(\bar{z}_{i+1,j}-\bar{z}_{i,j}\right)\end{aligned} \tag{3.38}$$

and

$$\begin{aligned}\frac{1}{\Delta\bar{x}_{1,I-2}+\Delta\bar{x}_{1,I-1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}=I-1,j} + \frac{1}{\Delta\bar{x}_{1,I-1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{1,I-2}}{\Delta\bar{x}_{1,I-2}+\Delta\bar{x}_{1,I-1}}\right)\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}=I,j} \\ = \frac{4}{\Delta\bar{x}^2_{1,I-1}}\left(\bar{z}_{i=I,j}-\bar{z}_{i=I-1,j}\right) - \frac{1}{2\Delta\bar{x}_{1,I-1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}}=I+1,j}\end{aligned}\,. \tag{3.39}$$

Analogously, for given derivatives $\left(\partial\bar{z}/\partial\bar{x}_2\right)_{x_1}$ at the lower and upper boundaries of the grid of knots

$$\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}} = \left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\left(\bar{x}_{1,i},\bar{x}^{\mathrm{K}}_{2,j}\right) \qquad i = 1,\ldots,I\,, \qquad j^{\mathrm{K}} = 1, J+1, \tag{3.40}$$

the derivatives at the remaining knots $\left(i, j^{\mathrm{K}}\right)$, with $i = 1,\ldots,I$ and $j^{\mathrm{K}} = 2,\ldots,J$, are obtained by solving for each grid line i, with $i = 1,\ldots,I$,

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{2,1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{2,2}}{\Delta\bar{x}_{2,1}+\Delta\bar{x}_{2,2}}\right)\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}=2}+\frac{1}{\Delta\bar{x}_{2,1}+\Delta\bar{x}_{2,2}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}=3} \\
&\qquad=\frac{4}{\Delta\bar{x}_{2,1}^2}\left(\bar{z}_{i,j=2}-\bar{z}_{i,j=1}\right)-\frac{1}{2\Delta\bar{x}_{2,1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}=1}
\end{aligned}, \tag{3.41}$$

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{2,j-1}+\Delta\bar{x}_{2,j}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}} \\
&+\frac{1}{\Delta\bar{x}_{2,j}}\left(2+\frac{\Delta\bar{x}_{2,j-1}}{\Delta\bar{x}_{2,j-1}+\Delta\bar{x}_{2,j}}+\frac{\Delta\bar{x}_{2,j+1}}{\Delta\bar{x}_{2,j}+\Delta\bar{x}_{2,j+1}}\right)\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}+1} \qquad j=2,\ldots,J-2, \\
&+\frac{1}{\Delta\bar{x}_{2,j}+\Delta\bar{x}_{2,j+1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}+2}=\frac{4}{\Delta\bar{x}_{2,j}^2}\left(\bar{z}_{i,j+1}-\bar{z}_{i,j}\right)
\end{aligned} \tag{3.42}$$

and

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{2,J-2}+\Delta\bar{x}_{2,J-1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}=J-1} \\
&+\frac{1}{\Delta\bar{x}_{2,J-1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{2,J-2}}{\Delta\bar{x}_{2,J-2}+\Delta\bar{x}_{2,J-1}}\right)\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}=J} \\
&=\frac{4}{\Delta\bar{x}_{2,J-1}^2}\left(\bar{z}_{i,j=J}-\bar{z}_{i,j=J-1}\right)-\frac{1}{2\Delta\bar{x}_{2,J-1}}\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}=J+1}
\end{aligned}. \tag{3.43}$$

Now, for given derivatives $\left(\partial^2\bar{z}/\left(\partial\bar{x}_1\partial\bar{x}_2\right)\right)$ at the four corners of the grid of knots

$$\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}}=\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\left(\bar{x}_{1,i}^{\mathrm{K}},\bar{x}_{2,j}^{\mathrm{K}}\right) \qquad i^{\mathrm{K}}=1,I+1, \qquad j^{\mathrm{K}}=1,J+1, \tag{3.44}$$

the derivatives at the remaining knots $\left(i^{\mathrm{K}},j^{\mathrm{K}}\right)$, with $i^{\mathrm{K}}=2,\ldots,I$ and $j^{\mathrm{K}}=1,J+1$, are obtained by solving for each grid line $j^{\mathrm{K}}=1$ and $j^{\mathrm{K}}=J+1$

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{1,1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{1,2}}{\Delta\bar{x}_{1,1}+\Delta\bar{x}_{1,2}}\right)\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}=2,j^{\mathrm{K}}}+\frac{1}{\Delta\bar{x}_{1,1}+\Delta\bar{x}_{1,2}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}=3,j^{\mathrm{K}}} \\
&\qquad=\frac{4}{\Delta\bar{x}_{1,1}^2}\left[\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i=2,j^{\mathrm{K}}}-\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i=1,j^{\mathrm{K}}}\right]-\frac{1}{2\Delta\bar{x}_{1,1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}=1,j^{\mathrm{K}}}
\end{aligned}, \tag{3.45}$$

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{1,i-1}+\Delta\bar{x}_{1,i}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}}\\
&+\frac{1}{\Delta\bar{x}_{1,i}}\left(2+\frac{\Delta\bar{x}_{1,i-1}}{\Delta\bar{x}_{1,i-1}+\Delta\bar{x}_{1,i}}+\frac{\Delta\bar{x}_{1,i+1}}{\Delta\bar{x}_{1,i}+\Delta\bar{x}_{1,i+1}}\right)\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}+1,j^{\mathrm{K}}}\\
&+\frac{1}{\Delta\bar{x}_{1,i}+\Delta\bar{x}_{1,i+1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}+2,j^{\mathrm{K}}}\\
&\qquad=\frac{4}{\Delta\bar{x}_{1,i}^2}\left[\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i+1,j^{\mathrm{K}}}-\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i,j^{\mathrm{K}}}\right]
\end{aligned}\qquad i=2,\ldots,I-2, \quad (3.46)$$

and

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{1,I-2}+\Delta\bar{x}_{1,I-1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}=I-1,j^{\mathrm{K}}}\\
&+\frac{1}{\Delta\bar{x}_{1,I-1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{1,I-2}}{\Delta\bar{x}_{1,I-2}+\Delta\bar{x}_{1,I-1}}\right)\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}=I,j^{\mathrm{K}}}\\
&=\frac{4}{\Delta\bar{x}_{1,I-1}^2}\left[\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i=I,j^{\mathrm{K}}}-\left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Bigg|_{i=I-1,j^{\mathrm{K}}}\right]-\frac{1}{2\Delta\bar{x}_{1,I-1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}}=I+1,j^{\mathrm{K}}}
\end{aligned}. \quad (3.47)$$

Now, all derivatives $\left(\partial^2\bar{z}/(\partial\bar{x}_1\partial\bar{x}_2)\right)$ at the knots $\left(i^{\mathrm{K}},j^{\mathrm{K}}\right)$, with $i^{\mathrm{K}}=1,\ldots,I+1$ and $j^{\mathrm{K}}=1,J+1$, are known. At the remaining knots $\left(i^{\mathrm{K}},j^{\mathrm{K}}\right)$, with $i^{\mathrm{K}}=1,\ldots,I+1$ and $j^{\mathrm{K}}=2,\ldots,J$, these derivatives are obtained by solving for each grid line $i^{\mathrm{K}}=1,\ldots,I+1$

$$\begin{aligned}
&\frac{1}{\Delta\bar{x}_{2,1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{2,2}}{\Delta\bar{x}_{2,1}+\Delta\bar{x}_{2,2}}\right)\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}=2}+\frac{1}{\Delta\bar{x}_{2,1}+\Delta\bar{x}_{2,2}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}=3}\\
&\qquad=\frac{4}{\Delta\bar{x}_{2,1}^2}\left[\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j=2}-\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j=1}\right]-\frac{1}{2\Delta\bar{x}_{2,1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}=1}
\end{aligned}, \quad (3.48)$$

$$\frac{1}{\Delta\bar{x}_{2,j-1}+\Delta\bar{x}_{2,j}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}}$$
$$+\frac{1}{\Delta\bar{x}_{2,j}}\left(2+\frac{\Delta\bar{x}_{2,j-1}}{\Delta\bar{x}_{2,j-1}+\Delta\bar{x}_{2,j}}+\frac{\Delta\bar{x}_{2,j+1}}{\Delta\bar{x}_{2,j}+\Delta\bar{x}_{2,j+1}}\right)\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}+1} \qquad j=2,\ldots,J-2, \quad (3.49)$$
$$+\frac{1}{\Delta\bar{x}_{2,j}+\Delta\bar{x}_{2,j+1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}+2}$$
$$=\frac{4}{\Delta\bar{x}_{2,j}^2}\left[\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j+1}-\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j}\right]$$

and

$$\frac{1}{\Delta\bar{x}_{2,J-2}+\Delta\bar{x}_{2,J-1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}=J-1}$$
$$+\frac{1}{\Delta\bar{x}_{2,J-1}}\left(\frac{5}{2}+\frac{\Delta\bar{x}_{2,J-2}}{\Delta\bar{x}_{2,J-2}+\Delta\bar{x}_{2,J-1}}\right)\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}=J} \quad . \quad (3.50)$$
$$=\frac{4}{\Delta\bar{x}_{2,J-1}^2}\left[\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j=J}-\left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Bigg|_{i^{\mathrm{K}},j=J-1}\right]-\frac{1}{2\Delta\bar{x}_{2,J-1}}\left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Bigg|_{i^{\mathrm{K}},j^{\mathrm{K}}=J+1}$$

Each set of equations, Eqs. (3.37) – (3.39), (3.41) – (3.43), and (3.48) – (3.50), provides a symmetric tridiagonal coefficient matrix. Such sets of equations can be solved through the use of simplified algorithms as discussed in [36].

Now, the 9 coefficients for each of the IJ cells $\{i,j\}$ are calculated from

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \left[\boldsymbol{V}\left(\bar{x}_{1,i}\right)\right]^{-1}\boldsymbol{C}\left[\boldsymbol{V}\left(\bar{x}_{2,j}\right)\right]^{-\mathrm{T}}, \quad (3.51)$$

where

$$\boldsymbol{C}=\begin{bmatrix} \left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Big|_{i^{\mathrm{K}},j^{\mathrm{K}}} & \left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Big|_{i^{\mathrm{K}},j} & \left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Big|_{i^{\mathrm{K}},j^{\mathrm{K}}+1} \\ \left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Big|_{i,j^{\mathrm{K}}} & \bar{z}_{i,j} & \left(\frac{\partial\bar{z}}{\partial\bar{x}_2}\right)_{\bar{x}_1}\Big|_{i,j^{\mathrm{K}}+1} \\ \left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Big|_{i^{\mathrm{K}}+1,j^{\mathrm{K}}} & \left(\frac{\partial\bar{z}}{\partial\bar{x}_1}\right)_{\bar{x}_2}\Big|_{i^{\mathrm{K}}+1,j} & \left(\frac{\partial^2\bar{z}}{\partial\bar{x}_1\partial\bar{x}_2}\right)\Big|_{i^{\mathrm{K}}+1,j^{\mathrm{K}}+1} \end{bmatrix}, \quad (3.52)$$

$$\left[\boldsymbol{V}\left(\overline{x}_{1,i}\right)\right]^{-1} = \begin{bmatrix} 0 & 1 & 0 \\ \dfrac{\Delta\overline{x}_{1,i}}{\Delta\overline{x}_{1,i-1}+\Delta\overline{x}_{1,i}} & 0 & \dfrac{\Delta\overline{x}_{1,i-1}}{\Delta\overline{x}_{1,i-1}+\Delta\overline{x}_{1,i}} \\ \dfrac{-1}{\Delta\overline{x}_{1,i-1}+\Delta\overline{x}_{1,i}} & 0 & \dfrac{1}{\Delta\overline{x}_{1,i-1}+\Delta\overline{x}_{1,i}} \end{bmatrix}, \tag{3.53}$$

and

$$\left[\boldsymbol{V}\left(\overline{x}_{2,j}\right)\right]^{-\mathrm{T}} = \begin{bmatrix} 0 & \dfrac{\Delta\overline{x}_{2,j}}{\Delta\overline{x}_{2,j-1}+\Delta\overline{x}_{2,j}} & \dfrac{-1}{\Delta\overline{x}_{2,j-1}+\Delta\overline{x}_{2,j}} \\ 1 & 0 & 0 \\ 0 & \dfrac{\Delta\overline{x}_{2,j-1}}{\Delta\overline{x}_{2,j-1}+\Delta\overline{x}_{2,j}} & \dfrac{1}{\Delta\overline{x}_{2,j-1}+\Delta\overline{x}_{2,j}} \end{bmatrix}. \tag{3.54}$$

In Eqs. (3.53) and (3.54), the values of $\Delta\overline{x}_{1,0}$, $\Delta\overline{x}_{1,I}$, $\Delta\overline{x}_{2,0}$, and $\Delta\overline{x}_{2,J}$ are

$$\Delta\overline{x}_{1,0} = \Delta\overline{x}_{1,1}, \qquad \Delta\overline{x}_{1,I} = \Delta\overline{x}_{1,I-1}, \tag{3.55, 3.56}$$

$$\Delta\overline{x}_{2,0} = \Delta\overline{x}_{2,1}, \quad \text{and} \qquad \Delta\overline{x}_{2,J} = \Delta\overline{x}_{2,J-1}. \tag{3.57, 3.58}$$

The continuous behavior of the resulting spline function and its first derivatives at the boundaries between the cells is mathematically proven in [36]. The number and distribution of nodes is optimized to ensure the required accuracy of $z^{\mathrm{SPL}}(x_1,x_2)$ over the whole range of validity. Once all the coefficients a_{ijkl} are determined, they are stored together with the values of the nodes and knots in a look-up table. This table and the associated algorithm for calculating the spline function is written to a source code file for application in computer programs (see Sec. 4).

In order to calculate $z^{\mathrm{SPL}}(x_1,x_2)$, the variables x_1 and x_2 are first transformed into $\overline{x}_1$ and $\overline{x}_2$ with the corresponding transformation functions. Equations (3.33, 3.34) give the indices *i* and *j* of the corresponding cell. The transformed variable $\overline{z}$ is then calculated from the spline polynomial $\overline{z}_{\{i,j\}}(\overline{x}_1,\overline{x}_2)$, Eq. (3.26), and is converted to *z* with the inverse transformation function $z(\overline{z})$.

3.2.2 Transformations

In order to increase the accuracy of a biquadratic spline function, the coordinates are transformed in such a way that the third derivatives, *i.e.*, the change in curvature, is reduced. Both independent variables x_1 and x_2, as well as the dependent variable *z*, can be transformed with functions of the form $\overline{x}_1(x_1)$, $\overline{x}_2(x_2)$, and $\overline{z}(z)$. If $\overline{z}(z)$ is nearly proportional to $\overline{x}_1(x_1)$ at constant $\overline{x}_2$ and $\overline{z}(z)$ is nearly proportional to $\overline{x}_2(x_2)$ at constant $\overline{x}_1$, then the change in curvature of the transformed function $\overline{z}(\overline{x}_1,\overline{x}_2)$ is reduced as compared to that of $z(x_1,x_2)$.

The transformation functions must be continuous and monotonic. An analytic solution for the inverse transformation function $z(\overline{z})$ is needed. For the inverse spline functions $x_1^{\mathrm{INV}}(z,x_2)$ and $x_2^{\mathrm{INV}}(x_1,z)$, the inverse transformation functions $x_1(\overline{x}_1)$ and $x_2(\overline{x}_2)$ should also be analytical.

In Secs. 5 and 6, where the SBTL method is applied to several property functions, the increased accuracy resulting from transformations is demonstrated. In many cases, several alternative analogous transformations of z, x_1, and x_2 are feasible. Due to more suitable node distributions, transformations of x_1 and x_2 into $\overline{x}_1$ and $\overline{x}_2$ are usually superior to the transformation of z. If required, accuracy and computing time of the spline function itself and its inverse spline functions must be assessed for the different transformation approaches to determine the trade-off between these criteria.

Fast, non-iterative algorithms to determine the cell $\{i,j\}$ for a given pair of transformed variables $(\overline{x}_1, \overline{x}_2)$ require a rectangular cell structure. In combination with the demands for the continuity of the biquadratic spline function and its first derivatives, this leads to a grid of nodes with a rectangular outer boundary in the $(\overline{x}_1, \overline{x}_2)$ plane. This rectangle must include the required range of validity. States beyond the range of validity must be extrapolated from the equation of state or with suitable extrapolation techniques.

In order to avoid extrapolations and to more efficiently control the node distribution across the grid within the range of validity, additional variable transformations can be applied. Through the use of these so-called scaling transformations of the form $\overline{x}_1(x_1, x_2)$ and/or $\overline{x}_2(x_2, x_1)$, the irregular shaped range of validity is converted into a rectangle. For this purpose, the boundaries of the range of validity are described with auxiliary spline functions of the form $x_{1,\min}(x_2)$, $x_{1,\max}(x_2)$, $x_{2,\min}(x_1)$, and $x_{2,\max}(x_1)$.

If, for instance, the variable x_1 is to be scaled between the boundary curves $x_{1,\min}(x_2)$ and $x_{1,\max}(x_2)$, see Fig. 31, the form of the scaled variable transformation reads

$$\overline{x}_1(x_1, x_2) = \overline{x}_1\left(x_1, x_{1,\min}(x_2), x_{1,\max}(x_2)\right). \tag{3.59}$$

For example, Eq. (3.59) could be expressed as a linear scaling function for x_1 between $x_{1,\min}(x_2)$ and $x_{1,\max}(x_2)$ with

$$\overline{x}_1(x_1, x_2) = \frac{\overline{x}_{1,\max} - \overline{x}_{1,\min}}{x_{1,\max}(x_2) - x_{1,\min}(x_2)}\left(x_1 - x_{1,\min}(x_2)\right) + \overline{x}_{1,\min}, \tag{3.60}$$

where $\overline{x}_{1,\min}$ and $\overline{x}_{1,\max}$ are free parameters chosen appropriately as the minimum and maximum values of the transformed coordinate. Figure 32 shows the range of validity and the grid of nodes in transformed coordinates.

The spline functions for the liquid phase in the (v,u) plane (see Sec. 5.1) are insightful examples for these transformation techniques. Another useful transformation approach results from the combination of the dependent variable z and the independent variables x_1 and/or x_2. A transformation of the form $\overline{z}(z, x_1, x_2)$ can be used in some cases to efficiently reduce the change in curvature. If, for instance, the specific volume in the gas phase is calculated from the pressure p and another property x_2, *i.e.*, $v(x_1 = p, x_2)$, the transformed specific volume $\overline{v}(v, p) = pv$ is preferably used as the dependent variable. In Sec. 5.2, the spline-based property function $v^{\mathrm{G}}(p,h)$ shows how this variable transformation technique is applied.

The concepts explained above offer several alternatives to create a spline function, and can be combined. Considering the requirements for accuracy, computing speed, range of validity, and memory consumption, transformation techniques must be assessed and the most suitable variant must be chosen.

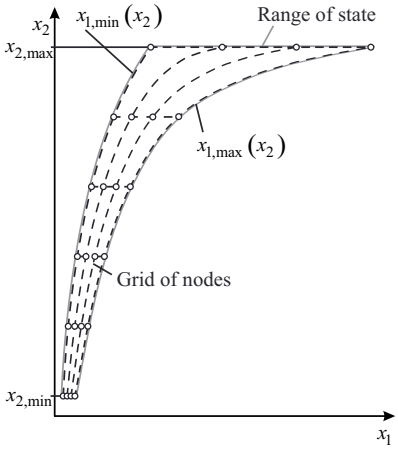

Figure 31: Projection of the grid of nodes in untransfomed coordinates.

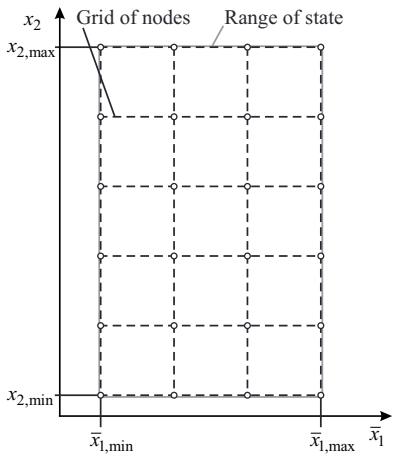

Figure 32: Projection of the grid of nodes in the $(\bar{x}_1, x_2)$ plane for a $\bar{x}_1(x_1, x_2)$ transformation.

3.2.3 Inverse Spline Functions

From the spline function $z^{\mathrm{SPL}}(x_1, x_2)$, the inverse spline functions $x_1^{\mathrm{INV}}(z, x_2)$ and $x_2^{\mathrm{INV}}(x_1, z)$ can be calculated with complete numerical consistency. This is demonstrated for $x_1^{\mathrm{INV}}(z, x_2)$. The transformed variable $\bar{x}_1$ is obtained by solving the polynomial $\bar{z}_{\{i,j\}}(\bar{x}_1, \bar{x}_2)$, Eq. (3.26), which results in

$$\bar{x}_{1,\{i,j\}}^{\mathrm{INV}}(\bar{z}, \bar{x}_2) = \frac{\left(-B_{ij} \pm \sqrt{B_{ij}^2 - 4A_{ij}C_{ij}(\bar{z})}\right)}{2A_{ij}} + \bar{x}_{1,i} \tag{3.61}$$

with

$$A_{ij} = a_{ij31} + \left(\bar{x}_2 - \bar{x}_{2,j}\right)\left(a_{ij32} + a_{ij33}\left(\bar{x}_2 - \bar{x}_{2,j}\right)\right),$$

$$B_{ij} = a_{ij21} + \left(\bar{x}_2 - \bar{x}_{2,j}\right)\left(a_{ij22} + a_{ij23}\left(\bar{x}_2 - \bar{x}_{2,j}\right)\right), \text{ and}$$

$$C_{ij}(\bar{z}) = a_{ij11} + \left(\bar{x}_2 - \bar{x}_{2,j}\right)\left(a_{ij12} + a_{ij13}\left(\bar{x}_2 - \bar{x}_{2,j}\right)\right) - \bar{z}\,.$$

For a monotonic function $\bar{z}_{\{i,j\}}(\bar{x}_1)_{\bar{x}_2}$ in the cell $\{i,j\}$, the sign (±) in Eq. (3.61) is negative if $\mathrm{sgn}\left(A_{ij}\right)\left(\partial\bar{z}/\partial\bar{x}_1\right)_{\bar{x}_2}\left(\partial^2\bar{z}/\partial\bar{x}_1{}^2\right)_{\bar{x}_2} < 0$, otherwise it is positive. The inequality yields $B_{ij} < 0$. Therefore, the sign (±) in Eq. (3.61) equals $\mathrm{sgn}(B_{ij})$ if the spline polynomial is monotonic in the cell $\{i,j\}$ for fixed values of $\bar{x}_2$.

In order to determine the cell indices *i* and *j* from Eqs. (3.33) and (3.34) in the $(\bar{x}_1, \bar{x}_2)$ plane for given values of $\bar{z}$ and $\bar{x}_2$, an auxiliary spline function $\bar{x}_1^{\mathrm{AUX}}(\bar{z}, \bar{x}_2)$ is used to calculate an estimate for $\bar{x}_1$.

To calculate the inverse spline function $x_1^{\mathrm{INV}}(z,x_2)$, z and x_2 are first transformed into $\bar{z}$ and $\bar{x}_2$. The cell indices *i* and *j* that belong to the given values for $(\bar{z},\bar{x}_2)$ are then determined with the auxiliary spline function $\bar{x}_1^{\mathrm{AUX}}(\bar{z},\bar{x}_2)$ and Eqs. (3.33) and (3.34). Then, the inverse spline polynomial $\bar{x}_{1,\{i,j\}}^{\mathrm{INV}}(\bar{z},\bar{x}_2)$, Eq. (3.61), is calculated. The result must fulfill the condition $\bar{x}_{1,i}^{\mathrm{K}} \le \bar{x}_1 \le \bar{x}_{1,i+1}^{\mathrm{K}}$; otherwise, the index *i* needs to be incremented or decremented and the calculation repeated. Eventually, $\bar{x}_1$ is converted to x_1 with the inverse transformation function $x_1(\bar{x}_1)$.

Non-monotonic functions have two valid solutions in the cell $\{i,j\}$ where the extremum of $\bar{z}_{\{i,j\}}(\bar{x}_1)_{\bar{x}_2}$ is located. This extremum is calculated from

$$\hat{\bar{x}}_{1,\{i,j\}} = -\frac{B_{ij}}{2A_{ij}} + \bar{x}_{1,i} \tag{3.62}$$

and

$$\begin{aligned}\hat{\bar{z}}_{\{i,j\}} = {} & A_{ij}\left(\hat{\bar{x}}_{1,\{i,j\}} - \bar{x}_{1,i}\right)^2 + B_{ij}\left(\hat{\bar{x}}_{1,\{i,j\}} - \bar{x}_{1,i}\right) \\ & + a_{ij11} + \left(\bar{x}_2 - \bar{x}_{2,j}\right)\left(a_{ij12} + a_{ij13}\left(\bar{x}_2 - \bar{x}_{2,j}\right)\right)\end{aligned} \tag{3.63}$$

If a scaling transformation (see Sec. 3.2.2) is applied with the dependent variable of the inverse spline function, *e.g.*, x_1, where x_2 is scaled with $\bar{x}_2(x_2,x_1)$, an analytic solution of the inverse spline function cannot be provided. Instead, a one-dimensional Newton iteration should be applied to solve

$$f(x_1) = 0 = z^{\mathrm{SPL}}(x_1)_{x_2} - z \tag{3.64}$$

with the following procedure

$$x_{1,k+1} = x_{1,k} - \frac{f(x_{1,k})}{\dfrac{\mathrm{d}f}{\mathrm{d}x_1}(x_{1,k})}, \tag{3.65}$$

where

$$\frac{\mathrm{d}f}{\mathrm{d}x_1}(x_{1,k}) = \left(\frac{\partial z}{\partial x_1}\right)_{x_2}(x_{1,k}). \tag{3.66}$$

The calculation of derivatives of spline functions is explained in Sec. 3.2.4.

The coefficients of the $\bar{x}_1^{\mathrm{AUX}}(\bar{z},\bar{x}_2)$ and $\bar{z}^{\mathrm{SPL}}(\bar{x}_1,\bar{x}_2)$ spline polynomials along with values of nodes and knots are stored together in a look-up table. This table, and the associated algorithm for calculating the inverse spline function, is written to a source code file for application in computer programs (see Sec. 4).

The inverse spline function $x_2^{\mathrm{INV}}(x_1,z)$ can be calculated in a similar manner with the equation

$$\bar{x}_{2,\{i,j\}}^{\mathrm{INV}}(\bar{x}_1,\bar{z}) = \frac{\left(-B_{ij} \pm \sqrt{B_{ij}^2 - 4A_{ij}C_{ij}(\bar{z})}\right)}{2A_{ij}} + \bar{x}_{2,j} \tag{3.67}$$

where

$$A_{ij} = a_{ij13} + \left(\bar{x}_1 - \bar{x}_{1,i}\right)\left(a_{ij23} + a_{ij33}\left(\bar{x}_1 - \bar{x}_{1,i}\right)\right),$$

$$B_{ij} = a_{ij12} + \left(\overline{x}_1 - \overline{x}_{1,i}\right)\left(a_{ij22} + a_{ij32}\left(\overline{x}_1 - \overline{x}_{1,i}\right)\right), \text{ and}$$

$$C_{ij}\left(\overline{z}\right) = a_{ij11} + \left(\overline{x}_1 - \overline{x}_{1,i}\right)\left(a_{ij21} + a_{ij31}\left(\overline{x}_1 - \overline{x}_{1,i}\right)\right) - \overline{z}\,.$$

For monotonic functions $\overline{z}_{\{i,j\}}\left(\overline{x}_2\right)_{\overline{x}_1}$ in the cell $\{i,j\}$, the sign (±) in Eq. (3.67) equals $\mathrm{sgn}(B_{ij})$, as described earlier in this section.

Algorithms for the calculation of inverse functions in the two-phase region depend on the formulation of the equilibrium condition. Practical examples are given in the Appendix.

3.2.4 Derivatives

The first derivatives of the spline function $z^{\mathrm{SPL}}(x_1, x_2)$ with respect to the independent variables x_1 and x_2 are calculated analytically from

$$\left(\frac{\partial z_{\{i,j\}}}{\partial x_1}\right)_{x_2} = \frac{\left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_1}\right)_{\overline{x}_2}\left(\frac{\partial x_2}{\partial \overline{x}_2}\right)_{\overline{x}_1} - \left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_2}\right)_{\overline{x}_1}\left(\frac{\partial x_2}{\partial \overline{x}_1}\right)_{\overline{x}_2}}{\left(\frac{\partial x_1}{\partial \overline{x}_1}\right)_{\overline{x}_2}\left(\frac{\partial x_2}{\partial \overline{x}_2}\right)_{\overline{x}_1} - \left(\frac{\partial x_1}{\partial \overline{x}_2}\right)_{\overline{x}_1}\left(\frac{\partial x_2}{\partial \overline{x}_1}\right)_{\overline{x}_2}} \tag{3.68}$$

and

$$\left(\frac{\partial z_{\{i,j\}}}{\partial x_2}\right)_{x_1} = \frac{\left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_2}\right)_{\overline{x}_1}\left(\frac{\partial x_1}{\partial \overline{x}_1}\right)_{\overline{x}_2} - \left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_1}\right)_{\overline{x}_2}\left(\frac{\partial x_1}{\partial \overline{x}_2}\right)_{\overline{x}_1}}{\left(\frac{\partial x_2}{\partial \overline{x}_2}\right)_{\overline{x}_1}\left(\frac{\partial x_1}{\partial \overline{x}_1}\right)_{\overline{x}_2} - \left(\frac{\partial x_2}{\partial \overline{x}_1}\right)_{\overline{x}_2}\left(\frac{\partial x_1}{\partial \overline{x}_2}\right)_{\overline{x}_1}}, \tag{3.69}$$

where

$$\left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_1}\right)_{\overline{x}_2} = \left(\frac{\partial \overline{z}_{\{i,j\}}}{\partial \overline{x}_1}\right)_{\overline{x}_2}\left(\frac{\partial z}{\partial \overline{z}}\right)_{\overline{x}_2} \text{ and} \tag{3.70}$$

$$\left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_2}\right)_{\overline{x}_1} = \left(\frac{\partial \overline{z}_{\{i,j\}}}{\partial \overline{x}_2}\right)_{\overline{x}_1}\left(\frac{\partial z}{\partial \overline{z}}\right)_{\overline{x}_1}. \tag{3.71}$$

The derivatives of the general transformation functions $\overline{z}(z, \overline{x}_1, \overline{x}_2)$ are simplified to

$$\left(\frac{\partial z}{\partial \overline{z}}\right)_{\overline{x}_1} = \left(\frac{\mathrm{d}\, z}{\mathrm{d}\, \overline{z}}\right) \text{ and} \tag{3.72}$$

$$\left(\frac{\partial z}{\partial \overline{z}}\right)_{\overline{x}_2} = \left(\frac{\mathrm{d}\, z}{\mathrm{d}\, \overline{z}}\right) \tag{3.73}$$

if the transformation of $\overline{z}$ is independent of $\overline{x}_1$ and $\overline{x}_2$, *i.e.*, $\overline{z}(z)$.

If no scaling transformations are applied, *i.e.*, if $\overline{x}_1$ is independent of x_2 and $\overline{x}_2$ is independent of x_1, the derivatives of the inverse transformation functions

$$\left(\frac{\partial x_1}{\partial \overline{x}_2}\right)_{\overline{x}_1} \text{ and } \left(\frac{\partial x_2}{\partial \overline{x}_1}\right)_{\overline{x}_2}$$

become zero, and Eqs. (3.68) and (3.69) are simplified to

$$\left(\frac{\partial z_{\{i,j\}}}{\partial x_1}\right)_{x_2} = \left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_1}\right)_{\overline{x}_2} \left(\frac{\mathrm{d}\,\overline{x}_1}{\mathrm{d}\,x_1}\right) \tag{3.74}$$

and

$$\left(\frac{\partial z_{\{i,j\}}}{\partial x_2}\right)_{x_1} = \left(\frac{\partial z_{\{i,j\}}}{\partial \overline{x}_2}\right)_{\overline{x}_1} \left(\frac{\mathrm{d}\,\overline{x}_2}{\mathrm{d}\,x_2}\right). \tag{3.75}$$

The derivatives of the spline function with transformed variables, Eq. (3.26), within cell $\{i,j\}$ are calculated from

$$\begin{aligned}\left(\frac{\partial \overline{z}_{\{i,j\}}}{\partial \overline{x}_1}\right)_{\overline{x}_2}\left(\overline{x}_1,\overline{x}_2\right) &= a_{ij21} + 2a_{ij31}\left(\overline{x}_1 - \overline{x}_{1,i}\right) \\ &+ \left[a_{ij22} + 2a_{ij32}\left(\overline{x}_1 - \overline{x}_{1,i}\right)\right]\left(\overline{x}_2 - \overline{x}_{2,j}\right) \\ &+ \left[a_{ij23} + 2a_{ij33}\left(\overline{x}_1 - \overline{x}_{1,i}\right)\right]\left(\overline{x}_2 - \overline{x}_{2,j}\right)^2\end{aligned} \tag{3.76}$$

and

$$\begin{aligned}\left(\frac{\partial \overline{z}_{\{i,j\}}}{\partial \overline{x}_2}\right)_{\overline{x}_1}\left(\overline{x}_1,\overline{x}_2\right) &= a_{ij12} + 2a_{ij13}\left(\overline{x}_2 - \overline{x}_{2,j}\right) \\ &+ \left[a_{ij22} + 2a_{ij23}\left(\overline{x}_2 - \overline{x}_{2,j}\right)\right]\left(\overline{x}_1 - \overline{x}_{1,i}\right) \\ &+ \left[a_{ij32} + 2a_{ij33}\left(\overline{x}_2 - \overline{x}_{2,j}\right)\right]\left(\overline{x}_1 - \overline{x}_{1,i}\right)^2\end{aligned} \tag{3.77}$$

3.2.5 Calculations in the Two-Phase Region

In order to calculate properties in the fluid two-phase region, the equilibrium condition must be described in a suitable manner. The saturation states could be calculated from the Maxwell criterion, *i.e.*, equal pressures and specific Gibbs energies at constant temperature for both phases; but for the sake of simplicity, a function for the relationship of pressure and temperature at saturation should be used instead.

If one of the variables x_1 or x_2 represents either pressure or temperature, the saturation curve can be described with the saturation temperature $T_\mathrm{s}(p)$ or the saturation pressure $p_\mathrm{s}(T)$, respectively. For example, if spline functions are needed for the (x_1, x_2) plane, where x_1 is the pressure p and x_2 is not the temperature, the saturation curve is described by $T_\mathrm{s}(p)$. Additionally, spline functions for both the liquid and the vapor phases, $T^\mathrm{L}(x_1 = p, x_2)$ and $T^\mathrm{G}(x_1 = p, x_2)$, must be provided. With their inverse spline functions $x_2^\mathrm{L}(x_1 = p, T)$ and $x_2^\mathrm{G}(x_1 = p, T)$, the saturated properties in the liquid phase x_2' and in the vapor phase x_2'' are calculated. Then, the desired mass-specific properties $z(x_1 = p, x_2)$ in the two-phase region can be calculated with the relation

$$z\left(x_1, x_2\right) = z' + \frac{x_2 - x_2'}{x_2'' - x_2'}\left(z'' - z'\right), \tag{3.78}$$

where $x_1 = p$, $z' = z^\mathrm{L}(x_1 = p, x_2 = x_2')$, and $z'' = z^\mathrm{G}(x_1 = p, x_2 = x_2'')$.

Consequently, the calculation of $z(x_1, x_2)$ in the two-phase region is numerically consistent with values in the single-phase regions, and a phase test to determine if a given state (x_1, x_2) is located either in the single-phase region or in the two-phase region is distinct and simple. As an example, an algorithm for calculating the properties in the two-phase region from (p,h) is given in Appendix A5. The inverse calculations from (p,s) and (h,s) are given in Appendices A6 and A7.

If x_1 and x_2 are neither pressure nor temperature, the properties in the two-phase region must be calculated by iteration. Again, the relationship between pressure and temperature at saturation can be described with a function $T_\mathrm{s}(p)$. Then, for given properties x_1 and x_2, the set of equations $\boldsymbol{F}(\boldsymbol{X})$, Eqs. (3.79) – (3.83),

$$\boldsymbol{F}_1(\boldsymbol{X}) = 0 = p^\mathrm{L}(x_1', x_2') - p_\mathrm{s}, \tag{3.79}$$

$$\boldsymbol{F}_2(\boldsymbol{X}) = 0 = p^\mathrm{G}(x_1'', x_2'') - p_\mathrm{s}, \tag{3.80}$$

$$\boldsymbol{F}_3(\boldsymbol{X}) = 0 = T^\mathrm{L}(x_1', x_2') - T_\mathrm{s}(p_\mathrm{s}), \tag{3.81}$$

$$\boldsymbol{F}_4(\boldsymbol{X}) = 0 = T^\mathrm{G}(x_1'', x_2'') - T_\mathrm{s}(p_\mathrm{s}), \text{ and} \tag{3.82}$$

$$\boldsymbol{F}_5(\boldsymbol{X}) = 0 = \frac{x_1 - x_1'}{x_1'' - x_1'} - \frac{x_2 - x_2'}{x_2'' - x_2'} \tag{3.83}$$

must be solved for the vector of unknowns $\boldsymbol{X} = (p_\mathrm{s}, x_1', x_2', x_1'', x_2'')^\mathrm{T}$. This can be done through the use of Newton's method for non-linear systems of equations by solving

$$\boldsymbol{J}(\boldsymbol{X}_k)\,\Delta\boldsymbol{X}_k = \boldsymbol{F}(\boldsymbol{X}_k) \text{ and} \tag{3.84}$$

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \Delta\boldsymbol{X}_k \tag{3.85}$$

in each iteration step k until convergence is reached. The Jacobian matrix $\boldsymbol{J}(\boldsymbol{X})$ is given as

$$\boldsymbol{J}(\boldsymbol{X}) = \tag{3.86}$$

$$\begin{bmatrix}
-1 & \left(\frac{\partial p^\mathrm{L}}{\partial x_1}\right)_{x_2}(x_1', x_2') & \left(\frac{\partial p^\mathrm{L}}{\partial x_2}\right)_{x_1}(x_1', x_2') & 0 & 0 \\
-1 & 0 & 0 & \left(\frac{\partial p^\mathrm{G}}{\partial x_1}\right)_{x_2}(x_1'', x_2'') & \left(\frac{\partial p^\mathrm{G}}{\partial x_2}\right)_{x_1}(x_1'', x_2'') \\
-\left(\frac{\mathrm{d}T_\mathrm{s}}{\mathrm{d}p}\right)(p_\mathrm{s}) & \left(\frac{\partial T^\mathrm{L}}{\partial x_1}\right)_{x_2}(x_1', x_2') & \left(\frac{\partial T^\mathrm{L}}{\partial x_2}\right)_{x_1}(x_1', x_2') & 0 & 0 \\
-\left(\frac{\mathrm{d}T_\mathrm{s}}{\mathrm{d}p}\right)(p_\mathrm{s}) & 0 & 0 & \left(\frac{\partial T^\mathrm{G}}{\partial x_1}\right)_{x_2}(x_1'', x_2'') & \left(\frac{\partial T^\mathrm{G}}{\partial x_2}\right)_{x_1}(x_1'', x_2'') \\
0 & \frac{(x_1 - x_1') - (x_1'' - x_1')}{(x_1'' - x_1')^2} & -\frac{(x_2 - x_2') - (x_2'' - x_2')}{(x_2'' - x_2')^2} & -\frac{(x_1 - x_1')}{(x_1'' - x_1')^2} & \frac{(x_2 - x_2')}{(x_2'' - x_2')^2}
\end{bmatrix}.$$

The derivatives in the Jacobian matrix are provided analytically as given in Sec. 3.2.4. Auxiliary spline functions for $p_s(x_1, x_2)$ and for x_1', x_1'', x_2', and x_2'' as functions of either temperature T or pressure p are recommended to provide initial values of the unknown variables. With the saturation properties, $z' = z^L(x_1', x_2')$ and $z'' = z^G(x_1'', x_2'')$, $z(x_1, x_2)$ is calculated from Eq. (3.78).

In situations where state points are calculated in the vapor region and the two-phase region only, such as in CFD simulations of steam turbines, or where small inconsistencies at the saturated liquid line are tolerable, the following additional phase boundary conditions are recommended. Instead of using $T_s(p)$, the properties at saturation are described with spline functions for

$$x_1''(p), \tag{3.87}$$

$$x_1'(x_2), \text{ and} \tag{3.88}$$

$$x_2'(T). \tag{3.89}$$

With this approach, the phase test at the saturation curves for a given state point (x_1, x_2) can be performed without iteration while the numerical consistency at the saturated vapor line is preserved.

Through the use of the inverse spline functions $x_2^G(x_1, p)$ and $p_s(x_1)$, obtained from $p^G(x_1, x_2)$ and Eq. (3.87), with

$$x_2''(x_1) = x_2^G(x_1, p = p_s(x_1)), \tag{3.90}$$

it can be determined if the state point is located in the vapor phase or in the two-phase region.

The properties in the two-phase region are calculated by solving

$$p_s = p^G(x_1'', x_2''), \tag{3.91}$$

$$T_s = T^G(x_1'', x_2''), \text{ and} \tag{3.92}$$

$$\frac{x_1 - x_1'}{x_1'' - x_1'} = \frac{x_2 - x_2'}{x_2'' - x_2'} \tag{3.93}$$

along with Eqs. (3.87) – (3.89). This can be carried out efficiently with Newton's iterative procedure for one-dimensional problems as shown for calculations from (v,u) in Appendix A8. The corresponding algorithms for the inverse functions of (p,v) and (u,s) are given in Appendices A9 and A10.

Alternatively, explicit spline functions for the desired properties in the two-phase region can be generated. This is the fastest approach, but will produce small inconsistencies at the phase boundaries.

4 FluidSplines – Software for Generating SBTL Property Functions

In order to apply the SBTL method to property functions of any fluid, the software FluidSplines [62] has been developed. This software implements all features of the SBTL method (see Sec. 3) and assists the user in generating spline functions for a given range of validity with a user-specified agreement with the underlying property formulations. For calculations from these underlying property formulations, an appropriate external property library must be connected to FluidSplines. For this purpose, FluidSplines provides an extensible low level interface. Currently, REFPROP [59] and the property libraries from the Zittau/Görlitz University of Applied Sciences can be used with FluidSplines.

4.1 Basic Structure

FluidSplines is written in C++. Its graphical user interface is based on the Microsoft Foundation Classes (MFC) and follows the classical document-view architecture with a single document interface (SDI). The drawing capabilities for two- and three dimensional diagrams are implemented through the use of the Microsoft Windows Graphics Device Interface (GDI) and OpenGL. Since MFC and GDI are Microsoft proprietary libraries, FluidSplines does not run on operating systems other than Microsoft Windows to date. However, FluidSplines produces pure C source code for the generated spline functions which is platform independent. Figure 33 gives a brief overview of the architecture of FluidSplines and its interfaces to external property libraries. As for every other MFC-SDI program, Windows creates the global application object (CFluidSplinesApp), which is derived from CWinAppEx, and starts the main function WinMain, which is part of the MFC library. WinMain calls a member function of CFluidSplinesApp to initialize the frame window (CMainFrame), the document (CFluidSplinesDoc), and one or more views derived from the CView base class. The frame window (CMainFrame) contains the menu bar, the toolbars, and the status bar. Its base class CFrameWndEx also implements dockable windows. In FluidSplines such dockable windows are used to display project information and error messages. All relevant data is stored in the document (CFluidSplinesDoc). The common base class of all objects in the document is CObject, which implements serialization and run-time class information. Depending on the chosen external property library, an instance of a CFluid-derived class is created. CFluid encapsulates property functions and other fluid-specific information and interfaces to the corresponding dynamic-link library (*.dll) of the external property library. The data related to the SBTL property function and the applied unit system are kept in an instance of CProjectData. One- and two-dimensional diagrams are represented by classes derived from CDiagram2D or CDiagram3D, respectively. These classes contain the relevant information to be depicted in the diagrams and provide functionalities to prepare the drawing elements, such as axes, curves, surfaces, text, markers, etc. Libraries for these drawing elements have been developed specifically for FluidSplines. The diagrams are drawn in the corresponding view. Instances of CFluidSplinesView show customizable two-dimensional diagrams for the SBTL property function to be generated. For two-dimensional SBTL property functions, these diagrams also allow for a flexible definition of the range of validity as described in Sec. 4.3. To visualize the grid of nodes in a three-dimensional view, an instance of CFluidSplinesView3D is created. The graphical user interface of FluidSplines is shown in Fig. 34.

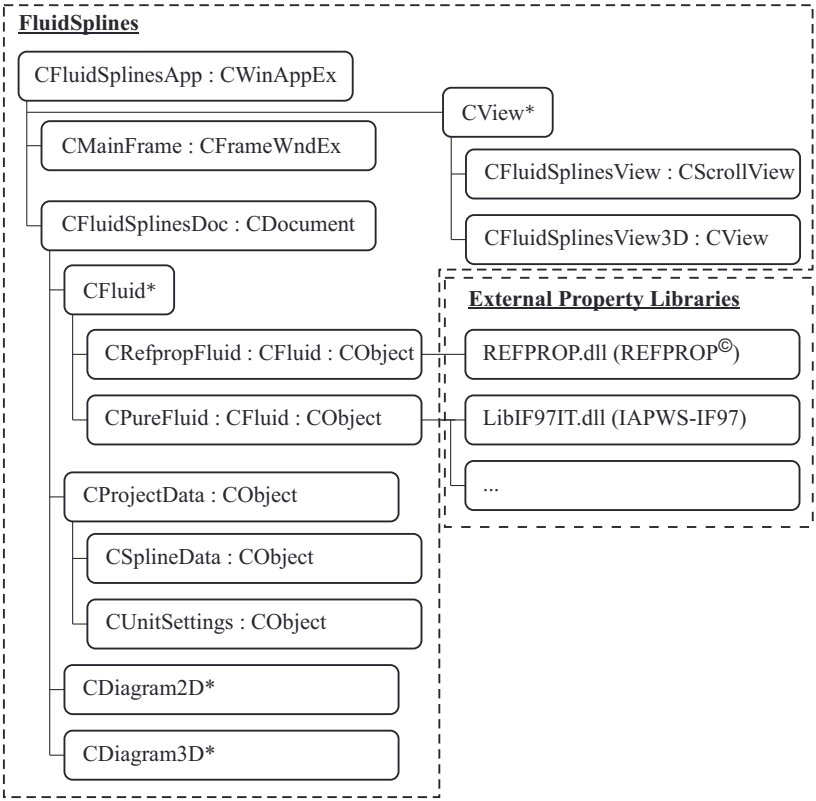

Figure 33: Brief overview of the architecture of FluidSplines and its interfaces to external property libraries. Class names begin with the character "C", the ":" operator separates the class from the base class it is derived from. The asterisk "*" denotes a pointer to an object.

In order to create an SBTL property function with FluidSplines, the underlying fluid property formulation must be chosen first. For this purpose, the external property library and the required fluid is selected first using the "Fluid" dialog box in the "Project" menu. If the external property library provides several equations of state for the selected fluid, then the relevant equation can be chosen by the user. Similarly, the reference state can be specified in the same dialog box. The unit system for the SBTL property functions to be generated can be defined using the "Units" dialog in the "Project" menu. The generation of one- and two-dimensional SBTL property functions is outlined in Secs. 4.2 and 4.3.

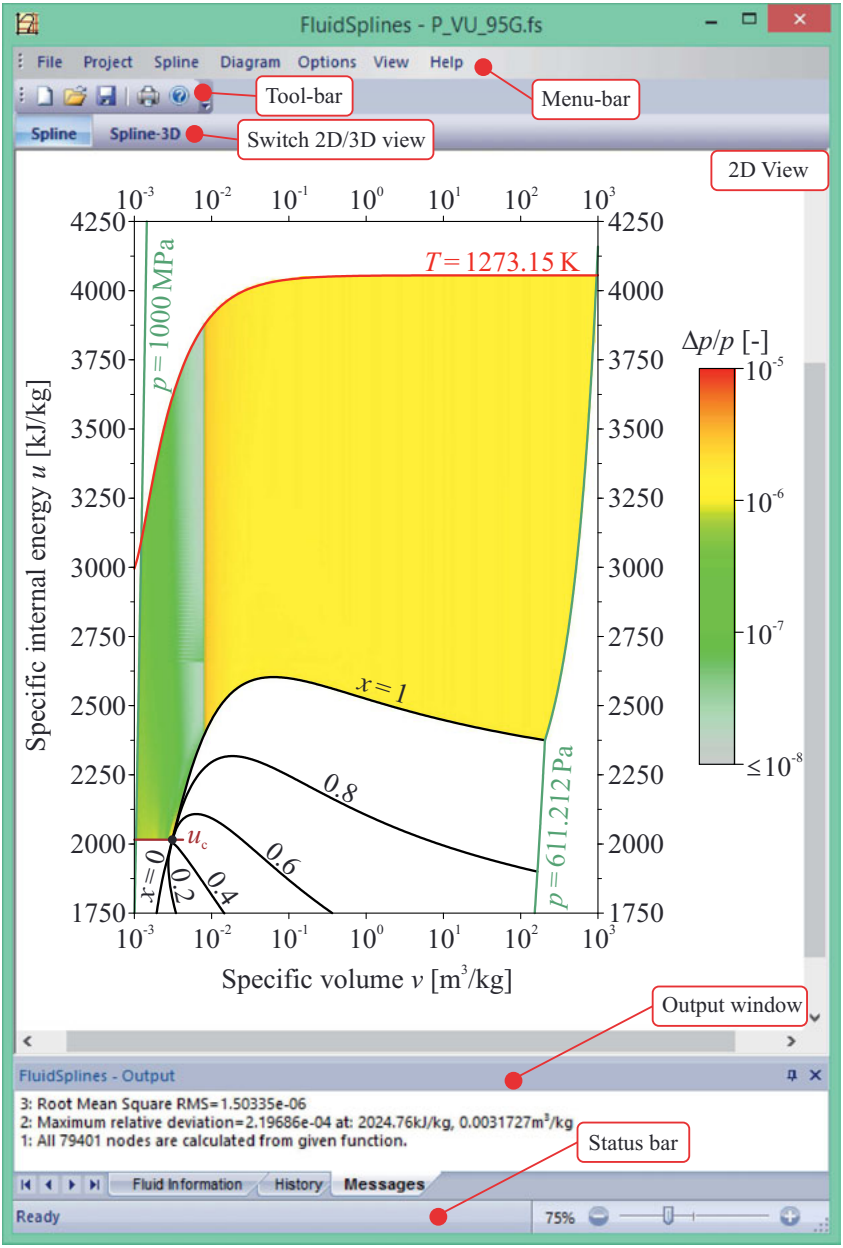

Figure 34: Graphical user interface of FluidSplines showing a *u*-*v* diagram with the deviations of the created spline function from the underlying formulation as a color plot.

4.2 Generation of One-Dimensional SBTL Property Functions

The generation of one-dimensional SBTL property functions in FluidSplines is demonstrated using the example of $T_s(p)$ based on IAPWS-95. The function to be interpolated, *i.e.*, $T_s(p)$ can be selected from the "Function" dialog in the "Spline" menu. Once the function is selected, FluidSplines will automatically prepare the corresponding *T*-*p* diagram as shown in the upper section of Fig. 35. The diagram can be customized using the "Diagram" menu items. For a one-dimensional property function, the desired range of validity $p_{\min} \le p \le p_{\max}$ can simply be entered into the program through the use of the "Spline"/"Region" dialog box. In this example, the pressure ranges between $p_{\min} = 6.11 \times 10^{-4}$ MPa and $p_{\max} = 22.064$ MPa. The spline interpolation algorithm, either quadratic or cubic, is chosen from the corresponding dialog under the "Spline"/"Interpolation" menu item. In the same dialog box, the coordinate transformations and the series of nodes are specified. In this example, a quadratic spline function is prepared, where the pressure *p* is transformed into $\overline{p}(p) = \sqrt{p}$.

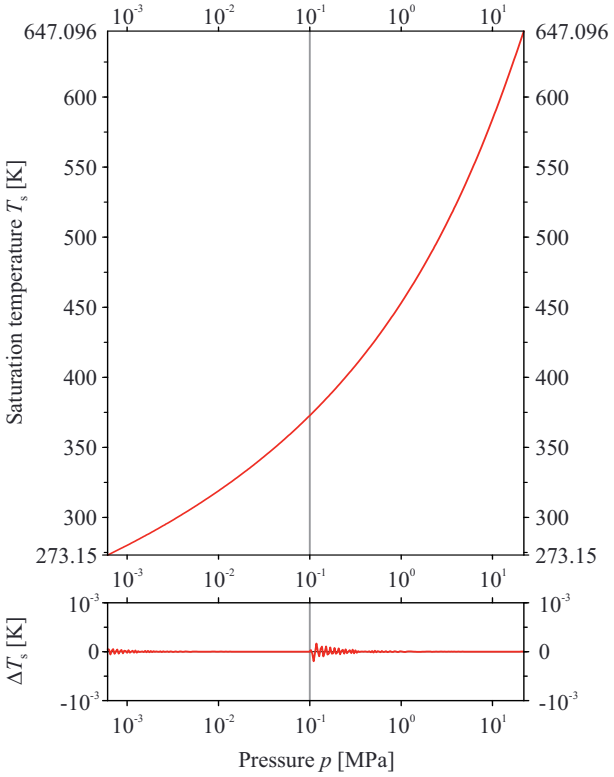

Figure 35: Spline function $T_s(p)$ and its deviations from IAPWS-95.

The series of piecewise equidistant nodes is created in transformed coordinates as follows:

$$\bar{p}\left(6.11\times10^{-4}\ \text{MPa}\right) \le \bar{p} \le \bar{p}\left(0.1\ \text{MPa}\right): \qquad 200\ \text{nodes},$$
$$\bar{p}\left(0.1\ \text{MPa}\right) \le \bar{p} \le \bar{p}\left(22.064\ \text{MPa}\right): \qquad 200\ \text{nodes}.$$

The generation of the spline function is then started from the "Interpolation" dialog box. The computation is carried out by a worker thread. In this way, the graphical user interface operates without delays and the spline generation can be stopped on user's demand. The spline generation process is multi-threaded to minimize its duration. During the spline generation, status and error messages are displayed in the "Output" window. A progress bar shows the progress of the spline generation. If the external property library fails to provide a required value, then this value is interpolated automatically within FluidSplines and a message is added to the "Output" window. Once a spline function is successfully generated, it can be used to provide starting values for the external property library. The accuracy test of the spline function can also be started from the "Interpolation" dialog box. For the example described here, a deviation diagram is added below the *T*-*p* diagram for $T_s(p)$ as shown in Fig. 35. The generated spline function can be exported from FluidSplines as pure C code.

4.3 Generation of Two-Dimensional SBTL Property Functions

The generation of two-dimensional SBTL property functions in FluidSplines is demonstrated using the example of $p(v,u)$ for the gas phase based on IAPWS-95. Upon the selection of this function in FluidSplines, a *u*-*v* diagram is created automatically. This diagram includes the isobars and isotherms bounding the range of validity of the underlying equation of state and the phase envelopes. Other isolines for *p*, *T*, *v*, *x*, *u*, *h*, or *s*, as well as spinodals and melting curves can be added using the "Contents" dialog from the "Diagram" menu. The curves in the diagram can be used to determine the required region, where the property function to be generated should be valid. In this example, the spline function is defined in the gas phase and is bounded by the isobars at 611.212 Pa and 1000 MPa, as well as by the isotherm at 1273.15 K and a constant internal energy of 2015.735 kJ/kg. The region determination is started from the "Region" dialog in the "Spline" menu. Messages in the "Output" window guide the user through the process. The bounding curves of the range of validity need to be selected one after another from the diagram. For every newly added curve, FluidSplines calculates the intersection with the previous curve. If two functions intersect more than once, the intended intersection can be picked from the diagram. Once the region is determined, FluidSplines automatically determines all local extreme values at the boundaries. If no scaling transformations are applied (see Sec. 3.2.2), the bounding rectangle of all intersections and extreme values defines the outer boundaries of the grid of nodes as shown in Fig. 36.

In many cases it is necessary to determine whether a point lies in the selected region or not. For polygons, the ray casting algorithm [63] can be used to determine whether a point lies inside or not. The intersections of a ray casted from the point in an arbitrary direction are counted. If the number of intersections is odd, then the point is located inside the polygon. For all other cases it is outside the polygon. The ray casting algorithm can also be used if the segments of the boundary are monotonic curves instead of straight lines. For this purpose, the selected region boundaries are decomposed into monotonic sub-boundaries (see Fig. 36). The intersections of the ray with the monotonic sub-boundaries must be determined by iteration.

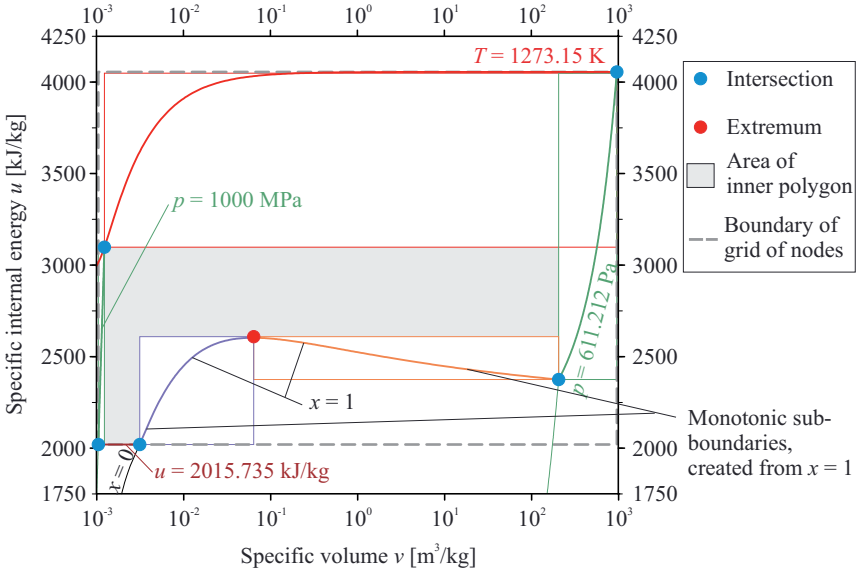

Figure 36: Determination of outer boundary of the rectangular grid of nodes and division of region boundaries into monotonic sub-boundaries for the region determination process.

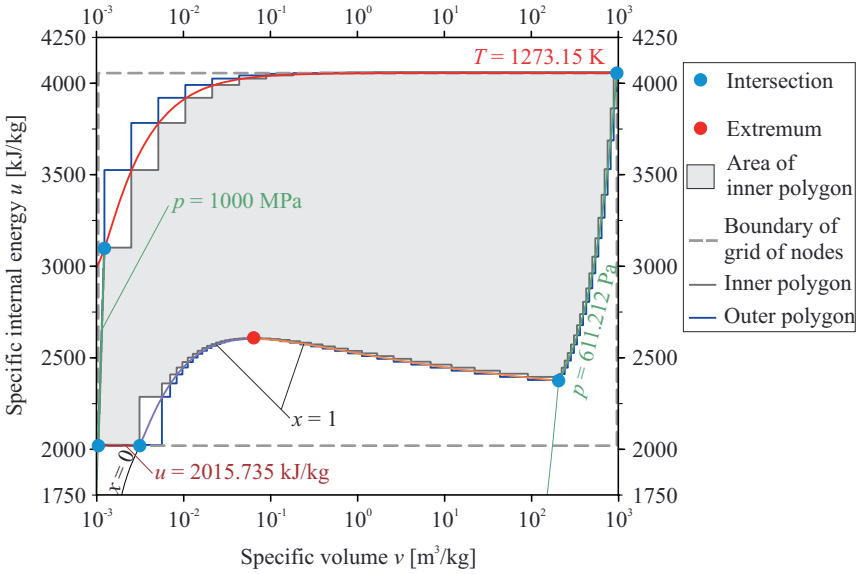

Figure 37: Inner and outer polygon for a larger number of monotonic sub-boundaries.

To minimize the calculation of intersections by iteration, the following procedure is applied. From the bounding rectangles of the monotonic sub-boundaries an inner polygon and an outer polygon are created as shown in Fig. 36. In some situations an inner polygon cannot be created this way. In other situations, the resulting inner polygon is self-intersecting and must be split into non-self-intersecting polygons. If the inner polygon exists, any point inside this polygon is also inside the selected region. Any point outside of the outer polygon is outside of the region. For all other cases, the ray casting algorithm for monotonic curves is applied. In order to increase the area of the inner polynomial, the monotonic sections of each boundary are divided into a larger number of sub-boundaries as shown in Fig. 37.

As for the generation of one-dimensional spline functions, the spline interpolation algorithm, the coordinate transformations, and the grid of nodes can be specified in the "Interpolation" dialog box. In this example, a biquadratic spline function is prepared, where the specific volume v is transformed into $\bar{v}(v) = \ln(v)$. The grid of piecewise equidistant nodes is created in transformed coordinates as follows:

$\bar{v}\left(1.05272\times10^{-3}\ \mathrm{m^3/kg}\right) \le \bar{v} \le \bar{v}\left(8\times10^{-3}\ \mathrm{m^3/kg}\right)$: 200 nodes,

$\bar{v}\left(8\times10^{-3}\ \mathrm{m^3/kg}\right) \le \bar{v} \le \bar{v}\left(961.341\ \mathrm{m^3/kg}\right)$: 200 nodes,

$2015.735\ \mathrm{kJ/kg} \le u \le 2650\ \mathrm{kJ/kg}$: 100 nodes,

$2650\ \mathrm{kJ/kg} \le u \le 4055.26\ \mathrm{kJ/kg}$: 100 nodes.

The generation of the spline function and the accuracy test are started as described in Sec. 4.2. For the example described here, the deviations of the spline function for $p(v,u)$ from the underlying equation of state, are displayed as a color plot in the u-v diagram (see Fig. 38).

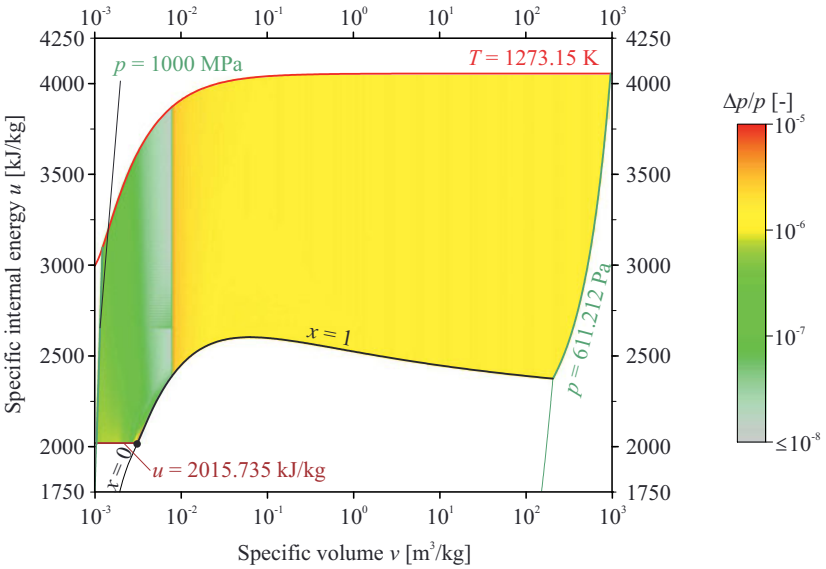

Figure 38: Deviations of the spline function for $p(v,u)$ from IAPWS-95.

The nodes are depicted in a three-dimensional diagram as shown in Fig. 39. In this way, the calculation of the nodes can be verified. Moreover, the effect of variable transformations on the property surface can be seen. The corresponding view is activated through the “Spline-3D” button as shown in Fig. 34.

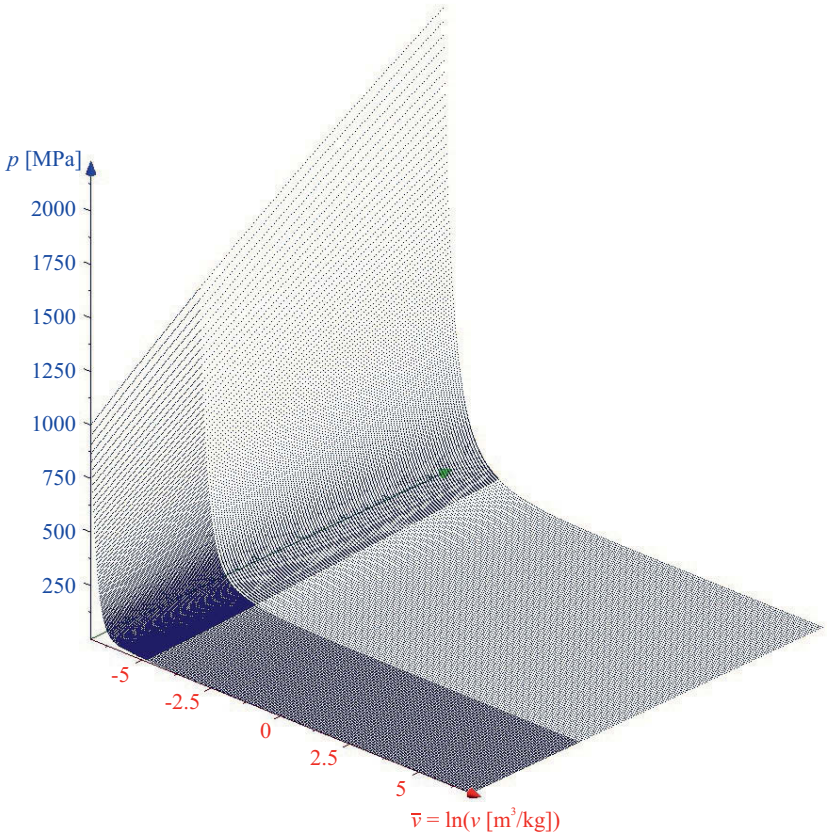

Figure 39: All nodes for the $p(v,u)$ spline function in a $\bar{v} - u - p$ diagram. The linear u-axis is hidden behind the nodes and ranges from 2015 kJ/kg to 4055 kJ/kg.

The nodes outside the range of validity of IAPWS-95 have been extrapolated from its equation of state. In some situations it is not possible to do so. For these situations the user may introduce extrapolation rules by picking curves in the 2D diagram and selecting extrapolation methods for the nodes beyond these curves. Extrapolations from a lower order Taylor series or mirroring the property surface at the specified curve are useful tools to obtain the required nodes beyond the range of validity. These techniques are only intended to provide suitable nodes for the spline interpolation algorithm but the resulting property functions may not produce reasonable values in the extrapolated areas.

The generated spline function can be exported from FluidSplines as pure C code.

5 SBTL Property Functions Based on IAPWS-IF97 for Water and Steam

The SBTL method has been applied to the industrial formulation IAPWS-IF97 through the use of FluidSplines (see Sec. 4). The documentation of the resulting SBTL property functions given in this section was originally published in [61].

5.1 Spline Functions of (*v*,*u*) and Inverse Functions

In order to provide fast and accurate property functions for extensive process simulations, where water and steam properties are frequently calculated from (*v*,*u*), the SBTL method has been applied to IAPWS-IF97. Spline functions have been created for the calculation of $p, T, s, w, \eta = f(v,u)$ in the single-phase region. Furthermore, numerically consistent property functions of (*p*,*v*) and (*u*,*s*) are calculable through the use of inverse spline functions as described in Sec. 3.2.3. The relationships between the spline and inverse spline functions are illustrated in Fig. 40. The properties in the two-phase region are calculated as explained in Sec. 5.1.3.

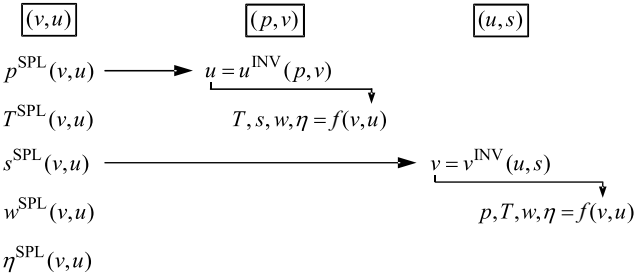

Figure 40: Property calculations from (*v*,*u*), (*p*,*v*), and (*u*,*s*).

5.1.1 Range of Validity

The range of validity is bounded as follows:

$273.15\ \text{K} \le T \le 1073.15\ \text{K} \qquad 611.212\ \text{Pa} \le p \le 100\ \text{MPa}$,

$1073.15\ \text{K} < T \le 2273.15\ \text{K} \qquad 611.212\ \text{Pa} \le p \le 50\ \text{MPa}$.

This range of validity corresponds to that of IAPWS-IF97, except for the lower pressure limit, which is set to $p_s(273.15\ \text{K}) = 611.212\ \text{Pa}$. Figure 41 shows the range of validity and the defined regions of the spline functions with the variables (*v*,*u*). The single phase is divided into the liquid region L, the gas region G, and the high-temperature region HT. With regard to regions defined in IAPWS-IF97, the current liquid region L covers region 1 and a part of region 3. Region 2 and the remaining part of region 3 are included in the gas region G. The spline functions are smoothed at the IF97 region boundaries 1-3 and 2-3. The two-phase region TP corresponds to region 4 of IAPWS-IF97 and the high temperature region HT matches region 5 of IAPWS-IF97.

The specific internal energy at the critical point $u_c = 2019.025\ 106\ \text{kJ/kg}$ is used to define the boundary between the L and G single-phase regions for supercritical state points. At the region boundaries L-G and G-HT in the single-phase region, small inconsistencies are unavoidable

(see Sec. 5.1.6). These should be negligible for most purposes, but if needed the transition at these boundaries can be smoothed with simple interpolation equations.

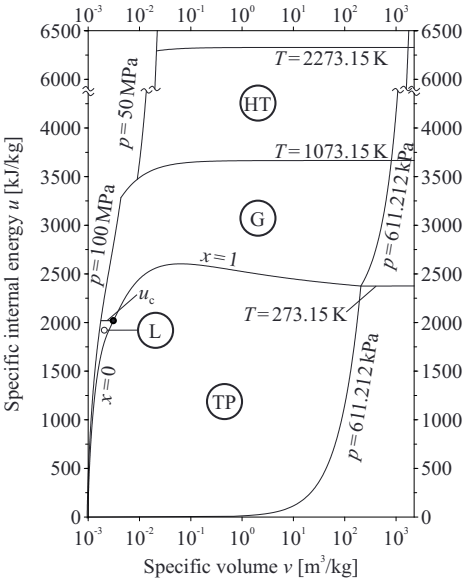

Figure 41: Range of validity in the (*u*,*v*) plane for spline functions based on IAPWS-IF97.

Note: For temperatures between 273.15 K and 273.16 K, the part of the range of validity of region L between the pressures on the melting line and on the saturation-pressure line corresponds to metastable liquid states. In the same temperature range, the part of the range of validity of region G between the pressures on the saturation-pressure line and on the sublimation line corresponds to metastable vapor states.

5.1.2 Spline Functions for the Single-Phase Regions

In each of the three single-phase regions, L, G, and HT, spline functions with the variables (*v*,*u*) are created. In the liquid region L, a scaling transformation (see Sec. 3.2.2) for the specific volume *v* with the boundary curves $v_{\min}(u) = v(p_{\max} = 100\ \text{MPa}, u)$ and $v_{\max}(u) = v'(u)$ is applied, so that

$$\overline{v}(v,u) = \frac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)} \left(v - v_{\min}(u)\right) + \overline{v}_{\min},$$

where the free parameters are set to $\overline{v}_{\min} = 1$ and $\overline{v}_{\max} = 100$. Thus, the shape of the grid of nodes corresponds to the shape of the liquid region L (see Fig. 41). In the single-phase regions G and HT, the specific volume is transformed as $\overline{v} = \ln(v)$. The grid dimensions of each (*v*,*u*) spline function are given in Tables A1, A2, and A3 in Appendix A11. Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation.

From the single-phase spline functions $p^{\mathrm{SPL}}(v,u)$ and $s^{\mathrm{SPL}}(v,u)$, the inverse spline functions $u^{\mathrm{INV}}(p,v)$ and $v^{\mathrm{INV}}(u,s)$ are determined as described in Sec. 3.2.3. With these inverse spline functions, all remaining properties are calculated from (*p*,*v*) and (*u*,*s*), as illustrated in Fig. 40.

5.1.3 Calculations in the Two-Phase Region

The properties in the two-phase region TP are calculated with the spline functions in the single-phase regions L and G, along with additional constraints for the phase equilibrium. For process simulations where the range of states does not include the liquid region L or where small inconsistencies at the saturated liquid line are tolerable, the calculation can be simplified with spline functions for $v''(p)$, $v'(u)$, and $u'(T)$ as discussed in Sec. 3.2.5. This simplification is applied to the spline functions of (*v*,*u*) and their inverse functions of (*p*,*v*) and (*u*,*s*) for the two-phase region TP described in this document. The algorithms are described in Appendices A8, A9, and A10. Auxiliary spline functions $p_{\mathrm{s}}^{\mathrm{AUX}}(v,u)$ and $p_{\mathrm{s}}^{\mathrm{AUX}}(u,s)$ were created to provide initial guesses for the calculations from (*v*,*u*) and (*u*,*s*).

5.1.4 Derivatives

The following derivatives are frequently required in CFD:

$$\left(\frac{\partial p}{\partial v}\right)_u, \qquad \left(\frac{\partial p}{\partial u}\right)_v, \qquad \left(\frac{\partial u}{\partial v}\right)_p,$$

$$\left(\frac{\partial T}{\partial v}\right)_u, \qquad \left(\frac{\partial T}{\partial u}\right)_v, \text{ and} \qquad \left(\frac{\partial u}{\partial v}\right)_T.$$

These derivatives are calculated analytically from $p^{\mathrm{SPL}}(v,u)$ and $T^{\mathrm{SPL}}(v,u)$. The derivatives are continuous and can therefore be applied in numerical calculations, *e.g.*, to prepare a Jacobian matrix in CFD. However, any thermodynamic property where high accuracy is required should be obtained from a dedicated spline function, rather than using derivatives of other spline functions. A description of the calculation of derivatives is given in Sec. 3.2.4.

5.1.5 Deviations from IAPWS-IF97

The maximum (max) and root-mean-square (RMS) deviations between the spline functions implemented as discussed in Secs. 5.1.2 and 5.1.3 and IAPWS-IF97, along with the permissible values (perm), are given in Tables 14 through 18. The permissible values were set by the IAPWS Task Group “CFD Steam Property Formulation” to ensure that the differences in the results of process simulations with the SBTL method from those obtained with the direct application of IAPWS-IF97 are negligible. The permissible values are less than or equal to the required numerical consistencies for the IAPWS-IF97 backward equations [3, 5, 6, 7, 8]. The values given in Tables 14 through 18 do not include the deviations caused by the inconsistencies of the IAPWS-IF97 basic equations at the IF97 region boundaries 1-3 and 2-3. The deviations of the SBTL property functions for $p(v,u)$, $T(v,u)$, and $s(v,u)$ from IAPWS-IF97 in the liquid region L and the gas region G are depicted in Appendix A12, Figs. A5-A10.

Table 14: Deviations in pressure $p(v,u)$ from IAPWS-IF97

IF97 Region		$\lvert\Delta p\rvert_{\text{perm}}$	$\lvert\Delta p\rvert_{\max}$	$(\Delta p)_{\text{RMS}}$
1	$p \le 2.5\,\text{MPa}$	0.6 %	0.12 %	0.012 %
	$p > 2.5\,\text{MPa}$	15 kPa	0.61 kPa	0.0044 kPa
2		0.001 %	0.000 48 %	0.000 12 %
3		0.001 %	0.000 95 %	0.000 04 %
4		0.0035 %	0.0035 %	0.000 28 %
5		0.001 %	0.000 53 %	0.000 15 %

Table 15: Deviations in temperature $T(v,u)$ from IAPWS-IF97

IF97 Region	$\lvert\Delta T\rvert_{\text{perm}}$ [mK]	$\lvert\Delta T\rvert_{\max}$ [mK]	$(\Delta T)_{\text{RMS}}$ [mK]
1	1	0.27	0.015
2	1	0.43	0.018
3	1	0.53	0.032
4	1	0.69 [a]	0.30 [a]
5	1	0.38	0.018

[a] Except for near-critical temperatures $[(T_c - T) < 1.5\,\text{K}]$.

Table 16: Deviations in specific entropy $s(v,u)$ from IAPWS-IF97

IF97 Region	$\lvert\Delta s\rvert_{\text{perm}}$ $\left[10^{-6}\,\text{kJ/(kg K)}\right]$	$\lvert\Delta s\rvert_{\max}$ $\left[10^{-6}\,\text{kJ/(kg K)}\right]$	$(\Delta s)_{\text{RMS}}$ $\left[10^{-6}\,\text{kJ/(kg K)}\right]$
1	1	0.74	0.049
2	1	0.34	0.045
3	1	0.52	0.022
4	1	0.34	0.044
5	1	0.87	0.056

Table 17: Deviations in speed of sound $w(v,u)$ from IAPWS-IF97

IF97 Region	$\lvert\Delta w\rvert_{\text{perm}}$	$\lvert\Delta w\rvert_{\text{max}}$	$(\Delta w)_{\text{RMS}}$
1	0.001 %	0.000 92 %	0.000 007 %
2	0.001 %	0.000 77 %	0.000 008 %
3	0.001 %	0.000 56 % [a]	0.000 031 % [a]
5	0.001 %	0.000 42 %	0.000 005 %

[a] In the vicinity of the critical point, the deviations of w are larger (< 0.02 %).

Table 18: Deviations in dynamic viscosity $\eta(v,u)$ from IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [65]

IF97 Region	$\lvert\Delta\eta\rvert_{\text{perm}}$	$\lvert\Delta\eta\rvert_{\text{max}}$	$(\Delta\eta)_{\text{RMS}}$
1	0.001 %	0.000 41 %	0.000 068 %
2	0.001 %	0.000 15 %	0.000 010 %
3	0.001 %	0.000 32 %	0.000 019 %

5.1.6 Numerical Consistency at Region Boundaries

The specific internal energy at the critical point $u_c = 2019.025\,106$ kJ/kg defines the region boundary between the liquid region L and the gas region G for supercritical state points (see Fig. 41). This boundary is within IAPWS-IF97 region 3. The numerical inconsistencies of the adjacent spline functions at the region boundary L-G result from the deviations between the spline functions and the basic equation of IAPWS-IF97 region 3 (see Sec. 5.1.5), and are given in Table 19.

Table 19: Numerical inconsistencies at the region boundaries L-G and G-HT

Region boundary	$\lvert\Delta p\rvert_{\text{max}}$	$\lvert\Delta T\rvert_{\text{max}}$	$\lvert\Delta s\rvert_{\text{max}}$	$\lvert\Delta w\rvert_{\text{max}}$	$\lvert\Delta\eta\rvert_{\text{max}}$
L-G [a]	0.0011 %	0.38 mK	4.8×10^{-7} kJ kg^{-1} K^{-1}	0.000 46 %	0.000 27 %
G-HT [b]	0.023 %	82 mK	8.2×10^{-5} kJ kg^{-1} K^{-1}	0.050 %	- [c]

[a] These values were obtained from the corresponding (*v*,*u*)-spline functions for regions L and G at constant specific internal energy $u_c = 2019.025\,106$ kJ/kg.

[b] These values were obtained from the corresponding (*v*,*u*)-spline functions for regions G and HT at constant temperature $T = 1073.15$ K.

[c] Since the upper temperature limit of the IAPWS viscosity release [65] is 1173.15 K, a spline function for the dynamic viscosity η in the high-temperature region is not provided.

The region boundary between the gas region G and the high-temperature region HT is identical to the IAPWS-IF97 region boundary 2-5 and follows the isotherm $T = 1073.15$ K. The underlying IAPWS-IF97 property functions have small discontinuities at the region boundary 2-5. The spline functions reproduce the results of the IAPWS-IF97 basic equations 2 and 5 with high accuracy. Thus, at the region boundary G-HT, the numerical inconsistencies of the IAPWS-IF97 basic equations (see [3]) are dominant; these are given in Table 19.

5.2 Spline Functions of (*p*,*h*) and Inverse Functions

In heat cycle calculations, water and steam properties are frequently calculated from (*p*,*h*). Therefore, another set of spline functions has been created for the calculation of $T, v, s, w, \eta = f(p,h)$ in the single-phase region. Furthermore, numerically consistent property functions of (*p*,*T*), (*p*,*s*), and (*h*,*s*) are required. These are calculated through the use of inverse spline functions as described in Sec. 3.2.3. The relationships between the spline and inverse spline functions are illustrated in Fig. 42. The properties in the two-phase region are calculated as explained in Sec. 5.2.3.

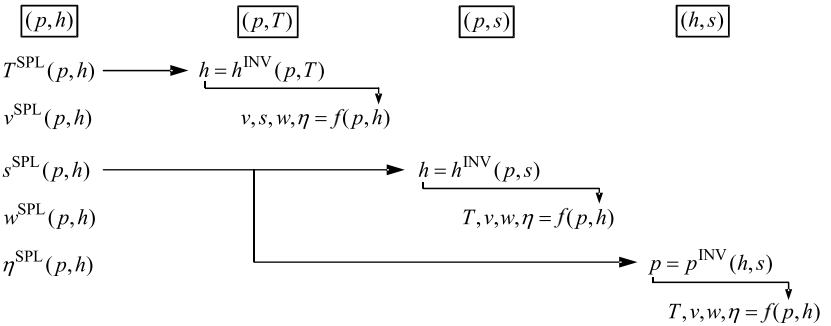

Figure 42: Property calculations from (*p*,*h*), (*p*,*T*), (*p*,*s*), and (*h*,*s*).

5.2.1 Range of Validity

The range of validity is bounded as follows:

$273.15\ \text{K} \le T \le 1073.15\ \text{K}$ $\quad 611.212\ \text{Pa} \le p \le 100\ \text{MPa}$,

$1073.15\ \text{K} < T \le 2273.15\ \text{K}$ $\quad 611.212\ \text{Pa} \le p \le 50\ \text{MPa}$.

This range of validity corresponds to IAPWS-IF97, except the lower pressure limit, which is set to $p_s(273.15\ \text{K}) = 611.212$ Pa. Figure 43 shows the range of validity and the defined regions of the spline functions with the variables (*p*,*h*). The single phase is divided into the liquid region L, the gas region G, and the high temperature region HT. With regard to IAPWS-IF97, the liquid region L covers region 1 and a part of region 3. Region 2 and the remaining part of region 3 are included in the gas region G. The spline functions are smoothed at the IF97 region boundaries 1-3 and 2-3. The two-phase region TP corresponds to region 4 of IAPWS-IF97, and the high-temperature region HT matches region 5 of IAPWS-IF97.

The specific enthalpy at the critical point $h_c = 2087.546\,845$ kJ/kg is used to describe the boundary between the L and G single-phase regions for supercritical state points. At the region boundaries L-G and G-HT in the single-phase region, small inconsistencies are unavoidable (see Sec. 5.2.6). These should be negligible for most purposes, but if needed, the transition at these boundaries can be smoothed with simple interpolation equations.

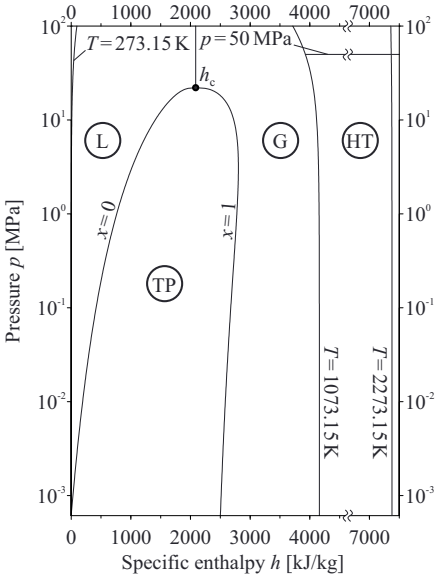

Figure 43: Range of validity in the (*p*,*h*) plane for spline functions based on IAPWS-IF97.

Note: For temperatures between 273.15 K and 273.16 K, see the note at the end of Sec. 5.1.1.

5.2.2 Spline Functions for the Single-Phase Regions

In each of the three single-phase regions, L, G, and HT, spline functions with the variables (*p*,*h*) are created. These spline functions are constructed on rectangular grids without scaling transformations. Variable transformations have been applied to $v(p,h)$, $s(p,h)$, and $w(p,h)$. The variable transformations and grid dimensions of each (*p*,*h*) spline function are given in Tables A4, A5, and A6 in Appendix A11. Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation.

From the spline functions $T^{\mathrm{SPL}}(p,h)$ and $s^{\mathrm{SPL}}(p,h)$ for the single phase, the inverse spline functions $h^{\mathrm{INV}}(p,T)$, $h^{\mathrm{INV}}(p,s)$, and $p^{\mathrm{INV}}(h,s)$ are determined as described in Sec. 3.2.3. With these inverse spline functions, all remaining properties are calculated from (*p*,*T*), (*p*,*s*), and (*h*,*s*), as illustrated in Fig. 42.

5.2.3 Calculations in the Two-Phase Region

The properties in the two-phase region TP are calculated with the spline functions in the single-phase regions L and G, along with additional constraints for phase equilibrium. For property calculations from (*p*,*h*) and (*p*,*s*) in the two-phase region, the saturation temperature T_s is calculated from a spline function $T_s(p)$ based on the corresponding equation of IAPWS-IF97. The enthalpies of the saturated liquid and the saturated vapor are determined from the inverse spline functions $h^L(p,T)$ and $h^G(p,T)$. The corresponding algorithms are described in Appendices A5 and A6. For a given enthalpy and entropy (*h*,*s*), fluid properties in the two-phase region must be determined by iteration as shown in Appendix A7. For this purpose, an auxiliary spline function $p_s^{AUX}(h,s)$ was created to provide an initial guess.

5.2.4 Derivatives

In heat cycle simulations, derivatives such as:

$$\left(\frac{\partial T}{\partial p}\right)_h, \left(\frac{\partial h}{\partial T}\right)_p, \text{ and } \left(\frac{\partial h}{\partial p}\right)_T$$

are frequently used. These derivatives are calculated analytically from $T^{SPL}(p,h)$. The derivatives are continuous and can therefore be applied in numerical calculations, *e.g.*, to prepare a Jacobian matrix in heat cycle simulation software. However, any thermodynamic property, where high accuracy is required, should be obtained from a dedicated spline function. A description of the calculation of derivatives is given in Sec. 3.2.4.

5.2.5 Deviations from IAPWS-IF97

The maximum (max) and root-mean-square (RMS) deviations between the spline functions and IAPWS-IF97, along with the permissible values (perm), are given in Tables 20 through 24. The permissible values were set by the IAPWS Task Group "CFD Steam Property Formulation" to ensure that the differences in the results of process simulations with the SBTL method from those obtained with the direct application of IAPWS-IF97 are negligible. The permissible values are less than or equal to the required numerical consistencies for the IAPWS-IF97 backward equations [3, 5, 6, 7, 8]. The values given in Tables 20 through 24 do not include the deviations caused by the inconsistencies of the IAPWS-IF97 basic equations at the IF97 region boundaries 1-3 and 2-3. The deviations of the SBTL property functions *T*(*p*,*h*), *v*(*p*,*h*), and *s*(*p*,*h*) from IAPWS-IF97 in the liquid region L and the gas region G are depicted in Appendix A12, Figs. A11-A13.

Table 20: Deviations in temperature *T*(*p*,*h*) from IAPWS-IF97

IF97 Region	$\lvert\Delta T\rvert_{perm}$ [mK]	$\lvert\Delta T\rvert_{max}$ [mK]	$(\Delta T)_{RMS}$ [mK]
1	25	0.63	0.073
2	10	0.81	0.026
3	25	0.65	0.045
5	10	0.34	0.042

Table 21: Deviations in specific volume $v(p,h)$ from IAPWS-IF97

IF97 Region	$\lvert\Delta v\rvert_{\text{perm}}$	$\lvert\Delta v\rvert_{\max}$	$(\Delta v)_{\text{RMS}}$
1	0.001 %	0.000 93 %	0.000 14 %
2	0.001 %	0.000 63 %	0.000 010 %
3	0.001 %	0.000 61 %	0.000 044 %
4	0.001 %	0.000 96 % [a]	0.000 10 % [a]
5	0.001 %	0.000 037 %	0.000 005 %

[a] Except for near-critical temperatures $[(T_c - T) < 4\ \text{K}]$ and for states near the saturated liquid curve $(0 \le x < 0.17)$ at pressures $p < 0.1$ MPa where small deviations in the calculated vapor fraction result in larger deviations in the calculated specific volume.

Table 22: Deviations in specific entropy $s(p,h)$ from IAPWS-IF97

IF97 Region	$\lvert\Delta s\rvert_{\text{perm}}$ $\left[10^{-6}\ \text{kJ/(kg K)}\right]$	$\lvert\Delta s\rvert_{\max}$ $\left[10^{-6}\ \text{kJ/(kg K)}\right]$	$(\Delta s)_{\text{RMS}}$ $\left[10^{-6}\ \text{kJ/(kg K)}\right]$
1	1	0.78	0.021
2	1	0.78	0.062
3	1	0.81	0.039
4	1	0.81	0.12
5	1	0.37	0.024

Table 23: Deviations in speed of sound $w(p,h)$ from IAPWS-IF97

IF97 Region	$\lvert\Delta w\rvert_{\text{perm}}$	$\lvert\Delta w\rvert_{\max}$	$(\Delta w)_{\text{RMS}}$
1	0.001 %	0.000 32 %	0.000 038 %
2	0.001 %	0.000 78 %	0.000 013 %
3	0.001 %	0.000 78 %	0.000 054 %
5	0.001 %	0.000 052 %	0.000 007 %

Table 24: Deviations in dynamic viscosity $\eta(p,h)$ from IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [65]

IF97 Region	$\lvert\Delta\eta\rvert_{\text{perm}}$	$\lvert\Delta\eta\rvert_{\max}$	$(\Delta\eta)_{\text{RMS}}$
1	0.001 %	0.000 63 %	0.000 077 %
2	0.001 %	0.000 77 %	0.000 014 %
3	0.001 %	0.000 80 %	0.000 033 %

5.2.6 Numerical Consistency at Region Boundaries

The specific enthalpy at the critical point $h_c = 2087.546\,845$ kJ/kg defines the boundary between the liquid region L and the gas region G above the critical pressure (see Fig. 43). This boundary is within IAPWS-IF97 region 3. The numerical inconsistencies of the adjacent spline functions at the region boundary L-G result from the deviations between the spline functions and the basic equation of IAPWS-IF97 region 3 (see Sec. 5.2.5) and are given in Table 25.

The region boundary between the gas region G and the high-temperature region HT is identical to the IAPWS-IF97 region boundary 2-5 and follows the isotherm $T = 1073.15$ K. The underlying IAPWS-IF97 property functions have small discontinuities at the region boundary 2-5. The spline functions reproduce the results of the IAPWS-IF97 basic equations 2 and 5 with high accuracy. Thus, at the region boundary G-HT, the numerical inconsistencies of the IAPWS-IF97 basic equations (see [3]) are dominant; these are given in Table 25 and are in agreement with those from the IAPWS-IF97 basic equations at the region boundary 2-5.

Table 25: Numerical inconsistencies at the region boundaries L-G and G-HT

Region boundary	$\lvert\Delta T\rvert_{\max}$ or $\lvert\Delta h\rvert_{\max}$	$\lvert\Delta v\rvert_{\max}$	$\lvert\Delta s\rvert_{\max}$	$\lvert\Delta w\rvert_{\max}$	$\lvert\Delta\eta\rvert_{\max}$
L-G [a]	$\lvert\Delta T\rvert_{\max} = 0.30$ mK	0.000 7 %	3.9×10^{-8} kJ kg^{-1} K^{-1}	0.000 51 %	0.000 33 %
G-HT [b]	$\lvert\Delta h\rvert_{\max} = 0.096$ kJ kg^{-1}	0.012 %	1.4×10^{-4} kJ kg^{-1} K^{-1}	0.046 %	- [c]

[a] These values were obtained from the corresponding (p,h)-spline functions for regions L and G at constant specific enthalpy $h_c = 2087.546\,845$ kJ/kg.

[b] These values were obtained from the inverse spline functions $h^{\text{G}}(p,T)$ and $h^{\text{HT}}(p,T)$ and the corresponding (p,h)-spline functions at constant temperature $T = 1073.15$ K.

[c] Since the upper temperature limit of the IAPWS viscosity release [65] is 1173.15 K, a spline function for the dynamic viscosity η in the high-temperature region is not provided.

5.3 Spline Functions for the Metastable-Vapor Region

The industrial formulation IAPWS-IF97 [3, 4] provides a supplementary equation for part of the metastable-vapor region. This equation is valid from the saturated vapor curve to the 5% equilibrium moisture line (determined from the equilibrium h' and h'' values) at pressures from the triple-point pressure up to 10 MPa.

Spline-based property functions of (v,u) and (p,h) have been developed for calculations in the metastable-vapor region described above. In order to avoid discontinuities at the saturated vapor curve, the range of validity of these spline functions has been extended to the gas region G as shown in Figs. 41 and 43. The spline functions described in this section are recommended for use in non-equilibrium process simulations. For simulating equilibrium processes, the spline functions described in Secs. 5.1 and 5.2 should be used.

5.3.1 Spline Functions of (v,u)

Spline-based property functions for calculating $p, T, s, w, \eta = f(v,u)$ in both the metastable-vapor region and the gas region G (see Fig. 41) have been created. For every spline-based property function of (v,u), the specific volume is transformed as $\overline{v} = \ln(v)$. The grid dimensions of these functions are equal to those given for the gas region G in Sec. 5.1.2 (see Table A2 in Appendix A11). Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation.

5.3.2 Spline Functions of (p,h)

Spline-based property functions for calculating $T, v, s, w, \eta = f(p,h)$ in both the metastable-vapor region and the gas region G (see Fig. 43) have been created. Variable transformations have been applied to $v(p,h)$, $s(p,h)$, and $w(p,h)$. The variable transformations and grid dimensions of each (p,h) spline function are given in Table A7 in Appendix A11. Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation.

5.3.3 Deviations from IAPWS-IF97

The deviations of the developed spline-based property functions from the IAPWS-IF97 supplementary equation for the metastable-vapor region and from the IAPWS-IF97 basic equation for region 2, along with the permissible (perm) values, are given in Tables 26 and 27. The values given in these tables do not include the deviations caused by the inconsistencies of the IAPWS-IF97 basic equations at the IF97 region boundary 2-3. At the saturated vapor curve for pressures $p < 10$ MPa, increased deviations due to the small inconsistency between the IAPWS-IF97 supplementary equation for the metastable-vapor region and the IAPWS-IF97 basic equation for region 2 cannot be avoided. The maximum deviations (max) in the metastable-vapor region and in region 2 of IAPWS-IF97 outside the temperature ranges $\left|T - T_s(p)\right|$ and the maximum deviations (max, sat) within these ranges are given in Tables 26 and 27. The root-mean-square deviations (RMS) of the spline-based property functions from the IAPWS-IF97 supplementary equation for the metastable-vapor region and from the IAPWS-IF97 basic equation for region 2 are also given in Tables 26 and 27.

Table 26: Deviations in pressure $p(v,u)$, temperature $T(v,u)$, specific entropy $s(v,u)$, speed of sound $w(v,u)$, and dynamic viscosity $\eta(v,u)$ from the supplementary equation for the metastable-vapor region and the basic equation for region 2 of IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [65]

Spline function	Permissible deviation	Maximum deviation in the metastable-vapor region and in region 2 of IAPWS-IF97 outside the range $\lvert T - T_s(p)\rvert$ defined in the next column	Range $\lvert T - T_s(p)\rvert$ along the saturated vapor curve for $p < 10$ MPa		RMS deviation in the metastable-vapor region and in region 2 of IAPWS-IF97
			$\lvert T - T_s(p)\rvert$	Maximum deviation	
$p(v,u)$	$\lvert\Delta p\rvert_{\text{perm}} = 0.001\,\%$	$\lvert\Delta p\rvert_{\max} = 0.000\,97\,\%$	7 K	$\lvert\Delta p\rvert_{\max,\,\text{sat}} = 0.016\,\%$	$(\Delta p)_{\text{RMS}} = 0.000\,34\,\%$
$T(v,u)$	$\lvert\Delta T\rvert_{\text{perm}} = 1\,\text{mK}$	$\lvert\Delta T\rvert_{\max} = 0.60\,\text{mK}$	10 K	$\lvert\Delta T\rvert_{\max,\,\text{sat}} = 25.2\,\text{mK}$	$(\Delta T)_{\text{RMS}} = 1.1\,\text{mK}$
$s(v,u)$	$\lvert\Delta s\rvert_{\text{perm}} = 1\times10^{-6}\,\text{kJ/(kg K)}$	$\lvert\Delta s\rvert_{\max} = 0.45\times10^{-6}\,\text{kJ/(kg K)}$	12 K	$\lvert\Delta s\rvert_{\max,\,\text{sat}} = 0.81\times10^{-4}\,\text{kJ/(kg K)}$	$(\Delta s)_{\text{RMS}} = 0.83\times10^{-6}\,\text{kJ/(kg K)}$
$w(v,u)$	$\lvert\Delta w\rvert_{\text{perm}} = 0.001\,\%$	$\lvert\Delta w\rvert_{\max} = 0.000\,88\,\%$	10 K	$\lvert\Delta w\rvert_{\max,\,\text{sat}} = 0.05\,\%$	$(\Delta w)_{\text{RMS}} = 0.0017\,\%$
$\eta(v,u)$	$\lvert\Delta\eta\rvert_{\text{perm}} = 0.001\,\%$	$\lvert\Delta\eta\rvert_{\max} = 0.000\,96\,\%$	6 K	$\lvert\Delta\eta\rvert_{\max,\,\text{sat}} = 0.0082\,\%$	$(\Delta\eta)_{\text{RMS}} = 0.000\,31\,\%$

Table 27: Deviations in temperature $T(p,h)$, specific volume $v(p,h)$, specific entropy $s(p,h)$, speed of sound $w(p,h)$, and dynamic viscosity $\eta(p,h)$ from the supplementary equation for the metastable-vapor region and the basic equation for region 2 of IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [65]

Spline function	Permissible deviation	Maximum deviation in the metastable-vapor region and in region 2 of IAPWS-IF97 outside the range $\lvert T - T_s(p)\rvert$ defined in the next column	Range $\lvert T - T_s(p)\rvert$ along the saturated vapor curve for $p < 10$ MPa		RMS deviation in the metastable-vapor region and in region 2 of IAPWS-IF97
			$\lvert T - T_s(p)\rvert$	Maximum deviation	
$T(p,h)$	$\lvert\Delta T\rvert_{perm} = 1\,\text{mK}$	$\lvert\Delta T\rvert_{max} = 0.90\,\text{mK}$	8 K	$\lvert\Delta T\rvert_{max,\,sat} = 18.4\,\text{mK}$	$(\Delta T)_{RMS} = 0.68\,\text{mK}$
$v(p,h)$	$\lvert\Delta v\rvert_{perm} = 0.001\,\%$	$\lvert\Delta v\rvert_{max} = 0.000\,60\,\%$	10 K	$\lvert\Delta v\rvert_{max,\,sat} = 0.018\,\%$	$(\Delta v)_{RMS} = 0.000\,28\,\%$
$s(p,h)$	$\lvert\Delta s\rvert_{perm} = 1\times10^{-6}\,\text{kJ/(kg K)}$	$\lvert\Delta s\rvert_{max} = 0.54\times10^{-6}\,\text{kJ/(kg K)}$	16 K	$\lvert\Delta s\rvert_{max,\,sat} = 0.84\times10^{-4}\,\text{kJ/(kg K)}$	$(\Delta s)_{RMS} = 0.85\times10^{-6}\,\text{kJ/(kg K)}$
$w(p,h)$	$\lvert\Delta w\rvert_{perm} = 0.001\,\%$	$\lvert\Delta w\rvert_{max} = 0.000\,89\,\%$	13 K	$\lvert\Delta w\rvert_{max,\,sat} = 0.045\,\%$	$(\Delta w)_{RMS} = 0.0011\,\%$
$\eta(p,h)$	$\lvert\Delta\eta\rvert_{perm} = 0.001\,\%$	$\lvert\Delta\eta\rvert_{max} = 0.000\,80\,\%$	6 K	$\lvert\Delta\eta\rvert_{max,\,sat} = 0.0061\,\%$	$(\Delta\eta)_{RMS} = 0.000\,15\,\%$

5.4 Computing-Time Comparisons

The computing times of the SBTL property functions have been evaluated and compared with those of IAPWS-IF97, where these functions are calculated from the basic equations, or, where available, from backward equations. The Computing-Time Ratio (*CTR*) is defined as:

$$CTR = \frac{\text{Computing time of the calculation from IAPWS-IF97 basic eq. or backward eq.}}{\text{Computing time of the calculation from the SBTL property function}}.$$

The IAPWS-IF97 property functions were computed from the Extended IAPWS-IF97 Steam Tables software [66]. Since the region definitions of the SBTL functions (see Figs. 41 and 43) are different from the regions of IAPWS-IF97 (see Fig. 7), the computing times of both formulations include the determination of the region that corresponds to the given state point. Neither IAPWS-IF97 nor the SBTL implementation takes advantage of information from previously calculated state points. The computing times were measured by means of software similar to NIFBENCH [3] with 100,000 randomly distributed state points in the corresponding region. All algorithms have been compiled into single-threaded software with the Intel Composer 2011 with default options. The tests were carried out on a Windows 8 computer equipped with an Intel Core i7-4500U CPU with 2.39 GHz and 8 GB RAM.

The results of the computing-time comparisons between the SBTL property functions of (v,u) presented in Sec. 5.1 and the iterative calculations from the IAPWS-IF97 basic equations are summarized in Table 28. In the single-phase region, the SBTL functions of (v,u) are between 130 and 471 times faster than those of the IAPWS-IF97 implementation. Computations from the inverse spline functions $u(p,v)$ and $v(u,s)$ with are between 2 and 134 times faster than those from the IAPWS-IF97 implementation.

Table 28: Computing-time ratios (*CTR*) of spline-based property functions of (v,u) and their inverse functions of (p,v) and (u,s) in comparison to the iterative calculations from the IAPWS-IF97 basic equations

	IAPWS-IF97 Region				
SBTL function	1	2	3	4	5
$p(v,u)$	130	271	161	19	470
$T(v,u)$	161	250	158	20	442
$s(v,u)$	164	261	160	17	449
$w(v,u)$	199	310	234	- [a]	471
$\eta(v,u)$	197	309	239	- [a]	- [b]
$u(p,v)$	2.0	6.4	2.8	5.6	3.2
$v(u,s)$	43	66	78	16	134

[a] Speed of sound w and dynamic viscosity η are not defined in the two-phase region.

[b] Since the upper temperature limit of the IAPWS viscosity release [65] is 1173.15 K, a spline function for the dynamic viscosity η in the high-temperature region is not provided.

The computing times of the SBTL property functions of (p,h) presented in Sec. 5.2 have been determined and compared to those of the iterative calculations from the IAPWS-IF97 basic equations that meet the demands for extremely high numerical consistency. As described in Sec. 2.3, backward equations are used to determine suitable starting values for these iterative calculations. The results of the computing-time comparisons are summarized in Table 29. In the single-phase region, the SBTL functions of (p,h) are between 17 and 1290 times faster than those of the IAPWS-IF97 implementation.

Table 29: Computing-time ratios (*CTR*) of SBTL property functions of (p,h) and their inverse functions of (p,T), (p,s), and (h,s) in comparison to the calculation from IAPWS-IF97 basic equations or backward equations (in parentheses)

SBTL function	IAPWS-IF97 Region				
	1	2	3	4	5
$T(p,h)$	17 (2.9)	23 (4.7)	580 (3.0)	17 (4.4)	53 (26)
$v(p,h)$	18 (3.8)	23 (6.1)	655 (5.1)	5.5 (2.6)	46 (25)
$s(p,h)$	28 (3.8)	31 (5.7)	918 (5.7)	9.0 (2.9)	39 (12)
$w(p,h)$	32 (5.0)	47 (10)	1160 (8.2)	-[a]	90 (30)
$\eta(p,h)$	31 (5.6)	45 (9.2)	1290 (7.9)	-[a]	-[b]
$h(p,T)$	0.94	0.71	1.5 (4.7)	-[c]	0.34
$h(p,s)$	4.6 (0.74)	8.1 (1.2)	88.1 (1.4)	5.2 (1.9)	9.3 (4.6)
$T(p,s)$	2.1 (0.50)	3.8 (0.94)	43 (0.76)	2.8 (1.0)	4.9 (3.8)
$p(h,s)$	36 (2.2)	207 (11)	474 (1.8)	25 (5.6)	396 (64)
$T(h,s)$	15 (2.0)	71 (8.6)	433 (1.7)	12 (5.8)	146 (52)

[a] Speed of sound w and dynamic viscosity η are not defined in the two-phase region.

[b] Since the upper temperature limit of the IAPWS viscosity release [65] is 1173.15 K, a spline function for the dynamic viscosity η in the high-temperature region is not provided.

[c] State points in the two-phase region are not uniquely defined with (p,T) inputs.

With the SBTL method, the specific enthalpy $h(p,T)$ is computed as an inverse spline function of $T(p,h)$, thus being numerically consistent with this function. This procedure is slower than the calculation of $h(p,T)$ in IAPWS-IF97 regions 1, 2, and 5, where the basic equations are directly applied. In heat cycle simulations, property functions are generally less frequently calculated from (p,T). The SBTL function $h(p,s)$ is computed as an inverse spline function of $s(p,h)$, and $T(p,s)$ is calculated from $T(p,h(p,s))$. Therefore, these property functions are also numerically consistent with each other. Analogously, the SBTL function $p(h,s)$ is computed as an inverse spline function of $s(p,h)$, and $T(h,s)$ is calculated from $T(p(h,s),h)$. For the IAPWS-IF97 implementation, the region boundaries are calculated by iteration from their original definitions in the (p,T) plane. The starting values for these calculations are obtained from the available region-boundary equations. Again, the property functions of (p,s) and (h,s) are calculated from the IAPWS-IF97 basic equations by iteration with starting values obtained from the available backward equations. The *CTR* values of the inverse spline functions of (p,T),

(p,s), and (h,s) with regard to computations from the IAPWS-IF97 basic equations are given in Table 29.

For the sake of completeness, the computing times of the SBTL functions have been compared to those obtained through the direct application of backward, auxiliary, and region-boundary equations. A suitable implementation of IAPWS-IF97 functions without iterative procedures is also available in [66]. The resulting *CTR* values are given in parentheses in Table 29. More details on computing-time comparisons between calculations from backward equations and iterative calculations from the IAPWS-IF97 basic equations can be found in [67].

The results of the computing-time comparisons for SBTL functions of (v,u) and (p,h) for the metastable-vapor region presented in Sec. 5.3 are summarized in Tables 30 and 31.

Table 30: Computing-time ratios (*CTR*) of SBTL property functions of (v,u) compared to the iterative calculations from the IAPWS-IF97 supplementary equation for the metastable-vapor region

	SBTL function				
	$p(v,u)$	$T(v,u)$	$s(v,u)$	$w(v,u)$	$\eta(v,u)$
CTR	88.3	86.4	89.5	87.0	90.0

Table 31: Computing-time ratios (*CTR*) of SBTL property functions of (p,h) compared to the iterative calculations from the IAPWS-IF97 supplementary equation for the metastable-vapor region

	SBTL function				
	$T(p,h)$	$v(p,h)$	$s(p,h)$	$w(p,h)$	$\eta(p,h)$
CTR	16.0	16.0	12.1	15.7	19.0

As discussed in Sec. 2.5.3, the results of computing-time comparisons are always implementation dependent. In order to compare the computing times of the SBTL property functions with fast implementations of the Peng-Robinson equation of state [14] (PR-EOS), IAPWS-95 [2], the short fundamental equation of state for water by Kunz et al. [21], IAPWS-IF97 [4], IAPWS-IF97 backward equations [4], and the TTSE method [10], the *CTR* values as defined in Sec. 2.5.3 are given in Table 32. For these comparisons it is assumed that the phase that corresponds to the given inputs is known for all calculations.

Table 32: Computing-time ratios (*CTR*) with regard to the Peng-Robinson equation of state for SBTL property functions of (v,u) and (p,h)

	CTR values as defined in Sec. 2.5.3 for SBTL functions			
IAPWS-IF97 Region	$p(v,u)$	$T(v,u)$	$T(p,h)$	$v(p,h)$
1	1.84	2.29	19.0	19.0
2	2.91	2.93	10.4	10.4

6 SBTL Property Functions Based on IAPWS-95 for Water and Steam

The IAPWS-95 formulation for general and scientific use [1, 2] is the most accurate representation of the thermodynamic properties of water and steam. The IAPWS-IF97 formulation for industrial use [3, 4] and the supplementary releases [5, 6, 7, 8] were developed based on IAPWS-95 to meet specific needs for higher computing speeds in many industrial applications, particularly for the steam power industry. The range of validity of IAPWS-IF97 is divided into five regions, resulting in small inconsistencies at the region boundaries. In situations where these inconsistencies cannot be tolerated, and/or for general and scientific use where the more accurate IAPWS-95 formulation is preferred, it may be useful to apply the SBTL method to IAPWS-95. In order to demonstrate the applicability of the SBTL method to IAPWS-95, several spline-based property functions for calculations from (v,u) and (p,h) have been developed. For simplicity in developing this example, spline functions covering the region of temperatures from 273.15 K to 1273.15 K and pressures up to 1000 MPa are described. This excludes a small portion of the range of validity of IAPWS-95 at high pressures and low temperatures, but application of the SBTL method in that region would be a straightforward extension. The SBTL method has been applied to the industrial formulation IAPWS-IF97 through the use of FluidSplines (see Sec. 4). The documentation of the SBTL property functions based on IAPWS-95 given in this section was originally published in [61].

6.1 Spline Functions of (v,u)

Spline functions based on IAPWS-95 for the calculation of $p, T, s, w = f(v,u)$ in the single-phase region were created analogously to those based on IAPWS-IF97 (see Sec. 5.1). The results of the computing-time comparisons are summarized in Sec. 6.3.

6.1.1 Range of Validity

The range of validity covers the fluid range of state bounded as follows:

$$273.15\ \text{K} \le T \le 1273.15\ \text{K} \qquad 611.212\ \text{Pa} \le p \le 1000\ \text{MPa}\,.$$

This range of validity corresponds to that of IAPWS-95, except for the lower temperature limit, which is 273.15 K, and the lower pressure limit, which is $p_s(273.15\ \text{K}) = 611.212\ \text{Pa}$. Figure 44 shows the range of validity and the defined regions of the spline functions with the variables (v,u). The range of validity is divided into the liquid region L, the gas region G, and the two-phase region TP. This division is similar to the division for IAPWS-IF97 shown in Fig. 41, except that no separate high-temperature region HT is needed.

The specific internal energy at the critical point $u_c = 2015.734\,524$ kJ/kg is used to define the boundary between regions L and G for supercritical state points. At the region boundary in the single-phase region, small inconsistencies are unavoidable. These should be negligible for most purposes, but if needed the transition at this boundary can be smoothed using simple interpolation equations.

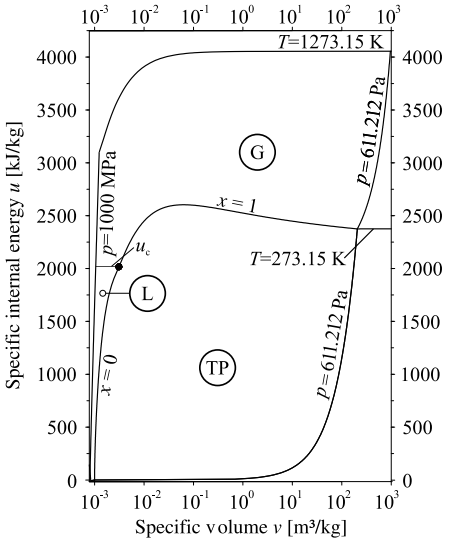

Figure 44: Range of validity in the (u,v) plane for spline functions based on IAPWS-95.

Note: For temperatures between 273.15 K and 273.16 K, see the note at the end of Sec. 5.1.1.

6.1.2 Spline Functions for the Single-Phase Region

In each of the single-phase regions L and G, spline functions with the variables (v,u) were created. In the liquid region L, a scaling transformation (see Sec. 3.2.2) for the specific volume v with the boundary curves $v_{\min}(u) = v(p_{\max} = 1000\ \text{MPa}, u)$ and $v_{\max}(u) = v'(u)$ is applied, so that

$$\overline{v}(v,u) = \frac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)} \left(v - v_{\min}(u)\right) + \overline{v}_{\min},$$

where the free parameters are set to $\overline{v}_{\min} = 1$ and $\overline{v}_{\max} = 100$. Thus, the shape of the grid of nodes corresponds to the shape of the liquid region L (see Fig. 44). In the gas region G, the specific volume is transformed as $\overline{v} = \ln(v)$. The grid dimensions of each (v,u) spline function are given in Tables A8 and A9 in Appendix A11. Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation.

6.1.3 Deviations from IAPWS-95

The maximum (max) and root-mean-square (RMS) deviations between the spline functions and IAPWS-95, along with the permissible values (perm), are given in Tables 33 through 36. The permissible values were set by the IAPWS Task Group "CFD Steam Property Formulation" to ensure that the differences in the results of process simulations with the SBTL method from those obtained with the direct application of IAPWS-95 are negligible. The permissible values are less than or equal to the required numerical consistencies for the IAPWS-IF97 backward equations [3, 5, 6, 7, 8]. The deviations in $p(v,u)$, $T(v,u)$, and $s(v,u)$ of the SBTL property

functions from IAPWS-95 in the liquid region L and the gas region G are depicted in Appendix A13, Figs. A14-A19.

Table 33: Deviations in pressure $p(v,u)$ from IAPWS-95

Region		$\lvert\Delta p\rvert_{\text{perm}}$	$\lvert\Delta p\rvert_{\max}$	$(\Delta p)_{\text{RMS}}$
L	$p \le 2.5\,\text{MPa}$	0.6 %	0.092 %	0.0080 %
	$p > 2.5\,\text{MPa}$	15 kPa	2.74 kPa	0.0090 kPa
G		0.001 %	0.001 % [a]	0.000 12 %

[a] Except for near-critical states, where $\lvert\Delta p\rvert_{\max} < 0.01\,\%$.

Table 34: Deviations in temperature $T(v,u)$ from IAPWS-95

Region	$\lvert\Delta T\rvert_{\text{perm}}$ [mK]	$\lvert\Delta T\rvert_{\max}$ [mK]	$(\Delta T)_{\text{RMS}}$ [mK]
L	1	0.34	0.029
G	1	1 [a]	0.017

[a] Except for near-critical states, where $\lvert\Delta T\rvert_{\max} < 10\,\text{mK}$.

Table 35: Deviations in specific entropy $s(v,u)$ from IAPWS-95

Region	$\lvert\Delta s\rvert_{\text{perm}}$ $\left[10^{-6}\,\text{kJ/(kg K)}\right]$	$\lvert\Delta s\rvert_{\max}$ $\left[10^{-6}\,\text{kJ/(kg K)}\right]$	$(\Delta s)_{\text{RMS}}$ $\left[10^{-6}\,\text{kJ/(kg K)}\right]$
L	1	0.53	0.017
G	1	0.26	0.045

Table 36: Deviations in speed of sound $w(v,u)$ from IAPWS-95

Region	$\lvert\Delta w\rvert_{\text{perm}}$	$\lvert\Delta w\rvert_{\max}$	$(\Delta w)_{\text{RMS}}$
L	0.001 %	0.001 % [a]	0.000 92 %
G	0.001 %	0.001 % [b]	0.000 039 %

[a] In the vicinity of the critical point, the deviations of w are larger but less than 0.4 %.

[b] In the vicinity of the critical point, the deviations of w are larger but less than 5 %.

6.2 Spline Functions of (*p*,*h*)

Spline functions based on IAPWS-95 for the calculation of $T, v = f(p,h)$ in the single-phase region were created analogously to those based on IAPWS-IF97 (see Sec. 5.2). The results of the computing-time comparisons are summarized in Sec. 6.3.

6.2.1 Range of Validity

The range of validity covers the fluid range of state bounded as follows:

$$273.15\ \text{K} \le T \le 1273.15\ \text{K} \qquad 611.212\ \text{Pa} \le p \le 1000\ \text{MPa}.$$

This range of validity corresponds to that of IAPWS-95, except for the lower temperature limit, which is 273.15 K, and the lower pressure limit, which is $p_s(273.15\ \text{K}) = 611.212\ \text{Pa}$. Figure 45 shows the range of validity and the defined regions of the spline functions with the variables (*p*,*h*). The range of validity is divided into the liquid region L, the gas region G, and the two-phase region TP. This division is similar to the division for IAPWS-IF97 shown in Fig. 43, except that no separate high-temperature region HT is needed.

The specific enthalpy at the critical point $h_c = 2084.256\,263$ kJ/kg is used to define the boundary between regions L and G for supercritical state points. At the region boundary in the single-phase region, small inconsistencies are unavoidable. These should be negligible for most purposes, but if needed the transition at this boundary can be smoothed using simple interpolation equations.

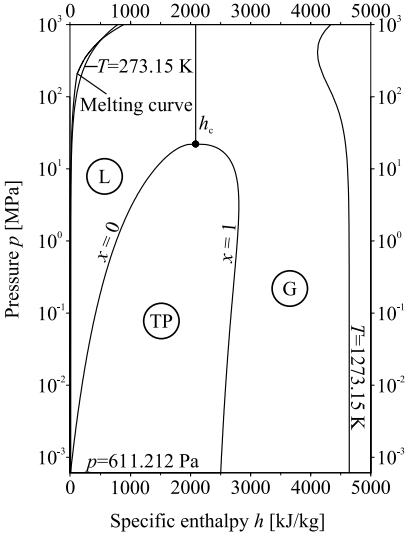

Figure 45: Range of validity in the (*p*,*h*) plane for spline functions based on IAPWS-95.

Note: For temperatures between 273.15 K and 273.16 K, see the note at the end of Sec. 5.1.1.

6.2.2 Spline Functions for the Single-Phase Region

In each of the single-phase regions L and G, spline functions with the variables (*p*,*h*) were created. In the gas region G, a transformation for the specific volume *v* of the form $\overline{v} = pv$ is applied. The grid dimensions of each (*p*,*h*) spline function are given in Table A10 in Appendix A11. Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation.

6.2.3 Deviations from IAPWS-95

The maximum (max) and root-mean-square (RMS) deviations between the spline functions and IAPWS-95, along with the permissible values (perm), are given in Tables 37 and 38. The permissible values were set by the IAPWS Task Group "CFD Steam Property Formulation" to ensure that the differences in the results of process simulations with the SBTL method from those obtained with the direct application of IAPWS-95 are negligible. The permissible values are less than or equal to the required numerical consistencies for the IAPWS-IF97 backward equations [3, 5, 6, 7, 8]. The deviations in *T*(*p*,*h*) and *v*(*p*,*h*) of the SBTL property functions from IAPWS-95 in the liquid region L and the gas region G are depicted in Appendix A13, Figs. A20 and A21.

Table 37: Deviations in temperature *T*(*p*,*h*) from IAPWS-95

Region	$\lvert\Delta T\rvert_{\text{perm}}$ [mK]	$\lvert\Delta T\rvert_{\max}$ [mK]	$(\Delta T)_{\text{RMS}}$ [mK]
L	1	1 [a]	0.033
G	1	1 [a]	0.025

[a] Except for near-critical states, where $\lvert\Delta T\rvert_{\max} < 10$ mK.

Table 38: Deviations in specific volume *v*(*p*,*h*) from IAPWS-95

Region	$\lvert\Delta v\rvert_{\text{perm}}$	$\lvert\Delta v\rvert_{\max}$	$(\Delta v)_{\text{RMS}}$
L	0.001 %	0.001 % [a]	0.000 062 %
G	0.001 %	0.001 % [a]	0.000 016 %

[a] Except for near-critical states, where $\lvert\Delta v\rvert_{\max} < 0.03$ %.

6.3 Computing-Time Comparisons

The computing times of the spline functions described in Secs. 6.1 and 6.2 have been evaluated and compared with those of IAPWS-95. The Computing-Time Ratio (*CTR*) is:

$$CTR = \frac{\text{Computing time of the calculation from IAPWS-95}}{\text{Computing time of the calculation from the SBTL algorithms}} .$$

The IAPWS-95 property functions were computed from the internal routines of REFPROP [59] where the phase (liquid or vapor) is known and no phase tests are performed. Calculations from the IAPWS-95 fundamental equation and its derivatives are computationally

expensive. In addition, depending on the considered property function, one- or two-dimensional iteration procedures are used in the REFPROP software. The resulting computing times are more than 100 times longer than for computations from SBTL functions. The computing times were measured by means of software similar to NIFBENCH [3] with 100,000 randomly distributed state points in the corresponding region. The compiler to build the test programs and the computer to run the test calculations are described in Sec. 5.4. The results of the computing-time comparisons are summarized in Table 39.

Table 39: Computing-time ratios (*CTR*) of spline-based property functions compared to calculations from IAPWS-95

	Region	
SBTL function	L	G
$p(v,u)$	243	434
$T(v,u)$	251	410
$T(p,h)$	≈15 000	6760
$v(p,h)$	≈14 500	6900

7 Bicubic Spline Functions for the Thermodynamic Potential $s(v,u)$ for Water and Steam

The biquadratic spline functions described in Secs. 5 and 6 represent the underlying property formulations with high accuracy. Therefore, such functions can be considered as quasi thermodynamically consistent, as discussed in Sec. 2.4.3. Table look-up methods providing full thermodynamic consistency have been described for instance by Schot [31], Herbst [44], Swesty [41], and Pini [37] (see Sec. 2.4.2). For comparisons, bicubic spline functions for the thermodynamic potential $s(v,u)$ have been prepared based on IAPWS-95 through the use of FluidSplines (see Sec. 4).

7.1 Range of Validity

Similarly to the SBTL property functions described in Sec. 6.1, the range of validity covers the fluid range of state bounded as follows:

$$273.15\ \text{K} \le T \le 1273.15\ \text{K} \qquad 611.212\ \text{Pa} \le p \le 1000\ \text{MPa}\,.$$

Again, the range of validity is divided into the liquid region L, the gas region G, and the two-phase region TP. The metastable-liquid and the metastable-vapor regions at the vapor-liquid phase transition are included in the liquid region L and the gas region G, respectively. For the metastable-vapor region no experimental data are available. IAPWS-95 produces reasonable values close to the saturation line. For calculations further away from the saturation line, the so-called gas equation [2] is recommended in [1]. Since the upper density limit of the gas equation is 55 kg/m^3, this equation cannot be applied to generate the nodes for the entire gas region G. To ensure a continuous property surface, all nodes were calculated solely from IAPWS-95. In the metastable-vapor region, the gas region G is bounded by the vapor spinodal and $u = 2000$ kJ/kg. The range of validity is shown in Fig. 46.

The specific internal energy at the critical point $u_c = 2015.734\,524$ kJ/kg is used to define the boundary between regions L and G for supercritical state points. At the region boundary in the single-phase region, small inconsistencies are unavoidable. These should be negligible for most purposes, but if needed the transition at this boundary can be smoothed using simple interpolation equations.

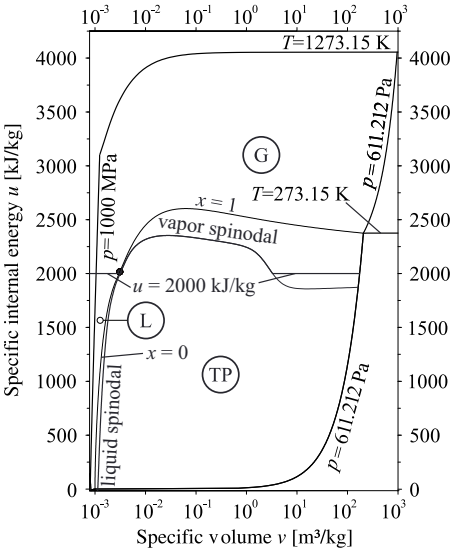

Figure 46: Range of validity in the (u,v) plane for $s(v,u)$ based on IAPWS-95.

Note: For temperatures between 273.15 K and 273.16 K, see the note at the end of Sec. 5.1.1.

7.2 Property Calculations in the Single-Phase Region

In each of the single-phase regions L and G, a bicubic spline function for the thermodynamic potential $s(v,u)$ was created. In the liquid region L, a scaling transformation (see Sec. 3.2.2) for the specific volume v between the 1000 MPa isobar $v_{\min}(u) = v(p_{\max} = 1000\ \text{MPa}, u)$ and the liquid spinodal $v_{\max}(u) = v_{\text{liq\_spin}}(u)$ is applied, so that

$$\overline{v}(v,u) = \frac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)} \left(v - v_{\min}(u)\right) + \overline{v}_{\min},$$

where the free parameters are set to $\overline{v}_{\min} = 1$ and $\overline{v}_{\max} = 100$. Thus, the shape of the grid of nodes corresponds to the shape of the liquid region L (see Fig. 46). In the gas region G, the specific volume is transformed as $\overline{v} = \ln(v)$. The grid dimensions of the $s(v,u)$ spline functions are given in Tables A11 and A12 in Appendix A11. Nodes outside the range of validity needed for the construction of a rectangular grid of nodes are obtained by appropriate extrapolation. The relationships between the thermodynamic potential $s(v,u)$ and the remaining thermodynamic properties are given in Table 40.

Table 40: Relationships of thermodynamic properties to specific entropy $s(v,u)$ and its derivatives

Property	Relationship
Pressure p $p = -(\partial u/\partial v)_s$	$p(v,u) = \left(\frac{\partial s}{\partial v}\right)_u \left(\frac{\partial s}{\partial u}\right)_v^{-1}$
Temperature T $T = (\partial u/\partial s)_v$	$T(v,u) = \left(\frac{\partial s}{\partial u}\right)_v^{-1}$
Gibbs free energy g $g = u + pv - Ts$	$g(v,u) = u + \left[v\left(\frac{\partial s}{\partial v}\right)_u + s\right]\left(\frac{\partial s}{\partial u}\right)_v^{-1}$
Specific enthalpy h $h = u + pv$	$h(v,u) = u + v\left(\frac{\partial s}{\partial v}\right)_u \left(\frac{\partial s}{\partial u}\right)_v^{-1}$
Specific isochoric heat capacity c_v $c_v = (\partial u/\partial T)_v$	$c_v(v,u) = -\left(\frac{\partial s}{\partial u}\right)_v^2 \left(\frac{\partial^2 s}{\partial u^2}\right)_v^{-1}$
Specific isobaric heat capacity c_p $c_p = (\partial h/\partial T)_p$	$c_p(v,u) = \frac{\left(\frac{\partial h}{\partial v}\right)_u \left(\frac{\partial s}{\partial u}\right)_v - \left(\frac{\partial h}{\partial u}\right)_v \left(\frac{\partial s}{\partial v}\right)_u}{\left(\frac{\partial T}{\partial v}\right)_u \left(\frac{\partial s}{\partial u}\right)_v - \left(\frac{\partial T}{\partial u}\right)_v \left(\frac{\partial s}{\partial v}\right)_u}$
Speed of sound w $w = v\sqrt{-(\partial p/\partial v)_s}$	$\frac{w^2(v,u)}{v^2} = -\left(\frac{\partial p}{\partial v}\right)_s = \left(\frac{\partial^2 u}{\partial v^2}\right)_s$

The derivatives of $s(v,u)$ with respect to the independent variables v and u are derived from the bicubic spline polynomial, Eq. (2.61) and the corresponding transformation functions. The remaining derivatives required for the relationships given in Table 40 read

$$\left(\frac{\partial h}{\partial v}\right)_u = \left(\frac{\partial s}{\partial v}\right)_u \left(\frac{\partial s}{\partial u}\right)_v^{-1} + v\left[\left(\frac{\partial^2 s}{\partial v^2}\right)_u \left(\frac{\partial s}{\partial u}\right)_v - \left(\frac{\partial s}{\partial v}\right)_u \left(\frac{\partial^2 s}{\partial v \partial u}\right)\right]\left(\frac{\partial s}{\partial u}\right)_v^{-2},$$

$$\left(\frac{\partial h}{\partial u}\right)_v = 1 + v\left[\left(\frac{\partial^2 s}{\partial v \partial u}\right)\left(\frac{\partial s}{\partial u}\right)_v - \left(\frac{\partial s}{\partial v}\right)_u \left(\frac{\partial^2 s}{\partial u^2}\right)_v\right]\left(\frac{\partial s}{\partial u}\right)_v^{-2},$$

$$\left(\frac{\partial T}{\partial v}\right)_u = -\left(\frac{\partial^2 s}{\partial v \partial u}\right)\left(\frac{\partial s}{\partial u}\right)_v^{-2},$$

$$\left(\frac{\partial T}{\partial u}\right)_v = -\left(\frac{\partial^2 s}{\partial u^2}\right)_v\left(\frac{\partial s}{\partial u}\right)_v^{-2},$$

and

$$\left(\frac{\partial p}{\partial v}\right)_s = \frac{\left(\frac{\partial^2 s}{\partial v^2}\right)_u\left(\frac{\partial s}{\partial u}\right)_v + \left(\frac{\partial s}{\partial v}\right)_u\left[\left(\frac{\partial s}{\partial v}\right)_u\left(\frac{\partial s}{\partial u}\right)_v^{-1}\left(\frac{\partial^2 s}{\partial u^2}\right)_v - 2\left(\frac{\partial^2 s}{\partial v \partial u}\right)\right]}{\left(\frac{\partial s}{\partial u}\right)_v^2}.$$

7.3 Deviations from IAPWS-95

The maximum (max) and root-mean-square (RMS) deviations of the bicubic spline functions for $s(v,u)$ and the derived property functions for $p(v,u)$, $T(v,u)$, and $c_v(v,u)$ from IAPWS-95 are given in Tables 41 through 44. In addition, the deviations are depicted in Appendix A14, Figs. A22-A29.

Table 41: Deviations in specific entropy $s(v,u)$ from IAPWS-95

Region	$\lvert\Delta s\rvert_{\max}$ $\left[\mathrm{kJ/(kg\,K)}\right]$	$(\Delta s)_{\mathrm{RMS}}$ $\left[\mathrm{kJ/(kg\,K)}\right]$
L	0.97×10^{-9}	0.21×10^{-9}
G	1.00×10^{-8} [a]	0.67×10^{-9}

[a] Except for near-critical states, where $\lvert\Delta s\rvert_{\max} < 5\times10^{-8}$ kJ/(kg K).

Table 42: Deviations in pressure $p(v,u)$ from IAPWS-95

Region		$\lvert\Delta p\rvert_{\max}$	$(\Delta p)_{\mathrm{RMS}}$
L	$p \le 2.5$ MPa	2.22 %	0.40 %
	$p > 2.5$ MPa	0.81 kPa	0.023 kPa
G		0.001 % [a]	0.000 011 %

[a] Except for near-critical states, where $\lvert\Delta p\rvert_{\max} < 0.012$ %.

Table 43: Deviations in temperature $T(v,u)$ from IAPWS-95

Region	$\lvert\Delta T\rvert_{\max}$ [mK]	$(\Delta T)_{\mathrm{RMS}}$ [mK]
L	0.79	0.0081
G	1 [a]	0.044

[a] Except for near-critical states, where $\lvert\Delta T\rvert_{\max} < 5$ mK .

Table 44: Deviations in specific isochoric heat capacity $c_v(v,u)$ from IAPWS-95

Region	$\lvert\Delta c_v\rvert_{\max}$	$(\Delta c_v)_{\mathrm{RMS}}$
L	0.067 % [a]	0.0018 %
G	0.1 % [b]	0.0035 %

[a] For 1900 kJ/kg $< u \le u_c$, the maximum deviations of c_v are larger but less than 4 %.

[b] In the vicinity of the critical point, the deviations of c_v are larger but less than 7 %.

Properties, which depend on the first derivatives of $s(v,u)$ only, such as p and T, exhibit comparatively small deviations from the underlying IAPWS-95 formulation. Other properties, which depend on higher order derivatives, such as c_v, show increased deviations. These deviations exceed the permissible values described in Sec. 6.1.3, but are smaller than the uncertainties of the underlying IAPWS-95 formulation.

7.4 Computing-Time Comparisons

The computing times of the bicubic spline functions for $s(v,u)$ and the derived property functions for $p(v,u)$, $T(v,u)$ have been evaluated and compared with those of IAPWS-95. The IAPWS-95 implementation, the definition of the Computing-Time Ratio (*CTR*), and the test procedure are equal to those described in Sec. 6.3. The computing times have been measured in the stable single-phase regions only. The spline functions are implemented to return p, T, and s at the same time. The resulting *CTR* values for the liquid region L and the gas region G are given in Table 45. Similarly, such functions can be implemented to efficiently compute other sets of thermodynamic properties at once (see Table 40). For inverse functions, the analytical solution of bicubic polynomials is computationally much more expensive than that of biquadratic polynomials.

Table 45: Computing-time ratios (*CTR*) of property functions derived from the bicubic spline functions for $s(v,u)$ compared to calculations from IAPWS-95

	Region	
SBTL function	L	G
$p,T,s = f(v,u)$	96	159

7.5 Property Calculations in the Two-Phase Region

To calculate the properties in the two-phase region, the set of equations $\boldsymbol{F}(\boldsymbol{X})$, namely

$$\boldsymbol{F}_1(\boldsymbol{X}) = 0 = p^{\mathrm{L}}(v',u') - p^{\mathrm{G}}(v'',u''), \tag{3.94}$$

$$\boldsymbol{F}_2(\boldsymbol{X}) = 0 = T^{\mathrm{L}}(v',u') - T^{\mathrm{G}}(v'',u''), \tag{3.95}$$

$$\boldsymbol{F}_3(\boldsymbol{X}) = 0 = g^{\mathrm{L}}(v',u') - g^{\mathrm{G}}(v'',u''), \text{ and} \tag{3.96}$$

$$\boldsymbol{F}_4(\boldsymbol{X}) = 0 = \frac{v-v'}{v''-v'} - \frac{u-u'}{u''-u'}. \tag{3.97}$$

must be solved for the vector of unknowns $\boldsymbol{X} = \left(v', u', v'', u''\right)^{\mathrm{T}}$. This can be done through the use of Newton's method for non-linear systems of equations by solving

$$\boldsymbol{J}(\boldsymbol{X}_k)\,\Delta\boldsymbol{X}_k = \boldsymbol{F}(\boldsymbol{X}_k) \text{ and} \tag{3.98}$$

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \Delta\boldsymbol{X}_k \tag{3.99}$$

in each iteration step k until convergence is reached. The Jacobian matrix $\boldsymbol{J}(\boldsymbol{X})$ is given as

$$\boldsymbol{J}(\boldsymbol{X}) = \tag{3.100}$$

$$\begin{bmatrix}
\left(\frac{\partial p^{\mathrm{L}}}{\partial v}\right)_u (v',u') & \left(\frac{\partial p^{\mathrm{L}}}{\partial u}\right)_v (v',u') & -\left(\frac{\partial p^{\mathrm{G}}}{\partial v}\right)_u (v'',u'') & -\left(\frac{\partial p^{\mathrm{G}}}{\partial u}\right)_v (v'',u'') \\
\left(\frac{\partial T^{\mathrm{L}}}{\partial v}\right)_u (v',u') & \left(\frac{\partial T^{\mathrm{L}}}{\partial u}\right)_v (v',u') & -\left(\frac{\partial T^{\mathrm{G}}}{\partial v}\right)_u (v'',u'') & -\left(\frac{\partial T^{\mathrm{G}}}{\partial u}\right)_v (v'',u'') \\
\left(\frac{\partial g^{\mathrm{L}}}{\partial v}\right)_u (v',u') & \left(\frac{\partial g^{\mathrm{L}}}{\partial u}\right)_v (v',u') & -\left(\frac{\partial g^{\mathrm{G}}}{\partial v}\right)_u (v'',u'') & -\left(\frac{\partial g^{\mathrm{G}}}{\partial u}\right)_v (v'',u'') \\
\frac{(v-v')-(v''-v')}{(v''-v')^2} & -\frac{(u-u')-(u''-u')}{(u''-u')^2} & -\frac{(v-v')}{(v''-v')^2} & \frac{(u-u')}{(u''-u')^2}
\end{bmatrix}.$$

8 Application of the SBTL Method in Computationally Expensive Process Simulations

8.1 Computational Fluid Dynamics

To consider the real-fluid behavior of water and steam in Computational Fluid Dynamics (CFD), SBTL property functions of (v,u) based on IAPWS-IF97 (see Sec. 5.1) have been implemented into the density-based CFD software TRACE [68, 69]. This part of the work was carried out in a joint project with the German Aerospace Center (DLR), where TRACE is being developed and maintained. The implementation was verified through the use of several test cases by di Mare [70]. As one of these test cases, the condensing steam flow around a fixed blade in a low-pressure turbine stage was simulated. The geometry and the boundary conditions were taken from a publication of White et al. [71], which also provides a detailed description of measurements for verification. The simulations with TRACE were carried out for three different property calculation algorithms, namely the existing IAPWS-IF97 implementation, the newly developed SBTL property functions of (v,u) based on IAPWS-IF97, and the ideal-gas model. The IAPWS-IF97 implementation is based on [66] and is extended for property functions of (v,u). The computation of these property functions is carried out by iteration from the IAPWS-IF97 basic equations. Starting values for property calculations at the current volume element are provided from the previous iteration step in TRACE. For the ideal-gas model, constant specific isobaric heat capacity is assumed. The findings obtained from the comparison of the simulations with the three different property calculation algorithms regarding their numerical results, computing times, and convergence behavior are published in [72]. In comparison to the direct IAPWS-IF97 implementation, the computing times of the CFD simulations were reduced by factors between 6 and 10 through the use of the SBTL functions. With regard to CFD calculations where steam is considered to be an ideal gas, the computing-time consumption with the SBTL functions is increased by a factor of only 1.4. The numerical results show negligible differences from those obtained from simulations with the direct IAPWS-IF97 implementation.

More recently, the flow solver applied in TRACE has been further modified by Post [73] to enhance its capabilities for simulations considering real fluid properties. To compare the different property calculation algorithms, the de Laval nozzle described by Moore et al. [74], see Fig. 47, was simulated [75] using the IAPWS-IF97 implementation, the SBTL functions, and the ideal gas model. The governing equations as well as the spatial discretization of the nozzle geometry and the solution scheme are described in [73]. At the inlet, the total pressure p_t is 500 kPa and the total temperature T_t is 520 K. Since the flow is supersonic at the outlet, the flow through the nozzle is fully determined. The nozzle height at the throat h_{throat} is 0.0315 m. For the different simulations using IAPWS-IF97 and the SBTL property functions, homogeneous two-phase flow with condensation at equilibrium conditions is assumed.

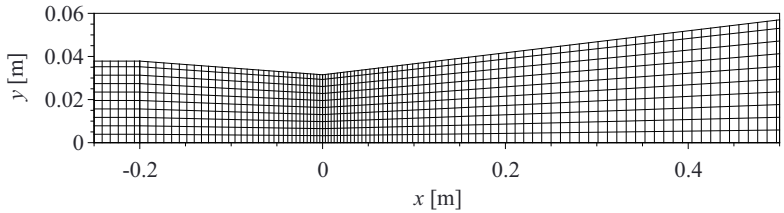

Figure 47: Computational grid for the de Laval nozzle with only every third grid line in each direction shown.

The nozzle height and the axial distribution of area averaged pressure ratio and liquid mass fraction are shown in Fig. 48. The results of the simulation using the SBTL property functions are practically equal to those obtained through the direct application of IAPWS-IF97.

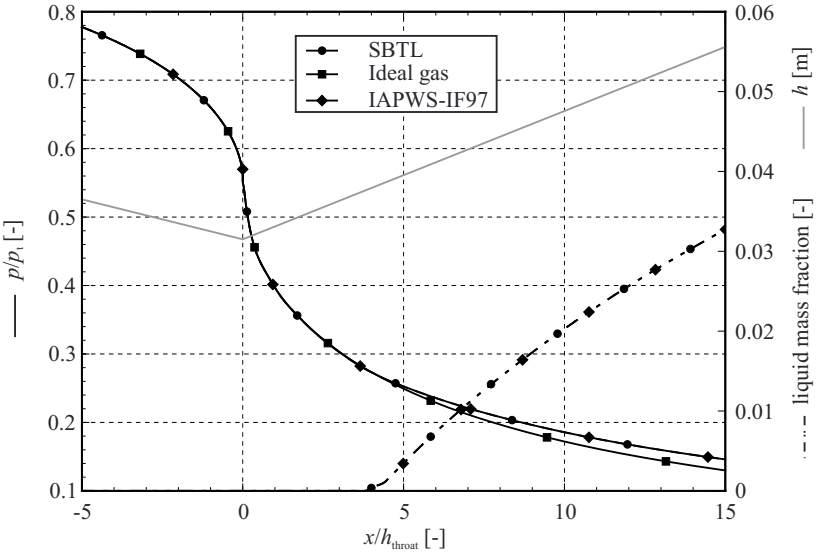

Figure 48: Nozzle height h and area averaged axial distributions of pressure p and liquid fraction along x.

The L1 residuals for the different simulations using the SBTL property functions, the IAPWS-IF97 implementation, and the ideal-gas model are shown in Fig. 49. Convergence is reached for all three property calculation methods after approximately 500 iteration steps. Due to the numerical consistency of the SBTL property functions with their inverse functions, the computation is very stable using this approach. With regard to the direct application of IAPWS-IF97, the overall computing times for flow simulations through the de Laval nozzle using the SBTL property functions are reduced by a factor of 5. In comparison to the ideal-gas model, the computing times of simulations with the SBTL property functions increased by a factor of 2.5. It seems likely that the numerical consistency of the property functions of the IAPWS-IF97

implementation causes the residuals to be larger. For some test calculations, the simulations using the IAPWS-IF97 even failed to converge. This could be avoided by decreasing the tolerances for the iterative calculations from the IAPWS-IF97 basic equations. Of course, this would extend the computing times even further. Currently, the tolerances for the internal energy u and the specific volume v are set to $|\Delta u| \le 0.001\,\text{kJ/kg}$ and $|\Delta v/v| \le 10^{-6}$.

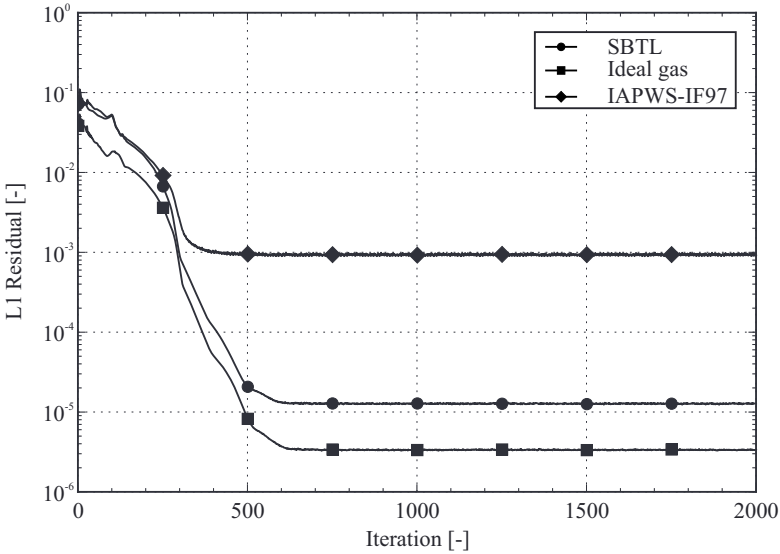

Figure 49: L1 residuals for the different simulations using the SBTL property functions, the IAPWS-IF97 implementation, and the ideal-gas model.

8.2 Heat Cycle Simulations

In software tools for heat cycle simulations, the balance equations of mass, energy, and entropy often lead to (p,h), (p,s), and (h,s) input variable combinations. To reduce the overall computing times of heat cycle simulations, the SBTL property functions of (p,h) based on IAPWS-IF97 and their inverse functions (see Sec. 5.2) have been implemented in two different software tools for heat cycle simulations.

The heat cycle simulation software KRAWAL-modular is being developed and maintained by SIEMENS PG. This software is primarily used as an in-house tool for power plant design calculations. In a first study performed by Bennoit [76], the computing times of $T(p,h)$, $s(p,h)$, $h(p,s)$, and $p(h,s)$ computed from IAPWS-IF97 and from the SBTL property functions were compared. In KRAWAL-modular, the industrial formulation IAPWS-IF97 is computed from the Extended Steam Tables software package [66]. The IAPWS-IF97 property functions of (p,h), (p,s), and (h,s) are calculated by iteration from the IAPWS-IF97 basic equations, where the starting values are determined from the available backward equations [3, 4, 5, 6, 7]. The region boundaries are calculated from their original definitions in the (p,T) plane, which requires iterative procedures in some cases (see Sec. 2.3). The iterative calculations from the

IAPWS-IF97 basic equations and region boundaries are required to ensure high numerical consistency of forward and backward functions, as pointed out in Sec. 2.3. As described in [76], computations of $T(p,h)$, $s(p,h)$, $h(p,s)$, and $p(h,s)$ from the SBTL functions are at least 10 times faster than those from the IAPWS-IF97 implementation [66]. It must be noted, that the computing times reported in [66] are given as averaged values for each property function but no distinction is made in what region these functions are called. Through the use of the complete set of SBTL property functions of (p,h) based on IAPWS-IF97 and their inverse functions (see Sec. 5.2) in KRAWAL-modular [77], the overall computing times are reduced on average by 50 % with regard to the direct application of IAPWS-IF97. The observed differences in the engineering design parameters are less than 0.02 % and therefore negligible.

The SBTL property functions of (p,h) based on IAPWS-IF97 and their inverse functions have also been implemented in the commercial heat cycle simulation software EBSILON® Professional [78]. In this software, the direct implementation of IAPWS-IF97 is derived from the LibIF97 property library [79] with a few modifications. In the original version of LibIF97, the backward functions are calculated from the available backward equations to avoid time consuming calculations from the IAPWS-IF97 basic equations by iteration. Where applicable, the region boundaries are calculated from the region-boundary equations (see Sec. 2.3). Since the numerical consistency of the backward equations for $T(p,h)$ with the IAPWS-IF97 basic equations does not meet the requirements in EBSILON® Professional, this property function is calculated by iteration from the corresponding basic equation. In contrast to KRAWAL-modular, some components, such as turbine stage groups and steam generators, are described with a lower level of detail in EBSILON® Professional. Thus, the calculation of fluid properties consumes a smaller share of the overall computing time in EBSILON® Professional. To compare the SBTL property functions with the direct application of IAPWS-IF97, a conventional steam power plant was simulated with both implementations independently [80]. For the direct application of IAPWS-IF97, 25% of the overall computing time were spent on property calculations for water and steam. With the SBTL property functions, the share of property calculations in the overall computing time is reduced to 8% and the overall computing times are reduced by 17% with regard to the direct application of IAPWS-IF97.

The practical application of SBTL property functions in software tools for heat cycle simulations demonstrates their usefulness for this kind of numerical process simulations. While the numerical results for the engineering design parameters obtained from simulations using the SBTL method differ negligibly from those obtained through the direct application of IAPWS-IF97, the overall computing times can be reduced significantly. The reduction of the overall computing times depends on the computing-time share of the previously implemented property functions and their computational speed. If the industrial formulation IAPWS-IF97 and its supplementary backward equations are in use, then the savings in computing time when switching to the SBTL property functions are relatively small. In cases where backward functions are calculated by iteration from the IAPWS-IF97 basic equations, the application of SBTL property functions leads to more significant computing-time savings. With regard to process simulations with IAPWS-95 or property formulations for other fluids, the overall computing times can be drastically reduced through the use of the SBTL method.

8.3 Nuclear Reactor System Safety Analysis

In RELAP-7 (Reactor Excursion and Leak Analysis Program) [81], the nuclear reactor system safety analysis code currently being developed at the Idaho National Laboratory (INL), the solution of the balance equations of mass, momentum, and total energy requires fluid property calculations from (v,u). For fully non-equilibrium, fully compressible, two-phase flows a novel 7-equation two-phase flow model is in use. This model describes both phases, liquid and vapor, independently. Each phase may be in a stable or in a metastable state. If both phases are in equilibrium (equal pressures, temperatures, Gibbs free energies, and velocities), the 7-equation two-phase flow model reduces to the 3-equation homogeneous equilibrium model. To consider the real fluid behavior of water and steam consistently for both models, the SBTL method has been applied to IAPWS-95 [1] and the latest IAPWS standards on viscosity [65] and thermal conductivity [82]. Differing from the SBTL property functions described in Sec. 6.1, the range of validity is bounded by:

$$273.15\ \mathrm{K} \le T \le 1273.15\ \mathrm{K} \qquad 611.212\ \mathrm{Pa} \le p \le 100\ \mathrm{MPa}\,.$$

The single-phase region is divided into the liquid region L (see Fig. 50) and the gas region G (see Fig. 51), where the corresponding metastable regions are included. For supercritical state points, the boundary between the liquid region L and the gas region G is defined by the specific internal energy at the critical point $u_c = 2015.734\,524$ kJ/kg.

In each of the regions L and G, spline functions for the calculation of $p, T, s, w, \eta, \lambda = f(v,u)$ were created based on IAPWS-95 and the current IAPWS standards on viscosity [65] and thermal conductivity [82]. As recommended in [1], metastable-vapor properties further away from the saturation line should be calculated from the so-called gas equation [2], rather than from IAPWS-95. Since the transition from IAPWS-95 to the gas equation along its maximum density of 55 kg/m^3 is not smooth in the metastable region, all nodes were calculated from IAPWS-95. The correlating equations for the dynamic viscosity η [65] and the thermal conductivity λ [82] contain critical enhancement terms to describe the behavior of these properties in the critical region. The critical enhancement terms depend on the derivative $(\partial v/\partial p)_T$, which is infinite along the spinodals. This causes numerical difficulties and therefore, the critical enhancement terms were omitted for the generation of the $\eta(v,u)$ and $\lambda(v,u)$ spline functions. For the dynamic viscosity η, the critical enhancement is significant in a very small region around the critical point only and its omission is recommended in [65] to simplify the calculation for industrial use. For the thermal conductivity λ, the critical enhancement is significant in a larger range around the critical point as discussed in [82].

In the liquid region L, a scaling transformation (see Sec. 3.2.2) for the specific volume v with the boundary curves $v_{\min}(u) = v(p_{\max} = 100\ \mathrm{MPa}, u)$ and $v_{\max}(u) = v_{\mathrm{liq\_spin}}(u)$ is applied, so that

$$\overline{v}(v,u) = \frac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)} \left(v - v_{\min}(u)\right) + \overline{v}_{\min}\,,$$

where the free parameters are set to $\overline{v}_{\min} = 1$ and $\overline{v}_{\max} = 100$. Thus, the shape of the grid of nodes corresponds to the shape of the liquid region L (see Fig. 50). In the gas region G, the specific volume is transformed as $\overline{v} = \ln(v)$. A detailed description of the (v,u) spline functions regarding their grid dimensions and the achieved accuracy is given in [83].

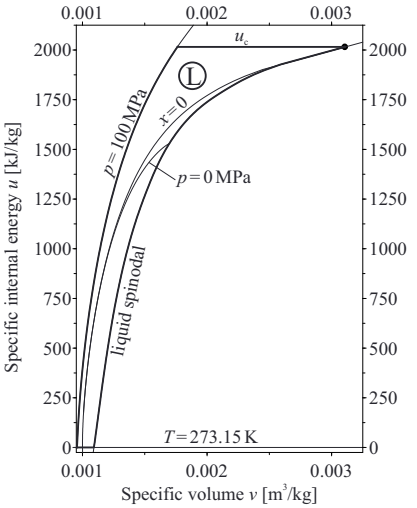

Figure 50: Liquid region L in the (u,v) plane.

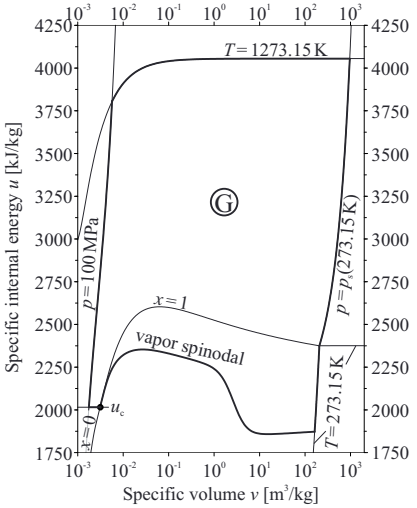

Figure 51: Gas region G in the (u,v) plane.

The specific enthalpy h and the specific Gibbs free energy g are calculated from $h = u + pv$ and $g = h - Ts$. Since the calculation of $g(v,u)$ involves the evaluation of $p(v,u)$, $T(v,u)$, and $s(v,u)$, additional spline functions are provided for even faster computations of $g(v,u)$. The isobaric and isochoric heat capacities are calculated from $p(v,u)$ and $T(v,u)$ according to their definitions

$$c_p = \left(\frac{\partial h}{\partial T}\right)_p = \frac{\left(\frac{\partial p}{\partial v}\right)_u - p\left(\frac{\partial p}{\partial u}\right)_v}{\left(\frac{\partial T}{\partial u}\right)_v \left(\frac{\partial p}{\partial v}\right)_u - \left(\frac{\partial T}{\partial v}\right)_u \left(\frac{\partial p}{\partial u}\right)_v} \quad \text{and} \qquad c_v = \left(\frac{\partial u}{\partial T}\right)_v .$$

Since c_p and c_v contain the first derivatives of the biquadratic spline functions for $p(v,u)$ and $T(v,u)$, the first derivatives of c_p and c_v are discontinuous. In many process simulations, these derivatives are not required. For all other cases, additional spline functions for the calculation of $c_p(v,u)$ and $c_v(v,u)$ are provided.

For calculations in the stable or metastable single-phase regions, inverse functions of (p,T), (p,v), (p,h), (p,s), and (h,s) are provided. In the gas phase, the inverse function (p,v) can be calculated without any iterations as an inverse spline function (see Sec. 3.2.3). For all other cases, the inverse functions are calculated by iteration from the (v,u) spline functions using Newton's method (see Appendix A3) and accurate auxiliary spline functions for the starting values. The saturation states could be calculated from the Maxwell criterion, *i.e.*, equal pressures, temperatures, and specific Gibbs free energies for both phases; but for the sake of simplicity, a spline function for $T_s(p)$ is used instead. For this function, the pressure p is

transformed as $\overline{p} = \sqrt{p}$. For given (v,u), the properties at saturation are calculated by solving the set of Eqs. (3.79) – (3.83) in Sec. 3.2.5 for the vector of unknowns $\boldsymbol{X} = \left(p_{\mathrm{s}}, v', u', v'', u'' \right)^{\mathrm{T}}$. The variables x_1 and x_2 in Eq. (3.83) are v and u. Calculations in the two-phase region for given (h,s) are performed in a similar fashion, but the variables x_1, x_1', x_1'', x_2, x_2', and x_2'' in Eq. (3.83) are h, $h' = u' + p_{\mathrm{s}} v'$, $h'' = u'' + p_{\mathrm{s}} v''$, s, $s' = s^{\mathrm{L}}\left(v', u'\right)$, and $s'' = s^{\mathrm{G}}\left(v'', u''\right)$. For given ($p$,$v$), ($p$,$h$), or ($p$,$s$), the properties at saturation are determined from the inverse functions of (p,T), where $T = T_{\mathrm{s}}\left(p\right)$. The property functions described above have been implemented into the newly developed property library LibSBTL_vu_95 [84]. An overview of calculable functions and analytical derivatives is given in Table 46.

The property library LibSBTL_vu_95 has been implemented into the nuclear reactor system safety analysis code RELAP-7 as reported in [81]. In this way, the thermodynamic and transport properties of water and steam are calculated with high accuracy and short computing times in the 7-equation two-phase flow model and in the 3-equation homogeneous equilibrium model. The test calculations documented in [83] show, that simulations using the SBTL property functions are only 2% slower than those with the stiffened gas equation of state, which was implemented in RELAP-7 during its development process.

Table 46: Property functions and derivatives exported from LibSBTL_vu_95 [84]

Independent variables	Dependent variables z	Derivatives
(v,u)	p, T, x, g, h, s, c_p, c_v, w, η, λ	$\left(\frac{\partial z}{\partial v}\right)_u, \left(\frac{\partial z}{\partial u}\right)_v, \left(\frac{\partial u}{\partial v}\right)_z$
(p,T)	v, u, h	$\left(\frac{\partial z}{\partial p}\right)_T, \left(\frac{\partial z}{\partial T}\right)_p, \left(\frac{\partial p}{\partial T}\right)_z$
(p,v)	u	$\left(\frac{\partial z}{\partial p}\right)_v, \left(\frac{\partial z}{\partial v}\right)_p, \left(\frac{\partial p}{\partial v}\right)_z$
(p,h)	T, x, v, u	$\left(\frac{\partial z}{\partial p}\right)_h, \left(\frac{\partial z}{\partial h}\right)_p, \left(\frac{\partial p}{\partial h}\right)_z$
(p,s)	v, u	$\left(\frac{\partial z}{\partial p}\right)_s, \left(\frac{\partial z}{\partial s}\right)_p, \left(\frac{\partial p}{\partial s}\right)_z$
(h,s)	p, T, x, v, u	$\left(\frac{\partial z}{\partial h}\right)_s, \left(\frac{\partial z}{\partial s}\right)_h, \left(\frac{\partial h}{\partial s}\right)_z$

9 Summary and Outlook

The aim of this work was the development of suitable property calculation algorithms for complex process simulations, such as flow analysis with Computational Fluid Dynamics (CFD), power-plant design with heat cycle calculations, and real-time process optimizations. In order to achieve the best possible results with these simulations, the applied property functions must be very accurate. Since these functions are called extremely often during the process simulation, their computing times must be very low. To ensure convergence, the property functions need to be continuously differentiable once. Furthermore, simulations with small spatial and time discretizations require the property functions to be numerically and thermodynamically consistent.

In order to find a suitable approach which meets the requirements, the currently available algorithms for calculating the properties of water and steam were compared with regard to their accuracy and their computing time. The attainable accuracy of an equation of state is determined by its functional form. The computing time required for evaluating the equation of state results from the necessary mathematical operations. Property functions of (v,u) and (p,h), which are frequently used in CFD and in heat cycle calculations, must be calculated by iteration from the equation of state, which typically depends on (T,v) or (p,T), respectively. This leads to extended computing times. Because of their comparatively short computing times, simple thermal equations of state, such as the ideal-gas equation or the Peng-Robinson equation, combined with an equation for the isobaric heat capacity of the ideal gas, are often used in CFD. However, depending on the range of state, these equations may be very inaccurate, which in turn leads to errors in the simulation results.

Many fundamental equations of state, such as IAPWS-95, contain numerous transcendental terms, such as exponential functions and logarithms. As the computation of these terms is very expensive, the property functions calculated from IAPWS-95 are between 30 and 100 times slower than those calculated from the Peng-Robinson equation. Therefore, such equations of state are inapplicable in extensive process simulations. This is also true for short equations of state for industrial applications, such as the equation of Kunz et al., whose property functions are between 10 to 26 times slower than those from the Peng-Robinson equation.

The industrial formulation IAPWS-IF97 enables fast and sufficiently accurate property calculations by combining fundamental equations of state for separate regions with backward equations, which are all optimized for computing speed. For example, property functions of (p,h) are calculated by iteration from the IAPWS-IF97 basic equations with starting values from the corresponding backward equation. Such calculations are numerically consistent and the computing times are comparable to those of calculations from the Peng-Robinson equation. Due to the absence of suitable backward equations, the IAPWS-IF97 property functions of (v,u) are up to 11 times slower than those of the Peng-Robinson equation. Thus, even IAPWS-IF97 is too slow for extensive process simulations, where the independent variables of the applied property functions are different from those of the basic equations and backward equations. The development of fast fundamental equations of state for separate regions and suitable backward equations is very time consuming and was therefore not pursued in this work.

Table look-up methods are a fast and accurate alternative for property calculations. The accuracy of a table look-up method can be controlled by the number and the distribution of the tabulated data points. Furthermore, table look-up methods can be flexibly applied to various property functions. The computing times of such methods are essentially dependent on the algorithms used for interval or cell search. With the TTSE method, which was studied as an example, property functions of (p,h) are more than 4 times faster than those of the Peng-Robinson equation. However, the discontinuities along the TTSE property functions lead to numerical problems in process simulations with very small spatial and time discretizations. The advantages and disadvantages of different table look-up methods have been studied and conclusions have been drawn for the development of an advanced table look-up method based on spline interpolation.

The newly developed Spline-Based Table Look-Up method (SBTL) approximates the results of existing equations of state or transport property equations, with high accuracy and low computing time. This is enabled by the combination of polynomial spline interpolation techniques, specialized variable transformations, and piecewise equidistant nodes for simplified search algorithms. Second order polynomial spline functions are continuously differentiable once and enable the fast calculation of numerically consistent inverse functions. Therefore, quadratic and biquadratic spline functions are preferred, but the method can also be extended to employ third or higher order polynomials. The mathematical details of the SBTL method are described comprehensively in this work.

The software tool FluidSplines has been developed to enable the fast application of the SBTL method to one- and two-dimensional property functions of any fluid. For this purpose, the software provides an extensible interface to calculate the underlying property functions from external property libraries, such as REFPROP for instance. Through the use of FluidSplines, SBTL property functions can be generated for the range of validity and the desired accuracy specified by the user. Currently, FluidSplines implements second- and third-order polynomial spline interpolation methods only, but it may be extended to other methods as well. For two-dimensional functions, an algorithm has been developed to determine whether or not a given state point is located in a region arbitrarily described by a set of bounding curves. Furthermore, several algorithms have been implemented to extrapolate nodes beyond the range of validity, if this is required during the spline-generation process. The calculated nodes and the deviations of the generated SBTL property function from the underlying property function can be assessed in detail by means of suitable two- and three dimensional diagrams. The generated SBTL property functions can be exported from FluidSplines as pure C source code, which is optimized regarding its computational speed.

To provide fast and accurate property functions for water and steam, the SBTL method has been applied to the industrial formulation IAPWS-IF97 and to the IAPWS-95 formulation for general and scientific use. For each of these formulations, two different sets of SBTL functions have been generated. The SBTL functions of (v,u) can be used for instance in CFD, whereas the SBTL functions of (p,h) are useful in heat cycle calculations, for example. The maximum deviations of these functions from their underlying property formulations are less than the permissible values. The permissible deviations in the single-phase region are 0.001 % for p, v, w, and η, except for the liquid region, where $|\Delta p|_{\mathrm{perm}} = 0.6\,\%$ for $p \le 2.5$ MPa and $|\Delta p|_{\mathrm{perm}} = 15$ kPa

for $p > 2.5$ MPa. The permissible deviations for T and s are 10 mK and 10^{-6} kJ/(kg K). These values were set by the IAPWS Task Group "CFD Steam Property Formulation" to ensure that the differences in the results of process simulations with the SBTL method from those obtained with the direct application of the corresponding IAPWS formulation are negligible. Furthermore, a certain thermodynamic consistency is ensured in this way. In addition to the SBTL functions of (v,u) based on IAPWS-IF97, numerically consistent inverse functions of (p,v) and (u,s) are provided. Similarly, numerically consistent inverse functions of (p,T), (p,s), and (h,s) are provided for the SBTL functions of (p,h) based on IAPWS-IF97.

The computing times of the SBTL functions have been evaluated and compared with those of the IAPWS-IF97 implementation given in [66]. Since the region definitions of the SBTL functions are different from those of IAPWS-IF97, the computing times of both formulations include the determination of the region that corresponds to the given state point. In the single-phase region, the SBTL functions of (v,u) are between 130 and 471 times faster than those of the IAPWS-IF97 implementation. The SBTL functions of (p,h) are at least 17 times faster than the corresponding functions of the IAPWS-IF97 implementation, in which backward and boundary equations are used to determine the starting values for the iterative calculations from the basic equations. With regard to computations from (p,h), where backward and boundary equations are directly applied, the SBTL functions are between 3 and 30 times faster.

The SBTL method has also been applied to the thermodynamic potential $s(v,u)$. For this purpose, bicubic spline functions have been generated for the liquid and the vapor phase based on IAPWS-95. From $s(v,u)$, the properties p, T, g, h, c_p, c_v, and w can be calculated consistently from their thermodynamic definitions. The resulting property functions were compared to the underlying IAPWS-95 formulation. In the gas phase, the maximum deviations in p and T are less than 0.001 % and 1 mK. The maximum deviation in c_v is 0.1 %, which is less than the uncertainty of the underlying IAPWS-95 formulation. Simultaneous computations of p, T, and s from the bicubic spline functions for $s(v,u)$ are between 100 and 160 times faster than those from IAPWS-95.

The applicability of the SBTL method in extensive process simulations has also been demonstrated in this work. In a joint project with the German Aerospace Center (DLR), SBTL functions of (v,u) and the corresponding inverse functions based on IAPWS-IF97 have been implemented into the CFD software TRACE. Several test simulations were carried out for three different property calculation algorithms, namely the existing IAPWS-IF97 implementation, the newly developed SBTL functions of (v,u) based on IAPWS-IF97, and the ideal-gas model. In comparison to simulations with the direct implementation of IAPWS-IF97, the computing times were reduced by factors between 5 and 10 through the use of the SBTL functions. With regard to simulations where steam is considered to be an ideal gas, the computing times are increased by factors between 1.4 and 2.5. The numerical results show negligible differences from those obtained through the direct application of the IAPWS-IF97 implementation in TRACE.

The SBTL functions of (p,h) and their corresponding inverse functions based on IAPWS-IF97 have been implemented into two different heat cycle simulation codes. In this way, the overall computing times of simulations with KRAWAL-modular, the heat cycle simulation software developed by SIEMENS PG, are reduced on average by 50 % with regard to the direct

application of IAPWS-IF97. The observed differences in the engineering design parameters are less than 0.02 % and therefore negligible. Similarly, the overall computing times of simulations with EBSILON® Professional, the heat cycle simulation software developed by STEAG Energy Services GmbH, are reduced by 17% with regard to the direct application of IAPWS-IF97.

For RELAP-7, the nuclear reactor system safety analysis code currently being developed at the Idaho National Laboratory (INL), SBTL functions of (v,u) and inverse functions of (p,T), (p,v), (p,h), (p,s), and (h,s) were developed based on IAPWS-95. The range of validity of these property functions includes the metastable-liquid and metastable-vapor regions at the vapor-liquid phase transition. Through the use of the SBTL property functions, the real fluid behavior of water and steam can be considered and the application of a novel 7-equation non-equilibrium two-phase flow model is enabled. Simulations using the SBTL functions based on IAPWS-95 are only 2% slower than those with the stiffened gas equation of state.

This work contributes to the activities within the IAPWS Task Group “CFD Steam Property Formulation”. The newly developed Spline-Based Table Look-up method (SBTL) has been adopted as an IAPWS-Guideline [61].

The application of the SBTL method to pure fluids, such as heavy water, carbon dioxide, helium, and others, is currently in progress. Future activities focus on the extension of the SBTL method to mixtures, such as humid air, humid combustion gas mixtures, etc.

Appendix

A1 Grid Optimization Algorithm of Kretzschmar et al.

The grid optimization algorithm proposed by Kretzschmar et al. [46, 47] is illustrated in Figs. A1-A4. Starting from the coarse grid depicted in Fig. A1, the accuracy is tested in each cell $\{i,j\}$. If the maximum deviation in each cell exceeds the required accuracy, the node density in both directions is increased independently and the resulting accuracy in each cell is tested again for both cases. This is illustrated in Figs. A2 and A3. In Fig. A2, at each grid line at constant $\bar{x}_2$, an additional node is inserted. The nodes along each of these grid lines are kept equidistant. Then, instead of inserting a node at each grid line at constant $\bar{x}_2$, the number of grid lines is increased by one as shown in Fig. A3. Again, the new grid lines are aligned equidistantly. The grid with the lowest maximum deviation in $z(x_1,x_2)$ is selected. If a range of cells meets the required accuracy, it is excluded from the optimization. If, for instance, the cells {1,1} and {2,1} in the grid depicted in Fig. A3 are sufficiently accurate, then the node density in the remaining grid is increased independently. This is shown for the grid of nodes in Fig. A4, where additional nodes are only inserted at those grid lines at constant $\bar{x}_2$, that bound the cells with higher deviations.

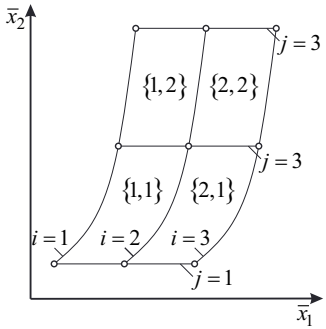

Figure A1: Initial grid of nodes for grid optimization.

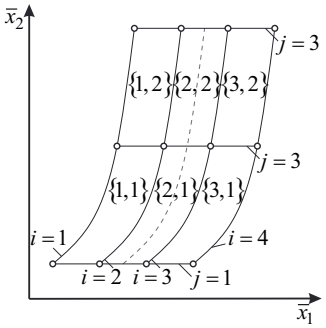

Figure A2: Grid of nodes with an additional node along $\bar{x}_1$ at each grid line at constant $\bar{x}_2$.

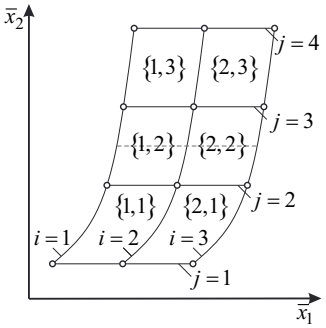

Figure A3: Grid of nodes with an additional grid line at constant $\bar{x}_2$.

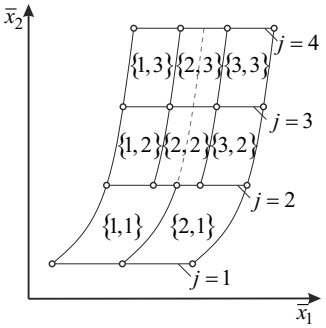

Figure A4: Grid of nodes with locally increased node density.

A2 Relationships between the Derivatives of the Residual Helmholtz Free Energy

The calculation of thermodynamic properties from a Helmholtz free energy equation often requires the computation of several derivatives. The residual part of the Helmholtz free energy equation $\Phi^{\mathrm{r}}(\delta,\tau)$ and its derivatives with respect to δ and τ, namely $\Phi^{\mathrm{r}}_{\delta(n)}$ and $\Phi^{\mathrm{r}}_{\tau(m)}$, can often be expressed as

$$\Phi^{\mathrm{r}}(\delta,\tau)=\sum_i \Phi^{\mathrm{r}}_i(\delta,\tau), \tag{A2.1}$$

$$\delta^n \Phi^{\mathrm{r}}_{\delta(n)}(\delta,\tau)=\sum_i \Phi^{\mathrm{r}}_i(\delta,\tau) F_i^{\delta(n)}(\delta,\tau), \tag{A2.2}$$

and

$$\tau^m \Phi^{\mathrm{r}}_{\tau(m)}(\delta,\tau)=\sum_i \Phi^{\mathrm{r}}_i(\delta,\tau) F_i^{\tau(m)}(\delta,\tau). \tag{A2.3}$$

The n-th derivative with respect to δ and the m-th derivative with respect to τ are multiplied by their respective powers δ^n and τ^m since the resulting expressions are frequently used for property calculations as can be seen in Table 3 in Sec. 2.2.1. In Eqs. (A2.2) and (A2.3), these expressions are calculated from the products of the term Φ^{r}_i itself and a factor $F_i^{\delta(n)}$ or $F_i^{\tau(m)}$, respectively.

For instance, the $(n-1)$-th derivative with respect to δ can be written as

$$\Phi^{\mathrm{r}}_{i,\delta(n-1)}=\delta^{1-n}\Phi^{\mathrm{r}}_i F_i^{\delta(n-1)}. \tag{A2.4}$$

The derivative of Eq. (A2.4) with respect to δ leads to the n-th derivative

$$\Phi^{\mathrm{r}}_{i,\delta(n)}=\delta^{1-n}\Phi^{\mathrm{r}}_{i,\delta} F_i^{\delta(n-1)}+\Phi^{\mathrm{r}}_i\left[\delta^{1-n}F_{i,\delta}^{\delta(n-1)}+(1-n)\delta^{-n}F_i^{\delta(n-1)}\right]. \tag{A2.5}$$

Inserting $\Phi^{\mathrm{r}}_{i,\delta}=\delta^{-1}\Phi^{\mathrm{r}}_i F_i^{\delta}$ and rearranging for $\delta^n\Phi^{\mathrm{r}}_{i,\delta(n)}$ yields

$$\delta^n\Phi^{\mathrm{r}}_{i,\delta(n)}=\Phi^{\mathrm{r}}_i\left\{\delta F_{i,\delta}^{\delta(n-1)}+F_i^{\delta(n-1)}\left[F_i^{\delta}-(n-1)\right]\right\}. \tag{A2.6}$$

Analogously, the expression

$$\tau^m \Phi^{\mathrm{r}}_{i,\tau(m)} = \Phi^{\mathrm{r}}_i \left\{ \tau F^{\tau(m-1)}_{i,\tau} + F^{\tau(m-1)}_i \left[F^{\tau}_i - (m-1) \right] \right\}. \tag{A2.7}$$

can be derived. The expressions in braces are the factors $F^{\delta(n)}_i$ and $F^{\tau(m)}_i$, namely

$$F^{\delta(n)}_i = \delta F^{\delta(n-1)}_{i,\delta} + F^{\delta(n-1)}_i \left[F^{\delta}_i - (n-1) \right] \tag{A2.8}$$

and

$$F^{\tau(m)}_i = \tau F^{\tau(m-1)}_{i,\tau} + F^{\tau(m-1)}_i \left[F^{\tau}_i - (m-1) \right]. \tag{A2.9}$$

Considering that $F^{\delta(0)}_i = 1$, $F^{\tau(0)}_i = 1$,

$$F^{\delta(n)}_{i,\delta} = \delta F^{\delta(n-1)}_{i,\delta\delta} + F^{\delta(n-1)}_{i,\delta} \left[F^{\delta}_i - n + 2 \right] + F^{\delta(n-1)}_i F^{\delta}_{i,\delta} \tag{A2.10}$$

and

$$F^{\tau(m)}_{i,\tau} = \tau F^{\tau(m-1)}_{i,\tau\tau} + F^{\tau(m-1)}_{i,\tau} \left[F^{\tau}_i - m + 2 \right] + F^{\tau(m-1)}_i F^{\tau}_{i,\tau} \tag{A2.11}$$

the factors $F^{\delta(n)}_i$ and $F^{\tau(m)}_i$ can be determined recursively.

A3 Newton's Method for Two Dimensions

The iteration procedure for Newton's method to solve two non-linear equations simultaneously reads

$$x_{1,k+1} = x_{1,k+1} - \frac{f_2 \left(\frac{\partial f_1}{\partial x_2} \right)_{x_1} - f_1 \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1}}{DEN} \tag{A3.1}$$

and

$$x_{2,k+1} = x_{2,k+1} - \frac{f_1 \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2} - f_2 \left(\frac{\partial f_1}{\partial x_1} \right)_{x_2}}{DEN} \tag{A3.2}$$

with

$$DEN = \left(\frac{\partial f_1}{\partial x_2} \right)_{x_1} \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2} - \left(\frac{\partial f_1}{\partial x_1} \right)_{x_2} \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1}. \tag{A3.3}$$

A4 Newton's Method for Three Dimensions

The iteration procedure for Newton's method to solve three non-linear equations simultaneously reads

$$x_{1,k+1} = x_{1,k+1} - \left\{ f_1 \left[\left(\frac{\partial f_2}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_3}{\partial x_3} \right)_{x_2,x_3} - \left(\frac{\partial f_2}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_3}{\partial x_2} \right)_{x_1,x_3} \right] \right.$$
$$+ f_2 \left[\left(\frac{\partial f_1}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_3}{\partial x_2} \right)_{x_1,x_3} - \left(\frac{\partial f_1}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_3}{\partial x_3} \right)_{x_1,x_2} \right] \quad , \tag{A4.1}$$
$$\left. + f_3 \left[\left(\frac{\partial f_1}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_2}{\partial x_3} \right)_{x_2,x_3} - \left(\frac{\partial f_1}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1,x_3} \right] \right\} DEN^{-1}$$

$$x_{2,k+1} = x_{2,k+1} - \left\{ f_1 \left[\left(\frac{\partial f_2}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_3}{\partial x_1} \right)_{x_2,x_3} - \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_3}{\partial x_3} \right)_{x_1,x_2} \right] \right.$$
$$+ f_2 \left[\left(\frac{\partial f_1}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_3}{\partial x_3} \right)_{x_1,x_2} - \left(\frac{\partial f_1}{\partial x_3} \right)_{x_2,x_3} \left(\frac{\partial f_3}{\partial x_1} \right)_{x_2,x_3} \right] \quad , \tag{A4.2}$$
$$\left. + f_3 \left[\left(\frac{\partial f_1}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2,x_3} - \left(\frac{\partial f_1}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_2}{\partial x_3} \right)_{x_1,x_2} \right] \right\} DEN^{-1}$$

and

$$x_{3,k+1} = x_{3,k+1} - \left\{ f_1 \left[\left(\frac{\partial f_2}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_3}{\partial x_2} \right)_{x_1,x_3} - \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_3}{\partial x_1} \right)_{x_2,x_3} \right] \right.$$
$$+ f_2 \left[\left(\frac{\partial f_1}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_3}{\partial x_1} \right)_{x_2,x_3} - \left(\frac{\partial f_1}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_3}{\partial x_2} \right)_{x_1,x_3} \right] \tag{A4.3}$$
$$\left. + f_3 \left[\left(\frac{\partial f_1}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1,x_3} - \left(\frac{\partial f_1}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2,x_3} \right] \right\} DEN^{-1}$$

with

$$DEN = \left[\left(\frac{\partial f_1}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1,x_3} - \left(\frac{\partial f_1}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2,x_3} \right] \left(\frac{\partial f_3}{\partial x_3} \right)_{x_2,x_3}$$
$$+ \left[\left(\frac{\partial f_1}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_2}{\partial x_1} \right)_{x_2,x_3} - \left(\frac{\partial f_1}{\partial x_1} \right)_{x_2,x_3} \left(\frac{\partial f_2}{\partial x_3} \right)_{x_1,x_2} \right] \left(\frac{\partial f_3}{\partial x_2} \right)_{x_1,x_3} \quad . \tag{A4.4}$$
$$+ \left[\left(\frac{\partial f_1}{\partial x_2} \right)_{x_1,x_3} \left(\frac{\partial f_2}{\partial x_3} \right)_{x_1,x_2} - \left(\frac{\partial f_1}{\partial x_3} \right)_{x_1,x_2} \left(\frac{\partial f_2}{\partial x_2} \right)_{x_1,x_3} \right] \left(\frac{\partial f_3}{\partial x_1} \right)_{x_2,x_3}$$

A5 Property Calculations in the Two-Phase Region from (*p*,*h*)

In order to calculate the properties in the two-phase region from (*p*,*h*), the following algorithm is recommended. In addition to the (*p*,*h*) spline functions in the liquid region L and gas region G, a function for $T_s(p)$ and the inverse spline functions $h^L(p,T)$ and $h^G(p,T)$ are provided. From these functions, $T_s = T_s(p)$, $h' = h^L(p,T_s)$, and $h'' = h^G(p,T_s)$ are determined without iteration. The vapor fraction x is calculated from $x = (h-h')/(h''-h')$ and the desired mass-specific properties are calculated from $z = z' + x(z''-z')$.

A6 Property Calculations in the Two-Phase Region from (*p*,*s*)

In order to calculate the properties in the two-phase region from (*p*,*s*), the following algorithm is recommended. In addition to the (*p*,*h*) spline functions in the liquid region L and gas region G, a function for $T_s(p)$ and the inverse spline functions $h^L(p,T)$ and $h^G(p,T)$ are provided. From these functions $T_s = T_s(p)$, $h' = h^L(p,T_s)$, $h'' = h^G(p,T_s)$, $s' = s^L(p,h')$, and $s'' = s^G(p,h'')$ are determined without iteration. The vapor fraction x is calculated from $x = (s-s')/(s''-s')$ and the desired mass-specific properties are calculated from $z = z' + x(z''-z')$.

A7 Property Calculations in the Two-Phase Region from (*h*,*s*)

In order to calculate the properties in the two-phase region from (*h*,*s*), the following algorithm is recommended. In addition to the (*p*,*h*) spline functions in the liquid region L and gas region G, a function for $T_s(p)$ and the inverse spline functions $h^L(p,T)$ and $h^G(p,T)$ are provided.

With the use of a one-dimensional Newton iteration scheme, the equation

$$f(p) = 0 = x_h(p) - x_s(p) \tag{A7.1}$$

is solved for the pressure p, where

$$x_h(p) = \frac{h - h'(p)}{h''(p) - h'(p)} \quad \text{and} \quad x_s(p) = \frac{s - s'(p)}{s''(p) - s'(p)}. \tag{A7.2, A7.3}$$

The iteration procedure is

$$p_{k+1} = p_k - \frac{f(p_k)}{\frac{\mathrm{d}f}{\mathrm{d}p}(p_k)}, \tag{A7.4}$$

where

$$\frac{\mathrm{d}f}{\mathrm{d}p}(p_k) = \frac{\mathrm{d}x_h}{\mathrm{d}p}(p_k) - \frac{\mathrm{d}x_s}{\mathrm{d}p}(p_k). \tag{A7.5}$$

In each iteration step k, $T_{s,k} = T_s(p_k)$ is calculated with the corresponding spline function. From the inverse spline functions $h^L(p,T)$ and $h^G(p,T)$, $h'(p_k)$ and $h''(p_k)$ are determined as $h'(p_k) = h^L(p_k, T_{s,k})$ and $h''(p_k) = h^G(p_k, T_{s,k})$. Then, $s'(p_k) = s^L(p_k, h'(p_k))$ and $s''(p_k) = s^G(p_k, h''(p_k))$ are subsequently calculated.

The derivatives in Eq. (A7.5) are calculated from

$$\frac{\mathrm{d}x_h}{\mathrm{d}p} = \frac{-\dfrac{\mathrm{d}h'}{\mathrm{d}p} - x_h\left(\dfrac{\mathrm{d}h''}{\mathrm{d}p} - \dfrac{\mathrm{d}h'}{\mathrm{d}p}\right)}{\left(h'' - h'\right)} \quad \text{and} \quad \frac{\mathrm{d}x_s}{\mathrm{d}p} = \frac{-\dfrac{\mathrm{d}s'}{\mathrm{d}p} - x_s\left(\dfrac{\mathrm{d}s''}{\mathrm{d}p} - \dfrac{\mathrm{d}s'}{\mathrm{d}p}\right)}{\left(s'' - s'\right)} \qquad \text{(A7.6, A7.7)}$$

where

$$\frac{\mathrm{d}h'}{\mathrm{d}p} = \left(\frac{\partial h}{\partial p}\right)_T + \left(\frac{\partial h}{\partial T}\right)_p \frac{\mathrm{d}T_\mathrm{s}}{\mathrm{d}p}, \qquad \frac{\mathrm{d}h''}{\mathrm{d}p} = \left(\frac{\partial h}{\partial p}\right)_T + \left(\frac{\partial h}{\partial T}\right)_p \frac{\mathrm{d}T_\mathrm{s}}{\mathrm{d}p}, \qquad \text{(A7.8, A7.9)}$$

$$\frac{\mathrm{d}s'}{\mathrm{d}p} = \left(\frac{\partial s}{\partial p}\right)_h + \left(\frac{\partial s}{\partial h}\right)_p \frac{\mathrm{d}h'}{\mathrm{d}p}, \quad \text{and} \quad \frac{\mathrm{d}s''}{\mathrm{d}p} = \left(\frac{\partial s}{\partial p}\right)_h + \left(\frac{\partial s}{\partial h}\right)_p \frac{\mathrm{d}h''}{\mathrm{d}p}. \qquad \text{(A7.10, A7.11)}$$

In Eqs. (A7.8) – (A7.11), the derivatives

$$\left(\frac{\partial h}{\partial p}\right)_T, \left(\frac{\partial h}{\partial T}\right)_p, \left(\frac{\partial s}{\partial p}\right)_h, \text{ and } \left(\frac{\partial s}{\partial h}\right)_p$$

are determined in the corresponding phase from $T^\mathrm{L}(p,h)$, $T^\mathrm{G}(p,h)$, $s^\mathrm{L}(p,h)$, and $s^\mathrm{G}(p,h)$. The temperature gradient along the saturation curve is derived from $T_\mathrm{s}(p)$. The iteration procedure is repeated until $|f| \le TOL$ and $p_\mathrm{s} = p_k$, T_s, x, h', h'', s', and s'' are determined. A spline function for $p_\mathrm{s}(h,s)$ is used to initialize p_k.

A8 Property Calculations in the Two-Phase Region from (*v*,*u*)

For property calculations where small inconsistencies at the saturated liquid line are tolerable, the following additional phase boundary conditions are recommended. In addition to the (*v*,*u*) spline functions in the liquid region L and the gas region G, spline functions for the properties at saturation $v''(p)$, $v'(u)$, and $u'(T)$ are required. With the use of a one-dimensional Newton iteration scheme, the equation

$$f(p) = 0 = x_v(p) - x_u(p) \qquad \text{(A8.1)}$$

is solved for the pressure *p*, where

$$x_v(p) = \frac{v - v'(p)}{v''(p) - v'(p)} \quad \text{and} \quad x_u(p) = \frac{u - u'(p)}{u''(p) - u'(p)}. \qquad \text{(A8.2, A8.3)}$$

The iteration procedure is

$$p_{k+1} = p_k - \frac{f(p_k)}{\dfrac{\mathrm{d}f}{\mathrm{d}p}(p_k)}, \qquad \text{(A8.4)}$$

where

$$\frac{\mathrm{d}f}{\mathrm{d}p}(p_k) = \frac{\mathrm{d}x_v}{\mathrm{d}p}(p_k) - \frac{\mathrm{d}x_u}{\mathrm{d}p}(p_k). \qquad \text{(A8.5)}$$

In each iteration step k, $v''(p_k)$ is calculated with the corresponding spline function. From the inverse spline function $u^{\mathrm{G}}(p,v)$, $u''(p_k)$ is determined as $u''(p_k) = u^{\mathrm{G}}(p_k, v''(p_k))$. Then, $T_{\mathrm{s}}(p_k) = T^{\mathrm{G}}(v''(p_k), u''(p_k))$, $u'(p_k) = u'(T_{\mathrm{s}}(p_k))$, and $v'(p_k) = v'(u'(p_k))$ are subsequently calculated. The derivatives in Eq. (A8.5) are calculated from

$$\frac{\mathrm{d}x_v}{\mathrm{d}p} = \frac{-\frac{\mathrm{d}v'}{\mathrm{d}p} - x_v\left(\frac{\mathrm{d}v''}{\mathrm{d}p} - \frac{\mathrm{d}v'}{\mathrm{d}p}\right)}{\left(v'' - v'\right)} \quad \text{and} \quad \frac{\mathrm{d}x_u}{\mathrm{d}p} = \frac{-\frac{\mathrm{d}u'}{\mathrm{d}p} - x_u\left(\frac{\mathrm{d}u''}{\mathrm{d}p} - \frac{\mathrm{d}u'}{\mathrm{d}p}\right)}{\left(u'' - u'\right)}, \quad \text{(A8.6, A8.7)}$$

where

$$\frac{\mathrm{d}v'}{\mathrm{d}p} = \frac{\mathrm{d}v'}{\mathrm{d}u'}\frac{\mathrm{d}u'}{\mathrm{d}T_{\mathrm{s}}}\frac{\mathrm{d}T_{\mathrm{s}}}{\mathrm{d}p} \quad \text{and} \quad \frac{\mathrm{d}u'}{\mathrm{d}p} = \frac{\mathrm{d}u'}{\mathrm{d}T_{\mathrm{s}}}\frac{\mathrm{d}T_{\mathrm{s}}}{\mathrm{d}p}. \quad \text{(A8.8, A8.9)}$$

The derivatives

$$\frac{\mathrm{d}v'}{\mathrm{d}u'} \text{ and } \frac{\mathrm{d}u'}{\mathrm{d}T_{\mathrm{s}}}$$

in Eqs. (A8.8) and (A8.9) are derived from the spline functions $v'(u)$ and $u'(T)$.

The saturation temperature gradient is calculated from

$$\frac{\mathrm{d}T_{\mathrm{s}}}{\mathrm{d}p} = \left(\frac{\partial T}{\partial v}\right)_u \frac{\mathrm{d}v''}{\mathrm{d}p} + \left(\frac{\partial T}{\partial u}\right)_v \frac{\mathrm{d}u''}{\mathrm{d}p}, \quad \text{(A8.10)}$$

where

$$\left(\frac{\partial T}{\partial v}\right)_u \text{ and } \left(\frac{\partial T}{\partial u}\right)_v$$

are determined in the gas phase from $T^{\mathrm{G}}(v,u)$. The derivative

$$\frac{\mathrm{d}v''}{\mathrm{d}p}$$

is derived from $v''(p)$, and

$$\frac{\mathrm{d}u''}{\mathrm{d}p} = \left(\frac{\partial u}{\partial p}\right)_v + \left(\frac{\partial u}{\partial v}\right)_p \frac{\mathrm{d}v''}{\mathrm{d}p} \quad \text{(A8.11)}$$

is calculated with

$$\left(\frac{\partial u}{\partial p}\right)_v \text{ and } \left(\frac{\partial u}{\partial v}\right)_p,$$

which are derived from $p^{\mathrm{G}}(v,u)$ in the gas phase. The iteration procedure is repeated until $|f| \le TOL$ and $p_{\mathrm{s}} = p_k$, T_{s}, x, v', v'', u', and u'' are determined. A spline function for $p_{\mathrm{s}}(v,u)$ is used to initialize p_k.

A9 Property Calculations in the Two-Phase Region from (*p*,*v*)

In order to calculate the properties in the two-phase region from (*p*,*v*) consistently with the calculations from (*v*,*u*) described in Appendix A8, the following algorithm is recommended. From the spline and inverse spline functions $v''(p)$, $u^{\mathrm{G}}(p,v)$, $T^{\mathrm{G}}(v,u)$, $u'(T)$, and $v'(u)$, the saturation properties are subsequently calculated from

$$v''=v''(p),\ u''=u^{\mathrm{G}}(p,v''),\ T_{\mathrm{s}}=T^{\mathrm{G}}(v'',u''),\ u'=u'(T_{\mathrm{s}}),\ \text{and}\ v'=v'(u')$$

without iteration. The vapor fraction x is calculated from $x=(v-v')/(v''-v')$, and the desired mass-specific properties are calculated from $z=z'+x\left(z''-z'\right)$.

A10 Property Calculations in the Two-Phase Region from (*u*,*s*)

In order to calculate the properties in the two-phase region from (*u*,*s*) consistently with the calculations from (*v*,*u*) described in Appendix A8, the following algorithm is recommended. With the use of a one-dimensional Newton iteration scheme, the equation

$$f\left(p\right)=0=x_u\left(p\right)-x_s\left(p\right) \tag{A10.1}$$

is solved for the pressure *p*, where

$$x_u\left(p\right)=\frac{u-u'\left(p\right)}{u''\left(p\right)-u'\left(p\right)} \qquad \text{and} \qquad x_s\left(p\right)=\frac{s-s'\left(p\right)}{s''\left(p\right)-s'\left(p\right)}. \tag{A10.2, A10.3}$$

The iteration procedure is

$$p_{k+1}=p_k-\frac{f\left(p_k\right)}{\dfrac{\mathrm{d}f}{\mathrm{d}p}\left(p_k\right)}, \tag{A10.4}$$

where

$$\frac{\mathrm{d}f}{\mathrm{d}p}\left(p_k\right)=\frac{\mathrm{d}x_u}{\mathrm{d}p}\left(p_k\right)-\frac{\mathrm{d}x_s}{\mathrm{d}p}\left(p_k\right). \tag{A10.5}$$

In each iteration step *k*, $v''(p_k)$ is calculated from the corresponding spline function. From the inverse spline function $u^{\mathrm{G}}(p,v)$, $u''(p_{\mathrm{s},k})$ is determined as $u''(p_{\mathrm{s},k})=u^{\mathrm{G}}(p_{\mathrm{s},k},v''(p_{\mathrm{s},k}))$. Then, $s''(p_{\mathrm{s},k})=s^{\mathrm{G}}(v''(p_{\mathrm{s},k}),u''(p_{\mathrm{s},k}))$, $T_{\mathrm{s}}(p_{\mathrm{s},k})=T^{\mathrm{G}}(v''(p_{\mathrm{s},k}),u''(p_{\mathrm{s},k}))$, $u'(p_{\mathrm{s},k})=u'(T_{\mathrm{s}}(p_{\mathrm{s},k}))$, $v'(p_{\mathrm{s},k})=v'(u'(p_{\mathrm{s},k}))$, and $s'(p_{\mathrm{s},k})=s^{\mathrm{L}}(v'(p_{\mathrm{s},k}),u'(p_{\mathrm{s},k}))$ are subsequently calculated. The derivatives in Eq. (A10.5) are calculated from

$$\frac{\mathrm{d}x_u}{\mathrm{d}p}=\frac{-\dfrac{\mathrm{d}u'}{\mathrm{d}p}-x_u\left(\dfrac{\mathrm{d}u''}{\mathrm{d}p}-\dfrac{\mathrm{d}u'}{\mathrm{d}p}\right)}{\left(u''-u'\right)} \qquad \text{and} \qquad \frac{\mathrm{d}x_s}{\mathrm{d}p}=\frac{-\dfrac{\mathrm{d}s'}{\mathrm{d}p}-x_s\left(\dfrac{\mathrm{d}s''}{\mathrm{d}p}-\dfrac{\mathrm{d}s'}{\mathrm{d}p}\right)}{\left(s''-s'\right)} \tag{A10.6, A10.7}$$

where

$$\frac{\mathrm{d}u'}{\mathrm{d}p}=\frac{\mathrm{d}u'}{\mathrm{d}T_{\mathrm{s}}}\frac{\mathrm{d}T_{\mathrm{s}}}{\mathrm{d}p} \qquad \text{and} \qquad \frac{\mathrm{d}s'}{\mathrm{d}p}=\left(\frac{\partial s}{\partial u}\right)_v\frac{\mathrm{d}u'}{\mathrm{d}p}+\left(\frac{\partial s}{\partial v}\right)_u\frac{\mathrm{d}v'}{\mathrm{d}p}. \tag{A10.8, A10.9}$$

In Eqn. (A10.8), the derivative

$$\frac{\mathrm{d}u'}{\mathrm{d}T_\mathrm{s}}$$

is derived from the spline functions $u'(T)$. The derivative

$$\frac{\mathrm{d}v'}{\mathrm{d}p} = \frac{\mathrm{d}v'}{\mathrm{d}u'}\frac{\mathrm{d}u'}{\mathrm{d}T_\mathrm{s}}\frac{\mathrm{d}T_\mathrm{s}}{\mathrm{d}p} \tag{A10.10}$$

is calculated with

$$\frac{\mathrm{d}v'}{\mathrm{d}u'},$$

which is derived from $v'(u)$. The temperature gradient at the saturation curve is calculated from

$$\frac{\mathrm{d}T_\mathrm{s}}{\mathrm{d}p} = \left(\frac{\partial T}{\partial v}\right)_u \frac{\mathrm{d}v''}{\mathrm{d}p} + \left(\frac{\partial T}{\partial u}\right)_v \frac{\mathrm{d}u''}{\mathrm{d}p}, \tag{A10.11}$$

where

$$\left(\frac{\partial T}{\partial v}\right)_u \text{ and } \left(\frac{\partial T}{\partial u}\right)_v$$

are determined in the gas phase from $T^\mathrm{G}(v,u)$. The derivative

$$\frac{\mathrm{d}v''}{\mathrm{d}p}$$

is derived from $v''(p)$, and

$$\frac{\mathrm{d}u''}{\mathrm{d}p} = \left(\frac{\partial u}{\partial p}\right)_v + \left(\frac{\partial u}{\partial v}\right)_p \frac{\mathrm{d}v''}{\mathrm{d}p} \tag{A10.12}$$

is calculated with

$$\left(\frac{\partial u}{\partial p}\right)_v \text{ and } \left(\frac{\partial u}{\partial v}\right)_p,$$

which are derived from $p^\mathrm{G}(v,u)$ in the gas phase. The iteration procedure is repeated until $|f| \le TOL$ and $p_\mathrm{s} = p_k$, T_s, x, v', v'', u', u'', s', and s'' are determined. A spline function for $p_\mathrm{s}(u,s)$ is used to initialize p_k.

A11 Transformations and Grid Dimensions

For each spline function described in Secs. 5.1, 5.2, 5.3, 6.1, 6.2, and 7, the transformations and dimensions of the grid of nodes are given in the tables below. For piecewise equidistant nodes, the domain of the considered transformed variable $\overline{x}_{\min} \le \overline{x} \le \overline{x}_{\max}$ is subdivided in several intervals with equidistant nodes. In the tables below, this is described with

$$\begin{bmatrix} \overline{x}_{\min} \\ \dots \\ \dots \\ \overline{x}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix},$$

where the boundaries of the intervals are given in the column on the left and the number of equidistant nodes between them is given in the column on the right.

Table A1: Transformations and dimensions of the grid of nodes of each (v,u) spline function for the liquid region L based on IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [13] (see Sec. 5.1)

Spline function	v [m³/kg]	u [kJ/kg]
	$\overline{v}(v,u) = \dfrac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)}\left(v - v_{\min}(u)\right) + \overline{v}_{\min}$ $\overline{v}_{\min} = 1 \qquad \overline{v}_{\max} = 100$	
	$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$p^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 95 \\ 100 \end{bmatrix} \begin{bmatrix} 100 \\ 200 \end{bmatrix}$	$\begin{bmatrix} -8.489\,68 \\ 250 \\ 2040.01 \end{bmatrix} \begin{bmatrix} 300 \\ 225 \end{bmatrix}$
$T^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 100 \end{bmatrix} [100]$	$\begin{bmatrix} -8.489\,68 \\ 2040.01 \end{bmatrix} [200]$
$s^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 100 \end{bmatrix} [100]$	$\begin{bmatrix} -8.489\,68 \\ 10 \\ 2040.01 \end{bmatrix} \begin{bmatrix} 10 \\ 200 \end{bmatrix}$
$w^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 90 \\ 100 \end{bmatrix} \begin{bmatrix} 100 \\ 50 \end{bmatrix}$	$\begin{bmatrix} -8.489\,68 \\ 10 \\ 2040.01 \end{bmatrix} \begin{bmatrix} 10 \\ 200 \end{bmatrix}$
$\eta^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 100 \end{bmatrix} [100]$	$\begin{bmatrix} -8.489\,68 \\ 300 \\ 2040.01 \end{bmatrix} \begin{bmatrix} 75 \\ 150 \end{bmatrix}$

Table A2: Transformations and dimensions of the grid of nodes of each (v,u) spline function for the gas region G based on IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [13] (see Sec. 5.1)

	v [m^3/kg]	u [kJ/kg]
	$\overline{v}(v) = \ln(v)$	
Spline function	$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$p^{\mathrm{G}}(v,u)$ $T^{\mathrm{G}}(v,u)$ $s^{\mathrm{G}}(v,u)$ $w^{\mathrm{G}}(v,u)$ $\eta^{\mathrm{G}}(v,u)$	$\begin{bmatrix} \overline{v}\left(1.698\,44\times10^{-3}\right) \\ \overline{v}\left(8\times10^{-3}\right) \\ \overline{v}\left(1004.42\right) \end{bmatrix} \begin{bmatrix} 150 \\ 200 \end{bmatrix}$	$\begin{bmatrix} 2009.99 \\ 2650 \\ 3693.67 \end{bmatrix} \begin{bmatrix} 100 \\ 75 \end{bmatrix}$

Table A3: Transformations and dimensions of the grid of nodes of each (v,u) spline function for the high-temperature region HT based on IAPWS-IF97 (see Sec. 5.1)

	v [m^3/kg]	u [kJ/kg]
	$\overline{v}(v) = \ln(v)$	
Spline function	$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$p^{\mathrm{HT}}(v,u)$ $T^{\mathrm{HT}}(v,u)$ $w^{\mathrm{HT}}(v,u)$	$\begin{bmatrix} \overline{v}\left(7.456\,81\times10^{-3}\right) \\ \overline{v}\left(2212.94\right) \end{bmatrix} [200]$	$\begin{bmatrix} 3432.75 \\ 6518.9 \end{bmatrix} [75]$
$s^{\mathrm{HT}}(v,u)$	$\begin{bmatrix} \overline{v}\left(7.344\,62\times10^{-3}\right) \\ \overline{v}\left(2112.08\right) \end{bmatrix} [200]$	$\begin{bmatrix} 3408.16 \\ 6364.93 \end{bmatrix} [100]$

Table A4: Transformations and dimensions of the grid of nodes of each (p,h) spline function for the liquid region L based on IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [13] (see Sec. 5.2)

Spline function	p [MPa]		h [kJ/kg]
	$\overline{p}(p)$	$\begin{bmatrix}\overline{p}_{\min}\\ \dots\\ \dots\\ \overline{p}_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$	$\begin{bmatrix}h_{\min}\\ \dots\\ \dots\\ h_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$
$T^{\mathrm{L}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 1\times10^{-2}\\ 20\\ 105\end{bmatrix}\begin{bmatrix}100\\ 75\\ 100\end{bmatrix}$	$\begin{bmatrix}-12.7192\\ 2140\end{bmatrix}\begin{bmatrix}125\end{bmatrix}$
$v^{\mathrm{L}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 1\times10^{-2}\\ 20\\ 105\end{bmatrix}\begin{bmatrix}100\\ 75\\ 125\end{bmatrix}$	$\begin{bmatrix}-12.7192\\ 2140\end{bmatrix}\begin{bmatrix}125\end{bmatrix}$
$s^{\mathrm{L}}(p,h)$	$\sqrt{p}$	$\begin{bmatrix}\overline{p}(5\times10^{-4})\\ \overline{p}(1\times10^{-2})\\ \overline{p}(20)\\ \overline{p}(105)\end{bmatrix}\begin{bmatrix}150\\ 100\\ 100\end{bmatrix}$	$\begin{bmatrix}-12.7192\\ 200\\ 2140\end{bmatrix}\begin{bmatrix}100\\ 150\end{bmatrix}$
$w^{\mathrm{L}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 1\times10^{-2}\\ 20\\ 25\\ 105\end{bmatrix}\begin{bmatrix}100\\ 75\\ 25\\ 75\end{bmatrix}$	$\begin{bmatrix}-12.7192\\ 200\\ 1700\\ 2140\end{bmatrix}\begin{bmatrix}25\\ 125\\ 50\end{bmatrix}$
$\eta^{\mathrm{L}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 1\times10^{-2}\\ 20\\ 105\end{bmatrix}\begin{bmatrix}100\\ 75\\ 100\end{bmatrix}$	$\begin{bmatrix}-12.7192\\ 300\\ 2140\end{bmatrix}\begin{bmatrix}75\\ 125\end{bmatrix}$

Table A5: Transformations and dimensions of the grid of nodes of each (p,h) spline function for the gas region G based on IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [13] (see Sec. 5.2)

Spline function	p [MPa]		h [kJ/kg]
	$\overline{p}(p)$	$\begin{bmatrix} \overline{p}_{\min} \\ \dots \\ \dots \\ \overline{p}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} h_{\min} \\ \dots \\ \dots \\ h_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$T^{\mathrm{G}}(p,h)$	p	$\begin{bmatrix} 5\times10^{-4} \\ 3\times10^{-2} \\ 0.5 \\ 20 \\ 105 \end{bmatrix} \begin{bmatrix} 150 \\ 75 \\ 100 \\ 75 \end{bmatrix}$	$\begin{bmatrix} 2040 \\ 2850 \\ 4195.88 \end{bmatrix} \begin{bmatrix} 75 \\ 75 \end{bmatrix}$
$v^{\mathrm{G}}(p,h) = \dfrac{\overline{v}^{\mathrm{G}}(p,h)}{p}$	p	$\begin{bmatrix} 5\times10^{-4} \\ 3\times10^{-2} \\ 0.5 \\ 20 \\ 105 \end{bmatrix} \begin{bmatrix} 125 \\ 50 \\ 50 \\ 75 \end{bmatrix}$	$\begin{bmatrix} 2040 \\ 2850 \\ 4195.88 \end{bmatrix} \begin{bmatrix} 75 \\ 50 \end{bmatrix}$
$s^{\mathrm{G}}(p,h)$	$\ln(p)$	$\begin{bmatrix} \overline{p}(5\times10^{-4}) \\ \overline{p}(20) \\ \overline{p}(110) \end{bmatrix} \begin{bmatrix} 150 \\ 75 \end{bmatrix}$	$\begin{bmatrix} 2040 \\ 2850 \\ 4219.44 \end{bmatrix} \begin{bmatrix} 75 \\ 75 \end{bmatrix}$
$w^{\mathrm{G}}(p,h)$	$\sqrt{p}$	$\begin{bmatrix} \overline{p}(5\times10^{-4}) \\ \overline{p}(5\times10^{-2}) \\ \overline{p}(20) \\ \overline{p}(105) \end{bmatrix} \begin{bmatrix} 100 \\ 75 \\ 100 \end{bmatrix}$	$\begin{bmatrix} 2040 \\ 2850 \\ 4195.88 \end{bmatrix} \begin{bmatrix} 150 \\ 50 \end{bmatrix}$
$\eta^{\mathrm{G}}(p,h)$	p	$\begin{bmatrix} 5\times10^{-4} \\ 2\times10^{-2} \\ 0.7 \\ 20 \\ 105 \end{bmatrix} \begin{bmatrix} 100 \\ 75 \\ 75 \\ 75 \end{bmatrix}$	$\begin{bmatrix} 2040 \\ 2850 \\ 4195.88 \end{bmatrix} \begin{bmatrix} 100 \\ 50 \end{bmatrix}$

Table A6: Transformations and dimensions of the grid of nodes of each (p,h) spline function for the high-temperature region HT based on IAPWS-IF97 (see Sec. 5.2)

Spline function	p [MPa]		h [kJ/kg]
	$\overline{p}(p)$	$\begin{bmatrix}\overline{p}_{\min}\\ \dots\\ \dots\\ \overline{p}_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$	$\begin{bmatrix}h_{\min}\\ \dots\\ \dots\\ h_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$
$T^{\mathrm{HT}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 60\end{bmatrix}[100]$	$\begin{bmatrix}3833.08\\ 7420.98\end{bmatrix}[100]$
$v^{\mathrm{HT}}(p,h)=\dfrac{\overline{v}^{\mathrm{HT}}(p,h)}{p}$	p	$\begin{bmatrix}5\times10^{-4}\\ 60\end{bmatrix}[100]$	$\begin{bmatrix}3833.08\\ 7420.98\end{bmatrix}[75]$
$s^{\mathrm{HT}}(p,h)$	$\ln(p)$	$\begin{bmatrix}\ln(5\times10^{-4})\\ \ln(60)\end{bmatrix}[125]$	$\begin{bmatrix}3817.25\\ 7450.34\end{bmatrix}[125]$
$w^{\mathrm{HT}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 60\end{bmatrix}[100]$	$\begin{bmatrix}3817.25\\ 7420.98\end{bmatrix}[75]$

Table A7: Transformations and dimensions of the grid of nodes of each (p,h) spline function for the metastable-vapor and gas region MG based on IAPWS-IF97 and the IAPWS viscosity release with recommendations for industrial use [13] (see Sec. 5.3)

Spline function	p [MPa]		h [kJ/kg]
	$\overline{p}(p)$	$\begin{bmatrix}\overline{p}_{\min}\\ \dots\\ \dots\\ \overline{p}_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$	$\begin{bmatrix}h_{\min}\\ \dots\\ \dots\\ h_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$
$T^{\mathrm{MG}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 9\times10^{-3}\\ 0.2\\ 2\\ 20\\ 105\end{bmatrix}\begin{bmatrix}150\\ 125\\ 75\\ 100\\ 75\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4195.88\end{bmatrix}\begin{bmatrix}100\\ 75\end{bmatrix}$
$v^{\mathrm{MG}}(p,h)=\dfrac{\overline{v}^{\mathrm{MG}}(p,h)}{p}$	p	$\begin{bmatrix}5\times10^{-4}\\ 9\times10^{-3}\\ 0.2\\ 2\\ 20\\ 105\end{bmatrix}\begin{bmatrix}125\\ 75\\ 50\\ 75\\ 75\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4195.88\end{bmatrix}\begin{bmatrix}100\\ 50\end{bmatrix}$
$s^{\mathrm{MG}}(p,h)$	$\ln(p)$	$\begin{bmatrix}\overline{p}(5\times10^{-4})\\ \overline{p}(8)\\ \overline{p}(20)\\ \overline{p}(105)\end{bmatrix}\begin{bmatrix}125\\ 75\\ 75\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4219.44\end{bmatrix}\begin{bmatrix}75\\ 75\end{bmatrix}$
$w^{\mathrm{MG}}(p,h)$	$\sqrt{p}$	$\begin{bmatrix}\overline{p}(5\times10^{-4})\\ \overline{p}(5\times10^{-2})\\ \overline{p}(20)\\ \overline{p}(105)\end{bmatrix}\begin{bmatrix}100\\ 100\\ 125\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4195.88\end{bmatrix}\begin{bmatrix}150\\ 50\end{bmatrix}$
$\eta^{\mathrm{MG}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 2\times10^{-2}\\ 0.7\\ 20\\ 105\end{bmatrix}\begin{bmatrix}150\\ 75\\ 100\\ 75\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4195.88\end{bmatrix}\begin{bmatrix}100\\ 50\end{bmatrix}$

Table A8: Transformations and dimensions of the grid of nodes of each (v,u) spline function for the liquid region L based on IAPWS-95 (see Sec. 6.1)

Spline function	v [m³/kg]	u [kJ/kg]
	$\overline{v}(v,u) = \dfrac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)} \left(v - v_{\min}(u)\right) + \overline{v}_{\min}$ $\overline{v}_{\min} = 1 \qquad \overline{v}_{\max} = 100$	
	$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$p^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 98 \\ 100 \end{bmatrix} \begin{bmatrix} 150 \\ 200 \end{bmatrix}$	$\begin{bmatrix} -20 \\ 250 \\ 2040 \end{bmatrix} \begin{bmatrix} 350 \\ 225 \end{bmatrix}$
$T^{\mathrm{L}}(v,u)$ $s^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 98 \\ 100 \end{bmatrix} \begin{bmatrix} 150 \\ 50 \end{bmatrix}$	$\begin{bmatrix} -20 \\ -10 \\ 2040 \end{bmatrix} \begin{bmatrix} 20 \\ 250 \end{bmatrix}$
$w^{\mathrm{L}}(v,u)$	$\begin{bmatrix} 1 \\ 98 \\ 100 \end{bmatrix} \begin{bmatrix} 150 \\ 50 \end{bmatrix}$	$\begin{bmatrix} -20 \\ -10 \\ 1750 \\ 2040 \end{bmatrix} \begin{bmatrix} 20 \\ 150 \\ 100 \end{bmatrix}$

Table A9: Transformations and dimensions of the grid of nodes of each (v,u) spline function for the gas region G based on IAPWS-95 (see Sec. 6.1)

	v [m^3/kg]	u [kJ/kg]
	$\overline{v}(v) = \ln(v)$	
Spline function	$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$p^{\mathrm{G}}(v,u)$ $T^{\mathrm{G}}(v,u)$ $s^{\mathrm{G}}(v,u)$ $w^{\mathrm{G}}(v,u)$	$\begin{bmatrix} \overline{v}\left(1.027\,96\times10^{-3}\right) \\ \overline{v}\left(8\times10^{-3}\right) \\ \overline{v}\left(1188.87\right) \end{bmatrix} \begin{bmatrix} 200 \\ 200 \end{bmatrix}$	$\begin{bmatrix} 2005 \\ 2650 \\ 4085.27 \end{bmatrix} \begin{bmatrix} 100 \\ 100 \end{bmatrix}$

Table A10: Transformations and dimensions of the grid of nodes of each (p,h) spline function for the liquid region L and the gas region G based on IAPWS-95 (see Sec. 6.2)

Spline function	p [MPa]		h [kJ/kg]
	$\overline{p}(p)$	$\begin{bmatrix}\overline{p}_{\min}\\ \dots\\ \dots\\ \overline{p}_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$	$\begin{bmatrix}h_{\min}\\ \dots\\ \dots\\ h_{\max}\end{bmatrix}\begin{bmatrix}\text{nodes}\\ \dots\\ \text{nodes}\end{bmatrix}$
$T^{\mathrm{L}}(p,h)$	p	$\begin{bmatrix}4.84693\times10^{-4}\\ 1\times10^{-2}\\ 20\\ 100\\ 1100\end{bmatrix}\begin{bmatrix}100\\ 75\\ 100\\ 125\end{bmatrix}$	$\begin{bmatrix}-13.3533\\ 2140\end{bmatrix}[150]$
$T^{\mathrm{G}}(p,h)$	p	$\begin{bmatrix}5\times10^{-4}\\ 3\times10^{-2}\\ 0.5\\ 20\\ 120\\ 1100\end{bmatrix}\begin{bmatrix}150\\ 75\\ 100\\ 100\\ 150\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4679.71\end{bmatrix}\begin{bmatrix}75\\ 100\end{bmatrix}$
$v^{\mathrm{L}}(p,h)$	p	$\begin{bmatrix}4.84693\times10^{-4}\\ 1\times10^{-2}\\ 20\\ 100\\ 1100\end{bmatrix}\begin{bmatrix}100\\ 75\\ 100\\ 125\end{bmatrix}$	$\begin{bmatrix}-13.3533\\ 2140\end{bmatrix}[150]$
$v^{\mathrm{G}}(p,h)=\dfrac{\overline{v}^{\mathrm{G}}(p,h)}{p}$	p	$\begin{bmatrix}5\times10^{-4}\\ 3\times10^{-2}\\ 0.5\\ 20\\ 120\\ 1100\end{bmatrix}\begin{bmatrix}125\\ 50\\ 50\\ 85\\ 90\end{bmatrix}$	$\begin{bmatrix}2040\\ 2850\\ 4679.71\end{bmatrix}\begin{bmatrix}75\\ 75\end{bmatrix}$

Table A11: Transformations and dimensions of the grid of nodes of the bicubic spline function $s(v,u)$ for the liquid region L based on IAPWS-95 (see Sec. 7)

v [m³/kg]	u [kJ/kg]
$\overline{v}(v,u) = \dfrac{\overline{v}_{\max} - \overline{v}_{\min}}{v_{\max}(u) - v_{\min}(u)}\left(v - v_{\min}(u)\right) + \overline{v}_{\min}$ $\overline{v}_{\min} = 1 \qquad \overline{v}_{\max} = 100$	
$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$\begin{bmatrix} 1 \\ 20 \\ 40 \\ 97.5 \\ 100 \end{bmatrix} \begin{bmatrix} 50 \\ 150 \\ 75 \\ 75 \end{bmatrix}$	$\begin{bmatrix} -12.356 \\ 250 \\ 1650 \\ 2000 \\ 2020 \end{bmatrix} \begin{bmatrix} 200 \\ 200 \\ 200 \\ 50 \end{bmatrix}$

Table A12: Transformations and dimensions of the grid of nodes of the bicubic spline function $s(v,u)$ for the gas region G based on IAPWS-95 (see Sec. 7)

v [m³/kg]	u [kJ/kg]
$\overline{v}(v) = \ln(v)$	
$\begin{bmatrix} \overline{v}_{\min} \\ \dots \\ \dots \\ \overline{v}_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$	$\begin{bmatrix} u_{\min} \\ \dots \\ \dots \\ u_{\max} \end{bmatrix} \begin{bmatrix} \text{nodes} \\ \dots \\ \text{nodes} \end{bmatrix}$
$\begin{bmatrix} \overline{v}\left(1.049\,57 \times 10^{-3}\right) \\ \overline{v}\left(8 \times 10^{-3}\right) \\ \overline{v}\left(961.672\right) \end{bmatrix} \begin{bmatrix} 200 \\ 200 \end{bmatrix}$	$\begin{bmatrix} 2000 \\ 2650 \\ 4055.26 \end{bmatrix} \begin{bmatrix} 100 \\ 100 \end{bmatrix}$

A12 Deviations of (*v*,*u*) and (*p*,*h*) Spline Functions from IAPWS-IF97

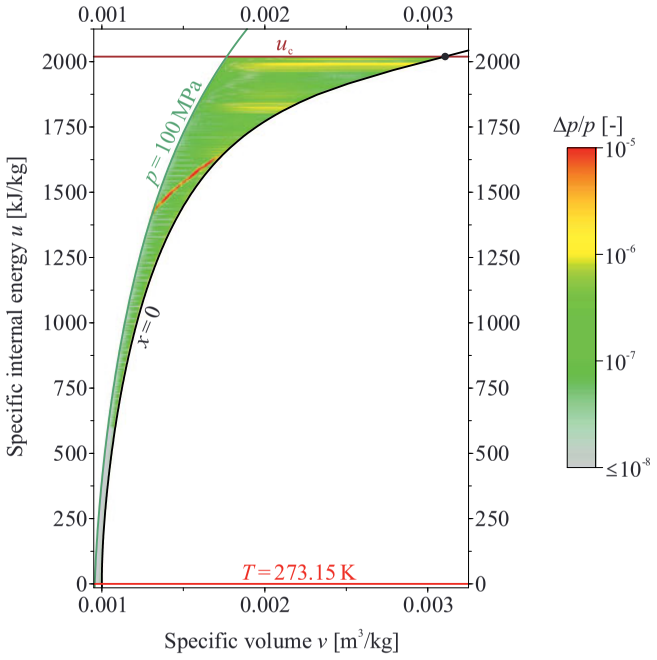

Figure A5: Deviations in pressure $p(v,u)$ from IAPWS-IF97 in the liquid region L.

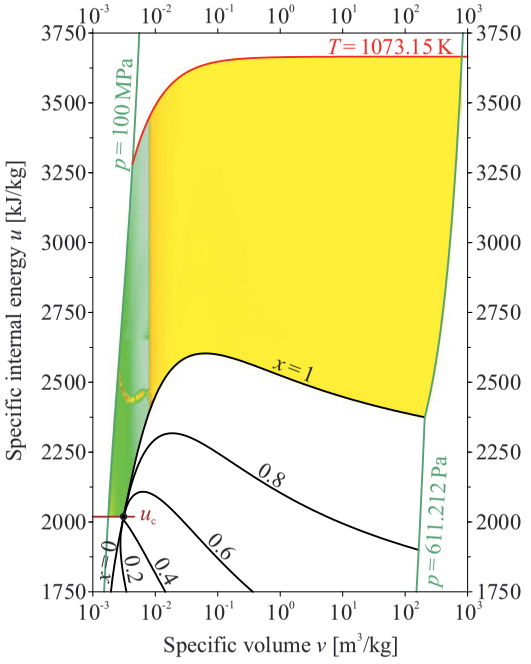

Figure A6: Deviations in pressure $p(v,u)$ from IAPWS-IF97 in the gas region G.

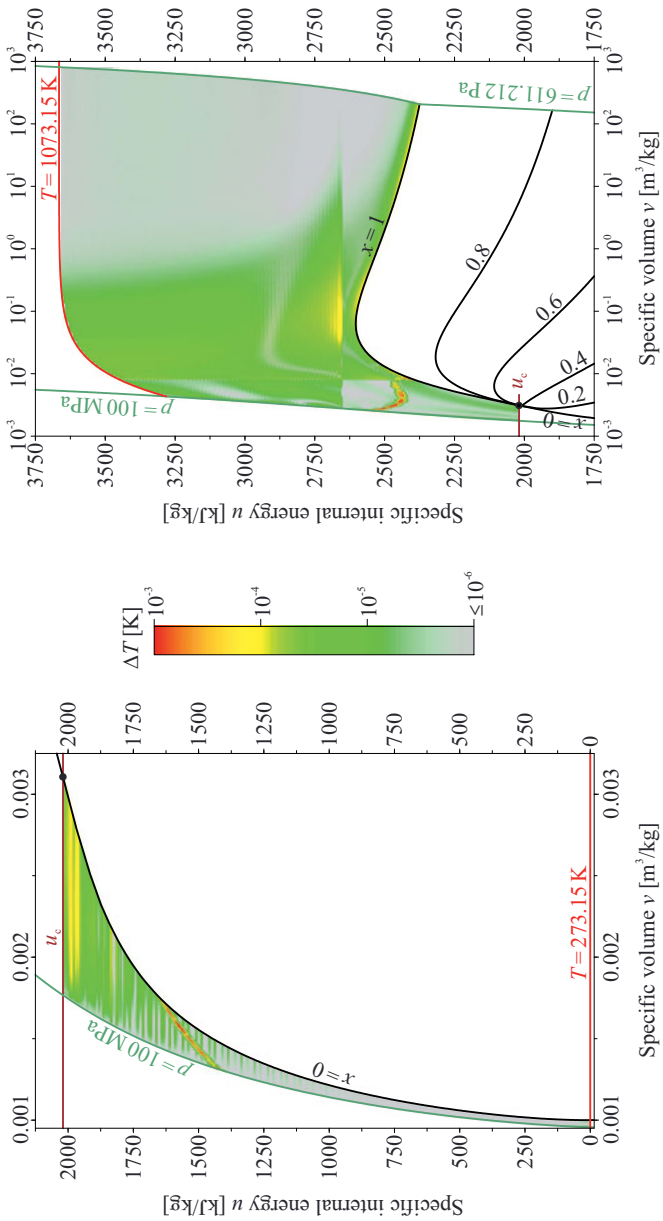

Figure A7: Deviations in temperature $T(v,u)$ from IAPWS-IF97 in the liquid region L.

Figure A8: Deviations in temperature $T(v,u)$ from IAPWS-IF97 in the gas region G.

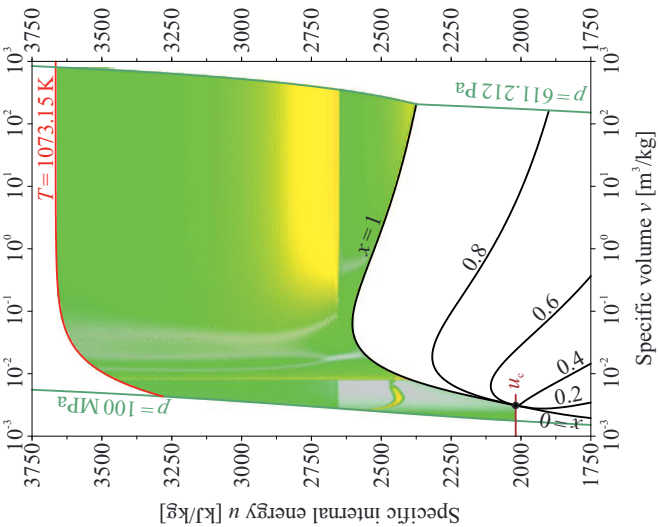

Figure A10: Deviations in specific entropy $s(v,u)$ from IAPWS-IF97 in the gas region G.

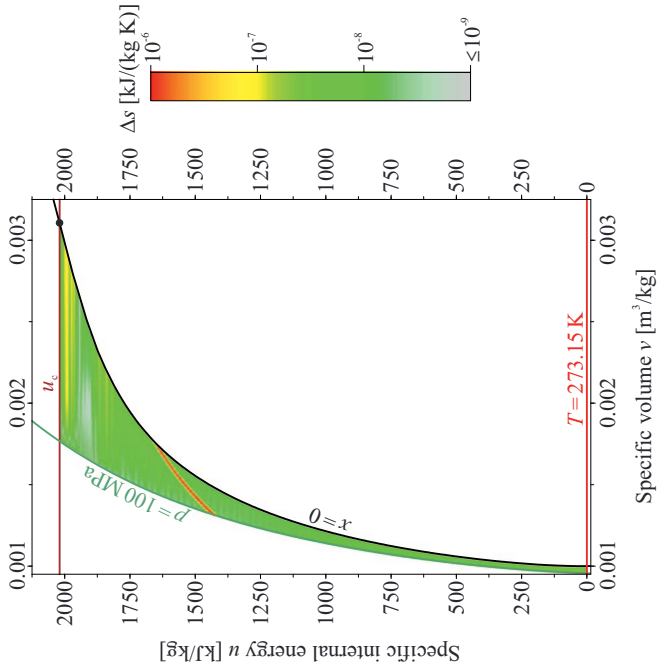

Figure A9: Deviations in specific entropy $s(v,u)$ from IAPWS-IF97 in the liquid region L.

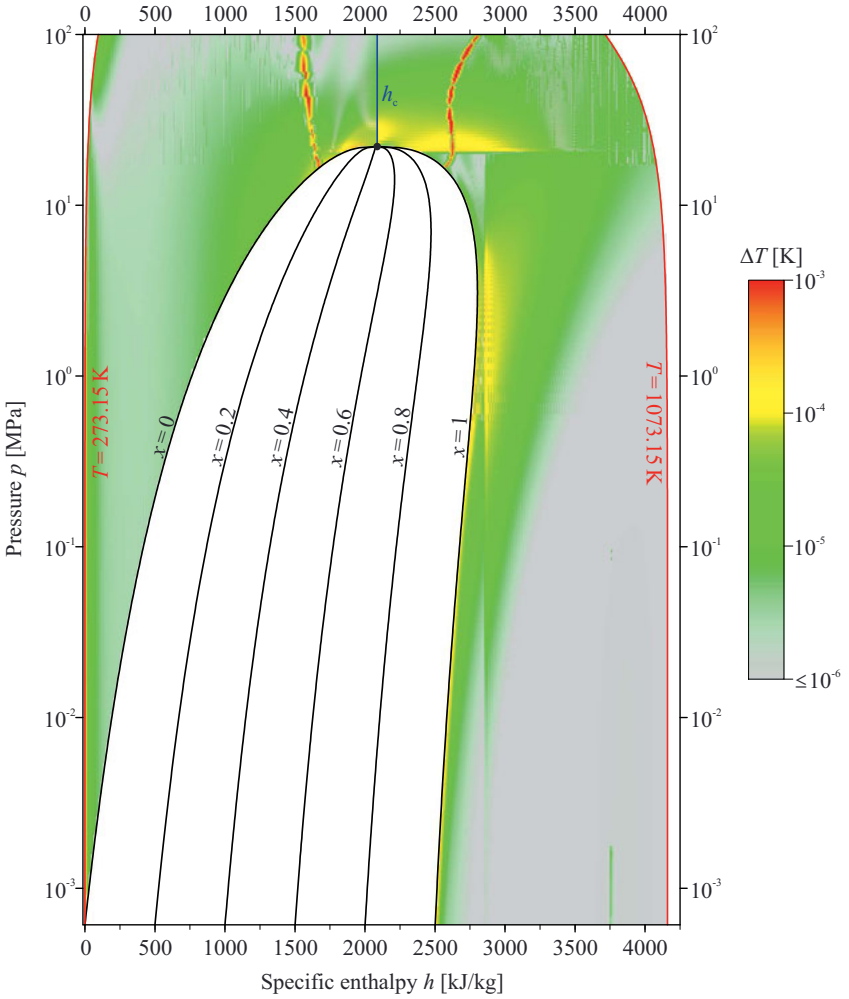

Figure A11: Deviations in temperature $T(p,h)$ from IAPWS-IF97 in the liquid region L and the gas region G.

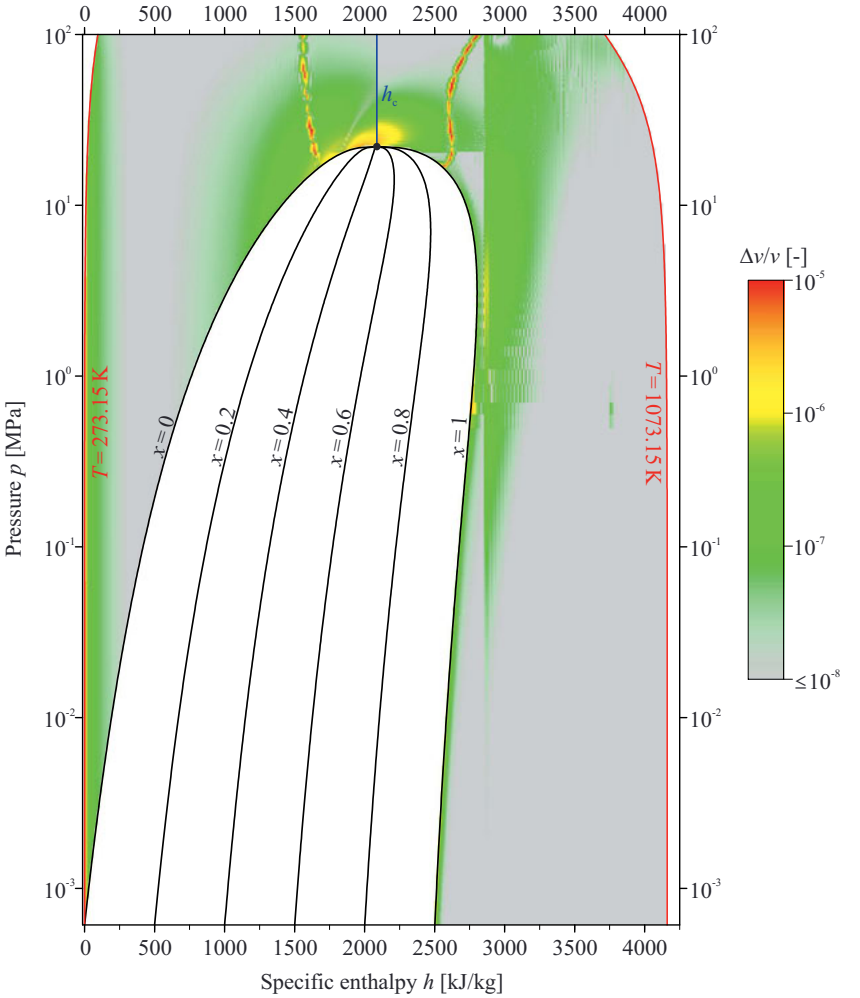

Figure A12: Deviations in specific volume $v(p,h)$ from IAPWS-IF97 in the liquid region L and the gas region G.

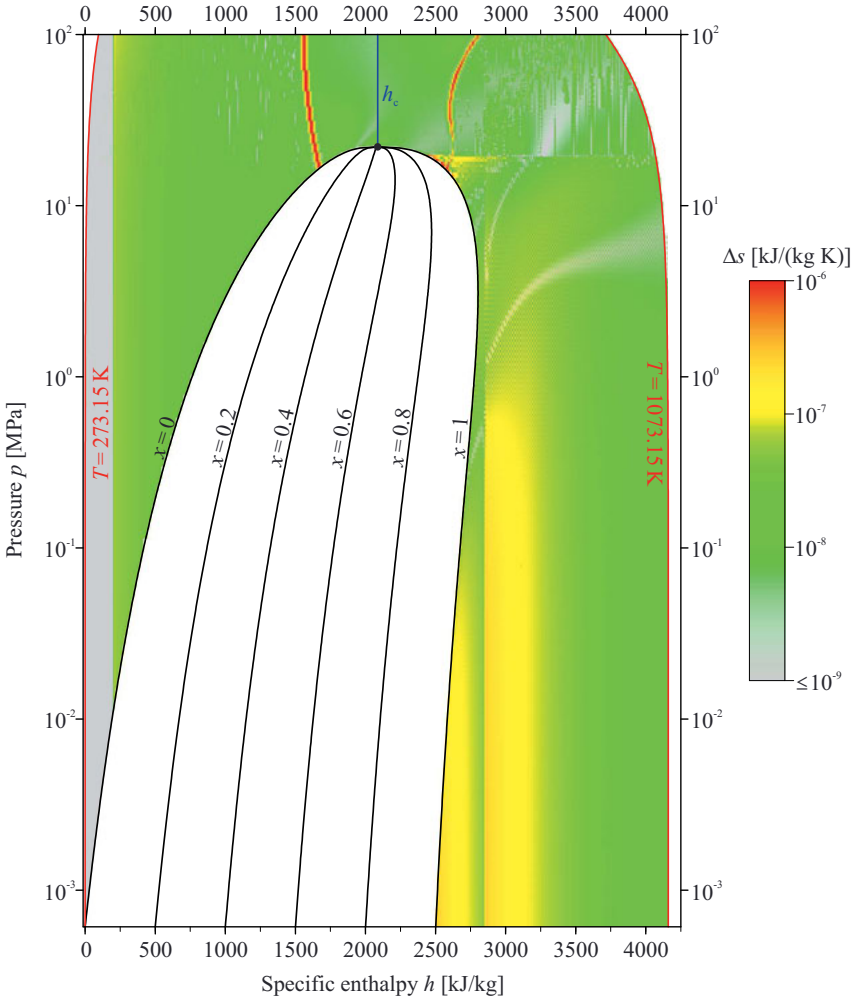

Figure A13: Deviations in specific entropy $s(p,h)$ from IAPWS-IF97 in the liquid region L and the gas region G.

A13 Deviations of (*v*,*u*) and (*p*,*h*) Spline Functions from IAPWS-95

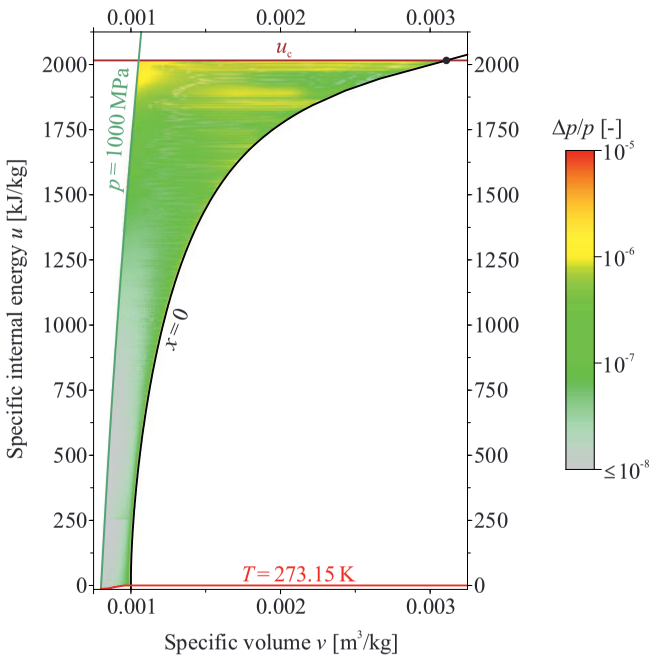

Figure A14: Deviations in pressure $p(v,u)$ from IAPWS-95 in the liquid region L.

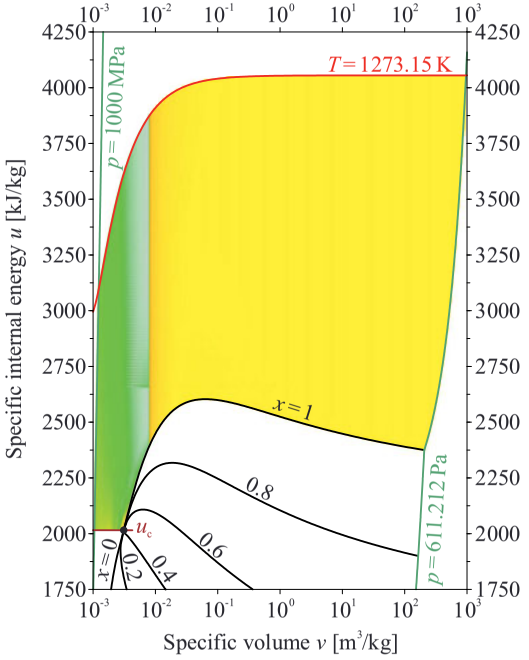

Figure A15: Deviations in pressure $p(v,u)$ from IAPWS-95 in the gas region G.

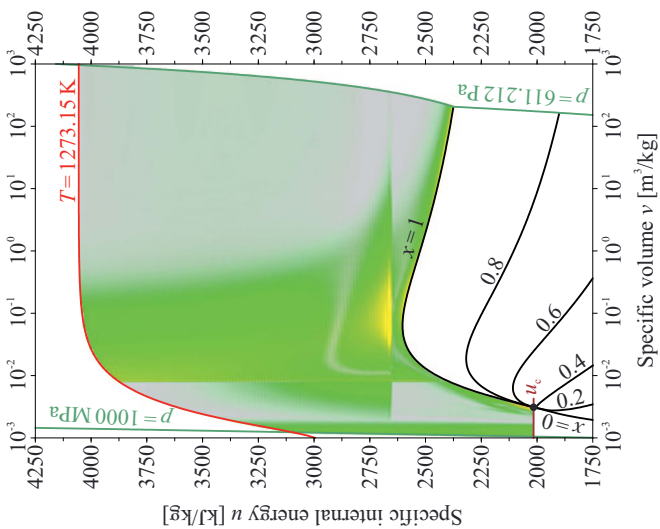

Figure A17: Deviations in temperature $T(v,u)$ from IAPWS-95 in the gas region G.

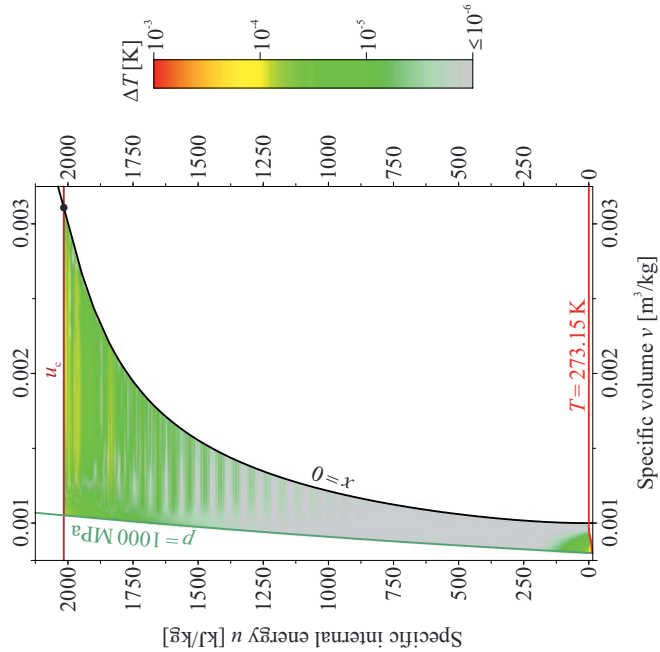

Figure A16: Deviations in temperature $T(v,u)$ from IAPWS-95 in the liquid region L.

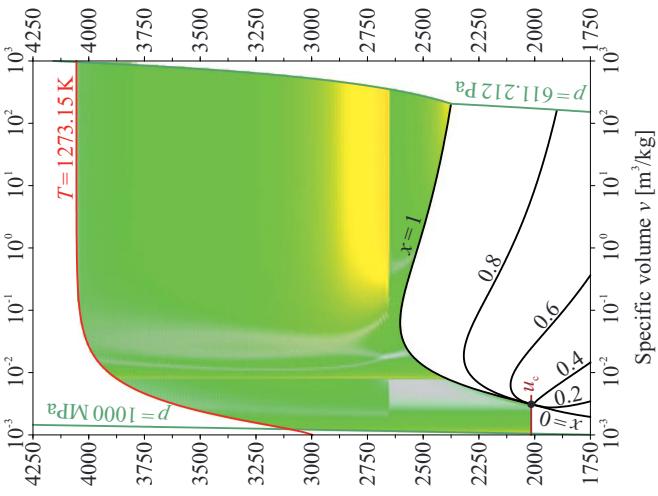

Figure A19: Deviations in specific entropy $s(v,u)$ from IAPWS-95 in the gas region G.

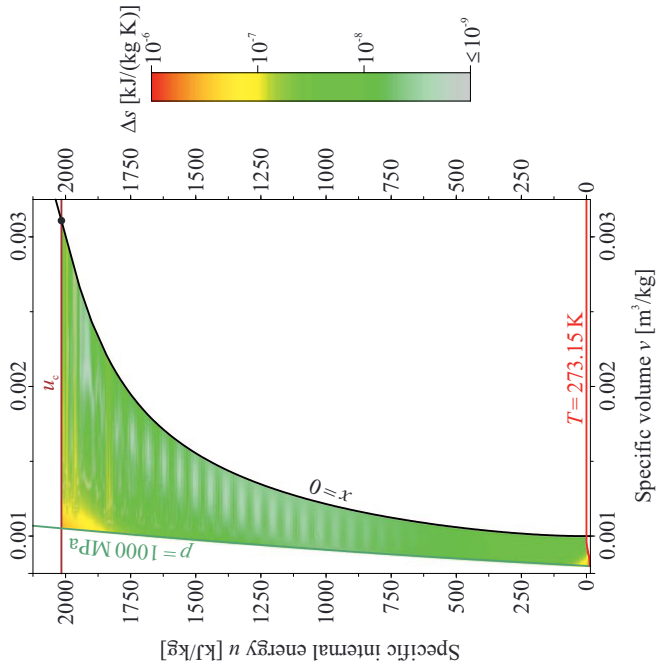

Figure A18: Deviations in specific entropy $s(v,u)$ from IAPWS-95 in the liquid region L.

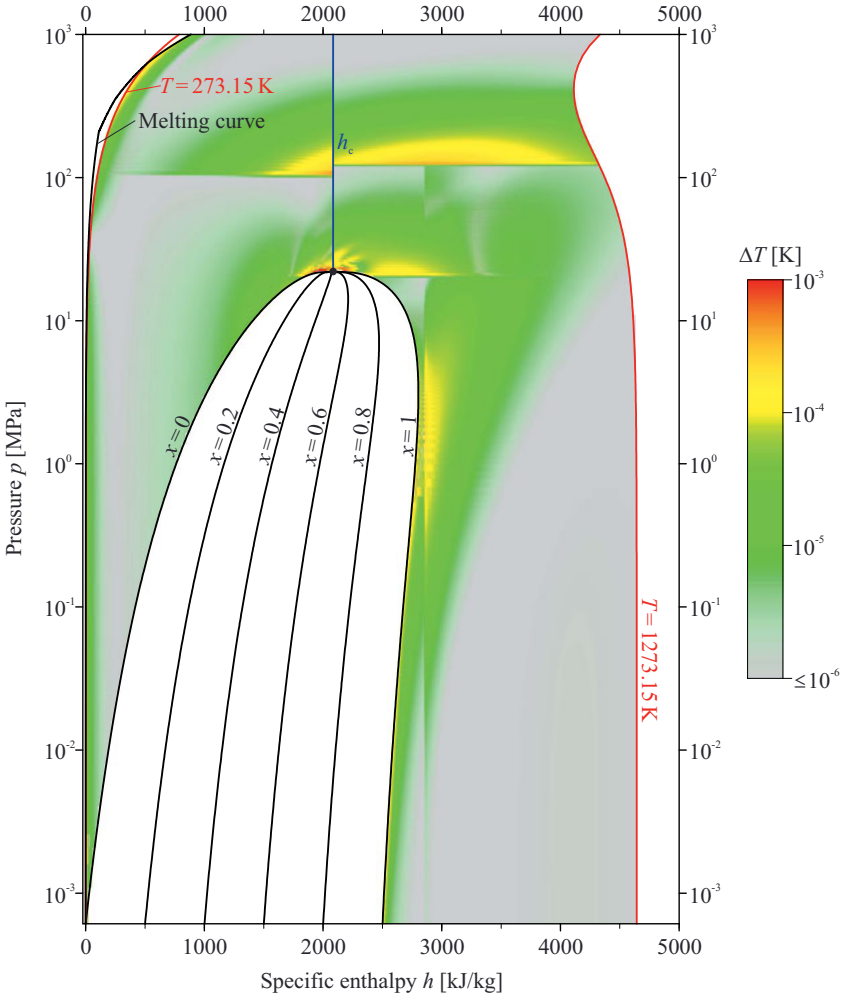

Figure A20: Deviations in temperature $T(p,h)$ from IAPWS-95 in the liquid region L and the gas region G.

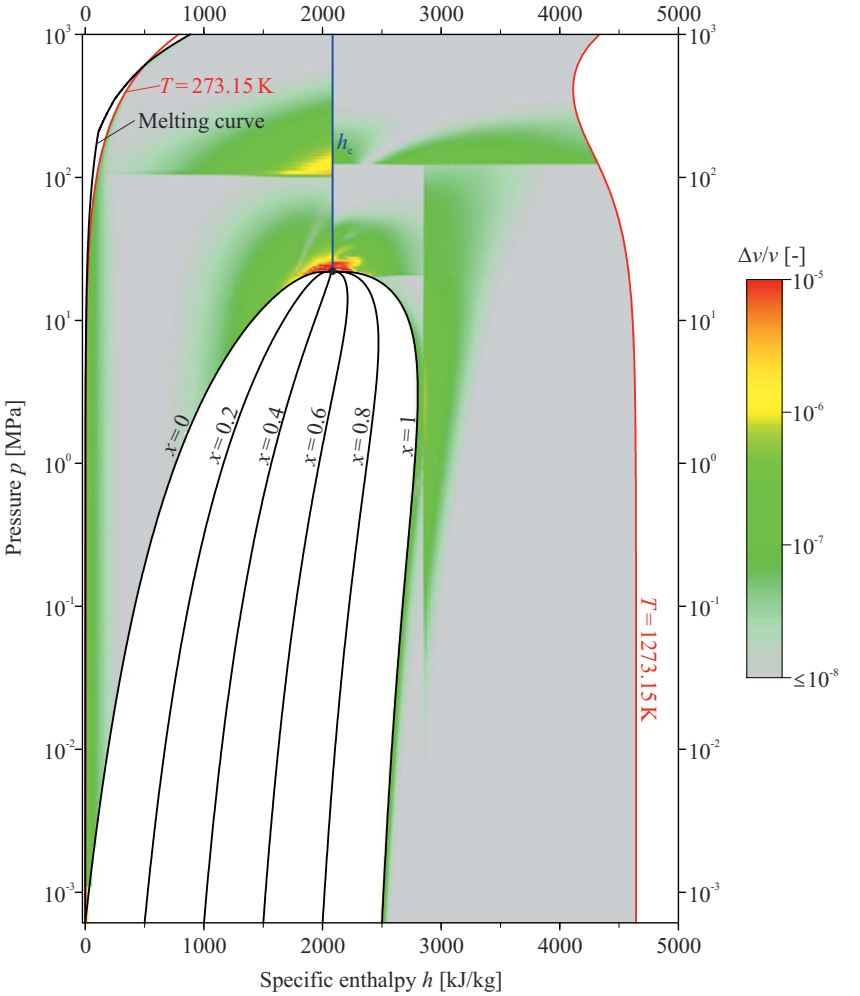

Figure A21: Deviations in specific volume $v(p,h)$ from IAPWS-95 in the liquid region L and the gas region G.

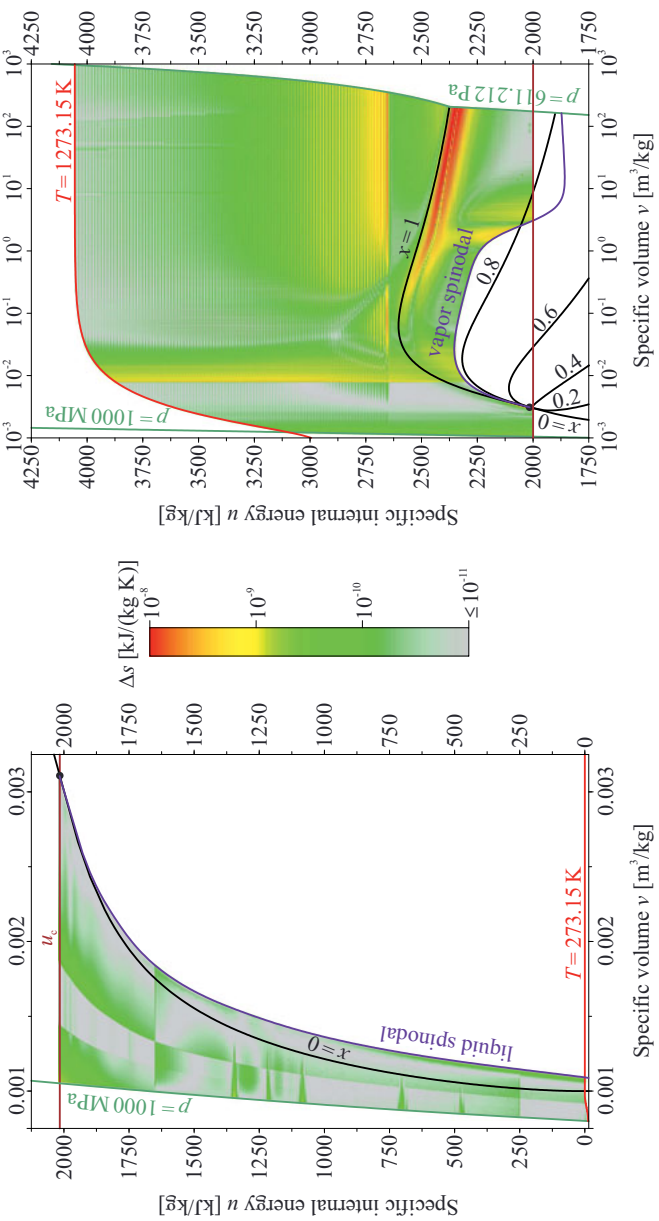

Figure A22: Deviations in specific entropy $s(v,u)$ from IAPWS-95 in the liquid region L (bicubic spline function).

Figure A23: Deviations in specific entropy $s(v,u)$ from IAPWS-95 in the gas region G (bicubic spline function).

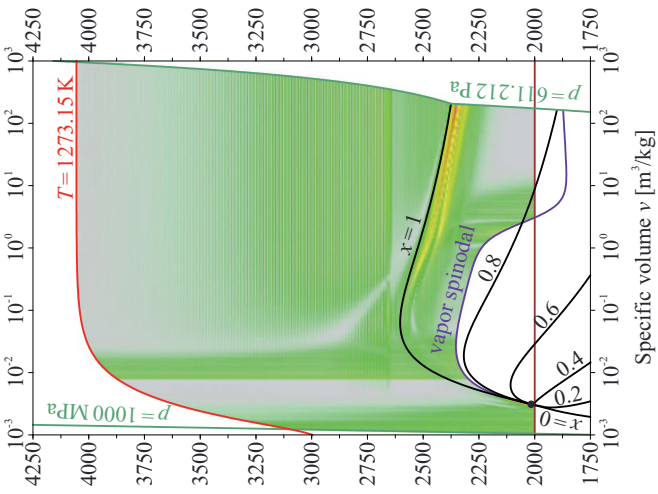

Figure A25: Deviations in pressure $p(v,u)$ from IAPWS-95 in the gas region G (derived from $s(v,u)$).

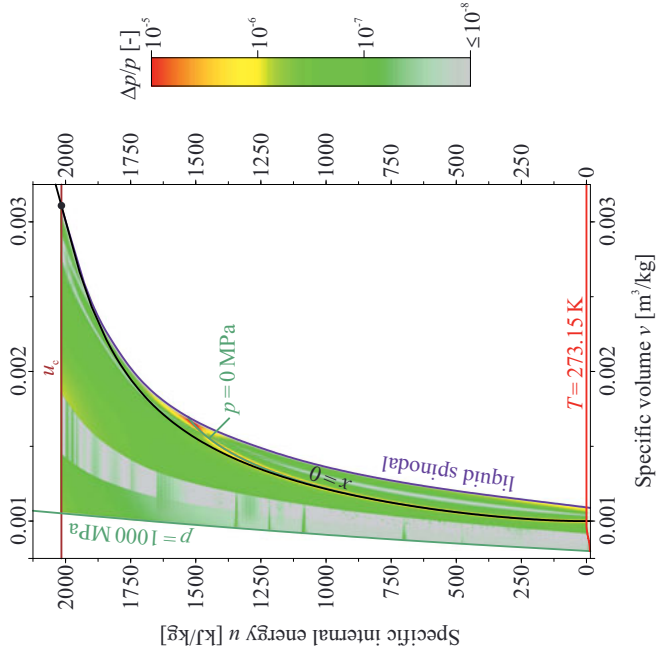

Figure A24: Deviations in pressure $p(v,u)$ from IAPWS-95 in the liquid region L (derived from $s(v,u)$).

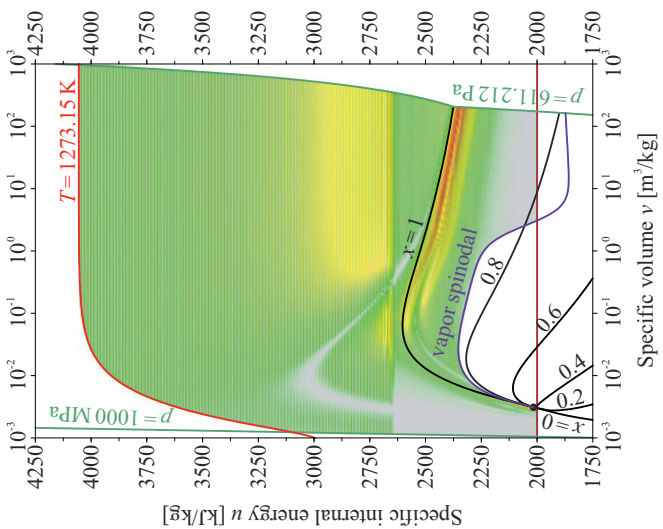

Figure A27: Deviations in temperature $T(v,u)$ from IAPWS-95 in the gas region G (derived from $s(v,u)$).

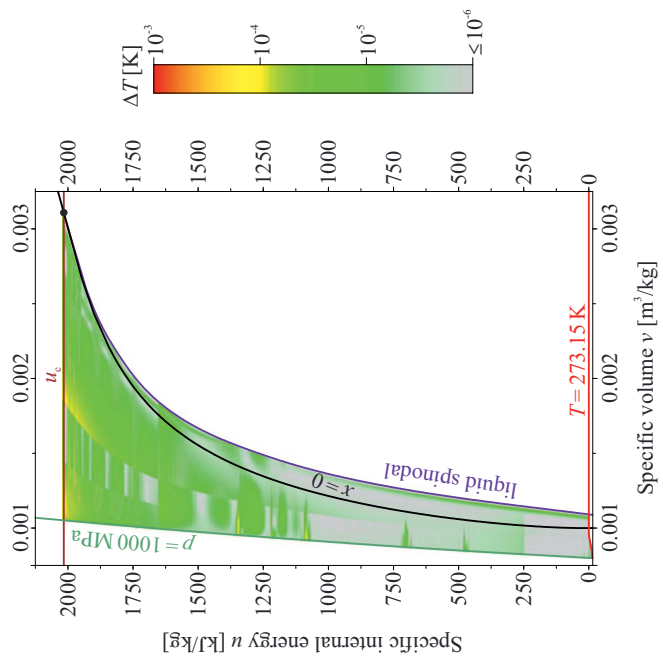

Figure A26: Deviations in temperature $T(v,u)$ from IAPWS-95 in the liquid region L (derived from $s(v,u)$).

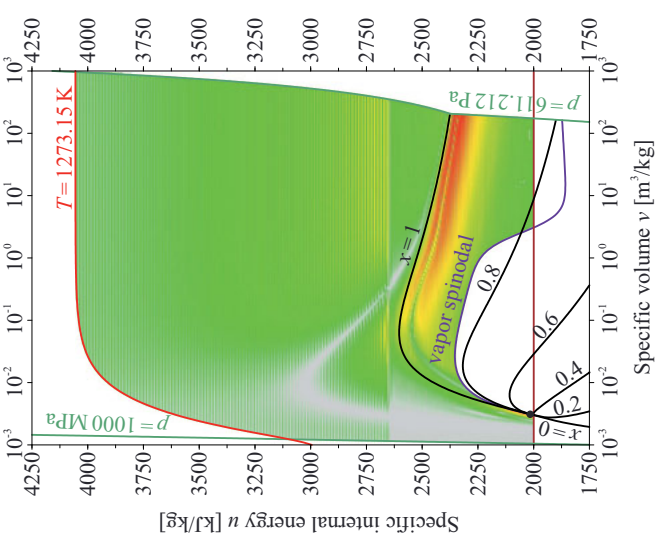

Figure A29: Deviations in spec. isochoric heat capacity $c_v(v,u)$ from IAPWS-95 in the gas region G (derived from $s(v,u)$).

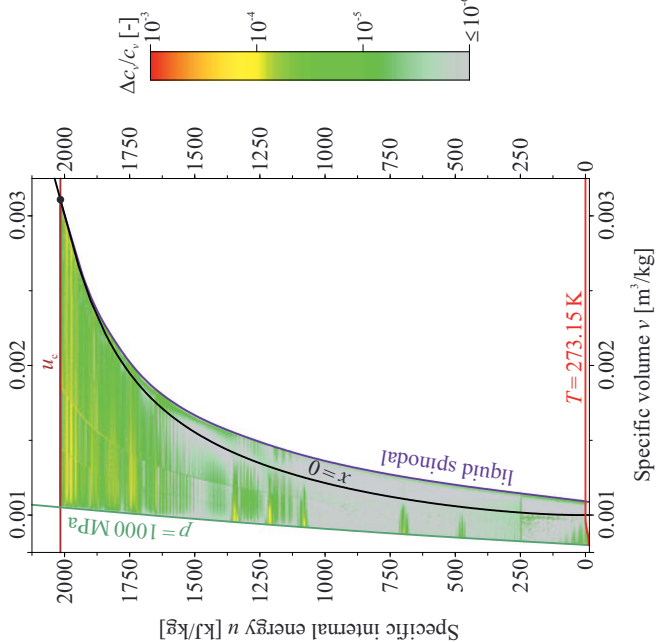

Figure A28: Deviations in spec. isochoric heat capacity $c_v(v,u)$ from IAPWS-95 in the liquid region L (derived from $s(v,u)$).

References

[1] IAPWS: *Revised Release on the IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use* (2014), available at http://www.iapws.org.

[2] Wagner, W., Pruß, A.: The IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use, *J. Phys. Chem. Ref. Data* **31**, 387-535 (2002).

[3] IAPWS: *Revised Release on the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam* (2007), available at http://www.iapws.org.

[4] Wagner, W., Cooper, J.R., Dittmann, A., Kijima, J., Kretzschmar, H.-J., Kruse, A., Mareš, R., Oguchi, K., Sato, H., Stöcker, I., Šifner, O., Takaishi, Y., Tanishita, I., Trübenbach, J., Willkommen, Th.: The IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam, *J. Eng. Gas Turbines & Power* **122**, 150-182 (2000).

[5] IAPWS: *Revised Supplementary Release on Backward Equations for Pressure as a Function of Enthalpy and Entropy p(h,s) for Regions 1 and 2 of the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam* (2014), available at http://www.iapws.org.

[6] IAPWS: *Revised Supplementary Release on Backward Equations for the Functions T(p,h), v(p,h), and T(p,s), v(p,s) for Region 3 of the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam* (2014), available at http://www.iapws.org.

[7] IAPWS: *Revised Supplementary Release on Backward Equations p(h,s) for Region 3, Equations as a Function of h and s for the Region Boundaries, and an Equation* $T_{sat}(h,s)$ *for Region 4 of the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam* (2014), available at http://www.iapws.org.

[8] IAPWS: *Revised Supplementary Release on Backward Equations for Specific Volume as a Function of Pressure and Temperature v(p,T) for Region 3 of the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam* (2014), available at http://www.iapws.org.

[9] IAPWS: *Guideline on the Tabular Taylor Series Expansion (TTSE) Method for Calculation of Thermodynamic Properties of Water and Steam Applied to IAPWS-95 as an Example* (2003), available at http://www.iapws.org.

[10] Miyagawa, K., Hill, P. G.: Rapid and Accurate Calculation of Water and Steam Properties Using the Tabular Taylor Series Expansion Method, *Journal of Engineering for Gas Turbines and Power*, **123** (3), 707-712 (2001).

[11] Weber, I., Kodl, I., Kretzschmar, H.-J., Knobloch, K.: Test Report of Documentation and Software of TTSE Method applied to IAPWS-95 as an Example, IAPWS Report (2002), available at http://www.iapws.jp.

[12] Poling, B. E., Prausnitz, J. M., O'Connell, J. P.: *Properties of Gases and Liquids*, Fifth Edition, DOI: 10.1036/0070116822, McGraw-Hill Professional (2001).

[13] van der Waals, J. D.: Over de Continuiteit van den Gas-en Vloeistoftoestand (Dutch; in English: On the Continuity of the Gas- and Liquid-State), University of Leiden (1873).

[14] Peng, D.-Y., Robinson, D.P.: A New Two-Constant Equation of State, *Ind. Eng. Chem. Fundam.* **15** (1), 59-64 (1976).

[15] Pitzer, K. S.: Corresponding States for Perfect Liquids, *J. Chem. Phys.*, **7**, 583–590 (1939).

[16] Deiters, U. K., Macías-Salinas, R.: The calculation of densities from cubic equations of state-revisited, *Ind. Eng. Chem. Res.*, **53**, 2529–2536, DOI: 10.1021/ie4038664 (2014).

[17] Span, R., Wagner, W.: Equations of State for Technical Applications. I. Simultaneously Optimized Functional Forms for Nonpolar and Polar Fluids, *Int. J. Thermophys.*, **24**, 1-39 (2003).

[18] Span, R., Wagner, W.: Equations of State for Technical Applications. II. Results for Nonpolar Fluids, *Int. J. Thermophys.*, **24**, 41-109 (2003).

[19] Span, R., Wagner, W.: Equations of State for Technical Applications. III. Results for Polar Fluids, *Int. J. Thermophys.*, **24**, 111-162 (2003).

[20] Lemmon, E.W., Span, R.: Short Fundamental Equations of State for 20 Industrial Fluids, *J. Chem. Eng. Data*, **51** (3), 785–850, DOI: 10.1021/je050186n (2006).

[21] Kunz, O., Klimeck, R., Wagner, W., Jaeschke, M.: The GERG-2004 wide-range equation of state for natural gases and other mixtures. GERG TM 15 2007. *Fortschr.-Ber. VDI*, Reihe 6, Nr. 557, VDI Verlag (2007).

[22] Jaeschke, M., Schley, P.: Ideal-gas thermodynamic properties for natural-gas applications, *Int. J. Thermophys.*, **16**, 1381-1392 (1995).

[23] Späth, H.: *One Dimensional Spline Interpolation Algorithms*, ISBN 1-56881-016-4, A K Peters (1995).

[24] Schoenberg, I. J.: Contributions to the problem of approximation of equidistant data by analytic functions, *Quart. Appl. Math.,* **4**, 45-99 and 112-141 (1946).

[25] Müller, W. C.: Fast and accurate water and steam properties programs for two-phase flow calculations, *Nuclear Engineering and Design*, **149** (1-3), 449-458 (1994).

[26] Kestin, J., Sengers, J.V., Kamgar-Parsi, B., Levelt Sengers, J.M.H.: Thermophysical Properties of Fluid H2O, *J. Phys. Chem. Ref. Data*, **13**, 175 (1984).

[27] Haar, L., Gallagher, J.S., Kell, G.S.: NBS/NRC Steam Tables, Hemisphere, New York (1984).

[28] Landis, F., Nielson, E. N.: The determination of thermodynamic properties by direct differentiation techniques, *Progress in International Research on Thermodynamic and Transport Properties, Second Symposium on Thermophysical Properties*, 218-227, Academic Press, New York (1962).

[29] Klaus, R. L., van Ness, H. C.: An Extension of the Spline Fit Technique and Applications to Thermodynamic Data, *AIChE Journal*, **13** (6), 1132-1136 (1967).

[30] Sokolnikoff, I. S., Redheffer, R. M.: *Mathematics of Physics and Modern Engineering*, McGraw-Hill, New York (1958).

[31] Schot, J.W.: A Spline-Function Method for Generating the Thermodynamic Properties of Water Substance, *Research and Development Report*, **2916**, Naval Ship Research and Development Center Washington D.C., Applied Mathematics Laboratory (1968).

[32] Wagner, W., Kleinrahm, R.: Densimeters for very accurate density measurements of fluids over a large range of temperature and up to high pressures, *Metrologia*, **41**(2), 24-39 (2004).

[33] Setzmann, U., Wagner, W.: A new method for optimizing the structure of thermodynamic correlation equations, *Int. J. Thermophys.*, **10**, 1103-1126 (1989).

[34] Lemmon, E. W., Jacobsen, R. T.: A New Functional Form and New Fitting Techniques for Equations of State with Application to Pentafluoroethane (HFC-125), *J. Phys. Chem. Ref. Data*, **34** (1), 69-108 (2005).

[35] Schumaker, L. L.: *Spline Functions: Basic Theory*, Third Edition, ISBN-13: 9780521705127, ISBN-10: 0521705126, Cambridge University Press (2007).

[36] Späth, H.: *Two Dimensional Spline Interpolation Algorithms*, ISBN 1-56881-017-2, A K Peters (1995).

[37] Pini, M., Spinelli, A., Persico, G., Rebay, S.: Consistent look-up table interpolation method for real-gas flow simulations, *Computers & Fluids*, **107**, 178–188 (2015).

[38] Schulze, C. W.: A Contribution to Numerically Efficient Modelling of Thermodynamic Systems, Dissertation, Institut für Thermodynamik, Fakultät Maschinenwesen, Technische Universität Braunschweig (2013).

[39] Bell, I. H., Wronski, J., Quoilin, S. Lemort, V.: Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp, *Industrial & Engineering Chemistry Research*, **53** (6), 2498-2508 (2014).

[40] Laughman, C. R., Zhao, Y., Nikovski, D.: Fast Refrigerant Property Calculations Using Interpolation-Based Methods, *International Refrigeration and Air Conditioning Conference at Purdue (Proceedings)* (2012).

[41] Swesty, F. D.: Thermodynamically Consistent Interpolation for Equation of State Tables, *Journal of Computational Physics*, **127** (1), 118-127 (1996).

[42] ASME: *ASME Steam Tables*, United Engineering Center, New York (1967).

[43] Kestin, J., Sage, B. H., Kennan, J. H., Keyes, F. G.: Comparisons and Comments on Existing Steam Data with Formulations, Moscow Meeting of the International Coordinating Committee on the Properties of Steam (1958).

[44] Herbst, G.: Zustandsgleichungen des Druckwassers für technische Anwendungen - Die Freie Enthalpie $g(T,p)$ in der nach $h(s,p)$ umformbaren Darstellung durch eine Spline-Funktion, Dissertation, Fakultät Maschinenwesen, Technische Universität Hannover (1976).

[45] White, R.: COMPLEX: A Least Squares Method for Fitting Quadratically Interpolated Tables to a Two-Dimensional Data Array, Lawrence Radiation Laboratory, University of California (1964).

[46] Kretzschmar, H.-J.: Zur Aufbereitung und Darbietung thermophysikalischer Stoffdaten für die Energietechnik, Promotion B (Habilitation), Fakultät Maschinenwesen, Technische Universität Dresden (1990).

[47] Kretzschmar, H.-J., Stöcker, I., Nabel, M., Dittmann, A.: A Method for Generating Interpolation Tables with Optimized Data Density for Fast Interpolations of Thermodynamic Properties in Process Modeling, *Proceedings of the 1995 ASME International Mechanical Engineering Congress in San Francisco*, No. 95-WA/HT-36, ASME, New York (1995).

[48] Gräber, M., Kirches, C., Schlöder, J. P., Tegethoff, W.: Nonlinear Model Predictive Control of a Vapor Compression Cycle Based on First Principle Models, *IFAC Proceedings Volumes*, **45** (2), 258-263 (2012).

[49] Kunick, M., Kretzschmar, H.-J., Gampe, U.: Fast Calculation of Thermodynamic Properties of Water and Steam in Process Modelling using Spline Interpolation, *Water, Steam, and Aqueous Solutions - Advances in Science and Technology for Power Generation, Proceedings of the 15th International Conference on the Properties of Water and Steam*, Ed. by R. Span and I. Weber, VDI GET, ISBN 978-3-931384-64-7 (2008).

[50] ANSYS Inc.: ANSYS CFX-Solver Theory Guide, Release 15.0, ANSYS Inc., Canonsburg (2013).

[51] ANSYS Inc.: ANSYS CFX-Pre User's Guide, Release 15.0, ANSYS Inc., Canonsburg (2013).

[52] Fog, A.: Optimizing Software in C++ - An Optimization Guide for Windows, Linux and Mac platforms, Technical University of Denmark (2015), available at http://www.agner.org.

[53] Fog, A.: Instruction Tables - Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs, Technical University of Denmark (2016), available at http://www.agner.org.

[54] Intel Corporation: Intel® 64 and IA-32 Architectures - Software Developer's Manual, Combined Volumes: 1, 2A, 2B, 2C, 3A, 3B, 3C and 3D, Intel Corporation (2016).

[55] Intel Corporation: Intel® Intrinsics Guide, interactive reference tool available at https://software.intel.com/sites/landingpage/IntrinsicsGuide/# (2017).

[56] Fog, A.: The microarchitecture of Intel, AMD and VIA CPUs - An Optimization Guide for Assembly Programmers and Compiler Makers, Technical University of Denmark (2016), available at http://www.agner.org.

[57] Levinthal, D.: Performance Analysis Guide for Intel® Core™ i7 Processor and Intel® Xeon™ 5500 Processors, Version 1.0, Intel Corporation (2009).

[58] Trübenbach, J.: Ein Algorithmus zur Aufstellung rechenzeitoptimierter Gleichungen für thermodynamische Zustandsgrößen, *VDI Fortschritt-Berichte*, Reihe 6, Nr. 417 (1999).

[59] Lemmon, E.W., Huber, M.L., McLinden, M.O.: NIST Standard Reference Database 23: Reference Fluid Thermodynamic and Transport Properties – REFPROP 9.1, National Institute of Standards and Technology, Standard Reference Data Program, Gaithersburg (2013).

[60] Brent, R. P.: *Algorithms for Minimization without Derivatives*, Prentice-Hall, Englewood Cliffs, New Jersey, ISBN 0-13-022335-2 (1973).

[61] IAPWS: *Guideline on the Fast Calculation of Steam and Water Properties with the Spline-Based Table Look-Up Method (SBTL)* (2015), available at http://www.iapws.org.

[62] Kunick, M., Kretzschmar, H.-J.: FluidSplines – Software for Generating Spline Functions, information available at http://www.thermodynamics-zittau.de.

[63] Shimrat, M.: Algorithm 112: Position of point relative to polygon, *Communications of the ACM*, **5** (8), (1962).

[64] Kunick, M., Kretzschmar, H.-J., Gampe, U., di Mare, F., Hrubý, J., Duška, M., Vinš, V., Singh, A., Miyagawa, K., Weber, I., Pawellek, R., Novy, A., Blangetti, F., Friend, D.G., Harvey, A.H.: Fast Calculation of Steam and Water Properties with the Spline-Based Table Look-Up Method (SBTL), *J. Eng. Gas Turbines & Power*, in preparation.

[65] IAPWS: *Release on the IAPWS Formulation 2008 for the Viscosity of Ordinary Water Substance* (2008), available at http://www.iapws.org.

[66] Wagner, W., Overhoff, U.: *Extended IAPWS-IF97 Steam Tables*, Springer-Verlag (2006).

[67] Wagner, W., Kretzschmar, H.-J.: *International Steam Tables*, Springer-Verlag (2008).

[68] Yang, H., Nürnberger, D., Kügeler, E.: Recent Progress in the hybrid-grid CFD simulation of turbomachinery flows, ISABE (2007).

[69] Becker, K., Heitkamp, K., Kügeler, E.: Recent progress in a hybrid-grid CFD solver for turbomachinery flows. *Proceedings Fifth European Conference on Computational Fluid Dynamics ECCOMAS CFD* (2010).

[70] di Mare, F.: Report on the Evaluation of the Spline-Based Table Look-Up Method (SBTL) in CPU-intensive 3D-CFD Applications, Report to IAPWS IRS Working Group (2014).

[71] White, A.J., Young, J.B., Walters, P.T.: Experimental Validation of Condensing Flow Theory for a Stationary Cascade of Steam Turbine Blades, *Phil. Trans. R. Soc. Lond. A*: *Mathematical, Physical and Engineering Sciences* **354**, 59-88 (1996).

[72] Kunick, M., Kretzschmar, H.-J., di Mare, F., Gampe, U.: CFD Analysis of Steam Turbines with the IAPWS Standard on the Spline-Based Table Look-Up Method (SBTL) for the Fast Calculation of Real Fluid Properties, *ASME Turbo Expo 2015: Turbine Technical Conference and Exposition, Vol. **8**: Microturbines, Turbochargers and Small Turbomachines; Steam Turbines*, ISBN: 978-0-7918-5679-6 (2015).

[73] Post, P.: Implementation and validation of a highly efficient thermodynamic algorithm for the 3D-CFD analysis of the steam flow in turbomachinery, Master's Thesis, Technische Universität Darmstadt (2015).

[74] Moore, M., Walters, P., Crane, R., Davidson, B.: Predicting the fog drop size in wet steam turbines, Institute of Mechanical Engineers (UK) , Wet Steam 4, Paper C 37/73, pp. 101–109 (1973).

[75] Post, P.: Simulation of a de Laval nozzle using three different property calculation algorithms for water and steam, Private communication, Technische Universität Darmstadt (2017).

[76] Bennoit, S.: Test of the IAPWS SBTL Guideline in KRAWAL-modular, Report to IAPWS IRS Working Group (2014).

[77] Weber, I.: Implementation and Test of SBTL property functions for water and steam in KRAWAL-modular, Private communication, SIEMENS AG, Siemens AG, Power and Gas Division (2017).

[78] Pawellek, R., Löw, T., Maltsev, A.: Evaluation of Spline Routines in EBSILON® Professional, Report to IAPWS IRS Working Group (2014).

[79] Kretzschmar, H.-J., Stöcker, I., Jähne, I., Kunick, M.: LibIF97 - Property Library for the Industrial Formulation IAPWS-IF97 for Water and Steam, Faculty of Mechanical Engineering, Zittau/Görlitz University of Applied Sciences (2016).

[80] Löw, T.: Implementation and Test of SBTL property functions for water and steam in EBSILON® Professional, STEAG Energy Services GmbH (2017).

[81] Berry, R. A., Zou, L., Zhao, H., Zhang, H., Peterson, J. W., Martineau, R. C., Kadioglu, S. Y., Andrs, D.: RELAP-7 Theory Manual, Idaho National Laboratory, Technical Report INL/EXT-14-31366 (Revision 2) (2016).

[82] IAPWS: *Release on the IAPWS Formulation 2011 for the Thermal Conductivity of Ordinary Water Substance* (2011), available at http://www.iapws.org.

[83] Kunick, M., Berry, R. A., Martineau, R. C., Kretzschmar, H.-J., Gampe, U.: Application of the new IAPWS Guideline on the fast and accurate calculation of steam and water properties with the Spline-Based Table Look-Up Method (SBTL) in RELAP-7, *KERNTECHNIK*, **82** (3) , 264-279 (2017).

[84] Kunick, M., Kretzschmar, H.-J.: Property Library LibSBTL_vu_95 for Water and Steam according to the IAPWS Guideline on the Fast Calculation of Steam and Water Properties with the Spline-Based Table Look-Up Method (SBTL) based on the Scientific Formulation IAPWS-95 and Current IAPWS Formulations for Transport Properties, Zittau/Görlitz University of Applied Sciences, Dept. of Technical Thermodynamics, Zittau, Germany (2015).

Online-Buchshop für Ingenieure

VDI nachrichten

BUCHSHOP

Online-Shops

Fachliteratur und mehr - jetzt bequem online recherchieren & bestellen unter: www.vdi-nachrichten.com/Der-Shop-im-Ueberblick

Täglich aktualisiert: Neuerscheinungen VDI-Schriftenreihen

Im Buchshop von vdi-nachrichten.com finden Ingenieure und Techniker ein speziell auf sie zugeschnittenes, umfassendes Literaturangebot.

Mit der komfortablen Schnellsuche werden Sie in den VDI-Schriftenreihen und im Verzeichnis lieferbarer Bücher unter 1.000.000 Titeln garantiert fündig.

Im Buchshop stehen für Sie bereit:

VDI-Berichte und die Reihe **Kunststofftechnik:**

Berichte nationaler und internationaler technischer Fachtagungen der VDI-Fachgliederungen

Fortschritt-Berichte VDI:

Dissertationen, Habilitationen und Forschungsberichte aus sämtlichen ingenieurwissenschaftlichen Fachrichtungen

Newsletter „Neuerscheinungen":

Kostenfreie Infos zu aktuellen Titeln der VDI-Schriftenreihen bequem per E-Mail

Autoren-Service:

Umfassende Betreuung bei der Veröffentlichung Ihrer Arbeit in der Reihe Fortschritt-Berichte VDI

Buch- und Medien-Service:

Beschaffung aller am Markt verfügbaren Zeitschriften, Zeitungen, Fortsetzungsreihen, Handbücher, Technische Regelwerke, elektronische Medien und vieles mehr – einzeln oder im Abo und mit weltweitem Lieferservice

Die Reihen der Fortschritt-Berichte VDI:

1 Konstruktionstechnik/Maschinenelemente
2 Fertigungstechnik
3 Verfahrenstechnik
4 Bauingenieurwesen
5 Grund- und Werkstoffe/Kunststoffe
6 Energietechnik
7 Strömungstechnik
8 Mess-, Steuerungs- und Regelungstechnik
9 Elektronik/Mikro- und Nanotechnik
10 Informatik/Kommunikation
11 Schwingungstechnik
12 Verkehrstechnik/Fahrzeugtechnik
13 Fördertechnik/Logistik
14 Landtechnik/Lebensmitteltechnik
15 Umwelttechnik
16 Technik und Wirtschaft
17 Biotechnik/Medizintechnik
18 Mechanik/Bruchmechanik
19 Wärmetechnik/Kältetechnik
20 Rechnerunterstützte Verfahren (CAD, CAM, CAE CAQ, CIM ...)
21 Elektrotechnik
22 Mensch-Maschine-Systeme
23 Technische Gebäudeausrüstung

ISBN 978-3-18-361806-4