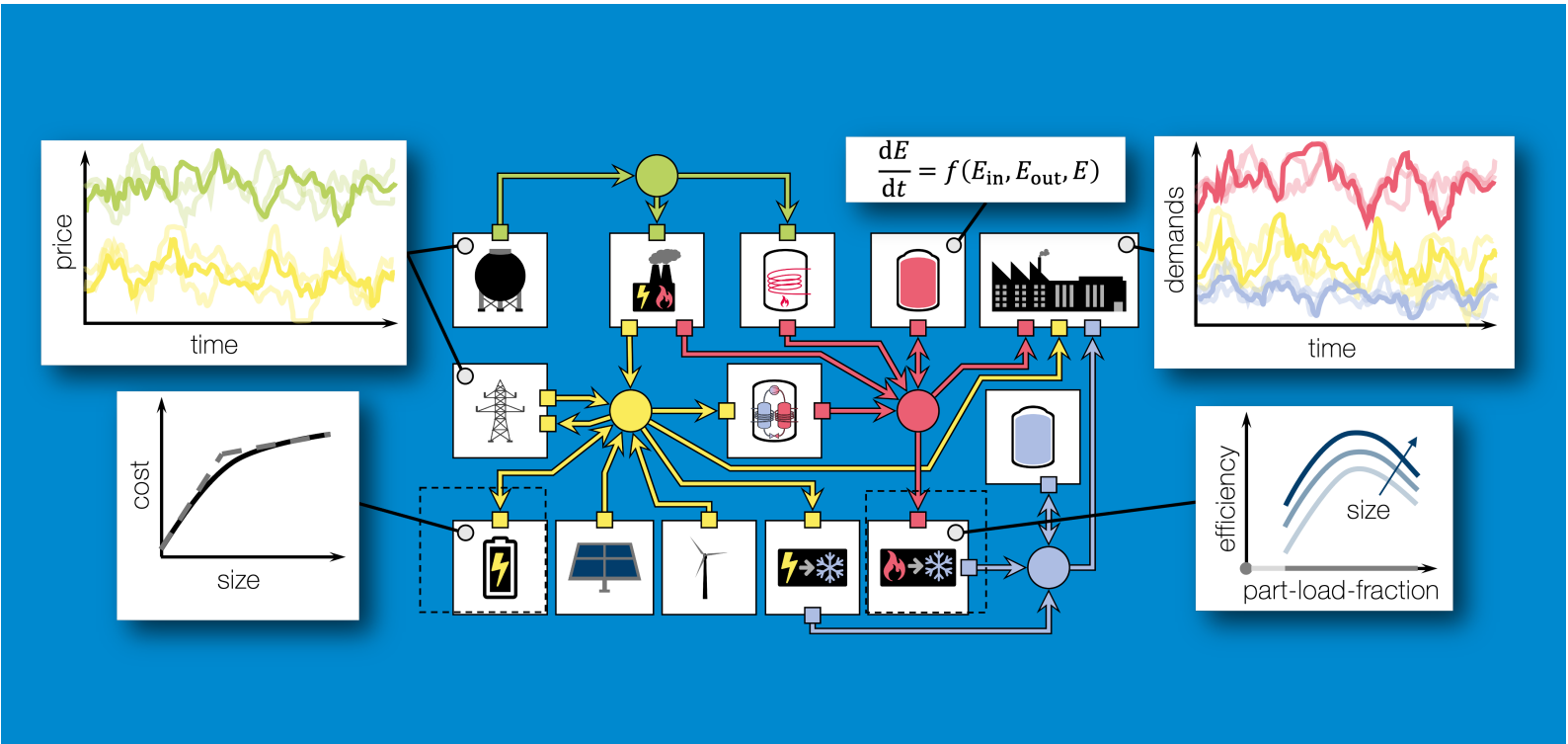
Aachener Verfahrenstechnik

RWTH AACHEN UNIVERSITY

Aachener Verfahrenstechnik Series
AVT.SVT – Process Systems Engineering
Volume 33 (2025)

Marco Langiu

Optimal Design and Operation of Energy Systems under Uncertainty via Two-Stage Stochastic Programming



Optimal Design and Operation of Energy Systems under Uncertainty via Two-Stage Stochastic Programming

Optimaler Entwurf und Betrieb von Energiesystemen unter Unsicherheit mittels Two-Stage Stochastic Programming

Von der Fakultät für Maschinenwesen der Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von

Marco Langiu

Berichter: Universitätsprofessor Alexander Mitsos, Ph.D.
Associate Professor Joseph Scott, PhD

Tag der mündlichen Prüfung: 22. November 2024

Titel: Optimal Design and Operation of Energy Systems under Uncertainty via Two-Stage Stochastic Programming

Autor: Marco Langiu

Reihe: Aachener Verfahrenstechnik Series
AVT.SVT - Process Systems Engineering
Band 33 (2025)

Herausgeber: Aachener Verfahrenstechnik
Forckenbeckstraße 51
52074 Aachen
Tel.: +49 (0)241 80 97717
Fax.: +49 (0)241 80 92326
E-Mail: secretary.svt@avt.rwth-aachen.de
http://www.avt.rwth-aachen.de

Volltext verfügbar: 10.18154/RWTH-2025-01116

Preface

This thesis was written during my time as a scientific employee, first at the Institute for Energy Systems Engineering (IEK-10) (now the Institute for Climate and Energy [ICE-1]) of the Forschungszentrum Jülich GmbH, and later at the Chair of Process Systems Engineering (AVT.SVT) of RWTH Aachen University.

First and foremost, I would like to express my sincere gratitude to my doctoral advisor, Professor Alexander Mitsos, Ph.D., for his unwavering support, for constantly challenging me, and for his integrity and high standards that served as an example to follow. I would also like to thank Professor Joseph Scott, Ph.D., for providing a recording of his talk on projection-based decomposition algorithms and for agreeing to serve as the co-examiner for this thesis. I am also grateful to Professor Dr.-Ing. Jakob Andert for chairing my examination.

The journey toward this thesis began even before my time at IEK-10, during my master's thesis under Hatim Djelassi, to whom I am deeply thankful for introducing me to many topics related to optimization and programming and for being an outstanding supervisor. Together with him, Dominik Bongatz, Jaromił Najman, and Susanne Sass, I not only had partners for insightful scientific discussions but also enjoyed many bouldering sessions.

I am especially grateful to my supervisor, Manuel Dahmen, for his guidance and unwavering positivity, which kept me motivated throughout this journey. Although I was the first member of the Optimization and Machine Learning group at IEK-10, I was fortunate to be only the second doctoral candidate at the institute, following Fritz Röben. With Fritz, I shared not only an office but also ideas, motivation, and many plants, making the onboarding a very enjoyable experience. The institute grew quickly, providing me with the rewarding opportunity to make my work on COMANDO a collaborative effort, spanning the different groups at IEK-10 and culminating in my first publication. In particular, I would like to thank my colleagues David Shu, Dominik Hering, and Florian Baader for their support and valuable contributions. Another enjoyable collaboration arose from the joint work with Philipp Glücker for the first OSMSES conference. I am also happy to have convinced him and Sonja Germscheid to join efforts in keeping the bouldering group alive after the original core parted ways following the completion of their degrees. I also enjoyed the frequent cooking with Philipp and Sonja and am deeply thankful for their food deliveries during the early days with my two children, which made this time so much easier and more enjoyable.

As my time in Jülich passed more quickly than I had anticipated, I continued my doctoral studies at the SVT, where I was kindly welcomed by my new colleagues, especially my office mates Jannik Lüthje and Jan Rittig, whom I thank for the good times and excellent working atmosphere. I thoroughly enjoyed exchanging ideas and engaging in stimulating discussions with the other members of the Energy Systems and MAiNGO Groups. I am particularly grateful to Jannik Lüthje, Clara Witte, and Aron Zingler for helping me refine and advance my ideas on numerous occasions, and to Marc-Daniel Stumm and Mohammad El Wajeh for their patience and support in setting up and testing the COMANDO-DyOS interface.

Finally, I would like to thank my family for their love, pride, and support throughout this journey. Most importantly, I am forever indebted to my wife, Sara, for her incredible patience, her encouragement, for always being there for me, and for giving me the two most precious gifts of all. Without you, I would not have made it this far.

Aachen, January 2025 *Marco Langiu*

Contents

Nomenclature

Acronyms

AML	algebraic modeling language
ANN	artificial neural network
API	application programming interface
B&B	branch & bound
CEPCI	chemical engineering plant cost index
COMANDO	**c**omponent-**o**riented **m**odeling and optimiz**a**tion for **n**onlinear **d**esign and **o**peration
COP	coefficient of performance
DAMF	differential-algebraic modeling framework
DR	demand response
ESMF	energy system modeling framework
GWI	global warming impact [US$/a]
(MI)DO	(mixed-integer) dynamic optimization
(MI)LP	mixed-integer linear programming
(MI)NLP	(mixed-integer) nonlinear programming
(MI)QCQP	(mixed-integer) quadratically constrained quadratic programming
MT	multiple temperature
MUSE-BB	**mu**lti-**se**ction **b**ranch & **b**ound
NAC	nonanticipativity constraints
PBDA	projection-based decomposition algorithm
ReLU	rectified linear unit
ST	single temperature
TAC	total annualized cost
TAR	total annualized revenue [US$/a]

Component Labels

AC	adsorption chiller
ACC	air-cooled condenser
B	boiler
BAT	battery
CC	compression chiller

CG	consumer group
CHP	combined heat and power unit
CON	condenser / condensation section of ACC
CS	cooling system
DEM	demand
DES	desuperheating section of ACC
ECO	economizer
EVA	evaporator
F	fan
GG	gas grid
HP	heat pump
HR	heating rod
HS	heat source
HT	heat transfer
HX	heat exchanger
L	linking subsystem
M	(thermal) mass
NW	network
P	pump
PG	power grid
PV	photovoltaic array
REC	recuperator
SUP	superheater
T	turbine
TES	thermal energy storage
WH	waste heat

Greek Symbols

α	fluid heat transfer coefficient [W/(m^2 K)]
β	convergence order
Δ	difference, gap
ε	tolerance
ϵ	specific entropy [J/kg]
$\boldsymbol{\zeta}$	element of the nonpositive orthant
η	efficiency [–]
θ	thickness [m]
κ	isentropic expansion coefficient
$\boldsymbol{\lambda}$, $\boldsymbol{\pi}$	dual multipliers
μ	kinematic viscosity [m/s^2]

$\boldsymbol{\xi}$	element of lifted space resulting from scenario relaxation
ρ	branching priority ratio
σ	strong-branching score
ϕ	reduced mass flow [–]
τ	strong-branching threshold

Latin Symbols

A	area [m^2]
c	coefficient
c_p	mass-specific heat capacity [J/(kg K)]
C	cost [US$]
$\mathcal{C}$	set of components / continuity class
d	diameter [m]
e	temperature effectiveness [–]
$E, \dot{E}$	energy [J], energy flow rate [W]
F	correction factor [–]
$\mathcal{F}$	feasible set of an optimization problem
h	specific enthalpy [kJ/(kg K)]
$\boldsymbol{H}$	NAC constraint matrix
$\mathbb{I}\bullet$	the set of nonempty, hyperrectangle subsets of some set $\bullet \subseteq \mathbb{R}^m$
k	thermal conductivity [W/(m K)] / number of partitions
K_S	coefficient from Stodola's ellipse law [m^2]
L	length [m] / Lipschitz constant
$\mathcal{L}$	list of k scenarios for which both sibling subproblems are feasible, the associated second-stage variable instances will be branched, producing 2^k orthant nodes
$\dot{m}$	mass flow rate [kg/s]
M	investment cost exponent
$\mathcal{M}$	map from scenarios producing exactly one infeasible sibling subproblem to the sibling n with a feasible subproblem, the associated second-stage variable instances will be branched, but do not add to the number of orthant nodes generated
n	node in a B&B tree
N	number, e.g., of first-stage variables (subscript x) or second-stage constraints (subscript II)
$\mathcal{N}$	set of open nodes of the B&B tree
Nu	Nusselt number [–]
$\mathcal{O}$	Landau notation
p	pressure [Pa]
$\boldsymbol{p}$	vector of parameters in component models

P	electrical power [W]
Pr	Prandtl number [–]
PR	pressure ratio [–]
q	vapor quality [–]
$\dot{Q}$	heat flow rate [W]
r	off-design reduction coefficient according to Ghasemi et al., 2013b, $\frac{\eta}{\eta_{\mathrm{T,d}}}$ [–]
$\boldsymbol{r}$	reduced cost multipliers
$\mathbb{R}$	real numbers
Re	Reynolds number [–]
s	scenario
$\mathcal{S}$	set of scenarios
SP	turbine size parameter $\frac{\dot{V}_{\mathrm{o,is,d}}^{0.5}}{\Delta h_{\mathrm{is,d}}^{0.25}}$ [m]
t	time [h]
T	temperature [K] / length of time horizon [h]
$\mathcal{T}$	time horizon
U	overall heat transfer coefficient [W/(m^2 K)]
v	velocity [m/s] / generic variable
$V, \dot{V}$	volume [m^3], volumetric flow rate [m^3/s]
VR	turbine volume ratio $\frac{\dot{V}_{\mathrm{o,is,d}}}{\dot{V}_{\mathrm{i,d}}}$ [–]
w	(scenario) weight
W	width
$\boldsymbol{x}$	first-stage / design variables
$\mathcal{X}$	domain of $\boldsymbol{x}$
$\boldsymbol{y}$	second-stage / operational variables
$\mathcal{Y}$	domain of $\boldsymbol{y}$
$\boldsymbol{z}$	aggregation of first- and second-stage variables, i.e., $\boldsymbol{z} \coloneqq (\boldsymbol{x}, \boldsymbol{y})$
$\mathcal{Z}$	domain of $\boldsymbol{z}$

Subscripts

$-$	nonpositive orthant
$\underline{\bullet}$	lower bound
0	initial state
amb	ambient conditions
ann	annuitiy
b	boiling
B	baffle
BOP	balance of plant
br	(geothermal) brine

C	Colburn
c	component / cold
crit	critical point
cs	cross-section
cv	convex relaxation
cw	cooling water
d	design state
D	Darcy
e	entry
E&D	exploration and development
el	electrical / electricity
eq	equivalent
f	fin
fix	fixed (cost)
fl	flow
g	generator
gr	ground
h	hot
I	first-stage / investment
ib	isobutane
II	second-stage
i	inlet / inside
is	isentropic conditions
lim	limit
liq	liquid
m	mechanical
mat	material
NAC	nonanticipativity constraints
nb	nucleate boiling
nom	nominal
op	operation / operational
o	outlet / outside
p	pipe / (tube) pitch
R	(fouling) resistance
re	return
ref	reference
rel	relative
s	shell
sat	saturation

S&T	shell and tube
sib	sibling
spec	specific (investment cost)
st	(turbine) stage
t	tube
tb	tube-bundle
th	thermal
tot	total
tp	tube passes
vap	vapor
var	variable (cost)
w	wall
wf	weighted fin

Superscripts

$\overline{\bullet}$	upper bound or average
d	node obtained from branching down
n	generic B&B node
o	orthant node
p	parent node
u	node obtained from branching up
a	algebraic (variable)
c	control (variable)
conv	conversion components
d	differential (variable)
†	incumbent
$*$	optimal / choked condition
sto	storage components

Functions

$\boldsymbol{e}$	denotes vectors of expressions of interest in component models, see Section 2.2.1
d	distance measure, see Definition 4.2
c	denotes an expression associated to a connector in component models, see Section 2.2.1
f	denotes objective functions
$f_{\mathrm{II},s}^{\mathcal{Y}_s}$	second-stage optimal value function, parametric optimal value of $\mathrm{RP}_s^{\mathcal{Y}_s}(\boldsymbol{x})$
$f_s^{\mathcal{X},\mathcal{Y}_s}$	scenario optimal value function, parametric optimal value of $\mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$
$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda})$	parametric optimal value of Lagrangian Relaxation $\mathrm{LR}^{\mathcal{X},\mathcal{Y}}$

$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda})$	parametric optimal value Lagrangian subproblems $\mathrm{LSP}_s^{\mathcal{X},\mathcal{Y}_s}$
$\boldsymbol{g}$	denotes vectors of inequality constraint functions
$\boldsymbol{h}$	denotes vectors of equality constraint functions
r_h	contribution to off-design reduction coefficient from relative enthalpy drop, see Section 3.1.3 and Chapter A
$r_{\dot{V}}$	contribution to off-design reduction coefficient from relative volumetric flow rate, see Section 3.1.3 and Chapter A
$\check{\nabla}$	subgradient operator, also see subtangent
$\boldsymbol{\psi}$	vector of state transition functions (right-hand side of differential equations) in D&O
sub_ϕ^n	subtangent of function ϕ at midpoint of node n
$\mathrm{vio}_{\mathrm{DE}}^n$	violation measure, see Definition 4.3
W	width of a multidimensional interval, see Definition 4.1

Optimization Problems

DE	deterministic equivalent form of TSP (restricted to the domain of node n)
$\mathrm{DE}_{\mathrm{NAC}}$	NAC formulation, equivalent to DE and TSP
D&O	Design and Operation problem
LP_s^n	linear relaxation of MC_s^n for node n (providing lower bounds)
LSP_s^n	relaxations of $\mathrm{DE}_{\mathrm{NAC}}$ for node n by dualizing NACs with multipliers $\boldsymbol{\lambda}_s$
MC_s^n	McCormick relaxation of SP_s^n for node n
$\mathrm{OBBT}_{s,v}^n$	OBBT problems for variable v and node n
RP_s^n	recourse problems for a given value of $\boldsymbol{x}$ for node n (providing upper bounds)
R_s^n	generic scenario relaxation for node n
SP_s^n	relaxations of $\mathrm{DE}_{\mathrm{NAC}}$ for node n by dropping NACs
TSP	two-stage (stochastic programming) problem, also see $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$

Kurzfassung

Die vorliegende Arbeit befasst sich mit der Modellierung von Energiesystemen für den optimalen Entwurf und Betrieb unter Unsicherheit, sowie mit der effizienten Lösung der resultierenden Optimierungsprobleme.

Zu diesem Zweck entwickelten wir das open-source Python-Paket COMANDO (**c**omponent-**o**riented **m**odeling and optimiz**a**tion for **n**onlinear **d**esign and **o**peration), welches eine strukturierte und flexible Modellierung von Energiesystemen unter detaillierter Berücksichtigung von Nichtlinearitäten, dynamischem Verhalten und diskreten Entscheidungen ermöglicht. COMANDO verbindet Ansätze zur strukturierten Modellierung aus algebraischen Modellierungssprachen und differentiell-algebraischen Modellierungsframeworks mit der Flexibilität einer allgemeinen Programmiersprache, um benutzerspezifische Problemformulierungen und Lösungsroutinen zu ermöglichen. Wir demonstrieren die Fähigkeiten von COMANDO anhand von Fallstudien, in denen automatische Linearisierung, dynamische und stochastische Optimierung sowie neuronale Netze als datengetriebene Ersatzmodelle in der deterministisch globalen Optimierung zur Anwending kommen.

In einem realistischen Anwendungsfall verwenden wir COMANDO um einen luftgekühlten geothermischen Organic Rankine Cycle für den Betrieb unter variablen Umgebungstemperaturen auszulegen. Dazu erstellen wir detaillierte Komponentenmodelle, von Pumpen, Wärmetauschern, Turbinen und Kondensatoren und verwenden künstliche neuronale Netze zur genauen Vorhersage von Fluideigenschaften sowie Off-designeigenschaften der Komponenten. Wir optimieren Design und Betrieb global, wobei wir gleichzeitig mehrere Betriebspunkte betrachten, um den erwarteten annualisierten Gesamtumsatz zu maximieren. Unsere Ergebnisse zeigen, dass die alleinige Betrachtung einzelner Betriebszustände im Allgemeinen zuSystemdesigns führt, die nicht über den gesamten Bereich der Umgebungstemperaturen betrieben werden können. Die Ergebnisse der Systemoptimierung für einzelne Temperaturen können jedoch bei der Suche nach einem optimalen Design, das den gesamten Betriebsbereich abdeckt, hilfreich sein.

Um realistische Design- und Betriebsprobleme effizienter und skalierbarer zu lösen, entwickeln wir einen neuen Dekompositionsalgorithmus namens MUSE-BB (**mu**lti-**se**ction **b**ranch & **b**ound), welcher mehrere potentielle Schwachstellen existierender Methoden adressiert. Zu diesem Zweck kombiniert MUSE-BB das gleichzeitige Branchen auf mehreren Betriebsvariablen mit einer Branch &Bound Suche, die explizit sowohl Design- als auch Betriebsvariablen berücksichtigt. Unsere theoretischen und numerischen Ergebnisse zeigen, dass MUSE-BB sowohl mit standard Branch & Bound, als auch mit bestehenden Dekompositionsalgorithmen konkurrieren kann.

Die Arbeit schließt mit einer Diskussion der Implikationen unserer Ergebnisse und mit Vorschlägen für die zukünftige Forschung auf dem Gebiet der Modellierung und Optimierung von Energiesystemen.

Summary

This thesis deals with the modelling of energy systems for optimal design and operation under uncertainty, as well as the efficient solution of the resulting optimization problems.

For this purpose, we have developed the open-source Python package COMANDO (**c**omponent-**o**riented **m**odeling and optimiz**a**tion for **n**onlinear **d**esign and **o**peration), which enables structured and flexible modelling of energy systems with detailed consideration of nonlinearities, dynamic behavior and discrete decisions. COMANDO combines structured modeling features of algebraic modelling languages and differential-algebraic modelling frameworks with the flexibility of a general-purpose programming language to allow for user-specific problem formulations and solution routines. We demonstrate the capabilities of COMANDO through case studies involving automatic linearization, dynamic and stochastic optimization, and the use of artificial neural networks as data-driven surrogate models in deterministic global optimization.

In a realistic use-case we employ COMANDO to design an air-cooled geothermal organic Rankine cycle for operation under variable ambient temperatures. For this, we create detailed component models for pumps, heat exchangers, turbines and condensers, and make use of artificial neural networks to accurately predict fluid properties and off-design characteristics of the components. We globally optimize the system design and operation under simultaneous consideration of multiple operating points, to maximize the expected total annualized revenue. Our results show that the consideration of individual operating points generally results in designs that cannot be operated over the full range of ambient temperatures. Nevertheless, the results from optimizing the system for individual temperatures can be used to support the search for an optimal design that covers the entire operating range.

For a more efficient and scalable solution of realistic design and operation problems, we develop a new decomposition-based algorithm called MUSE-BB (**mu**lti-**se**ction **b**ranch & **b**ound), which addresses several potential issues of existing methods. To do this, MUSE-BB combines simultaneous branching of multiple operational variables with a branch & bound search that explicitly considers both design and operational variables. Our theoretical and computational results indicate that MUSE-BB can be competitive with respect to both standard branch & bound, as well as existing decomposition algorithms.

The thesis concludes with a discussion of the implications of our results and with suggestions for future research in the field of power system modelling and optimization.

Publications and Copyrights

This thesis is partially based on research performed by the author during his time at the "Energy Systems Engineering Department" (IEK-10) at the Institute for Energy and Climate Research (now renamed to "Institute for Climate and Energy", ICE-1) of the Forschungszentrum Jülich GmbH (2018–2021) and at the Chair of Process Systems Engineering (AVT.SVT) of RWTH Aachen University (2021–2024). Parts of this thesis have already been published or submitted for publication and are reprinted with permission. These publications are listed below. Alexander Mitsos and Manuel Dahmen provided ideas and guidance as well as edits to all listed publications. Contributions of other co-authors are stated below the corresponding publications.

- Excerpts from the three following publications were used in Chapter 1.
- Chapter 2 is based on Langiu, Shu, Baader, Hering, Bau, Xhonneux, Müller, Bardow, Mitsos, and Dahmen (2021). "COMANDO: A Next-Generation Open-Source Framework for Energy Systems Optimization". In: *Comput. Chem. Eng.*, p. 107366. DOI: 10.1016/j.compchemeng.2021.107366
  © 2021 Elsevier Ltd.
  Contributions from Co-authors: David Shu contributed code for an automatic linearization and the case study in 2.3.1 with help and guidance from Manuel Dahmen and André Bardow. Florian Baader integrated Pyomo.DAE into the Pyomo interface of COMANDO and contributed the case study in 2.3.2 with help and guidance from Manuel Dahmen and André Bardow. Dominik Hering contributed to the case study in 2.3.3 with help and guidance from Manuel Dahmen, André Xhonneux, and Dirk Müller. Uwe Bau gave conceptual input for the creation of COMANDO.
- Chapter 3 is based on Langiu, Dahmen, and Mitsos (2022). "Simultaneous optimization of design and operation of an air-cooled geothermal ORC under consideration of multiple operating points". In: *Comput. Chem. Eng.* 161, p. 107745. DOI: 10.1016/j.compchemeng.2022.107745
  © 2022 Elsevier Ltd.
- Chapter 4 is based on Langiu, Dahmen, Bongartz, and Mitsos (2024). "MUSE-BB: A Decomposition Algorithm for Nonconvex Two-Stage Problems using Strong Multisection Branching". Submitted to *J. Glob. Optim.*, preprint available at https://optimization-online.org/?p=26862
  Contributions from Co-authors: Dominik Bongartz contributed to the conceptual development of the algorithm and provided guidance for the software implementation.

Additionally, the author contributed to the following publication that appeared during the author's time at the IEK-10, but do not form part of this thesis:

Glücker, Langiu, Pesch, Dahmen, and Benigni (2022). "Incorporating AC Power Flow into the Multi-Energy System Optimization Framework COMANDO". in: *2022 Open Source Modelling and Simulation of Energy Systems (OSMSES)* (Apr. 4–5, 2022). IEEE. DOI: 10.1109/osmses54027.2022.9769138

1. Introduction

Meeting the ever-increasing demand for energy while simultaneously reducing greenhouse gas emissions is one of the largest challenges faced by humanity today. Key characteristics of future energy systems, capable of achieving this goal, are high efficiency and flexibility in adapting to the variability of available feedstocks and demands (Liu, Georgiadis, and Pistikopoulos, 2010). Adequately (re-)designing energy systems to this end also requires accounting for their prospective operation (Frangopoulos, Von Spakovsky, and Sciubba, 2002). Naturally, future operating conditions, such as demands, prices, weather and other operational aspects are not known precisely, resulting in significant planning uncertainty that renders the design and operation of energy systems a challenging decision process (Pistikopoulos, 1995; Sahinidis, 2004). In the following, we formalize this decision process as a mathematical programming problem, give a overview of solution approaches for variants of this problem, and briefly discuss existing modeling tools. We highlight the need for advanced modeling tools and solution algorithms, motivating the presented work, and end this chapter with an outline of this thesis.

1.1. Energy System Design and Operation under Uncertainty

Addressing energy system design and operation requires system models that accurately capture the impact of individual decisions in the face of variable or uncertain operating conditions. For this purpose, energy systems can be seen as networks of interconnected components that transform and transport energy using a set of renewable or fossil resources to satisfy various kinds of demands (Beller, 1976). A model of a given energy system can thus be created by appropriately modeling all subsystems and individual components, their interaction, and their dependence on design and operating decisions as well as on uncertain parameters. While the subsequently discussed modeling tools and algorithms can be applied to a much wider range of systems – even outside of the energy domain – in this thesis we are primarily concerned with energy systems of intermediate size, e.g. buildings, industrial sites, and power plants.

In practice, models for such systems are frequently used for what is also known as *"what-if"* or *"scenario analysis"*. Here, system operation is simulated for a given (i.e., fixed) design, and operating conditions (Gilman, Lambert, and Lilienthal, 2006; Subramanian, Gundersen, and Adams, 2018), or an "ideal" operational-strategy is sought via operational optimization. Prospective costs, emissions, and other metrics can be estimated over the lifetime of the system by performing such scenario analyses for a range of different conditions. Repeating this procedure for different candidate designs provides alternatives among which the most promising option can be selected (e.g., Seeling-Hochmuth, 1997). While this sequential approach is pragmatic, the separate determination of design and operation is ad-hoc, and thus may yield suboptimal results.

An alternative is to formulate an optimization problem in which both design and operation are considered simultaneously (e.g., Papoulias and Grossmann, 1983; Andiappan, 2017; Frangopoulos, 2018; Demirhan et al., 2019). This is frequently done as a variant of the above scenario analysis, i.e., design and operation are optimized for a single, representative scenario, corresponding to "average", "nominal" or "design" conditions (Dembo, 1991). While in contrast to sequential approaches this will indeed result in a system design that is optimally suited for the selected operating point, there is still no guarantee of good overall performance. In particular, the resulting system may even be infeasible for operation in conditions that differ sufficiently from the assumed scenario. More importantly, even analyzing how changes in the uncertain parameters affect a given solution, e.g., via parametric optimization or sensitivity analysis (Pistikopoulos and Diangelakis, 2016; Mavromatidis, Orehounig, and Carmeliet, 2018; Ginocchi, Ponci, and Monti, 2021; Usher et al., 2023), generally is *not* sufficient to determine whether it is also good solution in the stochastic setting, see, e.g., Wallace, 2000 for simple counterexamples. Furthermore, even if many or all solutions obtained via scenario analysis share common features, e.g., energy storage is not part of any optimal scenario solution, this does *not* imply that these features are also part of an optimal solution in the uncertain setting. Ultimately, this is because scenario analysis implicitly assumes *perfect foresight*, i.e., the optimization is carried out in a deterministic setting. This disregards the changing availability of information over time, and as a result, flexibility has no value in these approaches (also see King and Wallace, 2012).

1.1.1. Two-Stage Stochastic Programming

To avoid the issues of sequential approaches and what-if analyses, it is possible to formulate an optimization problem for the combined design and operation, that considers multiple operating conditions simultaneously. A suitable approach for this is two-stage stochastic programming (Wallace and Fleten, 2003; Yunt et al., 2008; Birge and Louveaux, 2011; Li and Barton, 2015), typically applied when long-term ("here and now") decisions must be taken prior to the realization of a given scenario, in response to which recourse ("wait and see") decisions can be taken. The resulting *two stage problem* (TSP) can be expressed in the general form

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathcal{X}} f_{\mathrm{I}}(\boldsymbol{x}) + \sum_{s\in\mathcal{S}} \left[w_s \min_{\boldsymbol{y}_s\in\mathcal{Y}_s} \left\{ f_{\mathrm{II},s}(\boldsymbol{x},\boldsymbol{y}_s) \;\middle|\; \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x},\boldsymbol{y}_s) \leq \boldsymbol{0} \right\} \right] \\ &\text{s.t. } \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \end{aligned} \tag{TSP}$$

where design and operational decisions are captured by the variable vectors $\boldsymbol{x}$, and $\boldsymbol{y}_s$, with the associated domains $\mathcal{X}$, and $\mathcal{Y}_s$, respectively, the functions f_{I} and $f_{\mathrm{II},s}$ represent design and operational objectives, respectively, e.g., capital and operating expenditures, and the functions $\boldsymbol{g}_{\mathrm{I}}$ and $\boldsymbol{g}_{\mathrm{II},s}$ encode limitations that the taken decisions must adhere to, e.g., resulting from physical/thermodynamical laws, empirical rules or political requirements. For each scenario s from the overall set of scenarios $\mathcal{S}$, the corresponding instances of $f_{\mathrm{II},s}$ and $\boldsymbol{g}_{\mathrm{II},s}$ represent the operational objective and constraints associated to a different set of operating conditions, with appropriate weights w_s reflecting the frequency or probability with witch the respective conditions occur. In this sense, the above stochastic programming paradigm may also be applied to future events, that occur repeatedly, rather than just

once with a given probability, i.e., the scenarios do not not necessarily need to represent an uncertain, but may also represent a time-variable future (e.g., as in Yunt et al., 2008).

Early works addressing optimization problems of the form TSP, almost date back to the beginnings of mathematical programming, and consider linear programming (LP) variants, where all functions (f_{I}, $f_{\mathrm{II},s}$, $\boldsymbol{g}_{\mathrm{I}}$, $\boldsymbol{g}_{\mathrm{II},s}$) are linear in the variables $\boldsymbol{x}$ and $\boldsymbol{y}_s$, and $\mathcal{X}$ and $\mathcal{Y}_s$ do not impose integrality on any of the variables (Massé, 1946; Dantzig, 1955). Even today, LP formulations still constitute the most widely used variant of TSP, as they allow for the analysis of systems at the scale of entire nations, and the coupling with economical models (Beller, 1976; Kydes, 1978; Fishbone and Abilock, 1981; Schrattenholzer, 1981; Loulou and Labriet, 2007). Unfortunately, the arithmetic solution complexity of TSP scales at least polynomially with the number of considered scenarios (Anstreicher, 2001), so even LP variants can quickly become intractable when large numbers of scenarios are considered. To some extent, this issue can be addressed by specialized decomposition algorithms that exploit the inherent problem structure. Instead of solving the original problem directly, such algorithms employ problem transformation, such as projection, dualization, restriction, and relaxation (Geoffrion, 1970b; Geoffrion, 1970a) to derive various subproblems. As the resulting subproblems are much smaller than the original problem, they are significantly easier to solve and can be used to generate intermediate solutions that iteratively approach the solution of the original problem.

The two principal algorithmic approaches for exploiting the structure of TSP are stage-wise (or primal) and scenario-wise (or dual) decomposition. Stage-wise decomposition typically relies on variants of classical *Benders decomposition* (BD) (Benders, 1962) (better known in the stochastic programming community as L-shaped decomposition (Van Slyke and Wets, 1969)). While it only applies to LP variants of TSP in its original form, an generalization by Geoffrion (Geoffrion, 1972), known as *generalized BD* (GBD), extends this method to cover a particular subclass of convex nonlinear programming (NLP) problems, using nonlinear duality theory. BD-based approaches iteratively construct an outer-approximation of the *second-stage optimal value functions* $f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x})$, and thus fail to guarantee convergence if there are nonconvexities in the second stage. Whereas the focus on linear, or at least convex subclasses of TSP is natural from the perspective of computational tractability, and often sufficient for system analysis, ensuring an adequate level of detail for technical design and operation often requires introducing nonconvexities into a model. One example for this are discrete decisions, which may be required at the level of both, design (e.g., selection and connectivity of components), and operation (e.g., operating modes, unit commitment). Other sources of nonconvexity are nonlinearities that arise from the consideration of, e.g., component efficiencies, thermodynamic properties, or data-driven models such as ANNs. For problems of this type, several extensions of BD exist, that address mixed-integer LP (MILP) (e.g., Laporte and Louveaux, 1993; Angulo, Ahmed, and Dey, 2016), convex mixed-integer NLP (MINLP) (Li and Grossmann, 2018; Li and Grossmann, 2019a) as well as nonconvex MINLP (Li, Tomasgard, and Barton, 2011; Chen et al., 2011; Li, Sundaramoorthy, and Barton, 2014) variants of TSP.

Scenario-wise decomposition on the other hand typically relies on Lagrangian duality (Geoffrion, 2009; Fisher, 1981; Guignard and Kim, 1987; Dür and Horst, 1997) and are typically combined with branch and bound (B&B) (Horst and Tuy, 1996) to address nonconvexity. A classical example is Carøe and Schultz, 1999, which addresses MILP variants of TSP, while Karuppiah and Grossmann, 2007; Khajavirad and Michalek, 2009; Cao and Zavala, 2019 address the nonconvex MINLP subclass. Several methods also combine these

two principal decomposition approaches, resulting in powerful methods that can efficiently address general variants of TSP (e.g., Kannan, 2018; Li and Grossmann, 2019b).

The large number of existing decomposition algorithms essentially covers all of the different subclasses of TSP, however, efficiently solving large-scale problems arising from realistic applications remains a challenge. Furthermore, while the demand for good scaling with a large number of scenarios has resulted in a focus on methods that employ projection (see also Geoffrion, 1970b; Geoffrion, 1970a), in order to restrict the primary search to the design variables (Carøe and Schultz, 1999; Karuppiah and Grossmann, 2007; Cao and Zavala, 2019; Kannan, 2018; Li and Grossmann, 2019b), recent research suggest that this approach may face some theoretical issues (Robertson, Cheng, and Scott, 2020).

1.1.2. Alternative Approaches for Addressing Uncertainty

Apart from two-stage problems, the field of stochastic programming addresses many other related and generalized problems, relevant to energy system design and operation, such as chance constrained programming (e.g., Charnes and Cooper, 1959), robust optimization (e.g., Ben-Tal and Nemirovski, 2002), and in particular multistage stochastic programming (e.g., Massé, 1946; Birge, 1985; Carøe and Schultz, 1999; Pereira and Pinto, 1991; Rockafellar and Wets, 1991).

In addition to stochastic programming, several alternative approaches for addressing uncertainty exist. Examples include Markov decision processes and reinforcement learning (Puterman, 2014; Perera and Kamalaruban, 2021), Bayesian approaches (Sorourifar, Choksi, and Paulson, 2021; Borunda et al., 2016) and the concepts of fuzzy sets and numbers (Zadeh, 1965; Bellman and Zadeh, 1970; Martinsen and Krey, 2008), also see the reviews Wets, 1996; Sahinidis, 2004; Soroudi and Amraee, 2013; Aien, Hajebrahimi, and Fotuhi-Firuzabad, 2016; Grossmann et al., 2016. Finally, there are even attempts to unify all existing approaches to decision making under uncertainty in a single, overarching framework (Powell, 2021).

In this thesis, however, we restrict our focus to the two-stage formulation TSP. Also we assume a fixed, sufficiently representative scenario set $\mathcal{S}$ exists and thus do not address approaches relying on continuous random variables or repeated sampling (e.g. Norkin, Pflug, and Ruszczyński, 1998; Higle and Sen, 1991; Kleywegt, Shapiro, and Homem-de-Mello, 2002; Dantzig and Infanger, 2010; Shao and Scott, 2018). Our focus is motivated by the fact that combined design and operation is an "inherently two-stage process" (King and Wallace, 2012), where the primary interest is the resulting design. While alternative approaches to deal with uncertainty may result in more accurate representations of the real decision process, they primarily affect the resulting operational policies, whereas the effect on the optimal design is typically negligible, provided the set $\mathcal{S}$ is sufficiently representative.

1.1.3. Modeling Tools

After reviewing the theoretical aspects of formulating and solving optimization problems related to energy system design and operation, we next consider how a problem of the form TSP can be set up in practice. General-purpose algebraic modeling languages (AMLs), e.g., GAMS (Bussieck and Meeraus, 2004) or Pyomo (Hart, Watson, and Woodruff, 2011) are the de-facto standard for formulating optimization problems such as the optimal design and operation of energy systems, offering flexibility in the choice of algebraic formulation

and solution approach. However, more specialized energy system modeling frameworks (ESMFs), e.g., OSeMOSYS (Howells et al., 2011) or oemof (Hilpert et al., 2018) are available, that offer a component-oriented modeling approach, i.e., system models are created by specifying connections between component models. This approach simplifies the modeling process, model maintenance, and model re-use.

Established ESMFs typically employ linear programming (LP) (Schrattenholzer, 1981; Fishbone and Abilock, 1981; Loulou and Labriet, 2007; Bakken, Skjelbred, and Wolfgang, 2007; Howells et al., 2011; Hunter, Sreepathi, and DeCarolis, 2013; Dorfner, 2016) or mixed-integer linear programming (MILP) (Pfenninger and Keirstead, 2015; Hilpert et al., 2018; Atabay, 2017; Brown, Hörsch, and Schlachtberger, 2018; Johnston et al., 2019) formulations, well-suited for techno-economic analysis of large-scale systems (Connolly et al., 2010; Pfenninger, Hawkes, and Keirstead, 2014; Beuzekom, Gibescu, and Slootweg, 2015). In contrast, technical system design and operation requires a more detailed representation of system behavior, often giving rise to nonlinearities and dynamic effects that are difficult or impractical to represent with MILP formulations (see e.g., Li et al., 2011; Goderbauer et al., 2016; Schäfer et al., 2019b; Schäfer et al., 2019a). Decisions taken in the early stages of system design often have a large impact on overall cost and deviating from them at a later stage is generally very difficult (Pistikopoulos, 1995; Mitsos et al., 2018). This motivates the development of new modeling tools which are capable to provide a useful level of abstraction, capturing the modular nature of energy systems, while still being flexible enough to incorporate the necessary level of modeling detail to produce accurate results. Furthermore, while the presented stochastic programming literature indicates that a rich set of approaches for the design and operation problems is available, many existing modeling tools do not fully exploit these methods.

1.2. Goals of this Thesis

This thesis aims to address the challenges of structured and flexible modeling of energy system design and operation, as well as the efficient solution of the resulting optimization problems in the general nonlinear, nonconvex setting. To this end, we developed the next-generation ESMF for **c**omponent-**o**riented **m**odeling and optimiz**a**tion for **n**onlinear **d**esign and **o**peration (COMANDO), which we present in Chapter 2. COMANDO is an open-source Python package that enables modeling of energy systems at a high level of detail, including general nonlinearities, dynamic behavior, and discrete decisions. In addition to generic modeling capabilities available in AMLs, COMANDO offers a structured, modular generation of system models from subsystem and component models. Furthermore, it benefits from the full flexibility of a general, high-level programming language, useful for interfacing with a wide range of existing software, and formulating custom problem transformations and solution routines. We demonstrate features of COMANDO via case studies, including automated linearization, dynamic optimization, stochastic programming, and the use of nonlinear artificial neural networks as surrogate models in a reduced-space formulation for deterministic global optimization

Following this, we apply COMANDO to a realistic energy systems problem in Chapter 3, where we design an air-cooled geothermal organic Rankine Cycle (ORC) under consideration of the variable ambient temperature. For the resulting design and operation problem we consider realistic off-design behavior of the heat exchangers and turbine and employ

artificial neural networks (ANNs) as surrogate models for accurate fluid properties, and as computationally efficient representations of off-design parameters. We illustrate the issues with scenario analyses already discussed in Chapter 1, i.e., that designs resulting from problems considering only individual ambient temperatures are generally suboptimal or even infeasible for the entire operating range. However, in the considered case, the associated objective bounds guarantee global optimality of the the best feasible solution to the overall problem (where multiple ambient temperatures are considering) with sufficient accuracy. This is in contrast to the much weaker bounds, obtained directly from the state-of-the-art B&B solver addressing the overall problem.

Whereas our relatively simple solution approach ensures a sufficiently good solution in the specific case, considered in Chapter 3, this cannot always be expected. In general, reasonable solution times for realistic design and operation problems may only be achievable via suitable decomposition algorithms. However, existing algorithms, applicable to TSP in the general, nonconvex case may face certain practical and theoretical challenges. We therefore propose an alternative decomposition algorithm, that aims to address these potential issues in Chapter 4. The proposed algorithm combines simultaneous branching of multiple operational variables with a branch and bound search that explicitly considers both design and operational variables and is thus called **mu**lti-**se**ction **b**ranch & **b**ound (MUSE-BB). We prove finite termination with an ε^f-optimal solution, provide favorable convergence properties, relevant to the mitigation of clustering, and present promising numerical results that highlight the potential of MUSE-BB to outperform standard B&B algorithms on problems of the form TSP.

Finally, we draw conclusions and suggest directions for future research in Chapter 5.

2. COMANDO: Modeling Energy Systems for Optimal Design and Operation

In this chapter we consider the modeling of energy systems with a primary focus on optimizing their design and operation. To this end, we propose a next-generation ESMF for **c**omponent-**o**riented **m**odeling and optimiz**a**tion for **n**onlinear **d**esign and **o**peration (COMANDO), which we implemented in an open source Python package (COMANDO Repository). COMANDO borrows a generic, nonlinear representation of mathematical expressions and features for algorithm development from AMLs, and the representation of differential equations and more general system model aggregation from differential-algebraic modeling frameworks (DAMFs) such as gPROMS (Process Systems Enterprise, 1997–2023), MODELICA (Elmqvist and Mattsson, 1997), or DAE Tools (Nikolić, 2016). While the vast majority of existing ESMFs are implemented on top of an AML, COMANDO is based on the computer algebra systems SymPy (Meurer et al., 2017) and symengine (Čertík et al., 2019). These systems provide data structures for representing and manipulating mathematical expressions. These features facilitate the creation of automatic reformulation routines (e.g., automatic linearization), custom interfaces to AMLs or solvers, and user-defined solution algorithms.

With this combination of features, COMANDO incorporates flexible nonlinear and dynamic modeling into the modularity of an ESMF. Among other applications this enables the formulation and solution of two-stage stochastic programming problems for combined design and operation of energy systems under uncertainty. While this is the principal use-case of COMANDO, the framework can also be employed in many other settings such as, e.g., multi-objective optimization, scenario-analysis (i.e., sequential design and operation), or purely operational applications such as optimal scheduling and control, (stochastic) model-predictive control, or simulation.

This chapter is structured as follows: In Section 2.1, we give a brief review of the state of the art in optimization-based energy-system design and operation and identify the need for an open-source tool, dedicated specifically to the technical design and operation of energy systems. To this end, we present COMANDO in Section 2.2. In Section 2.3, we present four case studies highlighting important features of COMANDO. Section 2.4 concludes this chapter.

2.1. Optimization-based energy system design and operation

As discussed in Chapter 1, it is possible to cast the combined optimization of design and operation of energy systems as a two-stage stochastic programming problem of the form TSP. In Section 2.1.1 we present a variant of TSP that is specifically tailored towards combined design and operation problems, by additionally incorporating differential equations. This generalization additionally enables the representation of system dynamics, making the resulting problem formulation suitable for detailed operational problems such as optimal control.

In Section 2.1.2, we briefly summarize advantages and disadvantages of the three major classes of tools that can be used to formulate and solve variants of this problem, namely algebraic modeling languages (AMLs), energy system modeling frameworks (ESMFs) and differential-algebraic modeling frameworks (DAMFs). Subsequently we motivate the need for a new modeling framework, that specifically addresses the challenges of technical design and operation.

2.1.1. Optimal Design and Operation

In this chapter we consider a variant of the generic two-stage formulation TSP from Chapter 1 that is more specific to energy system *design and operation*. In particular, we explicitly account for dynamic effects that can be relevant for system operation and thus slightly adjust notation:

$$
\begin{aligned}
&\min_{\boldsymbol{x}} f_I(\boldsymbol{x}) + \sum_{s\in\mathcal{S}} w_s\, f^*_{II,s}(\boldsymbol{x}) \\
&\text{s.t. } \boldsymbol{g}_I(\boldsymbol{x}) \leq \boldsymbol{0} \\
&\quad \boldsymbol{h}_I(\boldsymbol{x}) = \boldsymbol{0} \\
&\quad f^*_{II,s}(\boldsymbol{x}) = \min_{\boldsymbol{y}_s(\cdot)} \; f_{II,s}(\boldsymbol{x}, \boldsymbol{y}_s(\cdot)) = \int_{\mathcal{T}_s} \dot{f}_{II}\big(\boldsymbol{x}, \boldsymbol{y}_s(t), \boldsymbol{p}_s(t)\big)\, \mathrm{d}t \\
&\qquad\qquad \text{s.t. } \boldsymbol{y}^{\mathrm{d}}_s(t=0) = \boldsymbol{y}^{\mathrm{d}}_{s,0} \\
&\qquad\qquad\quad \dot{\boldsymbol{y}}^{\mathrm{d}}_s(t) = \boldsymbol{\psi}\big(\boldsymbol{x}, \boldsymbol{y}_s(t), \boldsymbol{p}_s(t)\big) \\
&\qquad\qquad\quad \boldsymbol{g}_{II}\big(\boldsymbol{x}, \boldsymbol{y}_s(t), \boldsymbol{p}_s(t)\big) \leq \boldsymbol{0} \\
&\qquad\qquad\quad \boldsymbol{h}_{II}\big(\boldsymbol{x}, \boldsymbol{y}_s(t), \boldsymbol{p}_s(t)\big) = \boldsymbol{0} \\
&\qquad\qquad\quad \boldsymbol{y}_s(t) = [\boldsymbol{y}^{\mathrm{d}}_s(t), \boldsymbol{y}^{\mathrm{a}}_s(t), \boldsymbol{y}^{\mathrm{c}}_s(t)] \\
&\qquad\qquad\quad \boldsymbol{y}_s(t) \in \mathcal{Y}_s(t) \subset \mathbb{R}^{n_y} \times \mathbb{Z}^{m_y} \\
&\qquad\qquad\quad \mathcal{T}_s = [0, T_s] \\
&\quad \boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^{n_x} \times \mathbb{Z}^{m_x} \\
&\quad \mathcal{S} = \{s_1, s_2, \cdots, s_{|\mathcal{S}|}\}
\end{aligned}
\qquad \left.\begin{array}{l} \left.\begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \\ \end{array}\right\} \forall t \in \mathcal{T}_s \\ \\ \end{array}\right\} \forall s \in \mathcal{S} \\ \\ \end{array}\right. \tag{D\&O}
$$

The two-stage structure of (D&O) distinguishes between design- and operation-related variables, constraints, and objectives. We group design decisions into the vector $\boldsymbol{x}$ and operational decisions into one vector $\boldsymbol{y}_s(\cdot)$ (consisting of differential states, algebraic states, and controls, identified via the superscripts $^{\mathrm{d}}$, $^{\mathrm{a}}$, and $^{\mathrm{c}}$, respectively) for each scenario s, with associated probability of occurrence w_s. Further, the operational decisions are functions of time t from a continuous operating horizon $\mathcal{T}_s = [0, T_s]$ (in general, each scenario

may consider a different time horizon). Likewise, for different scenarios s the input data, i.e., the values of model parameters $\boldsymbol{p}_s(\cdot)$, may be functions of time t. The objective function of the first stage is comprised of design costs f_I and the expected value of the optimal operating costs. For a given design $\boldsymbol{x}$ and scenario s, the optimal operating costs $f^*_{II,s}$ correspond to the optimal objective value of the second stage. The operating costs are described by an integral over the operating horizon $\mathcal{T}_s$ of the momentary operating costs $\dot{f}_{II}$. The set of feasible design and operational decisions is described via constraints $\boldsymbol{g}_I$, $\boldsymbol{g}_{II}$, $\boldsymbol{h}_I$, and $\boldsymbol{h}_{II}$ (with an appropriate number of elements in $\boldsymbol{h}_{II}$, allowing for degrees of freedom), as well as bounds and integrality restrictions in the form of $\mathcal{X}$ and $\mathcal{Y}_s(t)$, with n and m corresponding to the number of continuous and discrete decisions, respectively. Additionally, for the subset of operational variables that correspond to differential states, $\boldsymbol{y}^{\mathrm{d}}_{s,0}$ denotes the associated initial state and $\boldsymbol{\psi}$ denotes the vector of state transition functions (right-hand side of the differential equations).

Formulation (D&O) covers both mixed design and operation problems, as well as pure operational problems (with fixed design decisions $\boldsymbol{x}$). If the values of w_s are interpreted as frequencies of occurrence for a certain operational setting, the corresponding scenarios can also be viewed as typical operating points or periods, as done e.g., in Yunt et al. (2008) and Baumgärtner et al. (2019a), respectively. Such scenarios can be derived from standardized reference load profiles, or via clustering of historical data (see, e.g., Schütz et al., 2018). While we do not consider this in the present work, additional constraints coupling different scenarios can be added to formulation (D&O), to incorporate long-term effects such as seasonal storage (see, e.g., Gabrielli et al., 2018; Baumgärtner et al., 2019b).

As discussed in Chapter 1, the two-stage formulation (D&O) can be cast into an equivalent single-stage formulation, also referred to as the *deterministic equivalent*, which can be solved with general-purpose solvers. While solvers interfaced from DAMFs, as well as some specialized dynamic optimization algorithms (Caspari et al., 2019; Scott and Barton, 2015), directly accept continuous-time problem formulations and take care of time-discretization internally, almost all solvers available via AMLs and ESMFs require discrete-time formulations as input. To obtain a discrete-time formulation, a particular discretization scheme must be chosen, and $\boldsymbol{y}_s(\cdot)$, $f_{II,s}(\boldsymbol{x}, \boldsymbol{y}_s(\cdot))$, and $\boldsymbol{\psi}\big(\boldsymbol{x}, \boldsymbol{y}_s(t), \boldsymbol{p}_s(t)\big)$ are replaced by corresponding discrete-time counterparts.

An alternative to solving the deterministic equivalent is to employ an algorithm capable of exploiting the special constraint structure of the two-stage formulation (D&O). Such an algorithm decomposes (D&O) into multiple subproblems that are solved iteratively to obtain increasingly tighter bounds on the solution of (D&O). As outlined in Chapter 1, many different decomposition algorithms are available, depending on the presence and location of nonlinearity, nonconvexity and integrality; for a concise overview, we also refer to Li and Grossmann (2019b).

2.1.2. Tools

Both deterministic equivalent formulations as well as suitable decomposition algorithms can be implemented in AMLs such as AMPL (Fourer, Gay, and Kernighan, 1990), GAMS (Bussieck and Meeraus, 2004), or AIMMS (Bisschop, 2006). In recent years, several AML extensions have been developed that can be leveraged for energy system modeling. In particular, stochastic programming related functionality has been incorporated widely, both in commercial AMLs such as AMPL (Fourer, Gay, and Kernighan, 1990; Valente et al.,

2009) and GAMS (Bussieck and Meeraus, 2004; Ferris et al., 2009), as well as in the open-source AMLs Pyomo (Hart, Watson, and Woodruff, 2011; Watson, Woodruff, and Hart, 2012) and JuMP (Dunning, Huchette, and Lubin, 2017; Huchette, Lubin, and Petra, 2014; Biel and Johansson, 2022). Further modeling constructs tailored towards special problem structures have been incorporated through block-oriented modeling (Friedman et al., 2013) in Pyomo and through Plasmo.jl (Jalving et al., 2017; Jalving, Cao, and Zavala, 2019) in JuMP. Finally, Pyomo.DAE (Nicholson et al., 2018) enables the direct representation of differential equations within optimization problems expressed in Pyomo and provides various options for automatic discretization. Through the combination of features offered by these extensions, newer AMLs are in principle well suited to model and optimize energy system design and operation. However, their abstract nature can complicate implementation, code maintenance, and re-use, and renders the resulting problem formulations difficult to comprehend. Development on the Pyomo AML has resulted in the modeling tool IDAES (Miller et al., 2018), which employs methodologies from process systems engineering with the aim of advancing fossil energy systems (IDAES homepage). In particular, IDAES provides models for thermal power plants and associated components. These systems are considered in the form of process flowsheets, i.e., components are modeled as control volumes with in- and outflows, whose steady-state and dynamic behavior can be specified via so-called property packages.

Compared to AMLs and their various extensions, ESMFs provide an even higher level of abstraction, allowing to model generic energy systems comprised of utilities for generating, converting, or storing different energy forms. This higher level of abstraction is commonly achieved via an interface layer on top of an AML that separates component and system modeling from problem formulation. In a first modeling step, models of energy system components, e.g., boilers, combined-heat-and-power units, or heat pumps are created. These component models contain variables, parameters and constraints specifying possible in- and outputs as well as the internal component behavior. In a second modeling step, system models are aggregated by specifying the connections between different components. Finally, component- and system-level constraints are combined with an objective, e.g., the minimization of total annualized cost (TAC) or global warming impact (GWI), yielding a problem formulation that can be passed to an appropriate solver.

The modular, object-oriented nature of modern ESMFs such as oemof (Hilpert et al., 2018) allows component and system models to be implemented as classes, inheriting reoccurring functionality, e.g., from generic models representing generation, transformation, storage or consumption of different energy commodities. Such inheritance allows for more structured modeling, thereby simplifying model maintenance and re-use compared to AMLs, e.g., through the creation of component libraries. However, the vast majority of ESMFs are based on either linear programming (LP) or mixed-integer linear programming (MILP) formulations, i.e., all participating functions must be linear in the decision variables $\boldsymbol{x}$ and $\boldsymbol{y}$. In such ESMFs, the user must provide linear approximations for all nonlinear expressions. While this is usually not considered a limitation in the context of *system analysis*, i.e., the principal focus of most ESMFs (cf. Pfenninger, Hawkes, and Keirstead, 2014), problems concerned with *technical design and operation* need to represent systems in more detail, often giving rise to nonlinearities that are difficult or impractical to linearize. In the presence of such nonlinearities, it is often sensible to use the original nonlinear equations or nonlinear surrogate models such as artificial neural networks (ANNs), as, e.g., in Schäfer et al. (2020), which however is not possible in MILP-based ESMFs.

Besides AMLs and ESMFs, differential-algebraic modeling frameworks (DAMFs) constitute a third class of tools that can be used to model energy systems. DAMFs also employ a component-oriented modeling approach, which, however, is more general than in a typical ESMF: In DAMFs, components may correspond to actual physical machinery or to a particular physical phenomenon (e.g., heat transfer) and can constitute subsystems, which are themselves composed of other components. Additionally, the information exchanged between components is not restricted to a particular kind of quantity, such as energy. DAMFs are particularly focused on detailed operational aspects, allowing for differential equations and nonlinear expressions within component models. They provide powerful features for operational *simulation* of the resulting models for which a fixed design is assumed. Design *optimization* is also possible in several DAMFs, (e.g., Smith, 1997; Pfeiffer, 2012), and the commercial tool gPROMS (Process Systems Enterprise, 1997–2023) even allows for the direct consideration of parametric uncertainty using formulations similar to Problem (D&O), (see, e.g., Bansal et al., 2000). In contrast, noncommercial, open-source tools such as Open-Modelica (Thieriot et al., 2011) or Optimica (Åkesson et al., 2010) are currently limited to a single set of operational data, impeding design optimization under uncertainty. Furthermore, DAMFs usually offer less freedom in the choice of problem formulation, solver or algorithm in comparison to AMLs. In particular, many tools employ gradient-based methods, (e.g., Pfeiffer, 2012; Navarro and Vassiliadis, 2014; Magnusson and Åkesson, 2015) yielding only local solutions, or heuristic global optimization methods, e.g., random search, genetic algorithms, or simulated annealing (Thieriot et al., 2011; Pfeiffer, 2012; Kim et al., 2018), which treat the system model as a black box and cannot reliably locate global solutions.

AMLs, ESMFs and DAMFs each exhibit strengths related to a particular aspect of modeling and optimizing energy system design and operation. ESMFs are tailored to energy systems modeling and offer a component-oriented approach that benefits model maintenance and re-usability. However, their principal focus is on system analysis. In particular, their restriction to LP or MILP formulations makes them less suited for applications concerned with technical design and operation. Both AMLs and DAMFs lift the restriction to (MI)LP formulations, but AMLs lack high-level component-oriented abstractions for generic energy systems and DAMFs lack control over the choice of problem formulation and optimization algorithm. We therefore propose a next-generation ESMF that allows for flexible, component-oriented modeling, including nonlinear and differential-algebraic formulations, parametric uncertainty, and the possibility to specify specialized solution algorithms. Its basic structure is presented in the following Section. A summary of the above discussion, highlighting the roles of each tool class and their influence on COMANDO is given in Tab. 2.1.

Tab. 2.1. Overview of the three tool classes that inspired COMANDO: algebraic modeling languages (AMLs), energy system modeling frameworks (ESMFs), and differential-algebraic modeling frameworks (DAMFs)

tool class	representative examples	typical domain of application	features adopted in COMANDO
AMLs	• AMPL (Fourer, Gay, and Kernighan, 1990) • GAMS (Bussieck and Meeraus, 2004) • AIMMS (Bisschop, 2006) • PYOMO (Hart, Watson, and Woodruff, 2011) • JuMP (Dunning, Huchette, and Lubin, 2017)	development of detailed, application-specific (MI)LP/(MI)NLP problem formulations for arbitrary applications and specialized solution routines	• free choice of modeling approach • possibility to specify alternative problem formulations • development of user-defined algorithms
ESMFs	• MESSAGE (Schrattenholzer, 1981) • MARKAL/TIMES (Fishbone and Abilock, 1981; Loulou and Labriet, 2007) • eTransport (Bakken, Skjelbred, and Wolfgang, 2007) • OSeMOSYS (Howells et al., 2011) • Temoa (Hunter, Sreepathi, and DeCarolis, 2013) • calliope (Pfenninger and Keirstead, 2015) • urbs (Dorfner, 2016) • ficus (Atabay, 2017) • oemof (Hilpert et al., 2018) • PyPSA (Brown, Hörsch, and Schlachtberger, 2018) • Switch 2.0 (Johnston et al., 2019)	system analysis (superstructure optimization, capacity expansion planning) for large scale national/international energy systems, typically using (MI)LP formulations	• component-oriented modeling • focus on energy systems • separation of modeling and problem formulation • open-source availability
DAMFs	• gPROMS (Process Systems Enterprise, 1997–2023) • MODELICA (Elmqvist and Mattsson, 1997) • DAE Tools (Nikolić, 2016)	detailed operational simulation; varying degrees of optimization capabilities, typically local solutions to NLP formulations	• modeling with differential equations • generic bidirectional connectivity • modularity for definition of subsystems

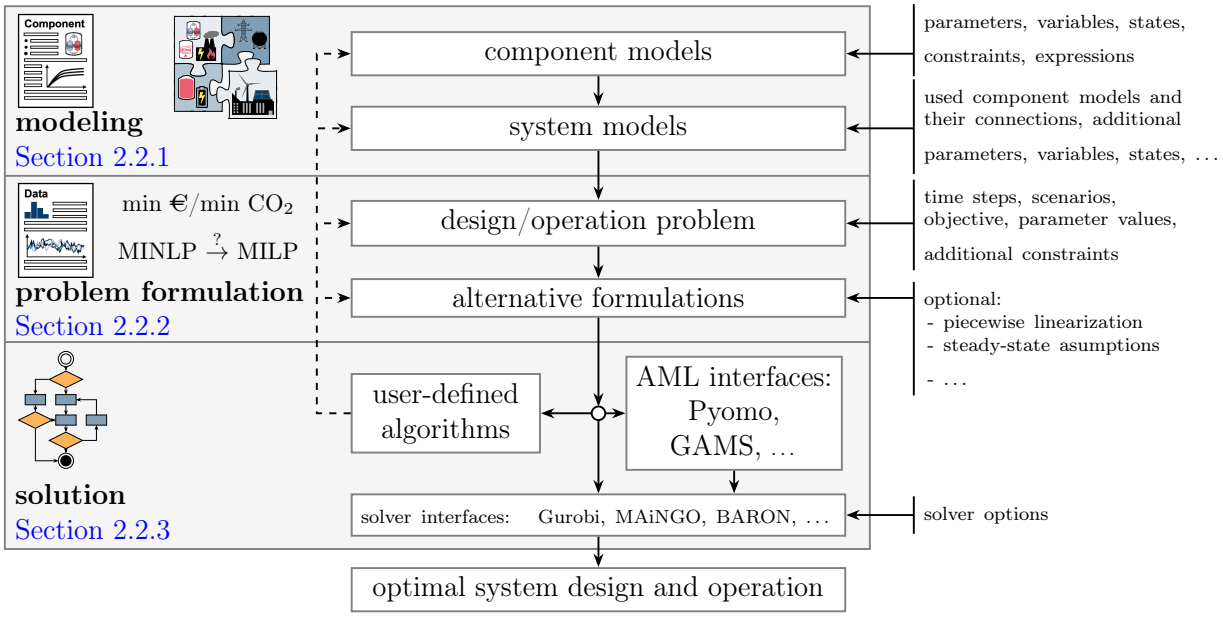


Fig. 2.1. Workflow for modeling, problem formulation, and optimization using COMANDO.

2.2. The COMANDO ESMF

The goal of COMANDO is to provide an open-source ESMF which allows to generate detailed models of energy system components, including differential-algebraic and nonlinear elements, and aggregate them to system models with the primary purpose of optimization. Traditional ESMFs are typically oriented towards techno-economic analysis of systems at national or international scales, where (MI)LP formulations are an asset that ensures computational tractability. In contrast, COMANDO is oriented towards the technical design and operation of small- to medium-scale systems, e.g., district energy systems, industrial sites, or energy conversion processes. At these scales, investigation of realistic component and system behavior is possible via the consideration of technically relevant effects such as part-load and dynamic behavior.

Unlike most ESMFs, which are commonly based on an AML, COMANDO is implemented as a flat, optimization-specific modeling layer on top of the computer algebra system SymPy (Meurer et al., 2017). This choice provides: i) data structures for the mathematical expressions used to describe components and systems, as well as ii) several routines useful for creating automatic reformulations and user-defined algorithms, such as automatic differentiation, substitution of expressions or solution of nonlinear systems of equations. As a modeling framework, COMANDO itself does not provide any specialized solution methods. Instead, it allows for component-oriented modeling at a high level of abstraction, while at the same time granting users access to low-level data structures. This allows for both intuitive modeling, as well as advanced use cases such as problem reformulations and the development of user-defined algorithms, simplifying the development of tailored solution approaches. During the development of COMANDO, we made an effort to maximize chances of its adoption by following best-practices for code development. This includes the creation of automated unit and integration tests, provision of documentation (both in the source code and as a standalone document (COMANDO Documentation), and

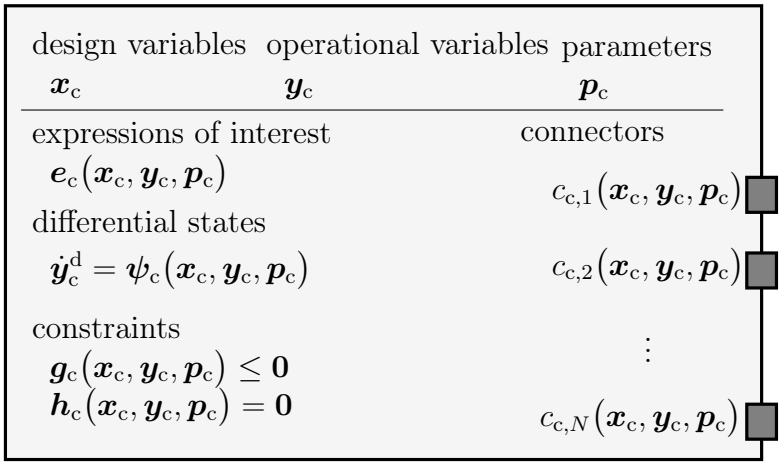


Fig. 2.2. Structure of a generic component c in COMANDO. Mathematical expressions are specified based on symbols that are either design variables ($\boldsymbol{x}_\mathrm{c}$), operational variables ($\boldsymbol{y}_\mathrm{c}$), or parameters ($\boldsymbol{p}_\mathrm{c}$). These expressions can be kept for later reference ($\boldsymbol{e}_\mathrm{c}$), constitute the right-hand side of differential equations ($\boldsymbol{\psi}_\mathrm{c}$), form part of algebraic constraints ($\boldsymbol{g}_\mathrm{c}, \boldsymbol{h}_\mathrm{c}$), or describe possible in- and/or outputs through connectors ($c_{\mathrm{c},i}$).

the inclusion of the full code for running the case studies as detailed usage examples. In order to further encourage adoption, we provide a generic parsing routine to translate individual COMANDO expressions to alternative textual or object-oriented representations, allowing users to easily link COMANDO to other software. An overview of the structure of COMANDO and the typical workflow of modeling, problem formulation and optimization is given in Fig. 2.1.

In Section 2.2.1 we describe the process of creating models for components and systems in COMANDO. Section 2.2.2 provides details on how optimization problems can be created from a system model and how alternative formulations of these problems can be obtained. Finally, the different options for solving the formulated problems are given in Section 2.2.3.

2.2.1. Modeling Process

The goal of the modeling phase is to generate a model describing the behavior of a given energy system. For the creation of such a system model, models for its constituting components as well as information on their connectivity are required.

We begin with the description of component models, which are used to represent elementary parts of an energy system. Fig. 2.2 depicts the structure of the `Component` class used for that purpose. A model of a component c consists of several types of mathematical expressions, given in symbolic form. Following the notation introduced in Section 2.1, the expressions describing the component contain different symbols corresponding to quantities which are either *parameters* ($\boldsymbol{p}_\mathrm{c}$), i.e., placeholders for values that are assumed to be given before an optimization, or design or operational *variables* ($\boldsymbol{x}_\mathrm{c}$ and $\boldsymbol{y}_\mathrm{c}$, respectively), i.e., placeholders for scalar and vector values that are to be determined during optimization.

To instantiate a `Component`, a unique name must be provided, which serves as an identifier for the component. The names of parameters, variables, and constraints associated to the component are prepended with this identifier, in order to distinguish quantities from different instances of the same component model. The `Component` class can either be used

directly or subclassed to specify specialized component classes with custom behavior. To create and add symbols to a component, the `Component` class provides three methods:

- `make_parameter`,
- `make_design_variable`, and
- `make_operational_variable`.

All three methods require a name for the symbol that is used to represent the quantity. The methods for creating variables provide optional arguments for the specification of variable bounds, domain (integer/real) and a scalar value for initialization, while the parameter creation method only provides a single optional argument for the specification of its value. Note that time- and scenario-specific values for operational quantities are set in the problem formulation phase after the time and scenario structure, has been specified, see Section 2.2.2.

Based on variables and parameters, mathematical expressions can be formed using the overloaded Python operators `+`, `-`, `*`, `/`, `**`, or any of the functions implemented in SymPy (e.g., exp, log, trigonometric, and hyperbolic functions). Any intermediate expressions $\boldsymbol{e}_\mathrm{c}$ that are of interest can be assigned an identifier and stored in the component using the `add_expression` method. These expressions can simply be used for evaluation or as parts of more complex expressions, e.g., system-level constraints, or an objective function, cf. Section 2.2.2. Vectors $\boldsymbol{g}_\mathrm{c}$ and $\boldsymbol{h}_\mathrm{c}$ contain inequality and equality constraints associated to the component c and their elements can be specified using the methods

- `add_le_constraint`,
- `add_eq_constraint`, and
- `add_ge_constraint`.

Each of these methods takes two expressions and an optional name for the resulting relation as arguments. Explicit distinction into first and second stage expressions and constraints is not necessary and occurs automatically, based on the symbols present in the respective expressions.

Dynamic behavior can be represented by specifying right-hand side expressions $\boldsymbol{\psi}_\mathrm{c}$ for the time derivatives of differential states $\boldsymbol{y}_\mathrm{c}^\mathrm{d}$ (recall that $\boldsymbol{y}_\mathrm{c}^\mathrm{d}$ constitutes a subset of the operational variables $\boldsymbol{y}_i$). Previously created operational variables may be declared differential states using the `declare_state` method or differential states may be created directly using the `make_state` method. The first method requires an existing variable and an expression, corresponding to entries of the vectors $\boldsymbol{y}_\mathrm{c}^\mathrm{d}$ and $\boldsymbol{\psi}_\mathrm{c}$ as mandatory arguments and allows for the specification of an initial state as well as bounds and an initial guess for the value of the derivative. The method results in the creation of a new operational variable, corresponding to an element in $\dot{\boldsymbol{y}}_\mathrm{c}^\mathrm{d}$, and an equality constraint, linking the time derivative with the given expression in $\boldsymbol{\psi}_\mathrm{c}$. An explicit relation between the state and its derivative is not specified at this point, as it depends on the desired time-discretization which is handled by the solution interfaces, cf. Section 2.2.2. The `make_state` method creates a new operational variable corresponding to the differential state and then calls `declare_state`.

To allow for the aggregation of components to systems, individual expressions in $\boldsymbol{c}_i$ can be assigned to connectors (cf. Fig. 2.2). Connectors are generally bidirectional, but may be specified to only allow for in-, or output. In- and output connectors restrict the assigned expression to a nonnegative or nonpositive range, respectively.

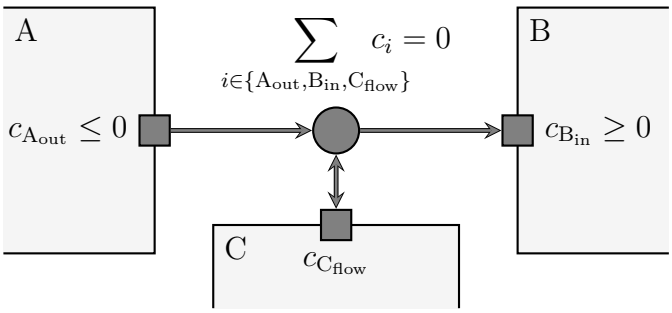


Fig. 2.3. A connection formed by connecting three connectors to a bus: The components A, B, and C each define a connector for a particular quantity. The connectors of A and B are marked as outputs and inputs, respectively, restricting the sign of the associated expression, while the connector of C is not restricted. The connection of $\mathrm{A_{out}}$, $\mathrm{B_{in}}$, and $\mathrm{C_{flow}}$ via a bus results in the creation of a balance constraint in the system model. This graphical notation is also used for the case studies in Section 2.3.

A system model can be created as an instance of the `System` class, whose instantiation again requires a unique label that serves as an identifier. Optionally, a list of components and the connections between them can be passed to the constructor of the `System` class. Each connection is specified via a label and a list of associated connectors. The connectors are connected to a "bus" at which the quantities associated to them are balanced and a corresponding constraint is created automatically, see the graphical notation in Fig. 2.3, which is also used for the case studies in Section 2.3. The elementary connections provided by COMANDO's `System` class, handle only simple balance equations. More complicated connectivities such as mixing streams with different temperatures, concentrations or other qualities are most naturally implemented as a dedicated component within COMANDO. Instead of specifying the complete structure during construction of a `System` instance, components and connections can also be added sequentially via corresponding methods, allowing for procedural model generation. As in DAMFs, a nested creation of systems from subsystems is possible by exposing connectors of individual components or extending existing connections via additional connectors. For instance, a neighborhood can be represented as a system composed of buildings as subsystems, which are in turn composed of heating, cooling and power equipment. As with component models, system models can be assigned their own variables, parameters, expressions and constraints describing their behavior. These two features are accomplished by letting the `System` class inherit from the `Component` class. The system superstructure can be considered explicitly by including appropriate design decisions within component models. More advanced approaches where the superstructure is not specified a-priori, e.g., superstructure-free synthesis (Voll et al., 2012), or automated superstructure generation and expansion (Voll et al., 2013), can be easily incorporated in the form of user-defined algorithms.

2.2.2. Problem Formulation

Based on a system model, different kinds of optimization problems considering system design and/or operation can be created. To this end, COMANDO provides the `Problem` class, instances of which can be created by the `create_problem` method of the `System` class.

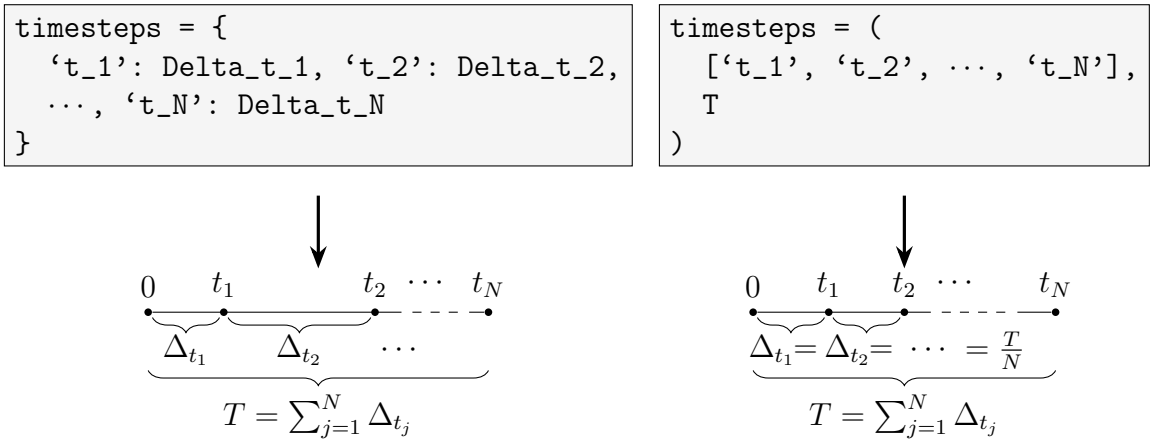


Fig. 2.4. Alternative ways to specify time steps for a particular scenario: For variable length an ordered mapping (left) and for constant length a list and the total length (right) can be specified. If multiple scenarios with different time structures are to be considered, one such description is given per scenario.

As the system model defines a constraint set which is parameterized by the parameters $\boldsymbol{p}$, only the objective terms f_I and $\dot{f}_{II}$ as well as a time and scenario structure and appropriate data (i.e., values for the parameters $\boldsymbol{p}$) need to be specified in the `create_problem` method to obtain a complete problem formulation, corresponding to (D&O). Note that the user may decide which units to use for data and time steps but must ensure they match. The units used for the case studies in Section 2.3 are given in the nomenclature at the beginning of this thesis.

To define the objective terms, the `System` class provides the `aggregate_component_expressions` method. For a given expression identifier, it returns the sum of all expressions stored under that identifier in the individual components. The resulting expressions can be used for the objective terms f_I and $\dot{f}_{II}$, depending on whether they consist exclusively of first stage (i.e., scalar) quantities or not. A second use for the `aggregate_component_expressions` method is to create expressions for system-level constraints involving contributions from multiple components.

The time and scenario structure is specified in terms of the considered scenarios $s \in \mathcal{S}$ and the corresponding discretized time horizons $\widehat{\mathcal{T}}_s$. The $\widehat{\mathcal{T}}_s$ are required by COMANDO's solver or AML interfaces for the automatic discretization of the differential equations. If more than one operational scenario is considered, the different scenarios can either be specified as a list of M scenario identifiers, corresponding to scenarios with probability $1/M$, or by a series of scenario identifiers and associated weights w_s. In the latter case, the weights are not required to sum to one, allowing for a more general weighting. Similarly, individual time points for each time horizon are either specified via a mapping of time point labels t to the corresponding lengths $\Delta_{s,t}$ or in the case of equidistant time steps via a list of labels and an end-time T_s, see Fig. 2.4. If the time horizons are identical for all scenarios, a single time horizon can be specified, otherwise, one specification per scenario is required.

Parameter values corresponding to the resulting time and scenario structure can be specified during problem creation and may later be updated using the `data` attribute of the `Problem` instance. Similarly, design and operational variable values can be updated using the `design` and `operation` attributes, respectively. Values for design variables must

be scalar while values for parameters and operational variables may be provided as scalars or as time- and/or scenario-dependent data.

After the abovementioned steps, a problem in the form of (D&O) is fully specified. However, it may be desirable to adapt the original problem formulation in different ways. Adaptations to the problem formulation range from simply adding further constraints to the reformulation of expressions in the problem. One generic reformulation routine implemented in COMANDO is the automatic linearization of arbitrary continuous multivariate expressions via convex-combination or multiple-choice linearization (Vielma, Ahmed, and Nemhauser, 2010). More generally, custom reformulations may be created making use of existing algorithms provided by SymPy (Meurer et al., 2017), e.g., for automatic differentiation, analytic solution of different kinds of nonlinear equation systems, or symbolic substitution of subexpressions. Note that reformulations do not have to result in approximations but can also be used to create alternative formulations that possess better properties than the original one, e.g., tighter relaxations for deterministic global optimization.

2.2.3. Problem Solution

A fully specified problem formulation can be directly passed to a suitable solver or to an AML. In this step, the problem structure and data are translated from the COMANDO representation to a new representation, matching the syntax of the target solver or AML. For this purpose, COMANDO contains a generic parsing routine that can be used to create new interfaces based on target-specific representations of the symbols and operations occurring within the different expressions of the problem formulation. Interfaces may be text-based, resulting in an input file for a solver or AML, or they can be object-oriented, resulting in a translation of the problem formulation using the target-API. Currently implemented interfaces are:

- text-based:
  - BARON (Sahinidis, 2020) (solver)
  - GAMS (Bussieck and Meeraus, 2004) (AML)
  - MAiNGO (Bongartz et al., 2018) (solver)
- API-based:
  - Pyomo (Hart, Watson, and Woodruff, 2011) (AML)
  - Pyomo.DAE (Nicholson et al., 2018) (AML)
  - Gurobi (Gurobi Optimization, LLC, 2020) (solver)
  - MAiNGO (Bongartz et al., 2018) (solver)

All of these interfaces provide methods to solve the deterministic equivalent formulation of Problem (D&O) with a given set of options, and to write back the obtained results to COMANDO. Note that a problem formulation may contain differential equations if states were defined in the component or system model. Since most solvers and AMLs do not support differential equations, the corresponding interfaces can specify different schemes for automatic time discretization. All existing interfaces implement implicit Euler discretization. More advanced schemes are available through the Pyomo.DAE interface.

Instead of directly solving a problem, it can also be addressed with a user-defined algorithm. User-defined algorithms can range from simple preprocessing routines based on the system model and available data to more advanced methods, such as decomposition techniques, commonly used in stochastic programming (see, e.g., Li and Grossmann, 2019b).

The architecture of COMANDO allows for manipulation at the level of component and system models as well as at the level of the resulting optimization problems. In particular the `Problem` class can also be used to specify the sub-problems that may occur within user-defined algorithms, allowing them to be passed to any of the available interfaces.

2.3. Case Studies

We now demonstrate key features of COMANDO in four case studies, which are illustrative of the kinds of design and operation problems we address with COMANDO. The case studies focus on different aspects of energy systems and vary in their approaches for modeling the considered systems and their components. The complete source code for all case studies can be found in the `examples` directory of the COMANDO Repository.

The first case study, based on our previous work (Voll et al., 2013; Sass and Mitsos, 2019), consists of the greenfield design and operation of an industrial energy system considering both economic and environmental impact. The component models account for nonlinearities in part-load behavior and investment cost, and differential equations for the state of charge of battery and thermal energy storage units, resulting in a mixed-integer dynamic optimization (MIDO) problem. Here, the automatic implicit Euler discretization as well as the automatic linearization implemented in COMANDO are employed to obtain a MILP formulation, and a simple user-defined algorithm for multi-objective optimization is demonstrated.

In the second case study, the operation of a simple building energy system is optimized, considering forecasts for electricity price and ambient temperature. The system model makes use of differential equations to describe the thermal behavior of the building, allowing to represent dynamic aspects of demand response via a MIDO problem. The interface to Pyomo.DAE (Nicholson et al., 2018) is used to apply orthogonal collocation on finite elements as an advanced time discretization method.

The third case study is a variation of the benchmark problem from (Saelens et al., 2019), integrating low-temperature waste heat into a district heating network via heat pumps. The explicit consideration of thermal losses and temperatures at different points of the network results in a nonconvex mixed-integer quadratically-constrained quadratic programming (MIQCQP) problem. For the implementation in COMANDO, repeated structures within the system are abstracted via subsystems, allowing for re-use of the models and reducing modeling effort. A stochastic formulation considering multiple operational scenarios based on clustered historical data is solved to obtain an optimal system design.

The fourth case study is a reimplementation of our previous work (Huster, Schweidtmann, and Mitsos, 2019), where the power production of an organic Rankine cycle is maximized, based on Ghasemi et al., 2013b. The detailed thermodynamic behavior of the working fluid is described via artificial neural networks (ANNs), capable of predicting fluid properties with high accuracy. The ANNs result in a highly nonconvex, but reduced-space NLP formulation that can be solved to global optimality with our in-house solver MAiNGO (Bongartz et al., 2018).

All case studies are solved on a desktop PC with an i7-8700 CPU (3.20GHz), 32 GB RAM, running Windows 10 Enterprise LTSC. An overview of the presented case studies and their key characteristics is given in Tab. 2.2.

Tab. 2.2. Overview of the presented case studies

case study	system structure	problem class (reformulations)	problem type	operational horizon representation	demonstrated features
industrial energy system (Section 2.3.1)	superstructure with CHP subsystem, 15 instances of 11 components	MINLP (MILP, NLP)	design	6 scenarios: 4 typical days with 4 time steps of varying length, each, and two isolated time steps representing extreme demands, implicit Euler time discretization	• superstructure optimization • automatic linearization • user-defined algorithm • re-use of model for multiple problem formulations
building demand response (Section 2.3.2)	9 instances of 4 components	MIDO (MILP)	operation	24 h horizon with 15 min time steps, each with 4 collocation points	• abstract components • modeling with differential equations • advanced time discretization via collocation in Pyomo.DAE (Nicholson et al., 2018)
low-temperature district heating network (Section 2.3.3)	superstructure with 9 instances of 2 subsystems (linking, consumer group), 26 instances of 6 components	MIQCQP	design	11 scenarios, each representing a static operating point	• modular model generation • superstructure optimization • stochastic programming
organic Rankine cycle (Section 2.3.4)	8 instances of 4 components	NLP	operation	single operating point	• hybrid modeling with ANNs • reduced space formulation • integration with different solver/AML interfaces

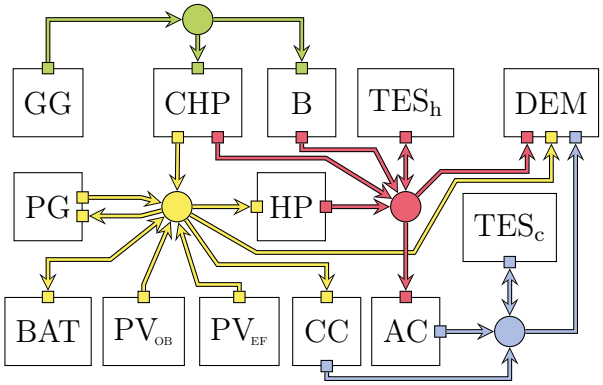


Fig. 2.5. Superstructure for the industrial energy system case study: gas-grid (GG), power-grid (PG), boiler (B), combined heat-and-power unit (CHP), compression chiller (CC), absorption chiller (AC), heat pump (HP), photovoltaic units on office buildings (PV_{OB}) and on experimental facilities (PV_{EF}), thermal energy storage for hot water (TES_h) and cooling water (TES_c), a battery (BAT), and a demand (DEM). Natural gas is shown in green, electricity in yellow, hot water in red, and cooling water in blue.

2.3.1. Case Study 1: Greenfield design of an industrial energy system

This case study is inspired by our previous work (Sass et al., 2020). For demonstration, we consider a simpler system, allowing only up to one component of each type. We make use of inheritance to abstract common model aspects of conversion and storage components into generic classes and then derive specialized variants that implement more specific behavior. Furthermore, we take advantage of automatic linearization and discretization routines to obtain MILP problems from the originally dynamic and nonlinear component models of Sass et al. (2020).

The industrial energy system needs to satisfy given time-dependent demands for heating, cooling, and electricity with minimal total annualized costs (TAC) and global warming impact (GWI). To satisfy these demands, multiple conversion and storage components are available in the superstructure of the system (Fig. 2.5). For self-containment, we briefly repeat the description of the conversion and storage components here. More detailed information can be found in Sass et al. (2020) and in the source code for this case study, available in the COMANDO Repository.

The conversion components $c \in \mathcal{C}^{conv}\{AC, B, CC, CHP, HP\}$ (cf. Fig. 2.5) are modeled with nonlinear investment cost and part-load efficiency curves. Additionally, minimal part-load requirements are considered by introducing binary variables. The investment cost reflect decreasing marginal investment costs $C_{I,c}$ with increasing nominal component output $\dot{E}_{nom,c}$, i.e.,

$$C_{I,c} = C_{ref,c}\dot{E}_{nom,c}^{M_c} \quad \forall c \in \mathcal{C}^{conv}, \tag{2.1}$$

where $C_{ref,c}$ and M_c are technology-specific parameters. The part-load efficiency η_c is expressed via a base efficiency multiplied with a rational function of the part-load fraction $\dot{E}_{o,c}/\dot{E}_{nom,c}$, and describes the relationship of input $\dot{E}_{i,c}$ and output $\dot{E}_{o,c}$:

$$\dot{E}_{o,c} = \eta_c \dot{E}_{i,c} \quad \forall c \in \mathcal{C}^{conv} \tag{2.2}$$

The HP and CHP models have variable base efficiencies that depend on temperatures and the nominal size, respectively. We create a generic conversion component class with an unparametrized nonlinear efficiency and investment cost model (Eqs. (2.1) and (2.2)).

From this conversion component class, we derive the individual conversion technologies as subclasses. Three instances of the CHP model with different ranges for the nominal size are considered, accounting for the size-dependence of the conversion efficiencies for heat and electricity. The three CHP models are aggregated into a subsystem which enforces that at most one of them is built. The subsystem can then be incorporated into other system models like any other component.

The storage components $c \in \mathcal{C}^{\text{sto}} = \{\text{BAT}, \text{TES}_\text{h}, \text{TES}_\text{c}\}$ are modeled with the differential equation

$$\frac{\mathrm{d}E_\text{c}}{\mathrm{d}t} = \eta_{\text{i,c}}\dot{E}_{\text{i,c}} - \frac{1}{\eta_{\text{o,c}}}\dot{E}_{\text{o,c}} - \frac{1}{c_{t,\text{c}}}E_\text{c} \quad \forall \text{c} \in \mathcal{C}^{\text{sto}}, \tag{2.3}$$

where the state E_c is the stored energy, $\eta_{\text{i,c}}$ and $\eta_{\text{o,c}}$ are constant charging and discharging efficiencies, $\dot{E}_{\text{i,c}}$ and $\dot{E}_{\text{o,c}}$ are the charging and discharging rates, and $c_{t,\text{c}}$ is a time constant describing self-discharging. As with the conversion components, we create a generic storage component class and derive technology-specific sub-classes, e.g., batteries. For each component we additionally consider a binary variable and associated constraints, representing whether the component is built or not.

We use the aggregated data from the supplementary material of Sass et al. (2020), which originate from clustering a full year of data for demands, weather, prices, and global warming impacts via the method described in Bahl et al., 2018. The aggregated data represent the full year via four typical days, each with four time steps of varying lengths (between 1 and 17 hours), and two isolated time points of length zero, representing peak heating and cooling demands. For a similar design problem, Bahl et al., 2018 showed that even coarse time resolutions such as this one provide optimal objective values, sufficiently close to those obtained with a full year at hourly resolution. In COMANDO we can consider such a time structure via six scenarios, corresponding to the four typical days and the two isolated time points for peak demands. The scenarios corresponding to typical days are weighted by number of days associated to them during clustering, and the scenarios for peak demands are assigned a weight of zero, i.e., they have no effect on the objective but are considered for feasibility, cf. formulation (D&O).

Due to the storage dynamics Eq. (2.3), problems derived from this system model will be MIDO problems. In our previous work (Sass et al., 2020), we manually implemented the MILP formulation resulting from explicit Euler discretization and a case-specific linearization in GAMS. As this process and subsequent changes are labor-intensive and error-prone, we instead make use of COMANDO's automatic routines for discretization and piecewise linearization.

The augmented ε-constraint method (Mavrotas, 2009) is implemented as a user-defined algorithm, in which two design optimization problems with either TAC or GWI as objective function are repeatedly solved. For the solution of the two problems we use Gurobi 9.1.1 with a relative optimality tolerance of 1%. Generating 8 designs from the Pareto front for TAC and GWI, shown in Fig. 2.6, takes about 3.6 hours. Note that a Pareto-optimal design can only improve upon one of the two objectives by worsening the other.

The total GWI can be reduced by 50% (from 1.152 to 0.577 kt/a) when accepting a fourfold increase in TAC (from 0.539 to 2.6 Mio. €) (Fig. 2.6, bottom). Solutions with lower TAC are characterized by small component capacities with lower investment costs, whereas solutions with lower GWI rely on large conversion and storage components (Fig. 2.6 inner bars, top). As these results were obtained with a linearization of the original

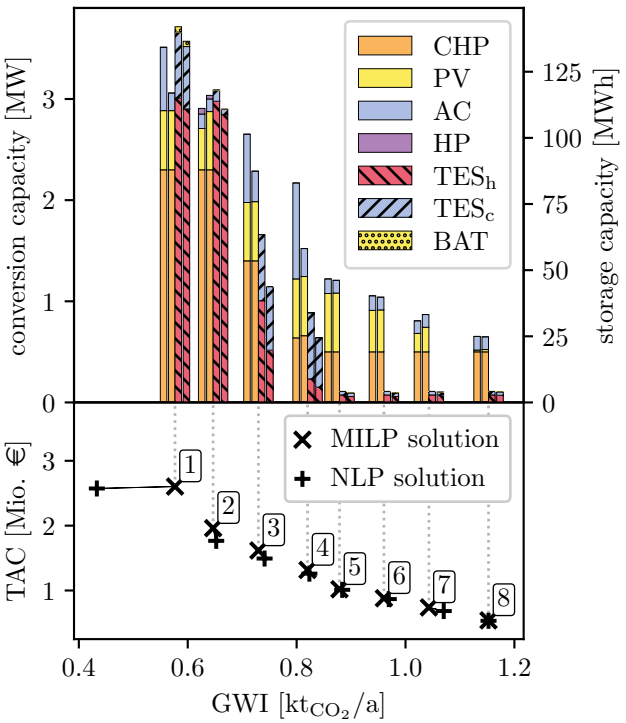


Fig. 2.6. Bottom: eight Pareto-optimal designs, determined from multi-objective optimization regarding total annualized cost (TAC) and global warming impact (GWI). Top: corresponding capacities of conversion (left) and storage components (right) from the MILP (inner bars) and NLP (outer bars) formulations. CHP: Combined heat and power unit, PV: photovoltaic array, AC: absorption chiller, HP: heat pump, TES$_h$: hot thermal energy storage, TES$_c$: cold thermal energy storage, BAT: battery. Note that boilers and compression chillers are not part of any design and thus excluded from the legend.

model, they are only approximate and the corrsponding designs may not be feasible with respect to the nonlinear model.

However, correcting the infeasibilities is straightforward in COMANDO as the original, nonlinear model formulation is available. We first obtain the MINLP problem resulting from implicit Euler discretization of the original formulation with TAC as the objective. We then repeat the multi-objective optimization with the same algorithm but using the MINLP formulation. For each iteration, we set the appropriate upper bound on GWI and fix binary variables to the values of the corresponding MILP solution, obtaining an NLP formulation. The values of the remaining variables are used as an initial point and the resulting formulation is passed to BARON 20.10.16 using default options, except for a relative optimality tolerance of 1% and a time limit of one hour for the subproblems.

In three cases the subproblems are terminated due to the time limit (with 3.5% relative gap for the TAC minimization of iteration 3 and 4, and 7.5% relative gap for the GWI correction of iteration 3). The remaining subproblems take at most 78 s to be solved to the desired optimality. Thus, all cases result in a design and an operational strategy that are feasible with respect to the original nonlinear formulation. The resulting solutions exhibit slightly lower TAC values and slightly higher GWI values than their MILP counterparts, with the exception of iteration 1, where the GWI value is 25% lower than for the MILP approach (433 t/a vs. 577 t/a). The corresponding designs can be seen in the outer bars in Fig. 2.6 (top). While the MILP and NLP solutions of iterations 2 and 5–8 are similar, iterations 1, 3 and 4 exhibit larger conversion components and smaller storages in the NLP case. In summary, the approach provides MINLP-feasible system designs that allow a trade-off between the TAC and GWI of the resulting system.

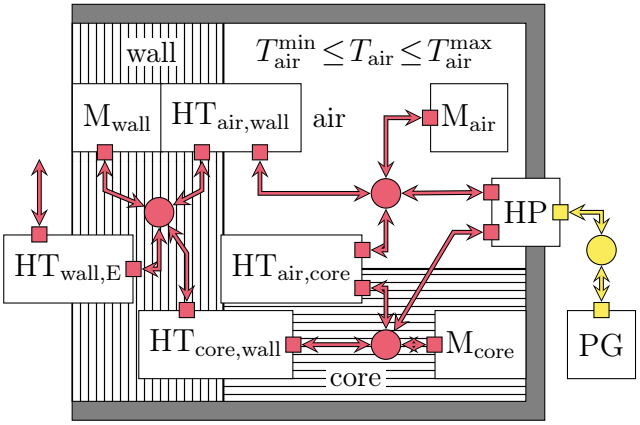


Fig. 2.7. Structure of the considered building energy system and implementation in COMANDO: three instances of the thermal mass class (M_{air}, M_{core}, M_{wall}), four instances of the heat transfer class ($HT_{air,wall}$, $HT_{wall,E}$, $HT_{air,core}$, $HT_{core,wall}$), heat pump (HP), and power grid (PG). Red arrows represent heat flows and yellow arrows electric power flows.

2.3.2. Case Study 2: Demand response of a building energy system

To illustrate how to formulate and solve optimization problems with more pronounced dynamic effects in COMANDO, we model an illustrative building energy system. The system is heated by a heat pump (HP) and is capable of performing load shifting via concrete core activation, i.e., a concrete core with a high thermal inertia can be heated directly. We investigate a demand response (DR) case, where we optimize the operation of the building energy system over the horizon of one day with given profiles for electricity price and ambient temperature.

The considered building energy system consists of three thermal zones: air, outside wall, and concrete core. Occupant comfort has to be ensured by maintaining the air temperature between minimal and maximal temperatures T_{air}^{min} and T_{air}^{max}, respectively. To do so, the air in the room can be heated via a direct heat flow to the air $\dot{Q}_{air,i}$, or indirectly through the concrete core, which can be heated via the heat flow $\dot{Q}_{core,i}$. We consider a zero-dimensional model of each thermal zone. For instance, the energy balance of the air zone is given by

$$\rho_{air} V_{air} c_{p,air} \frac{dT_{air}}{dt} = \dot{Q}_{core,air} - \dot{Q}_{air,wall} + \dot{Q}_{air,i}, \tag{2.4}$$

where $T_{air}, V_{air}, \rho_{air}$, and $c_{p,air}$ are the air temperature, volume, density, and specific heat capacity, respectively, and $\dot{Q}_{core,air}$ and $\dot{Q}_{air,wall}$ are heat exchange flows with the adjacent zones. The heat flow $\dot{Q}_{A,B}$ between two zones A and B is calculated depending on the temperatures T_A and T_B, the area $A_{A,B}$, and the heat transfer coefficient $U_{A,B}$:

$$\dot{Q}_{A,B} = U_{A,B} A_{A,B} (T_A - T_B) \tag{2.5}$$

The structure of the model is shown in Fig. 2.7. To model thermal masses, we introduce a component M, which is instantiated by specifying volume, density, and specific heat capacity, and optionally allows to specify minimal and maximal temperatures. The heat transfer is abstracted as a component HT, implementing Eq. (2.5), and the heat pump is again modeled with a temperature-dependent efficiency, but with the option of splitting the output to multiple connectors.

Based on the model of the building energy system, we define a DR optimization problem, i.e., we minimize the integral over the electricity costs for a given electricity price profile.

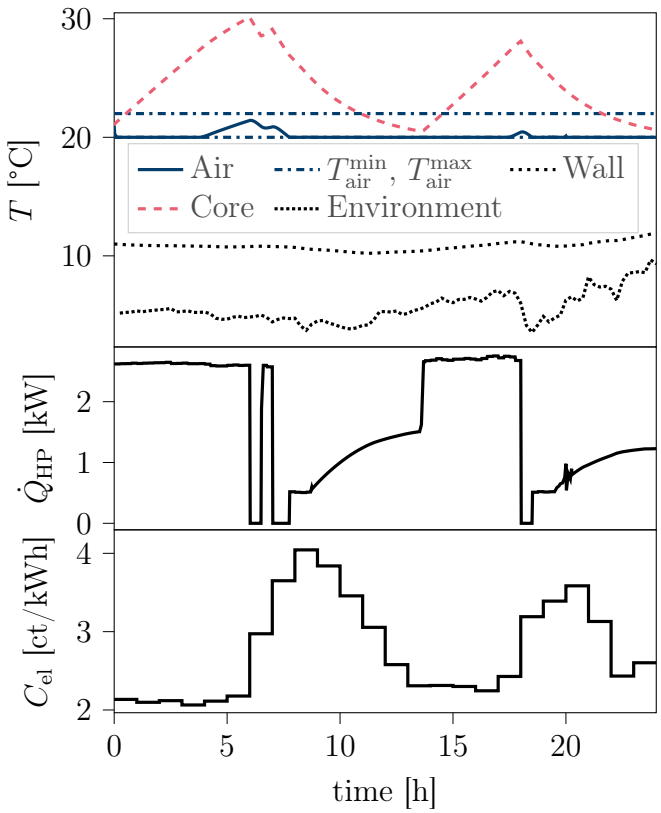


Fig. 2.8. Results of the demand response optimization for the building energy system: the temperatures of the different zones together with the air temperature comfort bounds $T_{\text{air}}^{\text{min}}$ and $T_{\text{air}}^{\text{max}}$ (top), the heat flow supplied by the heat pump $\dot{Q}_{\text{HP}}$ (center), and the electricity costs C_{el} (bottom).

The resulting operational objective function is thus chosen as $\dot{F}_{II} = C_{\text{el}} P_{\text{HP}}$, where C_{el} and P_{HP} are the electricity costs and electric input power of the heat pump, respectively.

As we consider a minimum part-load constraint for the heat pump, the resulting problem is a MIDO problem. The time horizon is a 24 hour period considered at quarter-hourly resolution and the input data consists of hourly electricity price data and ambient temperature data at quarter-hourly resolution. We use a full discretization approach (Cuthrell and Biegler, 1987) via the COMANDO interface to Pyomo.DAE (Nicholson et al., 2018). Specifically, we use Legendre-Radau collocation with four elements per hour and fourth-order polynomials. Since the model contains exclusively linear expressions and we use collocation with a fixed time grid, we obtain a MILP problem after discretization. The resulting formulation has 6931 constraints and 6257 variables, 96 of which are binary. The problem can be solved with Gurobi 9.1.1 to global optimality in less than one second of CPU time. Results are visualized in Fig. 2.8, where the temperatures of the three thermal zones, the ambient temperature, the heat flow supplied by the heat pump, and the electricity price are shown. During times of low prices, the concrete core is heated to store energy. During times of high prices, the concrete core transfers the stored heat to the air zone and cools down such that the heat pump has to supply less heat. Thus, load is shifted to times of favorable prices, while the air temperature remains within the comfort range.

Using the introduced component models for general thermal masses and heat transfers, the extension to a larger building energy system with several rooms, thermal masses, and heat transfers is straightforward. We note that it is also possible to perform rolling horizon optimization in COMANDO by defining an appropriate user-defined algorithm, e.g., as in our previous publication (Shu et al., 2019), where a preliminary version of COMANDO was used.

Tab. 2.3. Clustering of neighbouring buildings into consumer groups. Buildings within a group are assumed to have identical heating curves.

Consumer group	$T_{\mathrm{fl}}^{\max}$ ($T_{\mathrm{air}} = -12\,^{\circ}\mathrm{C}$)	$T_{\mathrm{fl}}^{\min}$ ($T_{\mathrm{air}} = 20\,^{\circ}\mathrm{C}$)
CG_{40}	40 °C	35 °C
CG_{50}	50 °C	40 °C
CG_{70}	70 °C	50 °C
CG_{85}	85 °C	60 °C

2.3.3. Case Study 3: Design of a low-temperature district heating network

In this case study, we extend components of previous work (Hering, Xhonneux, and Müller, 2021) to describe a district heating network and apply them to a design optimization of the network described by Saelens et al., 2019. The system comprises a source of waste heat, a distribution network, and 16 consumers. We aggregate the 16 consumers into four consumer groups, comprising four consumers each, and assume linear heating curves for the flow temperature T_{fl}. The heating curves are described by the flow temperatures $T_{\mathrm{fl}}^{\max}$ and $T_{\mathrm{fl}}^{\min}$, at ambient air temperatures of −12 °C and 20 °C, respectively, see Tab. 2.3.

The source of waste heat supplies heat to a network to which each consumer group may be connected or not. Both waste heat and consumer groups are linked to the network via a heat exchanger or a heat pump, and connecting a consumer group additionally requires the necessary pipes to be built. Independently of whether a consumer group is connected or not, it may also be equipped with a gas-fired boiler or an electric heating rod. The superstructure of the heating network is shown in Fig. 2.9.

The system is modeled using components for a source of waste heat (WH), the distribution network (NW), the power grid (PG) and the gas grid (GG). As both the linking unit and the consumer groups are composed of multiple components and occur more than once, they are modeled as subsystems. The linking subsystem (L) contains a heat pump (HP) and a heat exchanger (HX) and the consumer group subsystem (CG) contains a linking subsystem, a demand (DEM), and two instances of a generic heat source with different parametrizations, representing a boiler ($\mathrm{HS_B}$) and a heating rod ($\mathrm{HS_{HR}}$).

The design decisions comprise binary variables for the type of linking component (heat exchanger, heat pump, or none) and the type of additional heat source (gas boiler, electric heater, or none) to be built, as well as continuous variables for component sizing and the maximum and minimum return temperature of the network $T_{\mathrm{NW,re}}^{\max}$ and $T_{\mathrm{NW,re}}^{\min}$, respectively. Finally, four pipe segments can be added to the network model separately using the decision variables, $b_{\mathrm{NW,p}}$. The linking components for the consumer groups can only be built if all necessary pipe segments of the network are built. The demand component has a parameter for the required heat demand and computes the required flow temperature based on the ambient air temperature. The heat demand is based on Saelens et al. (2019), while the flow temperature is assumed to depend linearly on the ambient air temperature (cf. Tab. 2.3). The network return temperature $T_{\mathrm{NW,re}}$ also depends linearly on the ambient air temperature T_{air} and the design variables $T_{\mathrm{NW,re}}^{\max}$ and $T_{\mathrm{NW,re}}^{\min}$, while the network flow temperature $T_{\mathrm{NW,fl}}$ is assumed to be 15 K higher than $T_{\mathrm{NW,re}}$.

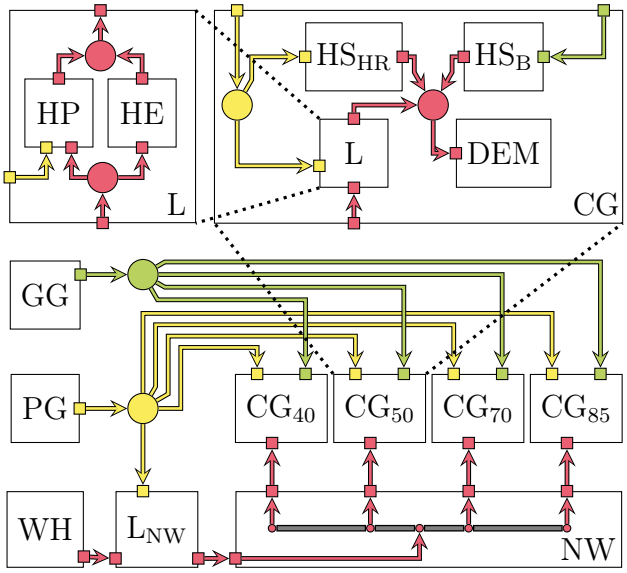


Fig. 2.9. Superstructure with components for the gas grid (GG), power grid (PG), waste heat source (WH) and network (NW) as well as subsystems for linking (L) and consumer groups (CG), see top. The superstructure of each linking subsystem contains a heat pump (HP) and a heat exchanger (HX); the superstructure of each consumer group subsystem contains two heat source (HS) components, parameterized as a heating rod ($\mathrm{HS_{HR}}$) and a boiler ($\mathrm{HS_B}$), a demand (DEM), and a decentralized linking subsystem. To connect the different consumer groups, the necessary pipe segments (depicted as gray bars within NW) need to be built.

We aggregate the whole network into one pipe network with two branches, cf. Fig. 2.9. The central linking component is connected to the center of the network with $T_{\mathrm{NW,fl}}$ and $T_{\mathrm{NW,re}}$. Despite being located at different distances from the center, we assume that all consumer groups receive and reject water at the same flow and return temperatures, $T_{\mathrm{NW,fl}} - \Delta T_{\mathrm{NW,fl,loss}}$ and $T_{\mathrm{NW,re}} + \Delta T_{\mathrm{NW,re,loss}}$, respectively. For this simplification to be conservative, we use the total length of the network, l_{NW}, calculated as

$$l_{\mathrm{NW}} = \sum b_{\mathrm{p}} l_{\mathrm{p}}, \tag{2.6}$$

to calculate the temperature drops, where b_{p} is the build decision and l_{p} is the length of each pipe segment p, cf. Fig. 2.9. To obtain the temperature differences in the flow and return pipes, $\Delta T_{\mathrm{NW,fl,loss}}$ and $\Delta T_{\mathrm{NW,re,loss}}$, respectively, we consider energy balances of the water for both the flow (fl) and return (re) pipe of the network, i.e.,

$$\dot{m}_{\mathrm{NW}}\, c_p\, \Delta T_{\mathrm{NW,fl,loss}} = U_{\mathrm{NW}}\, l_{\mathrm{NW}}\, (T_{\mathrm{NW,fl}} - T_{\mathrm{gr}}) \tag{2.7}$$

$$\dot{m}_{\mathrm{NW}}\, c_p\, \Delta T_{\mathrm{NW,re,loss}} = U_{\mathrm{NW}}\, l_{\mathrm{NW}}\, (T_{\mathrm{NW,re}} + \Delta T_{\mathrm{NW,re,loss}} - T_{\mathrm{gr}}) \tag{2.8}$$

where $\Delta T_{\mathrm{NW,fl,loss}}$ and $\Delta T_{\mathrm{NW,re,loss}}$ are operational variables describing the temperature drop in the respective pipe, c_p is the constant specific heat capacity of water, $U_{\mathrm{NW}} = 0.035 \frac{\mathrm{W}}{\mathrm{mK}}$ is the specific heat transfer coefficient and l_{NW} is the pipe network length, and $T_{\mathrm{gr}} = 8\,^{\circ}\mathrm{C}$ is the average ground temperature.

Tab. 2.4. Specific and fixed costs for heating equipment according to (BMVBS, 2012) and (BBSR, 2014)

Component	C_{spec}	C_{fix}
central HP	500 €/kW	0 €
decentral HP	620 €/kW	0 €
HX	90 €/kW	0 €
HS_{HR}	10 €/kW	100 €
HS_{B}	111 €/kW	4 300 €

The heat pump model in each linking component is modeled via the following set of equations:

$$\dot{Q}_{\text{HP}} \leq b_{\text{HP}}\, 400\,\text{kW} \tag{2.9}$$

$$P_{\text{HP}}\, T_{\text{CON,re}}\, \eta_{\text{COP}} = \dot{Q}_{\text{HP}} (T_{\text{CON,re}} - T_{\text{EVA,re}}) \tag{2.10}$$

$$\dot{m}_{\text{EVA}} c_p\, (T_{\text{EVA,fl}} - T_{\text{EVA,re}}) + P_{\text{HP}} = \dot{m}_{\text{CON}} c_p\, (T_{\text{CON,re}} - T_{\text{CON,fl}}) \tag{2.11}$$

Here, $\dot{Q}_{\text{HP}}$, P_{HP}, $\dot{m}_{\text{EVA}}$, $\dot{m}_{\text{CON}}$, $T_{\text{EVA,fl}}$, $T_{\text{EVA,fl}}$, $T_{\text{CON,re}}$ and $T_{\text{CON,fl}}$ are operational variables, and $\eta_{\text{COP}} = 0.6$ is the heat pump efficiency relative to the carnot efficiency. The outgoing heat flow for the heat pump ($\dot{Q}_{\text{HP}}$) is bounded by zero or the maximum allowable nominal size of 400 kW through Eq. (2.9). The input power P_{HP} is coupled to $\dot{Q}_{\text{HP}}$ via Eq. (2.10). In the energy balance Eq. (2.11), enthalpy differences at the evaporator and condenser side are described by the associated mass flows $\dot{m}_{\text{EVA}}$ and $\dot{m}_{\text{CON}}$, and flow and return temperatures $T_{\text{EVA,fl}}$, $T_{\text{EVA,fl}}$, $T_{\text{CON,fl}}$ and $T_{\text{CON,re}}$.

For the investment cost, we assume linear cost correlations with a specific cost C_{spec} and a fixed cost C_{fix} according to Tab. 2.4.

Additionally, we consider the costs for each pipe segment of the network based on Jentsch et al., 2008. Thus, the total investment costs of the system includes the investments into heating components and piping.

To obtain an economical design, we minimize TAC. We use k-means clustering (Pedregosa et al., 2011) to aggregate the original set of ambient temperatures and heat demands into representative clusters. Each resulting cluster center is a pair of daily mean values for temperature and heat demand and can be considered as a representative operating scenario. To reduce computational demand, the data is clustered into 11 such scenarios, including one scenario representing the maximum heat demand. Demand data with zero heat demand are dropped from the data set. Fig. 2.10 shows the resulting 11 clusters.

We use the clusters as scenarios in the COMANDO framework, with the fraction of data points in each cluster as the corresponding scenario weight. Considering the data in this way ensures that the final design is feasible for all considered scenarios and is optimized with regards to the expected value of TAC. The resulting problem is a MIQCQP with 526 continuous variables, 147 binary variables and 275 quadratic constraints. An optimal design with 0% optimality gap is obtained within six minutes of CPU time, using the Gurobi API interface with Gurobi 9.1.1 and 12 threads. The global optimal solution corresponds to the system shown in Fig. 2.11.

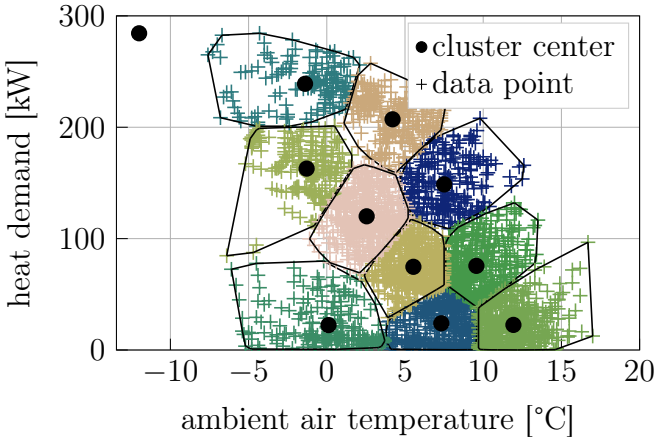


Fig. 2.10. Heat demand clusters: Each cross represents one pair of measurements of total daily mean heat demand and daily mean ambient air temperature. Colors and boundaries are used to aid visual distinction of the clusters whose centers are mean values depicted as black dots.

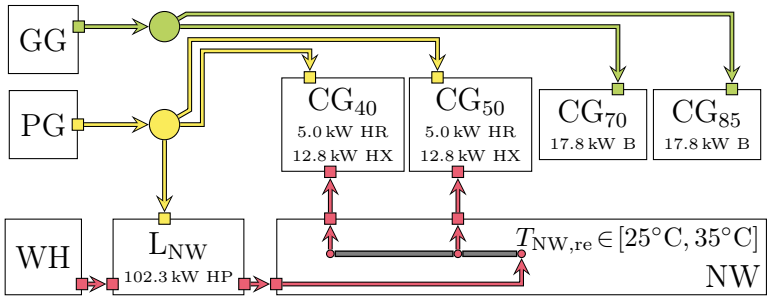


Fig. 2.11. Optimal system structure: A central heat pump HP supplies waste heat from WH to the network NW. Consumer groups CG_{40} and CG_{50} are connected to NW via heat exchangers (HX) and use heating rods (HR) for peak demands. Consumer groups CG_{70} and CG_{85} are not connected and satisfy their heat demand via boilers (B).

The network is designed with a variable return temperature between 25 °C and 35 °C and is connected to the waste heat source using a 102 kW heat pump. Consumer groups CG_{40} and CG_{50} are connected to the network using heat exchangers and have additional electric heating rods installed. Consumer groups CG_{70} and CG_{85} are not connected but satisfy their heat demand using gas-fired boilers instead. The TAC of this design are 22 095 €. At an annual heat demand of 322.7 MWh this corresponds to a specific heating cost of 68.5 €/MWh. In order to assure that the obtained system design is feasible for the original demand data, we perform a second optimization for which we fix the design (i.e., system structure and component sizes) and perform a purely operational optimization using the full set of demands. The design proves to be feasible, with the corrected TAC increasing by less than 2% to 22 414 €.

2.3.4. Case Study 4: Optimal operation of an organic Rankine cycle (ORC)

Finally, we consider a case study from our previous work (Ghasemi et al., 2013b; Huster, Schweidtmann, and Mitsos, 2019), where an optimal operating point of an organic Rankine cycle (ORC) with respect to net power production is sought. The original system under consideration was presented in Ghasemi et al., 2013b, where the off-design behavior for different ambient temperatures is analyzed. Subsequently, Huster, Schweidtmann, and

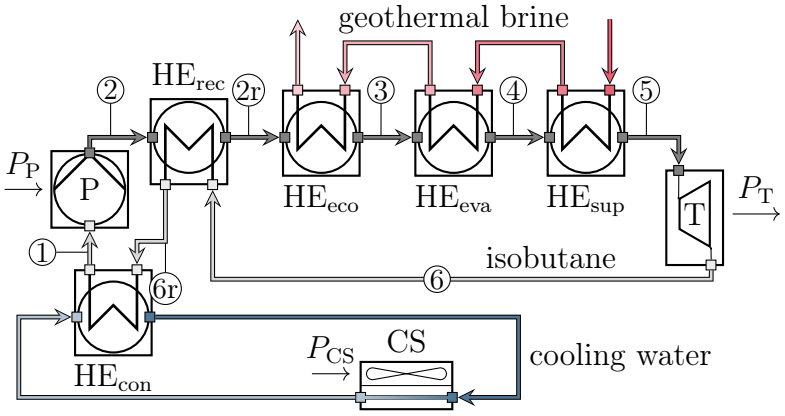


Fig. 2.12. System model of the ORC process from Huster, Schweidtmann, and Mitsos (2019). The components are a pump (P), a recuperator (HX_{REC}), an economizer (HX_{ECO}), an evaporator (HX_{EVA}), a superheater (HX_{SUP}), a turbine (T), a condenser (HX_{CON}), and a cooling system (CS). Flows of geothermal brine, the working fluid isobutane, and cooling water are depicted in red, gray, and blue, respectively. Electrical power is consumed by pump (P_P) and cooling system (P_{CS}) and produced by the turbine (P_T).

Mitsos, 2019 investigated the impact of accurate working fluid models via ANNs (Schweidtmann and Mitsos, 2018; Schweidtmann et al., 2019) on the system design. With this case study, we demonstrate how COMANDO can handle complex modeling features such as accurate fluid properties via ANNs and a sequential modeling approach that gives rise to reduced-space formulations beneficial for global optimization (Bongartz and Mitsos, 2017).

Again, we give a short overview of the case study for self-containment. In the considered process, the working fluid isobutane (ib) is first pressurized by a pump and then preheated in a recuperator before being heated to evaporation temperature, evaporated and superheated by cooling geothermal brine (br) from 408 K to 357 K. After expanding in a turbine, the working fluid is used in the recuperator to preheat the pressurized fluid and is finally condensed and cooled to its original state using cooling water at 288 K. The heat passed from the condenser to the cooling water (cw) is dissipated by a cooling system consisting of multiple fans.

The ORC is modeled as a system consisting of 4 types of components, i.e., a pump (P), a turbine (T), a cooling system (CS), and five heat exchangers (condenser HX_{CON}, recuperator HX_{REC}, economizer HX_{ECO}, evaporator HX_{EVA}, and superheater HX_{SUP}). All components have connectors for enthalpy in- and out-flows that are connected as depicted in Fig. 2.12 to obtain the system model.

As discussed in Bongartz and Mitsos (2017), reduced-space formulations, i.e., formulations in which a large number of variables and constraints are eliminated by substitution, are well suited for global optimization of power cycles such as the present ORC. To obtain a reduced-space formulation, model generation begins with an empty system model to which different component models are added sequentially. First, the decision variables are specified at the system level as follows: The mass flow $\dot{m}$ of the working fluid, the pressures p_1 and p_2 before and after the pump, and the specific enthalpy after the recuperator h_{2r}, as well as the isentropic specific enthalpy after the turbine $h_{6,is}$. All other quantities of interest are defined in terms of these five variables.

In our previous work (Schweidtmann and Mitsos, 2018; Schweidtmann et al., 2019), the

use of artificial neural networks (ANNs) in combination with our inhouse global MINLP solver MAiNGO (Bongartz et al., 2018) has been shown to result in tight relaxations, beneficial for deterministic global optimization. In Huster, Schweidtmann, and Mitsos (2019), we trained several ANNs to learn the relations between various quantities of different thermodynamic phases of the working fluid isubutane, using data generated from the equations of state implemented in the thermophysical property library CoolProp (Bell et al., 2014). The ANNs are used as data-driven surrogte models for the equations of state, which cannot be used directly within the optimization, as they are not available as analytical expressions (Schweidtmann et al., 2019). The validity of the ANNs used for this case study was extensively analyzed and discussed in the original publication (Huster, Schweidtmann, and Mitsos, 2019). Each ANN has two hidden layers with six neurons each, all of which use tanh as the activation function. The ANNs express individual output quantities in terms of either pressure p, pressure and specific enthalpy h, or pressure and specific entropy ϵ (note that we already use s to represent a scenario), as inputs. As a result of training, we thus obtain explicit analytical expressions for various quantities. In this case study, eight of the ANNs from Huster, Schweidtmann, and Mitsos (2019) are used as analytical surrogate models for the following quantities:

$h_{\text{liq}}(p, \epsilon)$	liquid enthalpy
$T_{\text{liq}}(p, h)$	liquid temperature
$h_{\text{sat,liq}}(p)$	enthalpy of saturated liquid
$\epsilon_{\text{sat,liq}}(p)$	entropy of saturated liquid
$T_{\text{sat}}(p)$	saturation temperature
$h_{\text{sat,vap}}(p)$	enthalpy of saturated vapor
$\epsilon_{\text{vap}}(p, h)$	vapor entropy
$T_{\text{vap}}(p, h)$	vapor temperature

The enthalpy flows of pump and turbine are described via mass flow and specific enthalpies, and the electrical power consumed by the pump (P_{P}) and provided by the turbine (P_{T}) are modeled as

$$P_{\text{P}} = \dot{m}\,\frac{h_{\text{P,is}} - h_{\text{P,i}}}{\eta_{\text{P,is}}}, \tag{2.12}$$

$$P_{\text{T}} = \dot{m}\,(h_{\text{T,i}} - h_{\text{T,is}})\,\eta_{\text{T,is}}, \tag{2.13}$$

where $\eta_{\text{P,is}}$ and $\eta_{\text{T,is}}$ are known, constant isentropic efficiencies and the required specific enthalpies h are determined via the appropriate ANNs.

For each heat exchanger, the differences of enthalpy flows at the hot (h) and cold (c) side are either defined in terms of a mass flow and specific enthalpies (ib) or in terms of a specific heat capacity flow $\dot{m}c_p$ and temperatures (cw and br):

$$\dot{Q}_{\text{h}} = \begin{cases} \dot{m}_{\text{h}}\,(h_{\text{h,i}} - h_{\text{h,o}}), & \text{h} = \text{ib} \\ (\dot{m}c_p)_{\text{h}}\,(T_{\text{h,i}} - T_{\text{h,o}}), & \text{h} \in \{\text{cw, br}\} \end{cases} \tag{2.14}$$

$$\dot{Q}_{\text{c}} = \begin{cases} \dot{m}_{\text{c}}\,(h_{\text{c,o}} - h_{\text{c,i}}), & \text{c} = \text{ib} \\ (\dot{m}c_p)_{\text{c}}\,(T_{\text{c,o}} - T_{\text{c,i}}), & \text{c} \in \{\text{cw, br}\} \end{cases} \tag{2.15}$$

As heat losses are neglected, the energy balance reduces to

$$\dot{Q}_\mathrm{h} = \dot{Q}_\mathrm{c}. \tag{2.16}$$

Since we aim for a reduced-space formulation, no variables are introduced for the left-hand sides of Eqs. (2.12)–(2.15) and the corresponding right-hand side expressions are used directly, avoiding the addition of constraints. In particular, where possible, Eq. (2.16) is automatically reformulated to obtain a definition for one of the temperatures or specific enthalpies in the right-hand sides of Eqs. (2.14) and (2.15) in terms of the other quantities. The heat-exchanger model is configured to perform the appropriate reformulation automatically, based on the provided quantities.

A pinch point is assumed in the condenser, i.e., the temperature of the cooling water at the pinch point, T_pinch, is assumed to lie $\Delta T_\mathrm{min} = 10\,\mathrm{K}$ below the evaporation temperature $T_\mathrm{sat}(p_1)$. Through this assumption, it is possible to compute the heat capacity flow of the cooling water, $(\dot{m}c_p)_\mathrm{cw}$, as

$$\begin{aligned}(\dot{m}c_p)_\mathrm{cw} &= \frac{\dot{m}\,(h_\mathrm{pinch} - h_1)}{\max(10^{-5}\,\mathrm{K},\ T_\mathrm{pinch} - T_\mathrm{cw,i})} \\ &= \frac{\dot{m}\left(h_\mathrm{sat,vap}(p_1) - h_\mathrm{sat,liq}(p_1)\right)}{\max\left(10^{-5}\,\mathrm{K},\ T_\mathrm{sat}(p_1) - 10\,\mathrm{K} - 288\,\mathrm{K}\right)}.\end{aligned} \tag{2.17}$$

Note that the max function and the constant 10^{-5} in Eq. (2.17) are introduced to avoid division by zero. The electrical power P_CS, required to run the fans of the cooling system, is modeled to be proportional to the specific heat capacity flow of the air $(\dot{m}c_p)_\mathrm{air}$ passing through them and is computed as

$$\begin{aligned}P_\mathrm{CS} &= \frac{\dot{V}_\mathrm{air}\,\Delta p_\mathrm{fan}}{\eta_\mathrm{fan}} \\ &= \frac{(\dot{m}c_p)_\mathrm{air}\,\Delta p_\mathrm{fan}}{c_{p,\mathrm{air}}\,\rho_\mathrm{air}\,\eta_\mathrm{fan}},\end{aligned} \tag{2.18}$$

where $\Delta p_\mathrm{fan} = 170\,\mathrm{Pa}$ and $\eta_\mathrm{fan} = 0.65$ are the pressure drop and efficiency of the fan, $\dot{V}_\mathrm{air}$, $c_{p,\mathrm{air}} = 1000\,\frac{\mathrm{J}}{\mathrm{kg\,K}}$ and $\rho_\mathrm{air} = 1.2\,\frac{\mathrm{kg}}{\mathrm{m}^3}$ are the volume flow, specific heat capacity and density of the air, respectively. With the assumption that

$$(\dot{m}c_p)_\mathrm{air} = (\dot{m}c_p)_\mathrm{cw}, \tag{2.19}$$

the power of the cooling system is fully determined. For the complete formulation, the reader is referred to the model source code.

The reduced space formulation results in a system model with relatively few expressions, however, since several quantities that are described by ANNs are themselves inputs to other ANNs or used in reformulations within the heat exchangers, the model expressions become deeply nested. For this particular use case, the standard SymPy backend (implemented in pure Python) proved to be inefficient as model generation takes about 45 minutes. Therefore, SymEngine (Čertík et al., 2019), a C++ implementation of a subset of SymPy, was integrated as an alternative backend for COMANDO. Although SymEngine has a

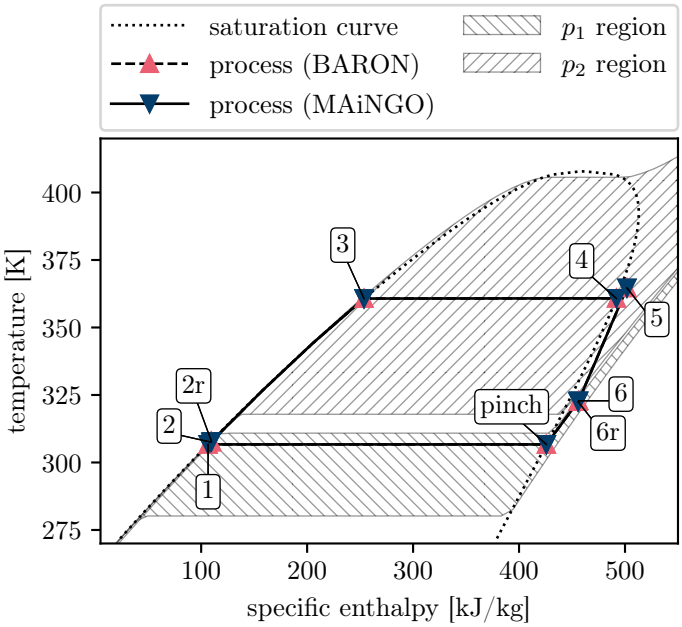


Fig. 2.13. Processes resulting from the optimization using BARON and MAiNGO and boundaries of pressure variables p_1 and p_2.

reduced feature set compared to SymPy, all functionality relevant for the presented case study is provided. The use of SymEngine reduces the model generation time to about 0.1 seconds. Nevertheless, the nested expressions in the model result in very large input files that can take substantial time when written to disk. For instance, when using only a single scenario and operating point and maximizing the net power production

$$P_{\text{net}} = P_{\text{T}} - P_{\text{P}} - P_{\text{CS}}, \tag{2.20}$$

the resulting optimization problem has only 5 variables and 32 constraints.

In order to solve this problem with BARON (Sahinidis, 2020), the nonsmooth max function in Eq. (2.17) is approximated with $\max(a,b) \approx 0.5(a+b+[(a-b+10^{-4})^2]^{0.5})$ and the $\tanh(x)$ function present in the ANNs is equivalently expressed as $1-2/[\exp(2x)+1]$. Generating the BARON input file takes around 1 minute and results in a file size of about 40 MB. This input file is passed to BARON 20.10.16 with absolute and relative optimality tolerances set to 1e-3. BARON reports finding a feasible solution with an objective value of $P_{\text{net}} = 16.48\,\text{MW}$ during preprocessing and terminates after the first iteration and 8 s of CPU time. Although a lower bound within the optimality tolerance is given in the log file, BARON states that it cannot guarantee global optimality due to missing bounds for certain nonlinear subexpressions.

To prove the global optimality of this solution, we use the COMANDO interface to the API of our inhouse solver MAiNGO (Bongartz et al., 2018). MAiNGO automatically provides relaxations of the nested expressions by propagating McCormick relaxations through subexpressions (Mitsos, Chachuat, and Barton, 2009). The COMANDO interface uses a SymEngine implementation of *common subexpression elimination* to find subexpressions that occur more than once within the problem description. By creating intermediate variables and replacing all occurrences of these subexpressions, a small (21 kB) input file for MAiNGO can be created. Since MAiNGO is capable of propagating McCormick relax-

ations, the user does not need to provide bounds on these intermediate variables and they are not treated as decision variables, maintaining the reduced-space formulation. Solving the resulting problem via MAiNGO version 0.3 with the solution returned by BARON as an initial point takes 22 s and confirms its global optimality (see Fig. 2.13), matching the results reported in Huster, Schweidtmann, and Mitsos (2019).

2.4. Conclusion

In this chapter, we presented COMANDO, our flexible open-source framework for **c**omponent-**o**riented **m**odeling and optimiz**a**tion for **n**onlinear **d**esign and **o**peration of energy systems. COMANDO combines desirable features of existing tools and provides layers of abstraction suitable for structured model generation and flexible problem formulation. The behavior of individual components can be represented with detailed models, including dynamic and nonlinear effects based on mechanistic, data-driven, or hybrid modeling approaches. The component models are then aggregated to energy system models, based on which different optimization problems concerning the design and/or operation of the energy system can be formulated. COMANDO natively allows to consider multiple operating scenarios via stochastic programming formulations, allowing to find system designs that are suitable for operation under uncertainty. The resulting problem formulations can either be manipulated in user-defined algorithms, or be passed to algebraic modeling languages or directly to solvers.

COMANDO allows for flexible model creation beyond the capabilities of existing MILP-based energy-system modeling tools and provides a wide range of options for problem formulation. Contrary to classical algebraic modeling frameworks, it allows for modular component and system representations, and is dedicated to energy system design and operation. Through four case studies, we demonstrate how COMANDO can be used to create modular and reusable component and system models of various types of energy systems. Further, we formulate and solve associated optimization problems. With COMANDO, we facilitate and enhance workflows of computer-based analysis of future integrated energy systems.

In the following chapter, we apply COMANDO to a significantly extended version of the case study from Section 2.3.4. To do this, we create more detailed component models that incorporate design decisions in addition to operational decisions, and furthermore take into account the variability of ambient conditions. Based on the resulting system model we simultaneously optimize the system design and operation, considering multiple operating points. COMANDO allows to create the detailed system model in a component-oriented manner, which greatly simplifies the modeling process and makes it easier to maintain and understand than a comparable implementation using an AML.

3. Optimal Design and Operation of an Air-cooled Geothermal Organic Rankine Cycle

In this chapter we employ the COMANDO modeling framework presented in Chapter 2 to create a detailed model of an air-cooled geothermal organic Rankine cycle (ORC). We employ deterministic global optimization to simultaneously optimize the system design and operation, while considering the effect of variable ambient temperature, resulting in a system that maximizes total annualized revenue (TAR).

ORCs are an established technology for the conversion of heat to electricity, with geothermal applications constituting around three quarters of installed ORC capacity worldwide (Tartière and Astolfi, 2017). For a recent review of geothermal energy systems in general see, e.g., Lee, Tester, and You, 2019. Due to their high availability, geothermal ORC systems can serve as a clean and renewable base load technology with the possibility for a high degree of autonomy (Kaplan, Sfar, and Shilon, 1999; Kyriakarakos, Ntavou, and Manolakos, 2020), with system capacities ranging from a few kW to several MW (Macchi and Astolfi, 2017a; Tartière and Astolfi, 2017). For inlet temperatures of the geothermal brine below 180 °C ORCs are economically preferable to dry or flash steam cycles (Nazif, 2011).

In regions where cooling water is not available, an air-cooled condenser (ACC) is the only option for heat rejection, making them a common choice for geothermal applications (Macchi and Astolfi, 2017a). As the heat transfer coefficient for air is low, large exchanger areas are needed, making ACC costs a major fraction of overall equipment costs (Mines and Wendt, 2013). In addition to ACC size, ambient temperature is another factor affecting cooling capacity, and parasitic losses of the ACC, and thus overall cycle efficiency. As a result, an optimal tradeoff between low investment costs and low parasitic losses in different operating conditions is crucial for a technically and economically viable system.

A large body of literature exists for optimization of ORCs in various fields of application, with focus on many different aspects, such as

- selection of optimal working fluids (e.g., Macchi, 2013; Lampe et al., 2014; Schilling et al., 2015; Schilling et al., 2017) or working fluid mixtures (e.g., Huster, Schweidtmann, and Mitsos, 2020a),
- turbine design (e.g., Macchi and Perdichizzi, 1981; Lazzaretto and Manente, 2014; Casartelli et al., 2015; Meroni et al., 2016; La Seta et al., 2016) or heat exchangers (e.g., Pierobon et al., 2013; El-Emam and Dincer, 2013; Erdogan, Colpan, and Cakici, 2017; Astolfi et al., 2017),
- superstructure optimization (e.g., Kalikatzarakis and Frangopoulos, 2016; Huster et al., 2020).

For preliminary design optimization, system operation is commonly represented by a single operating point, e.g., Astolfi et al., 2014a; Astolfi et al., 2014b perform thermodynamic and thermoeconomic optimizations of an air-cooled ORC system for different cycle configurations, fluids, and brine temperatures but consider a single fixed ambient temperature of 15 °C. To ensure reliable performance, it is important that such preliminary optimizations are followed by so-called off-design analyses, where the operation under conditions other than the design point is considered. For a review considering small- to medium-scale applications see Liu et al., 2018. These off-design analyses may again leverage optimization; often this is done via operational optimizations which seek optimal operating strategies for a fixed design at different operating conditions. For instance, Manente et al., 2013 investigate off-design control strategies for an air-cooled ORC, maximizing net power for variations of ambient and brine temperatures from the values assumed during the design phase. However, generalizations of conclusions from such off-design analyses to other systems must be done with care. An example is the frequently stated observation that superheating results in reduced thermodynamic performance for subcritical cycles and that the use of a superheater is of little use or even detrimental (e.g., Saleh et al., 2007; Mago et al., 2008; Astolfi et al., 2014b; Song et al., 2020). Other design assumptions or operating conditions can however yield contradicting results, e.g., Ghasemi et al., 2013c; Ghasemi et al., 2013b demonstrated that considering the off-design performance for an ORC using isobutane, superheat is valuable at high ambient temperatures. To obtain an overall optimum for a particular system, it is thus important to consider the effect of off-design explicitly during the design phase. A common approach for this is to repeatedly run a simulation model and select the design yielding the best results. Calise et al., 2014 perform multiple simulations of a solarthermal ORC at fixed design conditions, varying geometries of the heat exchangers. After determining the geometry that minimizes system costs they perform a second set of simulations to determine performance in off-design conditions. Similarly, Gómez-Aláez et al., 2017 consider an ORC recovering waste heat from hot flue gases of a gas turbine in a gas pipeline recompression station. They obtain a design point by fixing flue gas mass flow rate and temperature to their annual mean and maximizing net power. Subsequently they simulate off-design behavior, keeping turbine reduced mass flow and heat exchanger areas constant. In such simulation-based optimization approaches, all degrees of freedom must be pre-specified by appropriate assumptions or user inputs, hence the resulting designs are only optimal among the finite number of designs corresponding to the considered inputs.

An alternative approach is to let an optimization algorithm determine the optimal values for the degrees of freedom and other variables. For this, several works employ two-step approaches, first optimizing the system at design conditions, and subsequently with the obtained design at multiple off-design conditions. To avoid selecting a design that is suboptimal, such approaches are commonly iterated in different ways. Nusiaputra, Wiemer, and Kuhn, 2014 devise a modular ORC for operation in different wellhead and ambient conditions and develop a control strategy under the assumption of fixed nominal net power and exergy input. In the first phase, component sizes and a design point are calculated based on fixed wellhead and ambient temperature. In the second phase, turbine nozzle opening, pump speed, and fan speed are varied for different off-design conditions via an evolutionary algorithm. The two phases are repeated for a grid of design conditions and the design point yielding the best results is selected as the optimum. The procedure is applied to three different climate regions and specific investment costs or mean cash flow are used

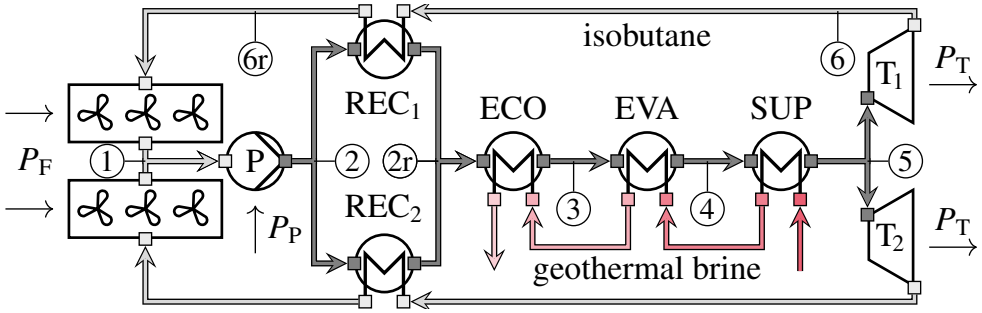


Fig. 3.1. Schematic of the considered ORC process. The working fluid isobutane is pressurized in a pump (P) requiring power P_P, split evenly, and preheated on the shell side of two recuperators (REC). The two streams are again mixed and heated, evaporated, and superheated on the shell side of an economizer (ECO), evaporator (EVA) and superheater (SUP), respectively, via geothermal brine. The working fluid is split again and expanded in two turbines (T), each producing power P_T, followed by heat recuperation. Finally it is desuperheated and condensed back to its original state in air cooled condenser, using electrically powered Fans (F), requiring a total power P_F.

as the selection criteria. Similarly, Capra and Martelli, 2015 extend their previous work (Martelli, Capra, and Consonni, 2015) to include the consideration of part-load operation during the design of an ORC for a combined heat and power application. The authors employ a sequential quadratic programming algorithm to optimize part-load in various operating conditions for fixed designs and a derivative-free black-box algorithm (Martelli and Amaldi, 2014) to search for better designs. Kalikatzarakis and Frangopoulos, 2016 consider an ORC recovering waste heat at different operating conditions of a marine diesel engine and additionally include the system synthesis in the optimization. The optimization is based on a genetic algorithm which switches to an sequential quadratic programming algorithm when progress stagnates. Pili, Spliethoff, and Wieland, 2019 consider an air-cooled ORC for waste heat recovery from a steel billet reheating furnace with variations in the mass flow and temperature of the heat source, and the ambient temperature. In addition to the typical two phases of design and off-design under quasi-steady state conditions, results from the off-design optimizations are interpolated and used as setpoints in a dynamic simulation considering the inertia of the heat exchangers. In this way, a more detailed performance evaluation over the entire operating profile becomes possible.

All of the mentioned two-phase approaches have in common that some aspects of the system model are kept hidden from the optimizer and are only used for subsequent evaluation. As a consequence, the design found by these approaches may not be optimal for the overall problem of design and operation. An alternative is to employ a mathematical programming model, which incorporates both system design and operation, and to give the optimizer access to all model equations. An early example of this approach is given in Yunt et al., 2008, where the design and operation of man-portable power generation systems is considered, and the equivalence of the resulting problem structure to stochastic programming problems (Birge and Louveaux, 2011; Kall and Wallace, 1994) is discussed.

In this chapter we apply such a stochastic programming formulation to simultaneously optimize the design and operation of an air-cooled geothermal ORC. Based on flexible component models, which incorporate the effects of size and operating condition on efficiency and cost, we formulate and solve optimization problems that maximize expected

total annualized return (TAR) for different sets of operating scenarios, represented by one or more ambient temperatures. Through the use of artificial neural networks (ANNs) as surrogate models, we incorporate detailed fluid properties and component characteristics while maintaining computational tractability. In Section 3.1 we present the models for the ORC system and its components, and our approach for quickly generating ANN surrogate models from data. In Section 3.2 we present computational results and Section 3.3 concludes this chapter.

3.1. Models and Methods

We consider the maximization of total annual return (TAR), calculated as

$$\mathrm{TAR} = -(1 + F_{\mathrm{op}})\, F_{\mathrm{ann}}\, C_{\mathrm{I,tot}} + C_{\mathrm{el}}\, \overline{P}_{\mathrm{net}}\, T_{\mathrm{op}}. \tag{3.1}$$

The first term corresponds to annualized capital expenses, where $C_{\mathrm{I,tot}}$ are the the total investment costs and we assume an operational cost factor $F_{\mathrm{op}} = 0.06$, a project lifetime of 20 years and a discount rate of 6%, resulting in an annuity factor of $F_{\mathrm{ann}} = 0.087\,1/\mathrm{a}$. The second term is the revenue from electricity production with an assumed electricity price $C_{\mathrm{el}} = 80\,\mathrm{US\$/(MW\,h)}$, the annual weighted average power production $\overline{P}_{\mathrm{net}}$ and the annual operating time $T_{\mathrm{op}} = 8\,000\,\mathrm{h/a}$.

The total investment costs $C_{\mathrm{I,tot}}$ are

$$C_{\mathrm{I,tot}} = C_{\mathrm{I,E\&D}} + (1 + F_{\mathrm{BOP}}) \sum_{c \in \mathcal{C}} C_{\mathrm{I,c}} \tag{3.2}$$

where $C_{\mathrm{I,E\&D}}$ are investments for exploration and development of the geothermal resource, assumed to be $15 \cdot 10^6$ US\$, $\mathcal{C}$ is the set of installed components (turbines, heat exchangers, pumps, etc.) and the factor $F_{\mathrm{BOP}} = 0.3$ takes into account balance of plant costs (Macchi and Astolfi, 2017a). Details on the investment costs for individual components, $C_{\mathrm{I,c}}$, will be given in the following sections. We point out that the assumed cost data are only given for replicability: they affect the numerical results, but not the proposed methodology, which is the focus of this work.

For a finite set of operating scenarios $\mathcal{S}$, with assumed relative likelihoods w_s, $s \in \mathcal{S}$, $\overline{P}_{\mathrm{net}}$ can be expressed as

$$\overline{P}_{\mathrm{net}} = \sum_{s \in \mathcal{S}} w_s\, P_{\mathrm{net},s}, \tag{3.3}$$

where $P_{\mathrm{net},s}$ is the net power for steady state operation in operational scenario s. Note that while dynamic effects play a crucial role for system control, quasi-steady state models as employed here are sufficiently accurate for techno-economic optimization as shown, e.g., by Pili et al., 2019b.

The system we consider is adapted from Ghasemi et al., 2013b. General assumptions are:

- A constant mass flow of $\dot{m}_{\mathrm{br}} = 660\,\mathrm{kg/s}$ of geothermal brine (an aqueous solution of various minerals) with an inlet temperature of $T_{\mathrm{br,i}} = 135\,^{\circ}\mathrm{C}$ and a pressure of $p_{\mathrm{br}} = 897\,\mathrm{kPa}$ is available.

- The minimum reinjection temperature is $T_{\mathrm{br,o,min}} = 60\,°\mathrm{C}$ and the specific heat capacity is assumed to be constant, $c_{p,\mathrm{br}} = 4.1\,\mathrm{kJ/(kg\,K)}$.
- The cycle to be considered is subcritical, recuperated, and uses the dry working fluid isobutane.
- Before expansion, the working fluid mass flow is split equally between two identical groups of turbines, recuperators and ACC banks.
- A direct ACC is employed, i.e., the working fluid is cooled via an air flow over finned condenser pipes, driven by a series of fans.
- In contrast to Ghasemi et al., 2013b, we consider a dedicated heat exchanger for superheating, and a single row of fans in each ACC bank instead of three. Additionally we only consider a single working fluid pump instead of three parallel ones, as power consumption of pumps – and thus their investment costs – are minor for subcritical cycles, see e.g., Huster et al., 2020. Consequently modeling multiple pumps is expected to have little effect on the TAR.
- Pressure drops for the working fluid are negligible, which means there are only two pressure levels.

The resulting system structure is depicted in Fig. 3.1. With the above assumptions, the net electric power in each operating scenario is

$$P_{\mathrm{net}} = 2\,P_{\mathrm{T}} - N_{\mathrm{F}}\,P_{\mathrm{F}} - P_{\mathrm{P}}, \tag{3.4}$$

where the subscripts T, F, and P identify turbine, ACC fans, and pump, respectively, and N_{F} is the total number of fans. The nominal power for each of these components must be an upper bound to all occurring power levels:

$$P_{\mathrm{c,nom}} \geq P_{\mathrm{c},s} \quad \forall s \in \mathcal{S}, \quad \forall \mathrm{c} \in \{\mathrm{T},\,\mathrm{F},\,\mathrm{P}\} \tag{3.5}$$

Together with these nominal values, the system design is specified via the heat exchanger geometries, the maximum pressure, the design enthalpy drop of the turbine and the design volumetric flow rate at its outlet, see the top of Tab. 3.1. System operation in each considered operating scenario is determined by the low and high pressure levels p_1 and p_2, the working fluid mass flow $\dot{m}$, specific enthalpies h at states 2r and 3 (cf. Fig. 3.1), the brine outlet temperature, the minimum temperature difference in the ACC, the electrical powers on the right-hand side of Eq. (3.4), and the relative enthalpy drop of the turbine as well as the relative volumetric flow rate at its outlet, see the bottom of Tab. 3.1.

To describe the remaining thermodynamical quantities of the working fluid in different states (circled labels in Fig. 3.1), we use existing ANNs, which have been trained and validated in Huster, Schweidtmann, and Mitsos, 2019. These ANNs are explicit, analytical representations of individual fluid properties, and thus avoid the need for lookups or iterative computations required when using database-based property models directly. While models incorporating database-based property models typically require black-box optimization, using local or stochastic global approaches, the functional form of the ANN representations allows for their incorporation into deterministic global optimization via our open-source solver MAiNGO.

Some additional functional relationships are expressed via newly generated ANN surrogate models as described in Section 3.1.1. This allows all model quantities to be explicitly

Tab. 3.1. Design and operational variables and their lower and upper bounds. The bounds for the specific enthalpies h_{2r} and h_3 correspond to the saturated liquid at $p = p_{2,\mathrm{LB}}$ and $p = p_{2,\mathrm{UB}}$, respectively.

design variable	**symbol**	**lower bound**	**upper bound**	**unit**
inner shell diameter ECO	$d_{\mathrm{s,ECO}}$	0.7	2.5	m
inner shell diameter EVA	$d_{\mathrm{s,EVA}}$	0.7	2.5	m
inner shell diameter SUP	$d_{\mathrm{s,SUP}}$	0.7	2.5	m
inner shell diameter REC	$d_{\mathrm{s,REC}}$	0.7	2.5	m
relative tube length ECO	$L_{\mathrm{t,ECO}}/d_{\mathrm{s,ECO}}$	4	12	–
relative tube length EVA	$L_{\mathrm{t,EVA}}/d_{\mathrm{s,EVA}}$	4	12	–
relative tube length SUP	$L_{\mathrm{t,SUP}}/d_{\mathrm{s,SUP}}$	4	12	–
relative tube length REC	$L_{\mathrm{t,REC}}/d_{\mathrm{s,REC}}$	4	12	–
relative baffle spacing ECO	$L_{\mathrm{B,ECO}}/d_{\mathrm{s,ECO}}$	0.2	1	–
relative baffle spacing SUP	$L_{\mathrm{B,SUP}}/d_{\mathrm{s,SUP}}$	0.2	1	–
relative baffle spacing REC	$L_{\mathrm{B,REC}}/d_{\mathrm{s,REC}}$	0.2	1	–
ACC heat transfer area	A_{ACC}	$1 \cdot 10^4$	$1 \cdot 10^6$	m^2
maximum pressure	p_{max}	3	25	bar
nominal fan power	$P_{\mathrm{F,nom}}$	37	200	kW
nominal pump power	$P_{\mathrm{P,nom}}$	0.1	2	MW
nominal turbine power	$P_{\mathrm{T,nom}}$	1	15	MW
design volumetric flow rate	$\dot{V}_{\mathrm{T,o,d}}$	1	50	m^3/s
design specific enthalpy drop	$\Delta h_{\mathrm{T,d}}$	10	65	kJ/kg
Stodola coefficient	K_{S}	0.01	0.05	m^2
operational variable	**symbol**	**LB**	**UB**	**unit**
mass flow rate of isobutane	$\dot{m}$	50	1500	kg/s
low pressure level	p_1	1.1	20	bar
high pressure level	p_2	3	25	bar
specific enthalpy before ECO	h_{2r}	76.105	335.50	kJ/kg
specific enthalpy after ECO	h_3	76.105	335.50	kJ/kg
fan power	P_{F}	0	200	kW
pump power	P_{P}	0.1	2	MW
turbine power	P_{T}	1	15	MW
relative volumetric flow rate	$\dot{V}_{\mathrm{o,rel}}$	0.2	1.2	–
relative specific enthalpy drop	Δh_{rel}	0.2	1.2	–
Brine outlet temperature	$T_{\mathrm{br,o}}$	333.15	403.15	K
ACC min. temperature difference	$\Delta T_{\mathrm{min,ACC}}$	1	55	K

expressed in terms of the variables from Tab. 3.1 and the ambient temperature. In Sections 3.1.2–3.1.4 we give additional detail on the models of the individual components.

Note that the component models are based on a combination of existing modeling approaches from different literature sources. While this allows for the conceptual process design and operation studied in this work, these models should be validated against real plant data before use in a real-world application.

3.1.1. Explicit Functions from Data via ANNs

Functional relationships between one or more input quantities and an output quantity may be represented as artificial neural networks (ANNs). The benefit of using ANNs as surrogate models within global optimization is twofold:

1. They can be used as explicit alternatives to functional relationships which are otherwise only given implicitly, e.g., in the form of raw data, or via functions containing iterative elements or control structures that cannot be handled by global optimizers directly (e.g., Huster, Schweidtmann, and Mitsos, 2019; Huster, Schweidtmann, and Mitsos, 2020b).

2. The propagation of tight relaxations for the used activation functions (e.g., tanh) via McCormick relaxations usually results in good relaxations for the overall functional relationship, cf. Schweidtmann and Mitsos, 2018. Furthermore, these relaxations are typically much tighter than those of alternative representations for regression models, such as polynomials, see e.g., Schweidtmann et al., 2019. These tighter relaxations generally improve convergence of global optimization.

By varying the number of neurons in each layer and their activation function, different ANN formulations can be generated. The choice of the activation is an active research topic, with common choices being the rectified linear unit (ReLU) and the hyperbolic tangent (tanh). While ReLU networks are inherently nonsmooth, they possess the desirable property of piecewise linarity, and thus can be cast as MILP formulations (see, e.g., Grimstad and Andersson, 2019; Lueg et al., 2021). However, for the present work we preferred the nonlinear but smooth tanh activation, as we already consider several other nonlinearities in our model. All resulting ANN surrogate models used in this work provide sufficient accuracy with a single hidden layer containing up to four neurons. More detailed information on the training is available in the supplementary material.

3.1.2. Pump

As noted above, the pump only has a minor effect on both P_{net} and TAR. Consequently we use a simple model with constant values for the mechanical efficiency $\eta_{\text{m}} = 0.95$, and the isentropic pump efficiency $\eta_{\text{P}} = 0.80$. With this, the electrical power consumption of the pump is given as:

$$P_{\text{P}} = \dot{m}\,\frac{h_{2,\text{is}} - h_1}{\eta_{\text{m}}\,\eta_{\text{P}}} \tag{3.6}$$

For the investment costs we use the correlation proposed by Astolfi et al., 2014b, converted from €$_{2014}$ to US$$_{2021}$:

$$C_{\text{I,P}} = 11\,066\,\text{US\$}\left(\frac{P_{\text{P,nom}}}{200\,000\,\text{W}}\right)^{0.67} \tag{3.7}$$

3.1.3. Turbines

As the overall mass flow rate is split equally between the turbines, we model the electrical power produced by a single turbine as

$$\dot{m}_{\mathrm{T}} = \frac{\dot{m}}{2}, \tag{3.8}$$

$$P_{\mathrm{T}} = \dot{m}_{\mathrm{T}} \; (h_5 - h_{6,\mathrm{is}}) \; \eta_{\mathrm{g}} \, \eta_{\mathrm{T}}, \tag{3.9}$$

with a generator efficiency of $\eta_{\mathrm{g}} = 0.95$. However, as the turbines are a major contributor to both overall cost and P_{net}, we consider the effect of design (i.e., turbine size) and operation (i.e., part-load) on the turbine efficiency η_{T}, which can be represented as

$$\eta_{\mathrm{T}} = \eta_{\mathrm{T,d}}(\mathrm{SP}, \mathrm{VR}, T_{\mathrm{crit}}) \; r\left(\Delta h_{\mathrm{rel}}, \dot{V}_{\mathrm{o,rel}}\right), \tag{3.10}$$

where the subscript D refers to the design point. Here the maximum achievable isentropic efficiency $\eta_{\mathrm{T,d}}$ is a function of the turbine size parameter $\mathrm{SP} = \frac{\dot{V}_{\mathrm{o,is,d}}^{0.5}}{\Delta h_{\mathrm{is,d}}^{0.25}}$, the turbine volume ratio $\mathrm{VR} = \frac{\dot{V}_{\mathrm{o,is,d}}}{\dot{V}_{\mathrm{i,d}}}$ and the used working fluid, represented by the respective critical temperature (Macchi and Perdichizzi, 1981; Lio, Manente, and Lazzaretto, 2016), and the reduction coefficient r is a function of the relative enthalpy drop Δh_{rel} and the relative volumetric flow $\dot{V}_{\mathrm{o,rel}}$ of the turbine (Ghasemi et al., 2013b; Pili et al., 2019a):

$$\Delta h_{\mathrm{rel}} = \frac{\Delta h}{\Delta h_{\mathrm{d}}} \tag{3.11}$$

$$\dot{V}_{\mathrm{o,rel}} = \frac{\dot{V}_{\mathrm{o}}}{\dot{V}_{\mathrm{o,d}}} \tag{3.12}$$

Lio, Manente, and Lazzaretto, 2016 computed $\eta_{\mathrm{T,d}}$ for isobutane and several other working fluids. The resulting values show little variation for low values of VR. As in all considered optimizations of the present work VR stayed below 4.25, we assume a fixed value of $\eta_{\mathrm{T,d}} = 0.88$, for simplicity.

Ghasemi et al., 2013b give a correlation for r[1] that is split into two separate factors, r_h, and $r_{\dot{V}}$, described by polynomials. In order to improve the relaxations of these expressions, we replace the original polynomial form with the following ANN surrogate models:

$$\begin{aligned} r_h = {} & 0.21395 \\ & + 0.77056 \, \tanh(0.064155 + 1.7140 \, \Delta h_{\mathrm{rel}}) \\ & + 0.10029 \, \tanh(2.8276 - 2.1628 \, \Delta h_{\mathrm{rel}}) \end{aligned} \tag{3.13}$$

$$\begin{aligned} r_{\dot{V}} = {} & 0.70472 \\ & - 0.27582 \, \tanh(0.39630 - 3.7982 \, \dot{V}_{\mathrm{o,rel}}) \\ & - 0.020017 \, \tanh(3.1786 - 5.0984 \, \dot{V}_{\mathrm{o,rel}}) \end{aligned} \tag{3.14}$$

The comparison with the original correlation and the absolute error can be seen in Fig. 3.2.

[1] Note that the journal publication is missing a 0 in one of the coefficients, the correct correlation can be found in the preprint (Ghasemi et al., 2013a).

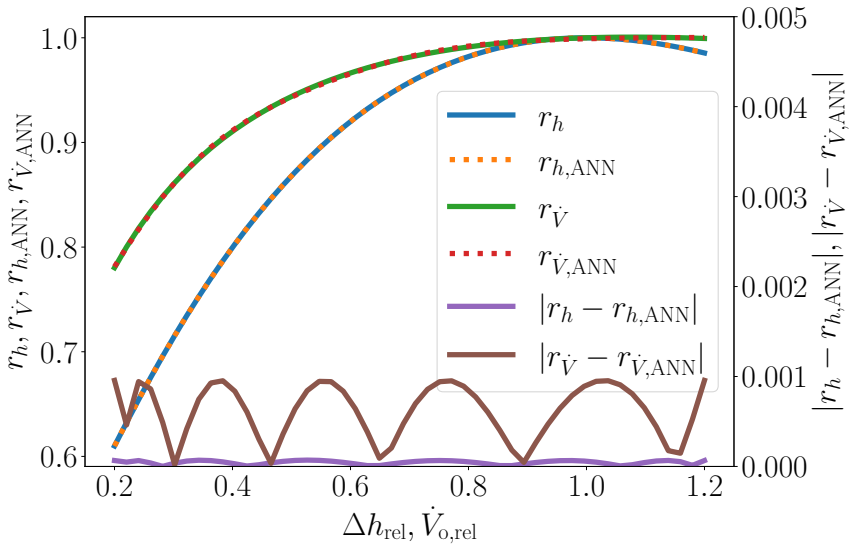


Fig. 3.2. The two factors r_h and $r_{\dot{V}}$, contributing to efficiency reduction coefficient r from Ghasemi et al., 2013b, their surrogate models $r_{h,\mathrm{ANN}}$ and $r_{\dot{V},\mathrm{ANN}}$, and the resulting errors as a function of the relative enthalpy drop Δh_{rel} and volumetric flow rate $\dot{V}_{\mathrm{o,rel}}$.

Another important performance aspect is the relationship between reduced mass flow rate ϕ, defined as

$$\phi = \frac{\dot{m}_{\mathrm{T}}}{\sqrt{\rho_5\, p_5}}, \tag{3.15}$$

and pressure ratio

$$\mathrm{PR} = \frac{p_6}{p_5} = \frac{p_1}{p_2}. \tag{3.16}$$

For classical Rankine cycles, Stodola's ellipse law is commonly used to describe this relationship. Despite the fact that this correlation is only valid for an infinite number of unchoked stages (Cooke, 1984), it is frequently used for modeling ORC turbines, see, e.g., Calise et al., 2014; Capra and Martelli, 2015; Mazzi, Rech, and Lazzaretto, 2015; Pili et al., 2017, even though ORC turbines commonly have between one and three stages (Macchi and Astolfi, 2017a).

As cycles operating with isobutane allow for high efficiencies at low volume ratios (see, e.g., Lio, Manente, and Lazzaretto, 2016), a reasonable initial assumption for the number of stages n_{st} is 1, as this keeps the turbine compact and thus cheap (Macchi and Astolfi, 2017b). We therefore assume a single stage turbine and use a generalization of Stodola's ellipse law proposed by Cooke, 1984. Cooke's generalization accounts for choking if PR sinks below the value corresponding to an isentropic expansion to sonic conditions PR^*, also see Fig. 3.3,

$$\mathrm{PR}^* = \left(\frac{2}{\kappa+1}\right)^{\frac{n_{\mathrm{st}}\,\kappa}{\kappa-1}}, \tag{3.17}$$

$$K_{\mathrm{S}} = \frac{\phi}{\sqrt{1 - \left(\frac{\max(0,\,\mathrm{PR}-\mathrm{PR}^*)}{1-\mathrm{PR}^*}\right)^2}}, \tag{3.18}$$

where we use a fixed isentropic expansion coefficient for isobutane of $\kappa = 1.08$, n_{st} is the number of turbine stages, and K_{S} is the Stodola coefficient, a design variable proportional to the flow cross-section of the turbine.

For the cost of turbine and generator, we use a correlation from Astolfi et al., 2014b

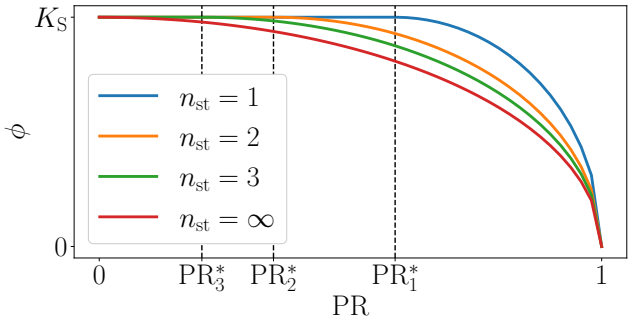


Fig. 3.3. Comparison of Stodola's ellipse law ($n_{st} = \infty$), presenting reduced mass flow rate ϕ, limited by the Stodola coefficient (K_S), as a function of pressure ratio (PR), and Cooke's adaption (Cooke, 1984) for finite stage number (n_{st}) and possibly choked conditions (*) for the working fluid isobutane ($\kappa = 1.08$).

that takes into account both the number of stages n_{st}, and the turbine size parameter SP for the costs of the turbine, as well as the turbine power P_T, for the costs of the electrical generator. Again costs have been converted from €$_{2014}$ to US$$_{2021}$:

$$C_{I,T} = 972\,241\,\text{US\$} \left(\frac{n_{st}}{2}\right)^{0.5} \left(\frac{\text{SP}}{0.18\,\text{m}}\right)^{1.1} + 15\,809\,\text{US\$} \left(\frac{P_{T,nom}}{5\,\text{MW}}\right)^{0.67} \tag{3.19}$$

Note that the problems solved in this work assume a fixed stage number of $n_{st} = 1$ for simplicity, however, the presented model can also handle higher numbers, or even the introduction of n_{st} as a design variable.

3.1.4. Heat Exchangers

We assume that the economizer (ECO), superheater (SUP), and recuperator (REC) are fixed tube sheet shell & tube exchangers, while the evaporator (EVA) is a shell and tube reboiler, and that heat is rejected via an air-cooled condenser (ACC).

It is common practice to model total heat transfer coefficients U_{HX} in off-design conditions as

$$U_{HX} = U_{HX,d} \left(\frac{\dot{m}}{\dot{m}_d}\right)^{F_{U,HX}}, \tag{3.20}$$

where $F_{U,HX}$ is a constant, obtained from simulations or measurements (see e.g., Capra and Martelli, 2015; Pili et al., 2019b). However, apart from $\dot{m}$, U_{HX} generally also depends on pressures, temperatures, and heat exchanger geometry, which is not reflected in Eq. (3.20). Instead of Eq. (3.20), we therefore use empirical correlations for the total heat transfer coefficients U_{HX}, which explicitly account for heat exchanger geometries. The heat transfer area A_{HX} is correlated with thermal quantities via

$$A_{HX} = \dot{Q}_{HX}\, U_{HX}^{-1} \Delta T_{ln,HX}^{-1} F_{T,HX}^{-1}. \tag{3.21}$$

Here $\dot{Q}_{HX}$ is the exchanged heat, $\Delta T_{ln,HX}$ the logarithmic mean temperature difference, and $F_{T,HX}$ a correction factor for a particular heat exchanger HX. For U_{HX}, the denominators of Eqs. (3.47), (3.73) and (3.78) are used for the respective inverse terms in Eq. (3.21). Further details on the component-specific correlations for U_{HX} and A_{HX} can be found in

Sections 3.1.4.1 and 3.1.4.2. In the following, we drop the subscript HX whenever it is not necessary for clarity.

The inverse of the logarithmic mean temperature difference is a convex function of $T_{\mathrm{h,i}} - T_{\mathrm{c,o}}$ and $T_{\mathrm{h,o}} - T_{\mathrm{c,i}}$, and takes the form

$$\Delta T_{\mathrm{ln}}^{-1} = \frac{\ln\left(\frac{T_{\mathrm{h,i}} - T_{\mathrm{c,o}}}{T_{\mathrm{h,o}} - T_{\mathrm{c,i}}}\right)}{(T_{\mathrm{h,i}} - T_{\mathrm{c,o}}) - (T_{\mathrm{h,o}} - T_{\mathrm{c,i}})}, \tag{3.22}$$

where the subscripts h and c refer to the hot and cold fluid, and i and o to the inlet and outlet, respectively. Instead of directly using the right-hand side of Eq. (3.22) in Eq. (3.21), we use an internal function, implementing $\Delta T_{\mathrm{ln}}^{-1}$ and making its convexity visible to the optimizer, resulting in better relaxations (cf. Mistry and Misener, 2016; Najman and Mitsos, 2016b). The logarithmic mean temperature difference is valid for pure cross-flow or for the case where one fluid is isothermal, i.e., for evaporation or condensation. For other situations a temperature correction factor $F_T \leq 1$ is used to adjust the temperature difference, see, e.g., Kuppan, 2013; Serth, 2007.

The cost for the heat exchangers is calculated via the established cost correlations from Turton et al., 2018, adjusted to US$$_{2021}$$ via the Chemical Engineering Plant Cost Index (CEPCI):

$$C_{\mathrm{I,HX,ref}} = C_{\mathrm{Guthrie}}(A_{\mathrm{o,HX}}, F_{\mathrm{HX},1}, F_{\mathrm{HX},2}, F_{\mathrm{HX},3}) \tag{3.23}$$

$$F_{\mathrm{HX,ref}} = F_{\mathrm{HX},4} + F_{\mathrm{HX},5}\, F_{\mathrm{HX,mat}}\, F_{\mathrm{HX},p} \tag{3.24}$$

$$C_{\mathrm{I,HX}} = \frac{\mathrm{CEPCI}_{2021}}{\mathrm{CEPCI}_{2001}} C_{\mathrm{I,HX,ref}}\, F_{\mathrm{HX,ref}}, \tag{3.25}$$

where C_{Guthrie} is the Guthrie cost function

$$C_{\mathrm{Guthrie}}(x, a, b, c) = 10^{a + b\,\log_{10}(x) + c\,\log_{10}(x)^2}, \tag{3.26}$$

and the pressure correction factor is calculated as

$$\hat{p}_{\mathrm{HX}} = p_{\max} - 1\,\mathrm{bar} \tag{3.27}$$

$$F_{\mathrm{HX},p} = C_{\mathrm{Guthrie}}(\hat{p}_{\mathrm{HX}}, 0.03881, -0.11272, 0.08183), \tag{3.28}$$

except for $F_{\mathrm{ACC},p}$ which is 1. The values of the numerical coefficients $F_{\mathrm{HX},i}$, $i \in \{1, \cdots, 5\}$, and $F_{\mathrm{HX,mat}}$ are given in Tab. 3.2. For the shell & tube exchangers, the overall outer tube area $A_{\mathrm{o,HX}}$ corresponds to the total heat exchange area A_{HX}, while for the ACC, the latter is larger due to the use of finned tubing, also see Section 3.1.4.2. The velocities of all fluids flowing within the tubes and the shells are limited to $v_{\max} = 3\,\mathrm{m/s}$ for liquids and to $v_{\max} = 20\,\mathrm{m/s}$ for gases.

3.1.4.1. Shell & Tube Heat Exchangers

For the shell and tube type exchangers, we assume triangular tube arrangement and fixed values for the tube pitch L_{p}, outer and inner tube diameter d_{o} and d_{i} according to Ghasemi et al., 2013b, while the tube length L_{t} and baffle spacing L_{B} are expressed via variable ratios with respect to the inner shell diameter d_{s}, c.f. Tab. 3.1. The number of tube passes

Tab. 3.2. Coefficients for heat exchanger cost correlations, taken from Turton et al., 2018. For ECO, SUP, and REC, we take the coefficients for fixed-tubesheet exchangers, and for EVA those for U-tube exchangers. Note that the coefficients for the kettle-reboilers which might be considered as an alternative for EVA are only valid for small units up to 100 m^2. We assume all shell & tube exchangers are manufactured from stainless steel and the ACC from carbon steel.

	ECO, SUP, REC	EVA	ACC
$F_{\mathrm{HX},1}$	4.324 7	4.464 6	4.033 6
$F_{\mathrm{HX},2}$	−0.303 0	−0.527 7	0.234 1
$F_{\mathrm{HX},3}$	0.163 4	0.395 5	0.049 7
$F_{\mathrm{HX},4}$	1.63	1.63	0.96
$F_{\mathrm{HX},5}$	1.66	1.66	1.21
$F_{\mathrm{HX,mat}}$	2.75	2.75	1

Tab. 3.3. Geometry for shell and tube type exchangers. The values for N_{tp} were assumed, the remaining values are taken from Ghasemi et al., 2013b.

HX	N_{tp}	L_{p} [mm]	d_{o} [mm]	d_{i} [mm]
EVA	2	20.64	15.88	14.23
ECO	1	20.64	15.88	14.23
SUP	1	20.64	15.88	14.23
REC	1	39.69	31.75	29.64

N_{tp} is assumed to be 2 for the evaporator and 1 for all other exchangers, resulting in $F_T = 1$ in all cases. The resulting geometry is summarized in Tab. 3.3.

With the assumed parameters, the tube bundle diameter d_{tb}, tube number N_{t} and heat exchanger area A_{o} can be obtained as discussed by Kuppan, 2013:

$$d_{\mathrm{tb}} = d_{\mathrm{s}} - \left(0.005\,\mathrm{m} + \frac{0.012\,\mathrm{m}}{d_{\mathrm{s}}}\right) \tag{3.29}$$

$$N_{\mathrm{t}} = \frac{1.56}{\sqrt{3}} \left(\frac{d_{\mathrm{tb}} - d_{\mathrm{o}}}{L_{\mathrm{p}}}\right)^2 \tag{3.30}$$

$$A_{\mathrm{o}} = \pi\, d_{\mathrm{o}} L_{\mathrm{t}} N_{\mathrm{t}} \tag{3.31}$$

Additionally, we consider the cross-sectional areas available for tube- and shell-side flow (index t and s, respectively):

$$A_{\mathrm{t,cs}} = \pi \frac{d_{\mathrm{i}}^2}{4} \frac{N_{\mathrm{t}}}{N_{\mathrm{tp}}} \tag{3.32}$$

$$A_{\mathrm{s,cs}} = L_{\mathrm{B}} \left(d_{\mathrm{s}} - d_{\mathrm{tb}} + (d_{\mathrm{tb}} - d_{\mathrm{o}}) \left(1 - \frac{d_{\mathrm{o}}}{L_{\mathrm{p}}}\right)\right) \tag{3.33}$$

Using these cross-sectional areas, we can express limits on the flow velocities

$$\dot{V}_{\mathrm{t,max}} \leq A_{\mathrm{t,cs}}\, v_{\mathrm{max}}, \tag{3.34}$$

$$\dot{V}_{\mathrm{s,max}} \leq A_{\mathrm{s,cs}}\, v_{\mathrm{max}}, \tag{3.35}$$

where v_{max} is set to the limit for liquids or gases, depending on the phase of the respective fluid.

Exchangers with Single Phase Fluids

In the economizer, superheater and recuperater both hot and cold fluids are all single phase shell and tube exchangers. For the hot fluid (subscript h), which flows on the tube-side, the Gnielinski correlation (Gnielinski, 1976; Gnielinski, 1983) is used:

$$\mathrm{Re}_{\mathrm{t}} = \frac{4\,\dot{m}_{\mathrm{h}}\,N_{\mathrm{tp}}}{\pi\,d_{\mathrm{i}}\,N_{\mathrm{t}}\,\overline{\mu}_{\mathrm{h}}} \tag{3.36}$$

$$F_{\mathrm{D}} = \left(0.782\,\ln(\mathrm{Re}_{\mathrm{t}}) - 1.51\right)^{-2} \tag{3.37}$$

$$F_{\mathrm{e}} = 1 + \left(\frac{d_{\mathrm{i}}}{L_{\mathrm{t}}}\right)^{2/3} \tag{3.38}$$

$$\mathrm{Nu}_{\mathrm{t}} = \frac{F_{\mathrm{D}}/8\,(\mathrm{Re}_{\mathrm{t}} - 1000)\overline{\mathrm{Pr}}_{\mathrm{h}}}{1 + 12.7\sqrt{F_{\mathrm{D}}/8}\left(\overline{\mathrm{Pr}}_{\mathrm{h}}^{2/3} - 1\right)}\, F_{\mathrm{e}} \tag{3.39}$$

$$\alpha_{\mathrm{t}} = \mathrm{Nu}_{\mathrm{t}}\frac{\overline{k}_{\mathrm{h}}}{d_{\mathrm{i}}} \tag{3.40}$$

Here F_{D} is the Darcy friction factor and F_{e} a correction for entry effects. The fluid properties $\overline{\mu}_{\mathrm{h}}$, $\overline{\mathrm{Pr}}_{\mathrm{h}}$, and $\overline{k}_{\mathrm{h}}$ are evaluated at $(T, p) = (\overline{T}_{\mathrm{h}}, p_{\mathrm{h}})$.

For the cold fluid (subscript c) flowing on the shell-side, we use the simplified Delaware method as described by Kern and Kraus, 1972 (also see Serth, 2007), to describe the heat transfer coefficient:

$$d_{\mathrm{eq}} = \frac{2\sqrt{3}\,L_{\mathrm{p}}^2}{\pi d_{\mathrm{o}}} - d_{\mathrm{o}} \tag{3.41}$$

$$\mathrm{Re}_{\mathrm{s}} = \frac{d_{\mathrm{eq}}\,\dot{m}_{\mathrm{c}}}{A_{\mathrm{s,cs}}\,\overline{\mu}_{\mathrm{c}}} \tag{3.42}$$

$$F_{\mathrm{C}} = \frac{1 + \frac{L_{\mathrm{B}}}{d_{\mathrm{s}}}}{2}\left(0.08\,\mathrm{Re}_s^{0.6821} + 0.7\,\mathrm{Re}_s^{0.1772}\right) \tag{3.43}$$

$$\mathrm{Nu}_{\mathrm{s}} = F_{\mathrm{C}}\,\overline{\mathrm{Pr}}_{\mathrm{c}}^{1/3}\left(\frac{\overline{\mu}_{\mathrm{c}}}{\mu_{\mathrm{c}}\left(\overline{T}_{\mathrm{w}}, p_{\mathrm{c}}\right)}\right)^{0.14} \tag{3.44}$$

$$\alpha_{\mathrm{s}} = \mathrm{Nu}_{\mathrm{s}}\frac{\overline{k}_{\mathrm{c}}}{d_{\mathrm{eq}}} \tag{3.45}$$

Here F_{C} is the Colburn factor and the fluid properties $\overline{\mu}_{\mathrm{c}}$, $\overline{\mathrm{Pr}}_{\mathrm{c}}$, and $\overline{k}_{\mathrm{c}}$ are evaluated at $(T, p) = (\overline{T}_{\mathrm{c}}, p_{\mathrm{c}})$ and

$$\overline{T}_{\mathrm{w}} = \frac{\overline{T}_{\mathrm{c}} + \overline{T}_{\mathrm{h}}}{2}. \tag{3.46}$$

The overall heat transfer coefficient is then given by

$$U_{\mathrm{S\&T}}^{-1} = \frac{d_\mathrm{o}}{d_\mathrm{i}} \frac{1}{\alpha_\mathrm{t}} + d_\mathrm{o} \frac{\ln(d_\mathrm{o}/d_\mathrm{i})}{2\,k_\mathrm{t}} + \frac{1}{\alpha_\mathrm{s}} + F_\mathrm{R}, \tag{3.47}$$

where we use $k_\mathrm{t} = 16\,\mathrm{W/(m\,K)}$ as the thermal conductivity of the tube material (stainless steel, Serth, 2007), and a factor for fouling resistance of $F_\mathrm{R} = 1.3 \cdot 10^4\,\mathrm{m^2\,K/W}$ (Hernandez-Galan and Plauchu, 1989).

Evaporator

For the tube side heat transfer coefficient, we again use Eqs. (3.36)–(3.40) while for the shell side, where the isobutane is boiled on horizontal tubes, we follow the approach proposed in Serth, 2007 and correct a coefficient for nucleate boiling (subsctipt nb) to account for convective effects. The resulting heat transfer coefficient for boiling (subscript b) can then be obtained through the following correlations:

$$p_\mathrm{c,rel} = \frac{p_\mathrm{c}}{p_\mathrm{c,crit}} \tag{3.48}$$

$$\Delta T_\mathrm{w} = \overline{T}_\mathrm{w} - T_\mathrm{sat} \tag{3.49}$$

$$F_p = 1.8\,p_\mathrm{c,rel}^{0.17} + 4\,p_\mathrm{c,rel}^{1.2} + 10\,p_\mathrm{c,rel}^{10} \tag{3.50}$$

$$\alpha_\mathrm{nb} = 1.469\,2 \cdot 10^{-15}\,p_\mathrm{c,crit}^{2.3}\,\Delta T_\mathrm{w}^{7/3}\,F_p^{10/3} \tag{3.51}$$

$$F_\mathrm{b} = 1 + 0.1\left(0.90644\,\frac{d_\mathrm{tb}\,d_\mathrm{o}}{L_\mathrm{p}^2} - 1\right)^{0.75} \tag{3.52}$$

$$\alpha_\mathrm{b} = \alpha_\mathrm{nb}\,F_\mathrm{b} + 250 \tag{3.53}$$

Eq. (3.51) is the Mostinski correlation for nucleate boiling, adjusted to SI units.

The overall heat transfer coefficient is again calculated via Eq. (3.47) with α_b instead of α_s. Following Serth, 2007, we use the saturation temperature for all calculations related to ΔT_ln and F_T, instead of the inlet temperature of the potentially subcooled liquid. Note that by fixing the geometry, the working fluid at the evaporator inlet will not necessarily be saturated when considering multiple operating points, but instead is determined by Eq. (3.21). We therefore introduce the enthalpy of the working fluid at the outlet of the economizer as an auxiliary variable, and limit its value to lie between 95%–100% of the saturated value at the given pressure.

3.1.4.2. Air-Cooled Condenser (ACC)

For the ACC, we assume horizontal bundles of finned tubes, arranged in multiple rows, again following Ghasemi et al., 2013b, but allowing for different heat exchanger areas via a variable number of tubes N_t. The resulting values for tube and fin dimensions and spacing are given in Tab. 3.4, for the assumed geometry, see Fig. 3.4.

Tab. 3.4. Assumed tube and fin geometry for air cooled condenser. The number of tube passes and tube length ($\hat{=}\,36\,\mathrm{ft}$) are typical values (Serth, 2007), the remaning values are taken from Ghasemi et al., 2013b.

N_{tp} [–]	L_{t} [m]	L_{p} [mm]	d_{f} [mm]	d_{o} [mm]	d_{i} [mm]	L_{f} [mm]	θ_{f} [mm]
4	10.97	69.85	63.5	31.75	27.53	1.9	0.41

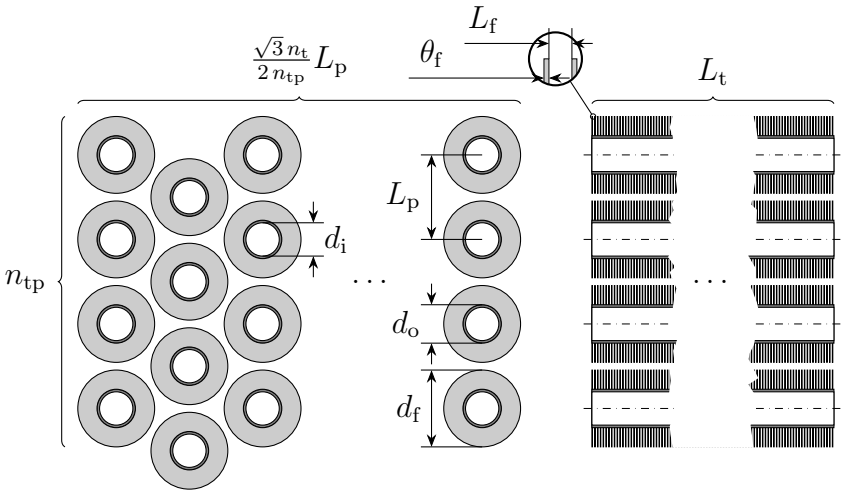


Fig. 3.4. Assumed geometry for the ACC: Number of tubes N_{t}, tube passes N_{tp}, tube length L_{t}, tube pitch L_{p}, diameters of inner tube d_{i}, outer tube d_{o}, and fin d_{f}, fin thickness θ_{f}, and spacing L_{f}. Left: cross-section, right: side view. Note that each tube pass corresponds to a single tube row.

The tube number N_{t} can then be expressed as a linear function of the overall heat transfer area A_{ACC}, i.e.,

$$\begin{aligned} N_{\mathrm{t}} &= \frac{A_{\mathrm{ACC}}}{L_{\mathrm{t}}\,\pi\left[d_{\mathrm{o}} + \frac{\frac{d_{\mathrm{f}}^2 - d_{\mathrm{o}}^2}{2} + \theta_{\mathrm{f}}\,(d_{\mathrm{f}} - d_{\mathrm{o}})}{L_{\mathrm{f}} + \theta_{\mathrm{f}}}\right]} \\ &= 0.050307\,A_{\mathrm{ACC}}, \end{aligned} \tag{3.54}$$

where d_{f} is the fin diameter, θ_{f} the fin thickness and L_{f} the fin spacing, also see Fig. 3.4. Note that since N_{t} is in the order of 1000 – 10 000, the effect of its integrality is negligible.

For the air-side heat transfer coefficient we first calculate the Reynolds number based on the maximum air velocity as given in Serth, 2007:

$$A_{\mathrm{face}} = \frac{\sqrt{3}\,N_{\mathrm{t}}}{2\,N_{\mathrm{tp}}}\,L_{\mathrm{p}} L_{\mathrm{t}} \tag{3.55}$$

$$v_{\mathrm{face}} = \frac{\dot{m}_{\mathrm{air}}}{\rho_{\mathrm{air}}(T_{\mathrm{air,i}})\,A_{\mathrm{face}}} \tag{3.56}$$

$$v_{\mathrm{air,max}} = \frac{v_{\mathrm{face}}\,L_{\mathrm{p}}\,(L_{\mathrm{f}} + \theta_{\mathrm{f}})}{L_{\mathrm{p}} - d_{\mathrm{o}} - (d_{\mathrm{f}} - d_{\mathrm{o}})\,\theta_{\mathrm{f}}} \tag{3.57}$$

$$\mathrm{Re}_{\mathrm{air}} = \frac{d_{\mathrm{o}}\,v_{\mathrm{air,max}}\,\overline{\rho}_{\mathrm{air}}}{\overline{\mu}_{\mathrm{air}}} \tag{3.58}$$

For the calculation of the heat transfer coefficient, we use the correlation of Ganguli, Tung,

and Taborek, 1985:

$$A_{\mathrm{o,rel}} = \frac{A_{\mathrm{o}}}{A_{\mathrm{ACC}}} \tag{3.59}$$

$$\mathrm{Nu}_{\mathrm{air}} = 0.38\,\mathrm{Re}_{\mathrm{air}}^{0.6}\,\overline{\mathrm{Pr}}_{\mathrm{air}}^{1/3}\,A_{\mathrm{o,rel}}^{0.15} \tag{3.60}$$

$$\alpha_{\mathrm{air}} = \mathrm{Nu}_{\mathrm{air}}\frac{\overline{k}_{\mathrm{air}}}{d_{\mathrm{o}}} \tag{3.61}$$

Here A_{ACC} is the total area that is in contact with air, including the fin surface, and $\overline{\rho}_{\mathrm{air}}, \overline{\mu}_{\mathrm{air}}, \overline{\mathrm{Pr}}_{\mathrm{air}}$, and $\overline{k}_{\mathrm{air}}$ are evaluated at $(T, p) = (\overline{T}_{\mathrm{air}}, 1\,\mathrm{atm})$.

The desuperheating (DES) and condensation (CON) sections need to be considered separately, as these regions may exhibit very different heat transfer coefficients and temperature differences.

While in the condensing section the isobutane is isothermal, and hence, $F_{T,\mathrm{CON}} = 1$, $F_{T,\mathrm{DES}}$ needs to be determined via an appropriate correlation. In preliminary calculations with various ambient temperatures, desuperheating occured mostly within the first tube-row of the ACC. We therefore determine $F_{T,\mathrm{DES}}$ based on the correlation of Schedwill, 1968 for a single row of finned tubes, also see Kuppan, 2013:

$$e_{\mathrm{h}} = \frac{T_{6r} - T_{\mathrm{pinch}}}{T_{6r} - T_{\mathrm{air,pinch}}} \tag{3.62}$$

$$e_{\mathrm{c}} = \frac{T_{\mathrm{air,o}} - T_{\mathrm{air,pinch}}}{T_{6r} - T_{\mathrm{air,pinch}}} \tag{3.63}$$

$$F_{T,\mathrm{DES}} = \frac{e_{\mathrm{c}}\,\ln\left(\frac{1-e_{\mathrm{h}}}{1-e_{\mathrm{c}}}\right)}{(e_{\mathrm{h}} - e_{\mathrm{c}})\,\ln\left(\frac{e_{\mathrm{c}}}{e_{\mathrm{h}}}\,\ln(1-e_{\mathrm{h}}) + 1\right)} \tag{3.64}$$

Here e_{h} and e_{c} are the effectiveness of the hot and the cold stream, i.e., isobutane and air, respectively. The temperatures T_{pinch} and $T_{\mathrm{air,pinch}}$ refer to the isobutane and air temperatures at the point of minimal temperature difference, i.e., the beginning of condensation. Note that Eq. (3.64) is indeterminate for $e_{\mathrm{h}} = e_{\mathrm{c}}$ as well as for

$$e_{\mathrm{c}} \geq e_{\mathrm{c,lim}} = -\frac{e_{\mathrm{h}}}{\ln(1-e_{\mathrm{h}})}, \tag{3.65}$$

also see Fig. 3.5. These two facts complicate the use of Eq. (3.64) in global optimization. We thus generate an ANN representing the inverse of $F_{T,\mathrm{DES}}$, needed in Eq. (3.21). As pointed out by Ahmad, Linnhoff, and Smith, 1988 and Smith, 2005, both low values of F_T, as well as regions where F_T has a steep slope should be avoided, as they correspond to an excessive temperature cross and the risk for larger errors in the predicted heat transfer, respectively. To achieve this, we scale the limit in Eq. (3.65) by 90% and consider only data that satisfies

$$e_{\mathrm{c}} \leq -\frac{e_{\mathrm{h}}}{\ln(1-e_{\mathrm{h}}/0.9)}. \tag{3.66}$$

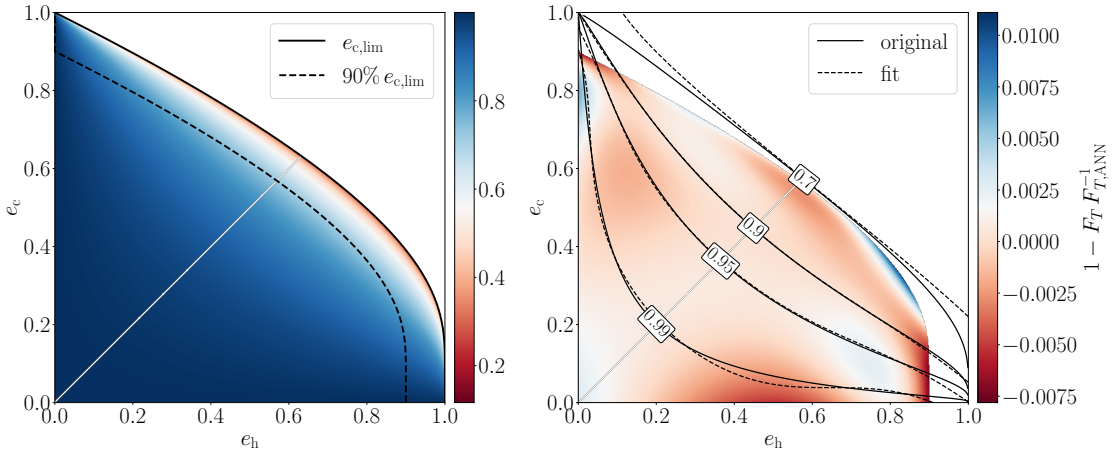


Fig. 3.5. Left: Schedwill correlation for F_T for cross-flow over a finned tube (Eq. (3.64)). Note that the function is indeterminate for $e_h = e_c$ as well as for $e_c \geq e_{c,lim}$ (Eq. (3.65)). Right: Error for the resulting ANN representing inverse F_T and contours for F_T based on the original correlation and the fit.

The resulting ANN surrogate model takes the form

$$\begin{aligned} F_{T,\mathrm{ANN}}^{-1}(e_\mathrm{h}, e_\mathrm{c}) = 0.60127 &- 1.9843 \tanh(3.1595 - 2.3103\, e_\mathrm{h} - 1.6979\, e_\mathrm{c}) \\ &+ 1.9843 \tanh(3.8096 + 2.0196\, e_\mathrm{h} - 2.3837\, e_\mathrm{c}) \\ &- 0.22447 \tanh(9.6024 - 6.5102\, e_\mathrm{h} - 9.3455\, e_\mathrm{c}) \\ &+ 0.61916 \tanh(2.8588 - 2.7623\, e_\mathrm{h} - 1.2156\, e_\mathrm{c}) \end{aligned} \tag{3.67}$$

and has its largest error of about 1% close to the boundary of the domain, also see Fig. 3.5.

For the desuperheating section, α_t is calculated using Eqs. (3.36), (3.37), (3.39) and (3.40) and $F_\mathrm{E} = 1$ since $d_\mathrm{i} \ll L_\mathrm{t}$. Neglecting fouling and the contact resistance between fins and tubing, the overall coefficient based on A_ACC can then be computed as described in Serth, 2007:

$$F_\mathrm{f} = \frac{d_\mathrm{f} + \theta_\mathrm{f} - d_\mathrm{o}}{2} \left[1 + 0.35 \ln\left(\frac{d_\mathrm{f} + \theta_\mathrm{f}}{d_\mathrm{o}}\right)\right] \tag{3.68}$$

$$F_\alpha = \sqrt{\frac{2\,\alpha_\mathrm{air}}{k_\mathrm{f}\,\theta_\mathrm{f}}} \tag{3.69}$$

$$\eta_\mathrm{f} = \frac{\tanh(F_\mathrm{f}\,F_\alpha)}{F_\mathrm{f}\,F_\alpha} \tag{3.70}$$

$$\eta_\mathrm{wf} = \frac{A_\mathrm{o,air} + \eta_\mathrm{f}\,A_\mathrm{f}}{A_\mathrm{ACC}} \tag{3.71}$$

$$A_\mathrm{i,rel} = \frac{A_\mathrm{i}}{A_\mathrm{ACC}} \tag{3.72}$$

$$U_\mathrm{DES}^{-1} = \frac{1}{\alpha_\mathrm{t}\,A_\mathrm{i,rel}} + \frac{A_\mathrm{ACC}}{\pi\,L_\mathrm{t}} \frac{\ln\left(d_\mathrm{o}/d_\mathrm{i}\right)}{2\,k_\mathrm{t}} + \frac{1}{\alpha_\mathrm{air}\,\eta_\mathrm{wf}} \tag{3.73}$$

Here $A_\mathrm{o,air}$ is the outer tube area that is in direct contact with air, i.e., excluding area in contact with fins, A_f is the total fin area, $k_\mathrm{f} = 237\,\frac{\mathrm{W}}{\mathrm{mK}}$ is the thermal conductivity of the aluminum fins and η_wf is the weighted fin efficiency.

In the condensation section, the local heat transfer coefficient within the tubes can be described by Shah's correlation (Shah, 1979; Shah, 2009):

$$F_x = (1-q)^{0.8} + \frac{3.8\, q^{0.76}(1-q)^{0.04}}{p_{\mathrm{h,rel}}^{0.38}} \tag{3.74}$$

$$\alpha_{\mathrm{CON}}(q) = 0.023\, \mathrm{Re}_{\mathrm{sat}}^{0.8} \underset{\mathrm{h,sat}}{\mathrm{Pr}}^{0.4} \frac{k_{\mathrm{h,sat}}}{d_{\mathrm{i}}} F_q, \tag{3.75}$$

where q is the vapor quality. Here $\mathrm{Re}_{\mathrm{sat}}$ is calculated as in Eq. (3.36) with $\overline{\mu}_{\mathrm{h}}$ replaced by $\mu_{\mathrm{h}}(T_{\mathrm{h,sat}}, p_{\mathrm{h}})$, and similarly $\mathrm{Pr}_{\mathrm{h,sat}}$ and $k_{\mathrm{h,sat}}$ are evaluated at $(T,p) = (T_{\mathrm{h,sat}}, p_{\mathrm{h}})$. As proposed by Shah, we assume a linear variation of the vapor quality and integrate Eq. (3.74) to obtain an average heat transfer coefficient that may be used for the complete condensation:

$$\overline{F}_q = 5/9 + \frac{2.0434}{p_{\mathrm{h,rel}}^{0.38}} \tag{3.76}$$

$$\overline{\alpha}_{\mathrm{CON}} = 0.023\, \mathrm{Re}_{\mathrm{sat}}^{0.8} \underset{\mathrm{h,sat}}{\mathrm{Pr}}^{0.4} \frac{k_{\mathrm{h,sat}}}{d_{\mathrm{i}}} \overline{F}_q \tag{3.77}$$

As in the desuperheating section the overall heat transfer coefficient can be computed as:

$$U_{\mathrm{CON}}^{-1} = \frac{1}{\overline{\alpha}_{\mathrm{CON}}\, A_{\mathrm{i,rel}}} + \frac{A_{\mathrm{ACC}}}{\pi\, L_{\mathrm{t}}} \frac{\ln(d_{\mathrm{o}}/d_{\mathrm{i}})}{2\, k_{\mathrm{t}}} + \frac{1}{\alpha_{\mathrm{air}}\, \eta_{\mathrm{wf}}} \tag{3.78}$$

With the respective quantities for the desuperheating and condensing sections, the overall area of the ACC must satisfy

$$\begin{aligned} A_{\mathrm{ACC}} &= A_{\mathrm{DES}} + A_{\mathrm{CON}} \\ &= \left(\dot{Q} U^{-1}\, \Delta T_{\mathrm{ln}}^{-1}\, F_T^{-1}\right)_{\mathrm{DES}} \\ &\quad + \left(\dot{Q} U^{-1}\, \Delta T_{\mathrm{ln}}^{-1}\right)_{\mathrm{CON}}. \end{aligned} \tag{3.79}$$

Note that for the condensation section F_T is equal to 1 due to the isothermal phase change.

The electrical power for each fan is calculated as

$$P_{\mathrm{F}} = \frac{\Delta p_{\mathrm{F}} \dot{V}_{\mathrm{air}}}{N_{\mathrm{F}}\, \eta_{\mathrm{F}}}, \tag{3.80}$$

where N_{F} is the total number of fans and an overall efficiency of $\eta_{\mathrm{F}} = 0.7$ is assumed. The necessary pressure difference Δp_{F}, provided by the fan is derived from the air-side pressure drop based on correlations from Ganguli, Tung, and Taborek, 1985:

$$\mathrm{Re}_{\mathrm{eq}} = \mathrm{Re}_{\mathrm{air}} \frac{2\, L_{\mathrm{f}}}{d_{\mathrm{f}} - d_{\mathrm{o}}} \tag{3.81}$$

$$F_{\mathrm{air}} = \frac{L_{\mathrm{p}} - d_{\mathrm{f}}}{d_{\mathrm{o}}} \tag{3.82}$$

$$F_{\mathrm{Re}} = \frac{1 + 2\,\frac{\exp(-F_{\mathrm{air}}/4)}{1+F_{\mathrm{air}}}}{0.021 + \frac{27.2}{\mathrm{Re}_{\mathrm{eq}}} + \frac{0.29}{\mathrm{Re}_{\mathrm{eq}}^{0.2}}} \tag{3.83}$$

$$\Delta p_{\text{air}} = 2\, F_{\text{Re}}\, N_{\text{tp}}\, \rho_{\text{air}}\, v_{\text{air,max}}^2 \tag{3.84}$$

$$\Delta p_{\text{F}} = 1.2\, \Delta p_{\text{air}} \tag{3.85}$$

In Eq. (3.85) we follow the assumption from Serth, 2007 and account for 20% of additional losses caused by the support structure, screens, etc.

The number of fans is determined from the total width of the ACC, see Fig. 3.4. Dividing this width by two gives the width of each of the banks, and as the fan bays are approximately square, dividing by L_{t} and flooring yields the number of bays per bank. Doubling this value gives the total number of bays which by assumption is equal to the number of fans N_{F}, i.e:

$$W_{\text{tot}} = \frac{\sqrt{3}\, n_{\text{t}}}{2\, n_{\text{tp}}} L_{\text{p}} \tag{3.86}$$

$$N_{\text{F}} = 2 \left\lfloor \frac{W_{\text{tot}}}{2\, L_{\text{t}}} \right\rfloor \tag{3.87}$$

Fan costs are modeled as in Smith, 2005, with costs adjusted from US$$_{2000}$ to US$$_{2021}$:

$$C_{\text{I,F}} = 18\,991\ \text{US\$}\ N_{\text{F}} \left(\frac{P_{\text{F,nom}}}{50\,\text{kW}} \right)^{0.76} \tag{3.88}$$

3.2. Computational Results

We implemented the component models presented in Section 3.1 and aggregated them to a system model representing the considered ORC, using our open-source modeling framework COMANDO (Langiu et al., 2021). The source code is available under `examples\ORC_off-design` in the COMANDO repository. Based on this system model, we define multiple optimization problems considering TAR as the objective. By slight abuse of notation, we occasionally use T_{amb} instead of s in the following, as our operational scenarios are fully characterized by the considered ambient temperatures. As in Ghasemi et al., 2013b, we consider the ORC system to be built at a location exhibiting an ambient temperature distribution as shown in Fig. 3.6, with a weighted average temperature of $\overline{T}_{\text{amb}} = 15.85\,°\text{C}$ (289 K). The different problems we consider are solved using our open-source deterministic global optimization solver MAiNGO (Bongartz et al., 2018) on an Intel i7-8700 CPU (3.20 GHz) with 32 GB of RAM. Solutions to these problems consist of optimal design decisions and one set of optimal operational decisions for each considered

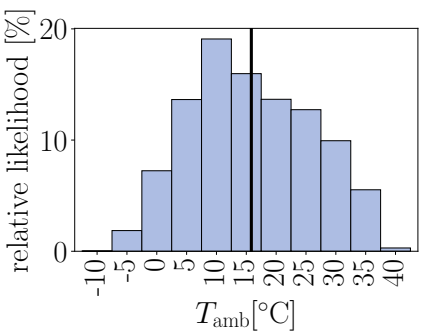


Fig. 3.6. Assumed distribution of daily ambient temperatures considered for the case study, adjusted from Ghasemi et al., 2013b. The vertical line depicts the weighted average of $\overline{T}_{\text{amb}} = 15.85\,°\text{C}$.

value of T_{amb}, according to Tab. 3.1.

In all performed optimizations, only a single local solution is found during preprocessing. Note that due to the nonconvexity, the problems may in principle have multiple local solutions. Given the large number of variables ($19 + 12 \times |\mathcal{S}|$), it is not possible to perform a dense search of the entire feasible space during preprocessing, and the ten randomly generated initial points might all converge to the same local minimum. To obtain a global solution, MAiNGO employs branch and bound, however, the level of detail considered in the system model results in a large formulation ($4+81 \times |\mathcal{S}|$ inequality, and $11 \times |\mathcal{S}|$ equality constraints) and as a consequence, in high computational cost. In particular, the upper bounds decrease very slowly for the considered optimizations, and the local solution from preprocessing is never improved, even for computation times up to one day.

Note that we use a reduced-space formulation, i.e., we directly use expressions describing intermediate quantities within other expressions instead of introducing optimization variables for them. A typical full-space formulation would introduce variables and add corresponding equality constraints for temperatures and enthalpies in all states, as well as for several other intermediate quantities, required for the calculation of component behavior. For the present problem this would amount to about $120 \times |\mathcal{S}|$ additional variables and constraints. While an even smaller reduced-space formulation than the used one is possible by eliminating the auxiliary variables A_{ACC}, p_{max}, P_{F}, $P_{\mathrm{F,nom}}$, P_{P}, $P_{\mathrm{P,nom}}$, K_{S}, $\dot{V}_{\mathrm{o,rel}}$, Δh_{rel}, P_{T}, and $P_{\mathrm{T,nom}}$, preliminary optimization results indicated that this does not improve computational time due to the resulting deterioration of relaxations.

To improve computational performance, we scaled all variables to a unit range and investigated different branching strategies. Compared to the default branching strategy, where all variables have branching priority 1, giving all design variables a higher priority of 2, 5, or 10 improves the performance somewhat, although the difference between choosing 5 and 10 is negligible. We found that a much better performance can be achieved by maintaining a priority 1 for the auxiliary variables mentioned above, whose value is determined by other variables or constraints, and setting the priority of all other variables to the heuristic value $1 + n^2$, where n is the number of functions each variable is present in.

3.2.1. Optimization for the average ambient temperature

Optimizing the system for any single ambient temperature results in a design that will generally be suboptimal, and may even be infeasible for other temperatures. To illustrate this point, we consider the optimization for a single operational scenario, corresponding to the weighted average of ambient temperatures from Fig. 3.6, i.e., $\mathcal{S} = \{\overline{T}_{\mathrm{amb}}\}, w_{\overline{T}_{\mathrm{amb}}} = 1$.

When fixing the optimal design decisions obtained for this average ambient temperature case, we can optimize the operational decisions for different ambient temperatures. For temperatures up to 15 °C MAiNGO finds a local optimum in preprocessing, and proves these local optima to be global within few seconds of branch and bound. The remaining 5 temperatures (cf. Fig. 3.6), however, are found to be infeasible within few seconds of branch and bound, also see Fig. 3.7. This is because the original optimization selects a design that is optimal for the average temperature but is operated at the boundary of the feasibility region for this temperature: The fans operate at their peak power and tube-side velocities for EVA and REC and shell-side velocities for ECO, REC, and SUP are at their maximum bounds for temperatures close to $\overline{T}_{\mathrm{amb}}$. As long as the brine mass flow $\dot{m}_{\mathrm{br}}$ is fixed to its maximum value, the states of the working fluid are constrained to remain identical for all

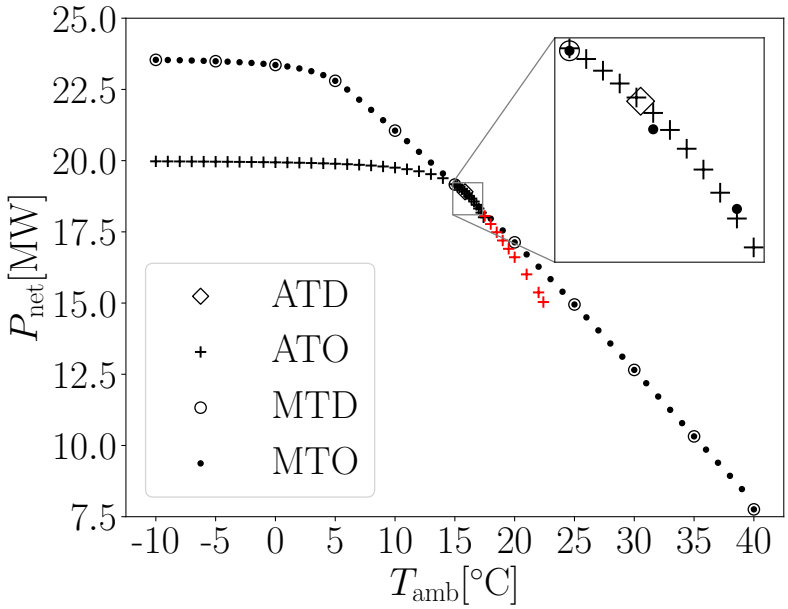


Fig. 3.7. Comparison of performance for the designs resulting from optimizations considering the average temperature only (ATD), or all 11 temperatures from Fig. 3.6 (MTD). The closeup shows that around the average temperature, the ATD slightly outperforms the MTD, however, running operational optimizations on a finer temperature resolution ($\leq 1\,\text{K}$) using the average (ATO) or multiple-temperature (MTO) design shows that the former is only feasible for a subset of all considered temperatures. When allowing for an increase in fluid speeds and a decrease in the brine mass flow, operation can be extended slightly (+). For ambient temperatures above 22.4 °C operation with the ATD is infeasible.

ambient temperatures. If the velocity limits are relaxed and $\dot{m}_{\text{br}}$ is allowed to be reduced by making it an operational variable, additional feasible points can be found for higher ambient temperatures (red crosses in Fig. 3.7). For ambient temperatures above 22.4 °C, however, operational problems remain infeasible. Similarly, optimizing the system for any other single operating scenario, results in designs that only allow for feasible operation close to the respective temperature. For example, optimizing for the maximum expected ambient temperature results in a design that becomes infeasible for ambient temperatures below 0 °C as Re_{air} approaches the lower limit of the validity range for the use of the Ganguli correlation, see Eq. (3.60).

3.2.2. Comparing results for single and multiple operating points

The results from the previous section suggest that we might find a more robust system design via an optimization for a single ambient temperature if we artificially restrict operational constraints, in order to obtain a more conservative design that leaves room for the necessary variations in operating points during off-design. While this may in fact produce a design that is feasible for all ambient temperatures, it is not clear which constraints to relax or how to balance conservatism and optimality. A more straightforward approach is to directly consider multiple operating points in a single optimization problem, e.g., as done in Yunt et al., 2008. By assigning appropriate weights to the objective contributions of different operational scenarios, the optimizer will automatically select the design that pro-

duces the best results based on the weighted average, also compare Fig. 3.7. A particular benefit of the presented model is that no explicit characterization of design and off-design operation is necessary. Instead, the design is automatically adjusted by the optimizer to account for all considered operating points.

Fig. 3.8 shows a qualitative comparison of the optimal variable values resulting from optimizations considering a single ambient temperature, each (ST), and an optimization that considers multiple operating points, corresponding to the 11 temperatures shown in Fig. 3.6 (MT), each weighted by the respective likelihood. It can be seen that some design quantities are similar or even identical for all cases while for other quantities, very different values are optimal depending on the considered ambient temperature. Furthermore, while the optimal design of the MT optimization is close to the weighted average of the ST designs for most quantities, the optimal value from the MT optimization for $d_{\mathrm{s,ECO}}$ is smaller and the $L_{\mathrm{t,SUP}}/d_{\mathrm{s,SUP}}$ and $L_{\mathrm{B,ECO}}/d_{\mathrm{s,ECO}}$ ratios are larger than in all ST cases. As with the optimal designs, the ranges of optimal operational values for the ST optimizations vary significantly for some of the variables. The solution of the MT optimization on the other hand exhibits approximately constant values for T_{amb} below 10 °C for all operational quantities, except for P_{F} and $\Delta T_{\mathrm{min,ACC}}$.

Generally, feasibility for operational scenarios not considered during the optimization determining the system design cannot be guaranteed. However, for the present case study the monotonicity of operational variable values for the MT case in Fig. 3.8 suggests that the MT design might be feasible for all ambient temperatures from −10 to 40 °C. Indeed, when fixing the design variables to the values from the MT solution and optimizing the operational strategies for intermediate temperatures in steps of 1 K, MAiNGO finds feasible solutions in all cases, also see Fig. 3.7. This is in contrast to the optimal design from the ST optimization considering only $\overline{T}_{\mathrm{amb}}$ (cf. Section 3.2.1), and demonstrates the robustness of the proposed approach.

3.2.3. Global optimization with reduced variable ranges

Even after 24 h of CPU time, relative gaps for the ST optimizations are between 13 and 124 %, and that of the MT optimization is at 121%. As a result, it is not clear whether a significantly better design is possible. To obtain better bounds on the optimal objective value, we repeat the optimizations from the previous section with the reduced ranges shown in Fig. 3.8.

As before, MAiNGO does not improve the feasible point found during preprocessing for any of the considered optimizations. The design values corresponding to the best feasible point of the MT optimization are given in Tab. 3.5. For the ST optimizations, both net power and overall efficiency increase almost linearly with decreasing temperature until −5 °C. In contrast, the values from the MT solution reach a plateau for temperatures below 10 °C in both cases, and always lie below those of the ST optimizations, see Fig. 3.9.

A well-known result from the literature on two-stage stochastic programming (see, e.g., Madansky, 1960) is that the weighted average of the globally optimal objective value of so-called *wait-and-see subproblems* (where it is assumed that separate first-stage decisions may be taken for each scenario) is an optimistic bound to the optimal objective value for the *here-and-now problem* (where first-stage decisions need to be taken before the realization of uncertain parameters). In the context of the present work, the first-stage decisions are

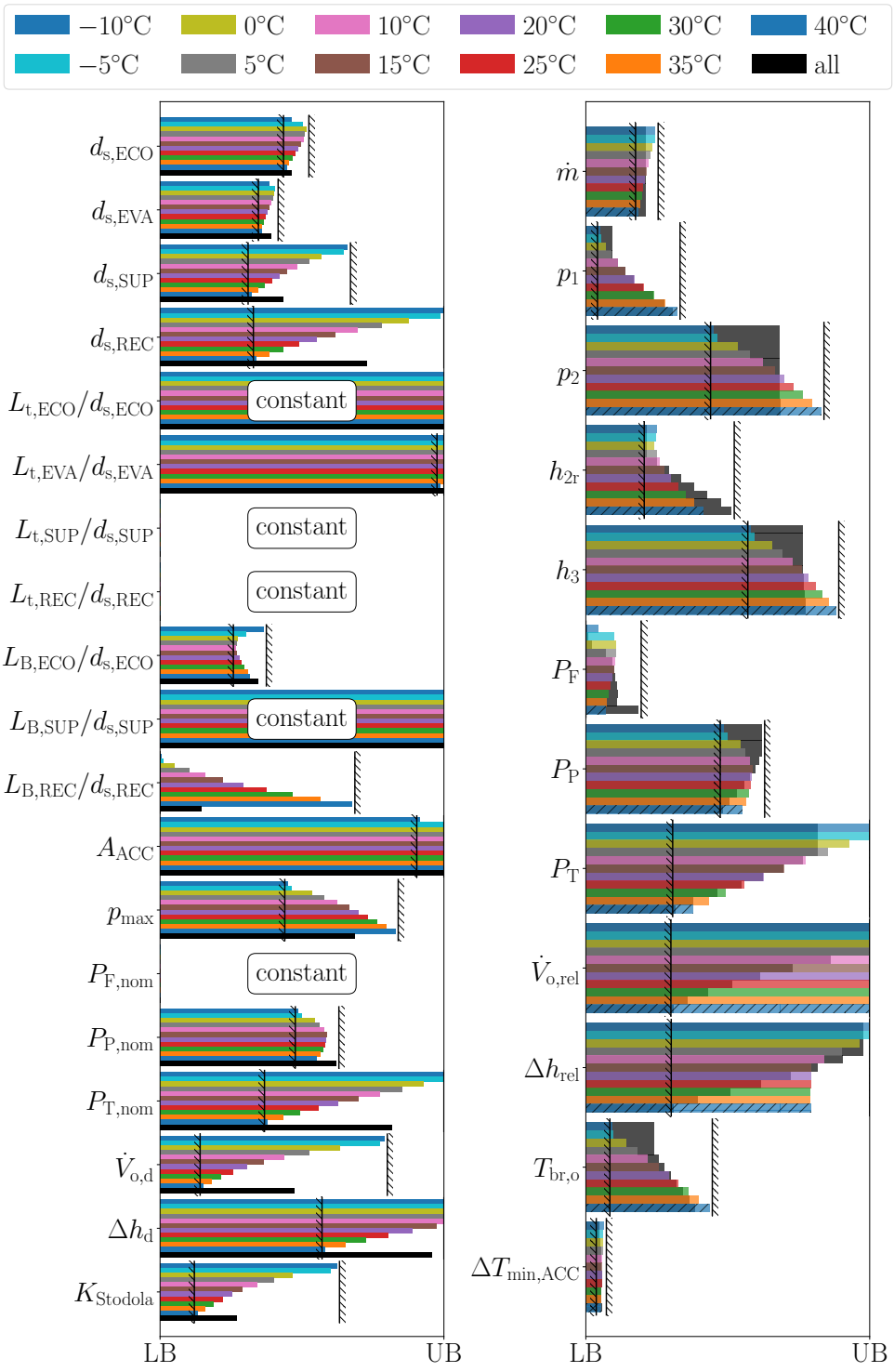


Fig. 3.8. Comparison of the variable ranges for single-temperature and multiple-temperature (all) optimizations, left: design, right: operation with results for single-temperature optimizations superimposed over those for the multiple-temperature optimization. A reduction of the considered variable ranges is proposed based on the results and indicated via hatched vertical lines (left hatch: new lower bound, right hatch: new upper bound). Design quantities for which the overall variation is less than 1% of the original range are considered constant, i.e., they are fixed to the value of the multiple-temperature optimization.

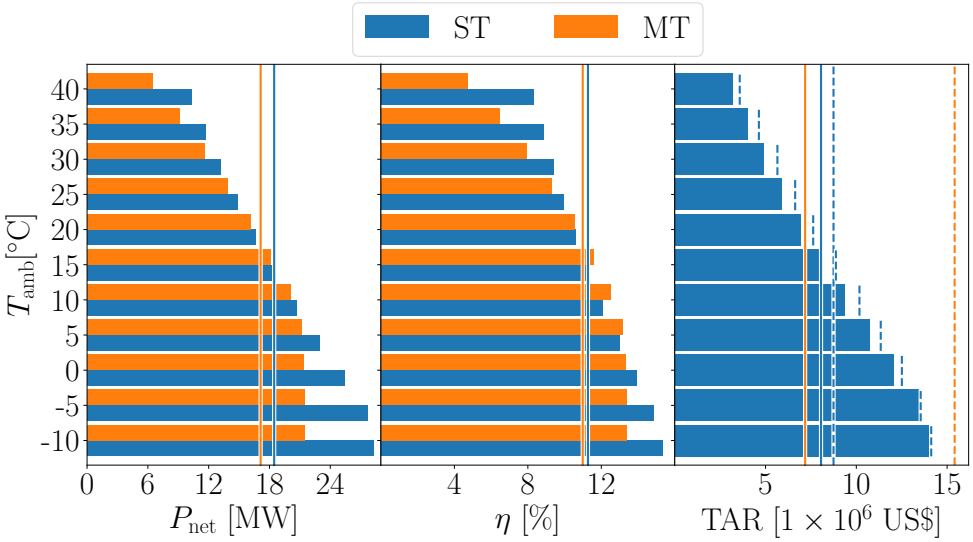


Fig. 3.9. Comparison of the net power P_{net}, total efficiency $\eta = P_{\text{net}}/\dot{Q}_{\text{br}}$, and TAR resulting from single-temperature (ST) and multiple-temperature (MT) optimizations. Solid vertical lines correspond to weighted averages, dashed vertical lines to upper bounds. The dashed blue line extending over the full height corresponds to the wait-and-see bound obtained by taking the weighted average of the ST bounds.

design decisions, the *wait-and-see problems* correspond to the ST design problems and the *here-and-now problem* corresponds to the MT problem. In lack of a globally optimal objective value, the upper bounds on the ST problems can be used to obtain the wait-and-see bound. In the following we show that within the same time-frame, this wait-and-see bound can be much tighter than the upper bound from the MT optimization.

After 24 h, the upper bound for the TAR obtained from the direct optimization of the MT problem (dashed orange line in the right plot in Fig. 3.9) is still 93% larger than the value for the best solution with a TAR of $7.81 \cdot 10^6$ US$/a (solid orange line in the right plot in Fig. 3.9). In contrast, the bound obtained by taking the weighted average of the upper bounds from the ST optimizations (dashed blue line in the right plot in Fig. 3.9) is only 11% larger than the best found TAR. Note that while multiple ST optimizations are necessary to generate this wait-and-see bound, it does not necessarily take more time than generating the direct bound from the MT problem, as the ST problems (and the MT problem) may be solved in parallel on separate CPUs.

The solid blue vertical line in the right plot in Fig. 3.9 corresponds to the lowest possible bound we can hope to obtain using the wait-and-see approach. It would be obtained, if the bounds of the subproblems converged to the respective objective values of the best feasible solutions. Unlike the MT optimization, the wait-and-see-approach is therefore not guaranteed to converge. If no better feasible point is found, the best possible wait-and-see bound would result in a gap of 7%. It is however possible to improve the wait-and-see approach via a decomposition algorithm as has been done by, e.g., Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b.

Tab. 3.5. Numerical results from the MT optimization. Top: geometry of the shell and tube exchangers, middle: remaining design variables, bottom: TAR and bounds (in million US$/a). Note that baffle spacing for the evaporator is missing as it does not occur in the considered correlations.

HX	$d_{\mathrm{s,HX}}$ [m]	$\frac{L_{\mathrm{t,HX}}}{d_{\mathrm{s,HX}}}$ [–]	$\frac{L_{\mathrm{B,HX}}}{d_{\mathrm{s,HX}}}$ [–]
ECO	1.43	12	0.476 5
EVA	1.285	12	–
REC	1.959	4	0.315 9
SUP	1.366	4	1

A_{ACC}	$P_{\mathrm{c,nom}}$ [MW]			$\dot{V}_{\mathrm{T,o,d}}$	$\Delta h_{\mathrm{T,d}}$	K_{S}
[m^2]	F	P	T	[m^3/s]	[kJ/kg]	[cm^2]
1×10^6	0.037	1.28	12.44	24.13	62.66	20.83

TAR	wait-and-see bound	direct bound
7.81	8.70	15.1

3.3. Conclusion

In this chapter, we presented a detailed model of an air-cooled geothermal ORC, simultaneously taking into account system design as well as operational decisions for multiple operational scenarios. Through the incorporation of ANNs as data-driven surrogate models, accurate working fluid properties as well as component characteristics are incorporated while maintaining computational tractability. We implement this model using our open-source modeling framework COMANDO and formulate a mathematical programming problem, maximizing total annualized revenue. We solve several instances of this problem for different sets of operating scenarios, corresponding to single or multiple ambient temperatures using our open-source global optimization solver MAiNGO. If only a single temperature is considered, e.g., the average or maximum ambient temperature, we obtain a system design that enables optimal operation for that temperature, but becomes infeasible for temperatures that are far from the considered one. In contrast, considering multiple operating points along with their relative likelihood during the optimization results in a robust design that can be operated over the entire operating range and allows for an operation providing maximum expected total annualized revenue.

Our contribution is twofold: first, we demonstrate the importance of considering multiple operating points within design problems, instead of a single one. Second, we show that single-temperature optimizations can still provide valuable bounding information that can be used to improve the bound obtained from multiple-temperature optimization. For the present case study, we are able to reduce the upper bound on the total annualized return from 93% to 11%.

Given the significant uncertainties in early cost estimates, the resulting solution can be deemed sufficiently accurate. However, it is unlikely that solutions of similar accuracy

can be obtained for other simultaneous design and operation problems in general. In such cases, obtaining solutions of sufficient quality in reasonable time may require the use of decomposition algorithms, which exploit the special problem structure for more a efficient solution. In the following chapter, we therefore review existing decomposition methods, identify potential issues with the state of the art, and develop a new decomposition algorithm applicable to general nonconvex design and operation problems.

4. MUSE-BB: A Decomposition Algorithm for Nonconvex Two-Stage Problems using Strong Multisection Branching

In this chapter, we develop a new decomposition algorithm, applicable to solve the two-stage stochastic programming problem TSP, introduced in Chapter 1. In particular, we consider the general case, where any of the participating functions may be nonconvex in both the first- and the second-stage variables. The generality of this problem class makes our algorithm ideally suited for the technical design and operation of energy systems, which may often include significant nonconvexities in both the first- and second-stage.

For ease of exposition, we restate the problem TSP here, considering the individual stages separately. The first-stage corresponds to the original two-stage problem.

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}} \coloneqq \min_{\boldsymbol{x}\in\mathcal{X}} \; & f_{\mathrm{I}}(\boldsymbol{x}) + \sum_{s\in\mathcal{S}} w_s \, f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x}) \\ \text{s.t. } & \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \end{aligned} \tag{TSP$^{\mathcal{X},\mathcal{Y}}$}$$

whereas the second-stage corresponds to the *recourse problem* (RP) for a fixed value of $\boldsymbol{x}$, and second-stage domain $\mathcal{Y}_s$:

$$\begin{aligned} f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x}) \coloneqq \min_{\boldsymbol{y}_s\in\mathcal{Y}_s} \; & f_{\mathrm{II},s}(\boldsymbol{x},\boldsymbol{y}_s) \\ \text{s.t. } & \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x},\boldsymbol{y}_s) \leq \boldsymbol{0}. \end{aligned} \tag{RP$_s^{\mathcal{Y}_s}(\boldsymbol{x})$}$$

In contrast to the original formulation, we introduced the *optimal value* $f^{\mathcal{X},\mathcal{Y}}$, the *second-stage optimal value functions* $f_{\mathrm{II},s}^{\mathcal{Y}_s}$ (also known as *optimal recourse functions*), as well as the explicit dependence of these functions and the associated optimization problems on the domains $\mathcal{X} \in \mathbb{IR}^{N_x}$, $\mathcal{Y}_s \in \mathbb{IR}^{N_y}$, and $\mathcal{Y} \in \mathbb{IR}^{N_s N_y}$, which are assumed to be bounded hyperrectangles, and thus compact sets. Throughout this work, we use the notation $\mathbb{I}\bullet$ to denote the set of nonempty, hyperrectangle subsets of some set $\bullet \subseteq \mathbb{R}^m$.

Here we use the overall domain $\mathcal{Y} \coloneqq \bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s$ for conciseness. Throughout this chapter, we omit such parametric dependencies from the labels of optimization problems, if they are immaterial, e.g., we occasionally still use the simpler notation TSP. As before, the decisions that need to be made in the first and second stage are captured by the variable vectors $\boldsymbol{x}$ and $\boldsymbol{y}_s$, and the functions $f_{\mathrm{I}} : \mathcal{X} \mapsto \mathbb{R}$ and $f_{\mathrm{II},s} : \mathcal{X} \times \mathcal{Y}_s \mapsto \mathbb{R}$ denote the scalar-valued first- and second-stage objective functions, and $\boldsymbol{g}_{\mathrm{I}} : \mathcal{X} \mapsto \mathbb{R}^{N_{\mathrm{I}}}$ and $\boldsymbol{g}_{\mathrm{II},s} : \mathcal{X} \times \mathcal{Y}_s \mapsto \mathbb{R}^{N_{\mathrm{II}}}$ the vector-valued first- and second-stage constraint functions. The set of considered scenarios $\mathcal{S}$ is assumed to have finite cardinality $N_s \coloneqq |\mathcal{S}| \geq 1$.

Since our formulation of $\text{TSP}^{\mathcal{X},\mathcal{Y}}$ is motivated by the application to design and operation of energy systems, it assumes an equal number of second-stage variables N_y, and second-stage constraints N_{II}, to be equal for all scenarios (which represent different operational conditions in our context). While the generalization to different numbers $N_{y,s}$, and $N_{\text{II},s}$, for each scenario s does not pose substantial complication, we herein restrict our attention to the simpler case for ease of exposition. Furthermore, we effectively assume that the weights w_s correspond to probabilities, i.e., $w_s \in (0,1], \sum_{s\in\mathcal{S}} w_s = 1$. This assumption makes the sum in the objective a convex combination, allowing for more concise definitions and proofs. Note that other weights can be equivalently used via appropriate scaling and redefinition of the objective. For conciseness, we aggregate the vectors $\boldsymbol{y}_s$ into the overall vector of second-stage variables $\boldsymbol{y} \in \mathcal{Y}$:

$$\boldsymbol{y} \coloneqq \begin{pmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_{N_s} \end{pmatrix} = \begin{pmatrix} y_{1,1} \\ \vdots \\ y_{1,N_y} \\ \vdots \\ y_{N_s,N_y} \end{pmatrix} \tag{$\boldsymbol{y}$}$$

We denominate any scalar element $y_{s,i}$ of $\boldsymbol{y}$ or $\boldsymbol{y}_s$ as a *second-stage variable* and refer to the collection of elements at the same position i in $\boldsymbol{y}_s$ for different values of s as *instances of a second-stage variable*. Furthermore, we define the *scenario objective functions* $f_s : \mathcal{X} \times \mathcal{Y}_s \mapsto \mathbb{R}$

$$f_s(\boldsymbol{x}, \boldsymbol{y}_s) \coloneqq f_{\text{I}}(\boldsymbol{x}) + f_{\text{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \tag{f_s}$$

and the *overall objective function* $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$

$$\begin{aligned} f(\boldsymbol{x}, \boldsymbol{y}) &\coloneqq f_{\text{I}}(\boldsymbol{x}) + \sum_{s\in\mathcal{S}} w_s\, f_{\text{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \\ &= \sum_{s\in\mathcal{S}} w_s (f_{\text{I}}(\boldsymbol{x}) + f_{\text{II},s}(\boldsymbol{x}, \boldsymbol{y}_s)) \\ &= \sum_{s\in\mathcal{S}} w_s\, f_s(\boldsymbol{x}, \boldsymbol{y}_s), \end{aligned} \tag{f}$$

where the equalities follow from our assumptions on the weights w_s.

Using these definitions, $\text{TSP}^{\mathcal{X},\mathcal{Y}}$ can be equivalently stated as the following single-stage optimization problem, also known as the *'extensive form'* or the *'deterministic equivalent'*:

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}} = \min_{\substack{\boldsymbol{x}\in\mathcal{X} \\ \boldsymbol{y}\in\mathcal{Y}}} & f(\boldsymbol{x}, \boldsymbol{y}) \\ \text{s.t. } & \boldsymbol{g}_{\text{DE}}(\boldsymbol{x}, \boldsymbol{y}), \end{aligned} \tag{$\text{DE}^{\mathcal{X},\mathcal{Y}}$}$$

where the vector-valued constraint function $\boldsymbol{g}_{\text{DE}} : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^{N_g^{\text{DE}}}$, groups all

$N_g^{\mathrm{DE}} \coloneqq N_{\mathrm{I}} + N_s N_{\mathrm{II}}$ constraints in DE, and is defined as

$$\boldsymbol{g}_{\mathrm{DE}}(\boldsymbol{x}, \boldsymbol{y}) = \begin{pmatrix} g_{\mathrm{DE},1}(\boldsymbol{x}, \boldsymbol{y}) \\ \vdots \\ g_{\mathrm{DE},N_g^{\mathrm{DE}}}(\boldsymbol{x}, \boldsymbol{y}) \end{pmatrix} \coloneqq \begin{pmatrix} g_{\mathrm{I},1}(\boldsymbol{x}) \\ \vdots \\ g_{\mathrm{I},N_{\mathrm{I}}}(\boldsymbol{x}) \\ g_{\mathrm{II},1,1}(\boldsymbol{x}, \boldsymbol{y}_1) \\ \vdots \\ g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\boldsymbol{x}, \boldsymbol{y}_{N_s}) \end{pmatrix}. \tag{$\boldsymbol{g}_{\mathrm{DE}}$}$$

The two problems $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ and $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ are equivalent in the sense that their globally and locally optimal solution points and optimal objective values coincide if they exist, whereas if one of the formulations is infeasible or unbounded, so is the other, see e.g., (Yunt et al., 2008). We are interested in the case where all functions in DE may be nonconvex. We limit the theoretical considerations, implementation, and numerical results to continuous variables. Thus, we do not explicitly address issues pertaining to discrete variables in the following. The presence of discrete variables would however not pose substantial complication.

The remainder of this chapter is structured as follows: Section 4.1 revisits solution approaches for TSP, highlights challenges of existing methods, and motivates the development of a new decomposition approach. Section 4.2 briefly reviews decomposable bounding subproblems used in scenario decomposition algorithms for TSP. In Section 4.3 we motivate the use of multisection branching of second-stage variables. Following this, we outline two alternative variants of multisection that allow for efficient incorporation of decomposable bounding problems in a B&B algorithm, branching on both $\boldsymbol{x}$ and $\boldsymbol{y}$. Section 4.4 presents the MUSE-BB algorithm, incorporating one variant of multisection branching. It includes implementation details followed by a formal statement of the MUSE-BB algorithm and subroutines. In Section 4.5 we present convergence results for both our lower bounding problems, and the overall algorithm. We show that under mild conditions MUSE-BB converges to an ε_f-optimal solution in finite time for any $\varepsilon_f > 0$. Section 4.6 presents the results of computational experiments on a small test problem, highlighting the effect of different parameters on MUSE-BB, and Section 4.7 concludes this chapter.

4.1. Solution Approaches: Challenges of Existing Methods and New Ideas

Solving $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ by applying general-purpose branch and bound (B&B) solvers (e.g., Androulakis, Maranas, and Floudas, 1995; Vigerske and Gleixner, 2017; Bongartz et al., 2018; Belotti, 2019; Sahinidis, 2024) to $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ is possible, typically amounting to solution of relaxations of $\mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ in every B&B node. Here $\mathcal{X}^n \in \mathbb{I}X$ and $\mathcal{Y}^n \in \mathbb{I}Y$, where $\mathbb{I}X$ and $\mathbb{I}Y$ denote the sets of nonempty, compact interval subsets of $\mathcal{X}$ and $\mathcal{Y}$. However, as B&B is intrinsically exponential in the number of (branched) variables, this approach has worst-case exponential runtime in the number of scenarios. This has motivated the development of decomposition algorithms capable of exploiting the special structure of TSP for a more efficient solution. In these algorithms, multiple independent subproblems are solved instead of instances of DE, which can result in a reduction of computational time required

for the solution, as the subproblems are generally much smaller and thus cheaper to solve. In the best case, such decomposition algorithms achieve linear scaling with the number of scenarios N_s, i.e., an arithmetic complexity of $\mathcal{O}(N_s)$. Furthermore, the subproblems are independent and may thus be solved in parallel, resulting in significant additional reductions of wall time.

Historically, decomposition strategies have predominantly been developed for certain subclasses of TSP, e.g., those restricted to linear functions and either only continuous (e.g., Dantzig and Wolfe, 1960; Benders, 1962) or mixed-integer variables (e.g., Laporte and Louveaux, 1993; Carøe and Schultz, 1999), or those restricted to convex nonlinear functions (e.g., Generalized Benders Decomposition (GBD) Geoffrion, 1972). More recently, algorithms addressing subclasses of TSP allowing for certain nonconvexities, but imposing additional structural assumptions have also been proposed (e.g., Li, Tomasgard, and Barton, 2011; Li, Sundaramoorthy, and Barton, 2014; Li and Cui, 2024). In the most general case, any of the functions in TSP may be nonconvex, and no additional structural assumptions are imposed. Two algorithm variants addressing this case are proposed by Ogbe and Li, 2019, however, both variants consider elements of $\boldsymbol{y}$ which introduce nonconvexity as complicating variables in addition to $\boldsymbol{x}$. Thus, in the worst case subproblems have a similar size as the original problem, diminishing the benefits of decomposition.

Three further recent algorithms all employ B&B exclusively on the first-stage variables: (*i*) Kannan, 2018 propose a modified Lagrangian relaxation in which so called nonanticipativity constraints (cf. Section 4.2) are dualized. The resulting Lagrangian problem is thus still a nonconvex two-stage problem but exhibits additional structure and can thus be solved in a decomposable manner using the algorithm proposed by Li, Tomasgard, and Barton, 2011; Li, Sundaramoorthy, and Barton, 2014. As a result, only the continuous first-stage variables need to be branched. (*ii*) Cao and Zavala, 2019 propose another B&B algorithm that obtains lower bounds in each node via global solutions to separate, but generally nonconvex scenario subproblems, resulting from simply dropping the nonanticipativity constraints. (*iii*) Li and Grossmann, 2019b use mixed integer linear or convex mixed integer nonlinear relaxations based on DE as lower bounding problems, which are solved via GBD. Cuts from Lagrangean subproblems are added to a Benders master problem and cutting planes for convexification are added to the Benders subproblems. All three algorithms (Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b), solve N_s independent subproblems on $\mathcal{X}^n \times \mathcal{Y}_s$ at each B&B node n, where $\mathcal{X}^n \in \mathbb{I}X$. While this implies that the computational work of the bounding operation scales linearly in N_s, and further, that the subproblems can be solved in parallel, linear scaling of the overall algorithms with N_s would additionally require that the number of nodes in the outer B&B search is independent of the number of scenarios. Note, however, that within a family of problems with variable number of scenarios, the quality of the lower bounds can be expected to depend on the number of scenarios. For a given tolerance, the number of nodes visited by the outer B&B search may therefore depend on the number of scenarios, despite branching only on $\boldsymbol{x}$. Of course, a similar argument also holds for other algorithms that are commonly thought to scale linearly with the number of scenarios, e.g., classical Lagrangian dualization for convex problems. While such algorithms are typically much more efficient for solving DE compared to general-purpose B&B, and empirically do exhibit linear scaling, they have not been rigorously proven to scale linearly with the number of scenarios in the general case.

Recently Robertson, Cheng, and Scott, 2020 observed that all three algorithms, ad-

dressing general nonconvex instances of TSP (Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b) fall into the category of *projection-based decomposition algorithms* (PBDAs). Algorithms in this category directly solve $\text{TSP}^{\mathcal{X},\mathcal{Y}}$ (which can be considered a projection of $\text{DE}^{\mathcal{X},\mathcal{Y}}$ onto the $\mathcal{X}$ space) by considering only the first-stage variables via second-stage optimal value functions $f_{\text{II},s}^{\mathcal{Y}_s}$. Robertson, Cheng, and Scott, 2024 argue that this approach likely suffers from the cluster effect, a phenomenon of some spatial B&B algorithms, where a large number of nodes may need to be visited near approximate global minimizers (Kearfott and Du, 1993; Du and Kearfott, 1994; Wechsung, Schaber, and Barton, 2014). To avoid this effect, the relaxations of both objective and constraints need to have a sufficiently high convergence order (Kannan and Barton, 2017b). Note that throughout the article we refer to convergence order in the sense of Hausdorff, unless stated otherwise. The convergence order of relaxations typically used in algorithms for (mixed-integer) nonlinear programs has been analyzed in a series of articles (cf. Bompadre and Mitsos, 2011; Najman and Mitsos, 2016a; Kannan and Barton, 2017b; Cao, Song, and Khan, 2019). Robertson, Cheng, and Scott, 2024 show that as a result of performing search in the $\mathcal{X}$ domain only, PBDAs need to construct relaxations of the so-called *scenario value functions*:

$$f_s^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{x}) \coloneqq \begin{cases} f_{\text{I}}(\boldsymbol{x}) + f_{\text{II},s}^{\mathcal{Y}_s}(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s} \\ +\infty, & \text{otherwise} \end{cases},$$

where $\mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s}$ are the feasible subsets of $\mathcal{X}$ in scenario s:

$$\mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s} \coloneqq \{\boldsymbol{x} \in \mathcal{X} \mid \boldsymbol{g}_{\text{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \exists \boldsymbol{y}_s \in \mathcal{Y}_s : \boldsymbol{g}_{\text{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \leq \boldsymbol{0}\}.$$

Adopting the convention for the minimum of an infeasible problem to be infinite, the weighted sum over the scenario value functions is equivalent to the objective of TSP. Robertson, Cheng, and Scott, 2024 demonstrate that only branching on $\boldsymbol{x}$ generally causes $f_s^{\mathcal{X},\mathcal{Y}_s}$ to be nonsmooth, which in turn limits the achievable convergence order. In particular, even the ideal PBDA, which uses the tightest-possible relaxation for each $f_s^{\mathcal{X},\mathcal{Y}_s}$, i.e., the convex envelope, generally has a convergence order below 1, and only achieves first-order convergence if all $f_s^{\mathcal{X},\mathcal{Y}_s}$ are Lipschitz. On the other hand, they show that this ideal relaxation has second-order convergence if $f_s^{\mathcal{X},\mathcal{Y}_s}$ are twice continuously differentiable, and furthermore, that the algorithm of Li and Grossmann, 2019b is equivalent to using this ideal relaxation, if optimal dual multipliers $\boldsymbol{\lambda}_s^*$ are used. Note that in general, generating convex envelopes of arbitrary $f_s^{\mathcal{X},\mathcal{Y}_s}$ (via optimal dual multipliers or otherwise) is prohibitively expensive. Furthermore, even for convex f, $\boldsymbol{g}_{\text{I}}$ and $\boldsymbol{g}_{\text{II},s}$, and even in the absence of discrete variables, the $f_s^{\mathcal{X},\mathcal{Y}_s}$ are not guaranteed to be smooth, but rather only lower semi-continuous (cf., e.g., Theorem 35, Chapter 3 of Birge and Louveaux, 2011). In summary, using PBDAs, i.e., branching on $\boldsymbol{x}$ only, limits convergence order to below one in general. As a result Robertson, Cheng, and Scott, 2024 state that PBDAs are expected to suffer from clustering, and suggest to search for alternative decomposition approaches, rather than for better relaxations in PBDAs. While a higher convergence order can certainly be advantageous, we point out that this conclusion might be overly pessimistic, as the occurrence of clustering is determined by the interplay of both convergence order, and growth order of the objective and constraint functions (also see Kannan and Barton, 2017b).

Nevertheless, the three aforementioned PBDAs (Kannan, 2018; Cao and Zavala, 2019;

Li and Grossmann, 2019b) may potentially have further issues. First, for each node n with domain $\mathcal{X}^n \in \mathbb{X}$, visited by the outer B&B algorithm searching on $\mathcal{X}$, an inner algorithm searches on $\mathcal{X}^n \times \mathcal{Y}_s$ during the solution of the subproblems. The consideration of $\boldsymbol{x}$ in both levels will therefore result in repeated consideration of the same domain, constituting a duplication of work. Second, in the general case, where there are nonconvexities in the second stage (through nonconvex objectives or constraints, or integer variables), the lower bounding subproblems must at least occasionally be solved globally to guarantee convergence. In addition, Kannan, 2018 and Cao and Zavala, 2019 also solve their upper bounding problems globally, while Li and Grossmann, 2019b do not explicitly state whether their solutions are local or global. Finally, the nesting of these expensive bounding routines in an outer B&B algorithm, bears resemblance to early ideas for solving general mixed-integer nonlinear programming problems, which considered branching on the integer variables and globally solving a continuous nonconvex problem in each node. However, such ideas have been abandoned since nested exponential approaches are considered computationally unfavorable (Smith and Pantelides, 1997).

To improve convergence orders of the relaxations, and to avoid duplication of work and the nesting of expensive search routines, we propose an alternative decomposition algorithm for TSP. Similar to solving DE via a classical B&B algorithm, we explicitly branch on first- and second-stage variables, however, we still make use of the structure inherent to TSP to obtain decomposable bounding subproblems for each scenario. We call our proposed algorithm *MUSE-BB*, as it combines classical scenario decomposition with **mu**lti**se**ction (Karmakar, Mahato, and Bhunia, 2009) in a **B&B** algorithm. Efficient branching on multiple instances of a particular second-stage variable is made possible by the fact that bounding subproblems for each scenario are independent of second-stage variable instances from other scenarios: While branching a node on N_s second-stage variables results in 2^{N_s} child nodes, only $2\,N_s$ independent subproblems need to be solved to update their lower bounds. Each child node can then be generated by combining bounds and variable domains from N_s out of the $2\,N_s$ independent subproblems. To limit memory requirements as well as the number of generated child nodes with poor lower bounds, we filter the N_s candidate bisections based on strong-branching scores, and allow for selecting a further subset of these bisections, ensuring an upper limit on the total number of generated child nodes.

Like classical B&B algorithms, MUSE-BB searches the full variable space. Thus in the worst-case, its runtime is expected to be exponential in N_s. However, the combination of decomposition with multisection allows for a more efficient exploration of the search space than with classical full-space algorithms. Moreover, we analyze the convergence order of the lower bounding scheme used in MUSE-BB. We show that while this convergence order is generally lower than in classical B&B algorithms, it is at least as high as in PBDAs, and can be strictly larger when the scenario value functions $f_s^{\mathcal{X},\mathcal{Y}_s}$ are not Lipschitz. In particular we show that the lower bounding scheme of MUSE-BB is (at least) first-order convergent if all functions and convex relaxations are Lipschitz. While our lower bounding scheme is generally not second-order convergent, we discuss a possible extension of MUSE-BB, whose lower bounding scheme achieves second-order convergence at unconstrained minimizers by dualizing nonanticipativity constraints instead of dropping them. Overall, the results indicate that MUSE-BB and its extension at least partially avoid issues with the cluster effect.

4.2. Decomposable Bounding Subproblems for TSP

In this section we review how bounds on TSP can be obtained from separate subproblems for each scenario. Since this approach trivially enables both parallelization and linear scaling of the computational work for bounding with N_s, its variants are the basis of many existing decomposition algorithms, as well as for MUSE-BB. The principal idea for decomposable bounding routines is that first-stage variables are complicating, because they appear in the objectives and constraints of all scenarios. Therefore, the problem can be decoupled by scenario, by either introducing independent copies of $\boldsymbol{x}$, or fixing its value. As shown in the following, these two cases result in subproblems which respectively provide lower and upper bounds on the optimal objective value $f^{\mathcal{X},\mathcal{Y}}$ of $\text{TSP}^{\mathcal{X},\mathcal{Y}}$.

An equivalent representation of $\text{DE}^{\mathcal{X},\mathcal{Y}}$ and thus $\text{TSP}^{\mathcal{X},\mathcal{Y}}$ is the lifting obtained by introducing a copy $\boldsymbol{x}_s$ of $\boldsymbol{x}$ for each scenario s and enforcing the equality of these copies, resulting in the following *nonanticipativity problem*.

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}} = \min_{\substack{\boldsymbol{x}_s \in \mathcal{X} \\ \boldsymbol{y} \in \mathcal{Y}}} & \sum_{s \in \mathcal{S}} w_s f_s(\boldsymbol{x}_s, \boldsymbol{y}_s) \\ \text{s.t.} \quad & \sum_{s \in \mathcal{S}} \boldsymbol{H}_s \boldsymbol{x}_s = \boldsymbol{0} \\ & \boldsymbol{g}_{\text{I}}(\boldsymbol{x}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S} \\ & \boldsymbol{g}_{\text{II},s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S}. \end{aligned} \tag{$\text{DE}^{\mathcal{X},\mathcal{Y}}_{\text{NAC}}$}$$

In DE_{NAC}, the first set of constraints enforces equality of all $\boldsymbol{x}_s$, thus, the coupling is moved to these so called *nonanticipativity constraints* (NACs), where $\boldsymbol{H}_s$ are appropriately shaped, sparse matrices.

For simplicity, we assume the following, specific form of the NACs, also used, e.g., in Li and Grossmann, 2019b:

$$\boldsymbol{x}_1 - \boldsymbol{x}_s = \boldsymbol{0} \quad \forall s \in \mathcal{S} \backslash \{1\}. \tag{NACs}$$

Due to the linearity of the NACs, dualizing them with $N_s - 1$ multiplier vectors $\boldsymbol{\pi}_s \in \mathbb{R}^{N_x}, s \in \mathcal{S}\backslash\{1\}$ removes the coupling, as it allows to define the vector $\boldsymbol{\lambda} \coloneqq (\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_{N_s})$, consisting of scenario-specific multiplier subvectors

$$\begin{aligned} \boldsymbol{\lambda}_1 &\coloneqq - \sum_{s \in \mathcal{S}\backslash\{1\}} \boldsymbol{\pi}_s / w_s, \\ \boldsymbol{\lambda}_s &\coloneqq \boldsymbol{\pi}_s / w_s \quad s \in \mathcal{S}\backslash\{1\}. \end{aligned} \tag{$\boldsymbol{\lambda}$}$$

Note that inherently,

$$\sum_{s \in \mathcal{S}} w_s \boldsymbol{\lambda}_s = \boldsymbol{0}. \tag{4.1}$$

The resulting dualization gives rise to the *Lagrangian relaxation*

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}}_{\text{LR}}(\boldsymbol{\lambda}) \coloneqq \min_{\substack{\boldsymbol{x}_s \in \mathcal{X} \\ \boldsymbol{y} \in \mathcal{Y}}} & \sum_{s \in \mathcal{S}} w_s \left[f_s(\boldsymbol{x}_s, \boldsymbol{y}_s) + \boldsymbol{\lambda}_s^{\intercal} \boldsymbol{x}_s \right] \\ \text{s.t.} \quad & \boldsymbol{g}_{\text{I}}(\boldsymbol{x}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S} \\ & \boldsymbol{g}_{\text{II},s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S}. \end{aligned} \tag{$\text{LR}^{\mathcal{X},\mathcal{Y}}$}$$

By weak duality, the value $f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda})$ provides a lower bound to $f^{\mathcal{X},\mathcal{Y}}$ for any $\boldsymbol{\lambda}$ satisfying Eq. (4.1) (cf. e.g., Dür and Horst, 1997). Furthermore, this bound can be obtained by solving the N_s separate *Lagrangian subproblems*

$$\begin{aligned} f_{\mathrm{LSP},s}^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{\lambda}_s) \coloneqq \min_{\substack{\boldsymbol{x}_s \in \mathcal{X} \\ \boldsymbol{y}_s \in \mathcal{Y}_s}} & f_s(\boldsymbol{x}_s, \boldsymbol{y}_s) + \boldsymbol{\lambda}_s^\intercal \boldsymbol{x}_s \\ \text{s. t. } & \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}_s) \leq \mathbf{0} \\ & \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \mathbf{0}, \end{aligned} \qquad \mathrm{LSP}_s^{\mathcal{X},\mathcal{Y}_s}$$

and calculating the Lagrangian relaxation based lower bound as

$$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}) \coloneqq \sum_{s \in \mathcal{S}} w_s \, f_{\mathrm{LSP},s}^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{\lambda}_s) \leq f^{\mathcal{X},\mathcal{Y}}. \tag{LRLB}$$

The best such bound is obtained by solving the Lagrangian dual, which can be written as

$$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}^*) \coloneqq \max_{\substack{\boldsymbol{\lambda} \in \mathbb{R}^{N_s N_x} \\ \sum_{s \in \mathcal{S}} \boldsymbol{\lambda}_s = \mathbf{0}}} f_{\mathrm{LSP},s}^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{\lambda}_s). \qquad \mathrm{L}^{\mathcal{X},\mathcal{Y}}$$

It can be shown that if the sets $\mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s}$ have a nonempty intersection, the resulting bound corresponds to the minimum of the weighted sum of convex envelopes of scenario value functions, (Robertson, Cheng, and Scott, 2024), i.e:

$$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}^*) = \min_{\boldsymbol{x} \in \mathcal{X}} \sum_{s \in \mathcal{S}} w_s \operatorname{conv} f_s^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{x}).$$

In that sense $f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}^*)$ constitutes the best bound obtainable via convex relaxation in the framework of scenario decomposition. Unfortunately, obtaining optimal dual multipliers $\boldsymbol{\lambda}^*$ is both computationally expensive and numerically challenging (Oliveira et al., 2013). We therefore only consider the implications of updating the dual multipliers in Section 4.5, whereas in the remainder of this chapter, we focus on the simpler case, also considered by Cao and Zavala, 2019, where all multipliers are fixed to zero. In that case, the scenario relaxation of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ consists of N_s *scenario problems* of the form

$$\begin{aligned} f_{\mathrm{SP},s}^{\mathcal{X},\mathcal{Y}_s} \coloneqq \min_{\substack{\boldsymbol{x}_s \in \mathcal{X} \\ \boldsymbol{y}_s \in \mathcal{Y}_s}} & f_s(\boldsymbol{x}_s, \boldsymbol{y}_s) \\ \text{s. t. } & \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}_s) \leq \mathbf{0} \\ & \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \mathbf{0}. \end{aligned} \qquad \mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$$

In Section 4.4.1 we will introduce further subproblems, obtained from additional relaxations of SP_s. To distinguish the different optimal objective values, we use corresponding subscripts. The globally optimal objective values $f_{\mathrm{SP},s}^{\mathcal{X},\mathcal{Y}_s}$ of problems $\mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$ can be used to obtain a lower bound $f_{\mathrm{SP}}^{\mathcal{X},\mathcal{Y}}$ on the optimal objective value $f^{\mathcal{X},\mathcal{Y}}$ of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$, i.e.,

$$f_{\mathrm{SP}}^{\mathcal{X},\mathcal{Y}} \coloneqq \sum_{s \in \mathcal{S}} w_s \, f_{\mathrm{SP},s}^{\mathcal{X},\mathcal{Y}_s} \leq f^{\mathcal{X},\mathcal{Y}}. \tag{SPLB}$$

While the resulting first-stage solutions obtained for each scenario will generally differ from

each other, the bound can be made arbitrarily tight by exhaustive branching on $\boldsymbol{x}$. PBDAs like Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b use this fact: while they branch on $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ during the global solution of the subproblems SP_s, the outer B&B search only requires branching on $\boldsymbol{x}$ to ensure convergence. As shown by Robertson, Cheng, and Scott, 2024, however, the convergence order of such lower bounding schemes is inherently limited due to the nonsmoothness of $f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x})$, incurred by projection, also cf. TSP and Section 4.1.

Upper bounds on $f^{\mathcal{X},\mathcal{Y}}$ can generally be obtained by evaluating any feasible point. Fixing $\boldsymbol{x}$ to an arbitrary point $\widetilde{\boldsymbol{x}} \in \mathcal{X}$ that is feasible with respect to $\boldsymbol{g}_{\mathrm{I}}$, gives rise to N_s instances of RP_s. If each of these problems has at least one feasible point $\widetilde{\boldsymbol{y}}_s$, the function values $f_{\mathrm{II},s}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}_s)$ provide an upper bound on $f_{\mathrm{II},s}^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}})$, and thus the upper bounding function $\overline{f}$, defined as

$$\overline{f}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}) \coloneqq f_{\mathrm{I}}(\widetilde{\boldsymbol{x}}) + \sum_{s \in \mathcal{S}} w_s \, f_{\mathrm{II},s}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}_s) \geq f^{\mathcal{X},\mathcal{Y}} \tag{UB}$$

provides an upper bound on the optimal objective value $f^{\mathcal{X},\mathcal{Y}}$. Thus, given a candidate for $\widetilde{\boldsymbol{x}}$, values for $\widetilde{\boldsymbol{y}}_s$ can be obtained by local or global solutions of $\mathrm{RP}_s^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}})$. Candidates proposed in the literature are commonly based on the individual solutions $\boldsymbol{x}_s^*$ from the lower bounding subproblems. A common candidate is the (w_s-weighted) average $\widetilde{\boldsymbol{x}} = \boldsymbol{x}^{\mathrm{avg}} = \sum_{s \in \mathcal{S}} w_s \, \boldsymbol{x}_s^*$ (Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b). However, since the feasible set of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ is generally nonconvex, this point may be infeasible. An alternative candidate that is at least guaranteed to be feasible with respect to $\boldsymbol{g}_{\mathrm{I}}$, and $\boldsymbol{g}_{\mathrm{II},s^{\mathrm{rep}}}$, is $\widetilde{\boldsymbol{x}} = \boldsymbol{x}_{s^{\mathrm{rep}}}^*$, such that s^{rep} is a *representative* scenario for which $\boldsymbol{x}_{s^{\mathrm{rep}}}^*$ is closest to $\boldsymbol{x}^{\mathrm{avg}}$ with respect to the relative Euclidean distance, i.e,

$$s^{\mathrm{rep}} \in \arg\min_{s \in \mathcal{S}} \sum_{i=1}^{N_x} \left(\frac{x_i^{\mathrm{avg}} - x_{s,i}^*}{\overline{x}_i - \underline{x}_i} \right)^2, \tag{s^{rep}}$$

where $\overline{x}_i$, and $\underline{x}_i$ denote the original lower and upper bounds of x_i (Li and Grossmann, 2019b). Note that while $\boldsymbol{x}_{s^{\mathrm{rep}}}^*$ is trivially feasible in s^{rep}, it is generally not in other scenarios. Furthermore, if a candidate $\boldsymbol{x}_{s^{\mathrm{rep}}}^*$ does happen to be feasible, there is no guarantee that local solutions to the corresponding instances of RP_s are found.

As with any spatial B&B method, in the general nonconvex case, a guarantee to find a feasible point allowing for termination is only given if the feasible set of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ has a nonempty interior at a global minimizer, also compare with the analysis for single-stage programs in Kirst, Stein, and Steuermann, 2015. Furthermore, Example 3.1 in Kirst, Stein, and Steuermann, 2015, where upper bounds are obtained simply from feasible lower bounding solutions, shows that even when the interior is nonempty, certain combinations of problem instances, branching, and node selection rules can lead to sequences of lower bounding solutions that never include a feasible point. In such cases, an adaption of the tested candidates, such as the approach proposed by Kirst, Stein, and Steuermann, 2015 may be necessary. Unfortunately such approaches may not address the more general situation of an empty interior, e.g., due to the presence of equality constraints, although there are approaches that produce upper bounds without guaranteeing to find feasible points (Füllner, Kirst, and Stein, 2020). On the other hand, feasible, and even (approximately) globally optimal solutions can often be produced relatively easily for many applications.

Because of this, we neither implement the methods presented in Kirst, Stein, and Steuermann, 2015 in MUSE-BB, nor analyze this issue further. Instead we follow the common approach to perform upper bounding via local solutions from candidate points, and concentrate this work on the issues pertaining to lower bounding.

In summary, by solving instances of the separable subproblems $\mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$ and $\mathrm{RP}_s^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}})$, we can bound the desired optimal solution value of the original problem $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ from below and above:

$$f_{\mathrm{SP}}^{\mathcal{X},\mathcal{Y}} \leq f^{\mathcal{X},\mathcal{Y}} \leq \overline{f}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}).$$

Assuming that arbitrarily good feasible points are found during the successive partitioning of the variable domains, the bounds can be tightened until some satisfactory accuracy $\varepsilon_f > 0$ is reached. The upper bound $\overline{f}$ then serves as an ε-optimal solution to problem $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$. In the following section we present a special branching scheme that efficiently combines the decomposable subproblems with partitioning of both $\mathcal{X}$ and $\mathcal{Y}$.

4.3. Multisection Branching for Decomposable Bounding Schemes

To avoid several issues associated with the nested branching of PBDAs (cf. Section 4.1), we propose to combine decomposable bounding schemes with explicit branching of both first- and second-stage variables. As argued below, standard branching of individual variables would eliminate some of the benefits of decomposable bounding schemes. We therefore propose a special branching scheme that either partitions a single first-stage variable or multiple second-stage variable instances in each iteration. To refer to the partition elements containing the lower/upper part of a branched variable domain, we say the respective variable was *branched down/up*.

We first present the concept of multisection as used in the MUSE-BB in Section 4.3.1. Following this, in Section 4.3.2 we outline an alternative idea that may be seen as a hybrid between the variant presented in Section 4.3.1, and existing PBDAs.

4.3.1. Multisection in MUSE-BB

In a B&B algorithm for TSP using separable lower and upper bounding problems, branching on elements of $\boldsymbol{x}$ and $\boldsymbol{y}$ has different implications for the resulting nodes: each node n is characterized by the domains $\mathcal{X}^n \in \mathbb{I}\mathcal{X}$ and $\mathcal{Y}^n \in \mathbb{I}\mathcal{Y}$, where $\mathcal{Y}^n := \bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s^n$; $\mathcal{Y}_s^n \in \mathbb{I}\mathcal{Y}_s$. To obtain a lower bound on n, variants of the N_s subproblems $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$ are solved. While branching on an element of $\boldsymbol{x}$ bisects $\mathcal{X}^n$ into two subdomains, e.g., $\mathcal{X}^d$ and $\mathcal{X}^u$, which generally results in changed bound contributions from all subproblems (compare cases **a)** and **b** in Fig. 4.1), branching on an element of $\boldsymbol{y}$, e.g., $y_{s,i}$, only bisects the second-stage variable domain $\mathcal{Y}_s^n$ of the associated scenario s (compare cases **b)** and **c** in Fig. 4.1). Thus, if we were to only branch on $y_{s,i}$, each of the two resulting child nodes would have $N_s - 1$ unchanged subproblems with respect to n. An example for this situation is given by the case **c)** of Fig. 4.1. In the parallel setting, where at least two subproblems can be solved simultaneously, this implies that standard branching on second-stage variables leaves some processing capacity unused. In other words, we could only exploit the parallelizable solution of subproblems when processing nodes obtained from branching on

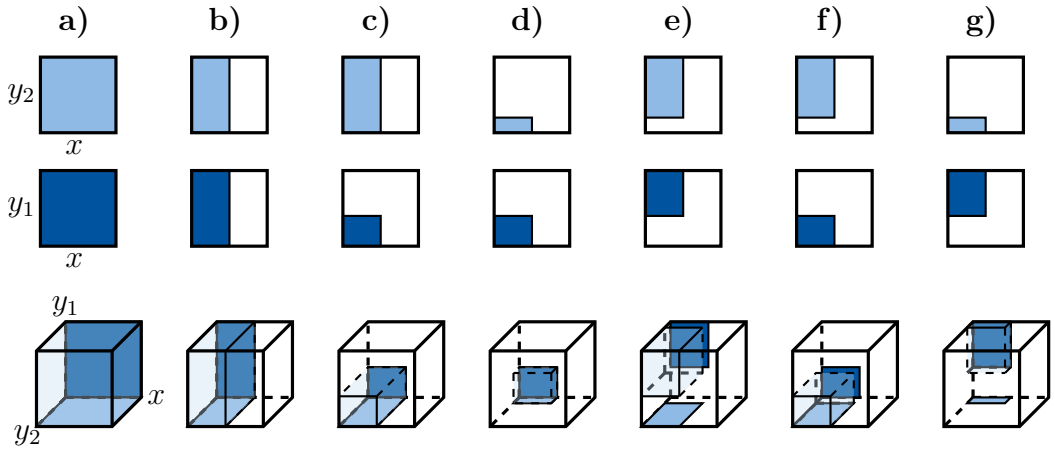


Fig. 4.1. Implications of branching in scenario decomposition. We consider nodes from solving an instance of DE with $N_x = N_y = 1$, and $N_s = 2$. In this case, each node corresponds to a 3D domain (bottom) and updating the lower bounds requires solving two bounding subproblems on a 2D domain (top). These subproblems can be considered projections on the $\mathcal{X} \times \mathcal{Y}_1$ and $\mathcal{X} \times \mathcal{Y}_2$ faces of the node domain (dark and light blue colors, respectively). **b)**: Branching the node from **a)** on x affects both subproblem domains. **c)**: Branching the node from **b)** on a single instance of y, here y_1, only affects the associated subproblem domain, while the subproblem for the second scenario remains unchanged. **d)** through **g)**: Alternatively to **c)**, branching on all instances of y simultaneously results in four nodes instead of two. However, out of the eight subproblems associated with these nodes only four are distinct. When processing two complementary nodes, e.g., node **d)** (where both y_1 and y_2 are branched down), and node **e)** (where both y_1 and y_2 are branched up), all distinct subproblems are solved. Thus, explicitly processing the remaining nodes, i.e., **f)** and **g)** in our example, is unnecessary. Instead, bounds for these nodes can be generated by combining the results from the subproblems solved for **d)** and **e)**.

first-stage variables.

To enable parallelism when processing nodes produced from second-stage branching, we can branch on all N_s instances of a particular second-stage variable, instead of a single one. Note that such a multisection is equivalent to N_s sequential bisections, i.e., it splits the original node into 2^{N_s} child nodes instead of two, also see the cases **d)** through **g)** of Fig. 4.1. Multisection has previously been used in different B&B algorithms for general nonlinear problems. Mostly this was in the form of branching the domain of a single variable at multiple points (also called "multisplitting") (Csallner, Csendes, and Markót, 2000; Markót, Csendes, and Csallner, 2000; Kazazakis, 2017), but there are also examples of using multisection in the present sense, i.e., branching once on multiple variables (Karmakar, Mahato, and Bhunia, 2009). While these works showed that multisplitting and multisection can result in better computational performance than bisection, the considered B&B algorithms used standard bounding procedures and thus needed to process all of the resulting nodes individually. In contrast, when solving TSP, the use of separable bounding subproblems such as SP_s^n allows us to generate bounds for the exponential number of nodes resulting from multisection without explicitly processing each one individually: for each scenario s, branching on the associated second-stage variable instance bisects the domain $\mathcal{Y}_s^n$ into two subdomains, $\mathcal{Y}_s^d$, and $\mathcal{Y}_s^u$. Combining these new domains with the unchanged domain $\mathcal{X}^n$ therefore results in two different subproblems (i.e., $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^d}$, and $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^u}$) per

scenario, i.e., multisection of second-stage variables only results in $2\,N_s$ distinct subproblems. Each child node simply corresponds to one of the possible combinations of selecting one of the two subproblems for each scenario. This means that to update lower bounds on all 2^{N_s} child nodes, only the $2\,N_s$ distinct subproblems need to be solved. Note that this can be achieved by processing any two of the 2^{N_s} nodes that contain complementary subproblems. One such choice consists of the pair of nodes resulting from branching all instances of the selected second-stage variable down, or up. In the following we respectively call these two nodes the *lower and upper sibling nodes*.

In summary, if a first-stage variable is selected for branching, we perform standard bisection resulting in two child nodes, whereas if a second-stage variable is selected, we instead perform multisection branching of all associated variable instances for different scenarios, resulting in 2^{N_s} nodes. In both cases, only two nodes need to be processed after branching a given node n: after first-stage branching, these two nodes are simply the child nodes with domains $(\mathcal{X}^d, \mathcal{Y}^n)$ and $(\mathcal{X}^u, \mathcal{Y}^n)$. After second-stage branching, we process the sibling nodes, with domains $(\mathcal{X}^n, \mathcal{Y}^d)$ and $(\mathcal{X}^n, \mathcal{Y}^u)$, where $\mathcal{Y}^d \coloneqq \bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s^d$ and $\mathcal{Y}^u \coloneqq \bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s^u$. While theoretically one could generate 2^{N_s} nodes after each second-stage branching, this poses several issues in an actual implementation. To address this, it is possible to filter the candidate bisections contributing to the final multisection, i.e., to keep only a "promising" subset and thus produce a small number of high quality nodes. The process we use for this will be presented in Section 4.4.4.

We point out that while the proposed multisection procedure may appear to avoid a computational cost for node processing that is exponential in N_s, this is only true at the level of an individual iteration. Whereas the cost for generating the child nodes from multisection branching is indeed linear in N_s, their number, and thus the overall computational cost for further processing is still exponential in N_s. In the following, we outline an alternative use of our multisection idea that may avoid this exponential scaling but bears resemblance to PBDAs. While we do not pursue this idea further in the present work, we believe it to be fruitful for future research.

4.3.2. Projected Multisection

The only conceivable path to avoid exponential scaling with N_s in the context of multisection-based second-stage branching is to avoid the explicit generation of the resulting child nodes. One approach for this is to maintain information related to second-stage variable domains and objective bounds at the subproblem level, without combining this information from different scenarios into individual B&B nodes. One can still compute a lower bound related to the first-stage domain by combining the lowest lower bounds from each scenario. Furthermore this bound can be refined by further partitioning of the resulting subproblem domains and organizing the resulting subproblem nodes in a separate B&B tree for each scenario. Similar to PBDAs, this results in 'nested' B&B trees: The outer tree contains nodes based on first-stage domains, each of which maintains a state of progress for the search in the second-stage domains and associated lower bounds in N_s separate second-stage trees. In contrast to existing PBDAs, branching of first- and second-stage variables is carried out exclusively in the outer and inner trees, respectively. This not only avoids duplication of first-stage variables, but also the exhaustive exploration of the second-stage trees in each iteration, since their state is passed on to child nodes from branching on first-stage variables in the outer tree. The number of nodes in the outer

tree is exponential in N_x. In the worst case, each such outer node is associated to N_s full second-stage trees, the size of which is exponential in N_y. Thus a total of $N_s a^{N_x} b^{N_y}$ nodes may need to be processed, i.e., it appears that this approach would avoid the exponential scaling with N_s and indeed scale linearly with N_s.

While this linear scaling appears promising, avoiding the generation of nodes from explicit combinations of subproblem data results in lower bounds based on the lowest lower bounds from the second-stage trees. Since this may be seen as a 'projection' of bounding information, it is unclear whether this approach can be considered a full-space or rather a projected search. We suspect the convergence order of this approach to be limited as with PBDAs, and therefore focus the remainder of this work on the previously presented idea.

4.4. Proposed Algorithm

We now present the spatial multisection B&B algorithm MUSE-BB for the solution of TSP$^{\mathcal{X},\mathcal{Y}}$. Algorithm 1 presents a formal statement of MUSE-BB; the relevant subroutines will be presented in the following. For conciseness, we assume throughout this section that given a node n, we have access to its domains $\mathcal{X}^n$ and $\mathcal{Y}^n$, and lower bound $\underline{f}^n$, as well as the domains $\mathcal{X}_s^n$ and $\mathcal{Y}_s^n$, and lower bounds $\underline{f}_s^n$ of the corresponding subproblems. Under this assumption, it suffices to provide nodes to the subroutines instead of all associated data. If a node n can be fathomed by infeasibility, we set its lower bound to ∞.

On a high level, MUSE-BB only differs from a standard B&B algorithm in the use of different kinds of iterations for nodes obtained from branching on first- and second-stage variables. In both cases, the bounds of unprocessed nodes are updated, and the nodes are either fathomed (by infeasibility or value dominance, i.e., $\underline{f}^n \geq \overline{f}$) or branched.

Each iteration begins with the selection of a node n from a list of nodes $\mathcal{N}$ (Algorithm 1 in Algorithm 1). The selected node is either an unprocessed node (the root node or one of the two child nodes obtained from standard bisection of a first-stage variable) which is processed via a *"normal iteration"*, similar as in a standard B&B algorithm (Algorithms 1–1), or it is a placeholder for two unprocessed sibling nodes that will be addressed via a special *"sibling iteration"* (Algorithms 1–1). In the latter case, n is the parent of the sibling nodes that was processed and multisected, (i.e., branched on N_s second-stage variables as presented in Section 4.3) in a previous iteration. In that case, there is an entry in $\mathcal{M}_{\text{sib}}$, mapping n to the sibling nodes d and u (Algorithm 1), which are processed together, using some of the bound and domain information from their parent, n (Algorithm 1). During this step we also search for an upper bound within the domain of the parent node. If possible, we use such a bound to update the best known upper bound (Algorithm 1) and if this does not allow the parent node to be fathomed (Algorithm 1), we use the results from the sibling iteration to generate processed child nodes whose number is exponential in the number of branched variables. However, instead of using all N_s bisections of the original multisection, we filter them, selecting only a subset for the final multisection (Algorithm 1, also see Section 4.4.4). We only consider branching via partitioning of the original domain through hyperplanes, orthogonal to the branched variable dimensions. Because of this and the related concept of an orthant, i.e., the intersection of k mutually orthogonal half-spaces in k-dimensional Euclidean space, we refer to the nodes resulting from the filtered multisection as *"orthant nodes"* in the following. The map $\mathcal{M}$ and the list $\mathcal{L}$, returned from the filtered multisection, determine the subproblem data (from n, d, or u) to be used for a

Algorithm 1: MUSE-BB

Input : Instance of $\text{TSP}^{\mathcal{X},\mathcal{Y}}$, tolerance ε_f, effective bisection limit $k_{\max}$, strong-branching threshold τ
Output: Incumbent point $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$, incumbent objective value $\overline{f}$, certificate $\underline{f}$

$n \leftarrow \mathcal{X} \times \mathcal{Y}$; $\underline{f}^n \leftarrow -\infty$; $\mathcal{N} \leftarrow \{n\}$; $\overline{f} \leftarrow \infty$; $\mathcal{M}_{\text{sib}} \leftarrow$ empty Map;
while $\mathcal{N} \neq \emptyset$ **do** // there are nodes to be processed
  $n \leftarrow$ select a node and remove from $\mathcal{N}$;
  if $n \in \mathcal{M}_{\text{sib}}$ **then** // do a "sibling iteration"
    // n is the previously processed parent node, re-entered into $\mathcal{N}$ as a placeholder for the sibling nodes
    $(d, u) \leftarrow \mathcal{M}_{\text{sib}}[n]$; // recover the sibling nodes to be processed
    // see Subroutine 3 in Section 4.4.3
    $(\underline{f}^n, \overline{f}^n, \boldsymbol{x}^n, \boldsymbol{y}^n) \leftarrow$ `processSiblings`(n, d, u);
    if $\overline{f}^n < \overline{f}$ **then** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}) \leftarrow (\boldsymbol{x}^n, \boldsymbol{y}^n, \overline{f}^n)$; // update best upper bound
    if $\underline{f}^n < \overline{f}$ **then**
      // see Subroutine 4 in Section 4.4.4
      $(\mathcal{M}, \mathcal{L}) \leftarrow$ `filteredMultiSection`$(n,\ d,\ u)$;
      foreach $i \in \{0, \cdots, 2^{|\mathcal{L}|} - 1\}$ **do**
        // see Subroutine 5 in Section 4.4.4
        $o \leftarrow$ `generateOrthantNode`$(i, n, d, u, \mathcal{M}, \mathcal{L})$;
        // see Subroutine 1 in Section 4.4.3
        if $\underline{f}^o < \overline{f}$ **then** `branchNode`(o);
      end
    end
  else // do a "normal iteration"
    $(\underline{f}^n, \overline{f}^n, \boldsymbol{x}^n, \boldsymbol{y}^n) \leftarrow$ `processNode`(n) ; // see Subroutine 2 in Section 4.4.3
    if $\overline{f}^n < \overline{f}$ **then** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}) \leftarrow (\boldsymbol{x}^n, \boldsymbol{y}^n, \overline{f}^n)$; // update best upper bound
    if $\underline{f}^n < \overline{f}$ **then** `branchNode`(n); // see Subroutine 1 in Section 4.4.3
  end
  $\underline{f} \leftarrow \min_{n \in \mathcal{N}} \underline{f}^n$; // update lowest lower bound
  if $\underline{f} + \varepsilon_f > \overline{f}$ **then return** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}, \underline{f})$;
end
return $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}, \underline{f})$;

particular orthant node o. The number k of orthant nodes to be generated is determined by the length of $\mathcal{L}$ (Algorithm 1). As the orthant nodes are already in a processed state, we immediately branch them, provided they cannot be fathomed (Algorithm 1).

In the course of the algorithm, unprocessed nodes are either fathomed or branched, until the lower and upper bounds converge to ε_f optimality (Algorithm 1) or the list $\mathcal{N}$ is exhausted (Algorithm 1, possibly indicating infeasibility of $\text{TSP}^{\mathcal{X},\mathcal{Y}}$). On termination, MUSE-BB either provides an incumbent $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$, with an associated objective value $\overline{f} = f(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$ that is at most ε_f larger than the global lower bound $\underline{f}$, or a certificate of infeasibility ($\underline{f} = \infty$).

We implement MUSE-BB as an extension of our deterministic global optimization solver and open-source project MAiNGO (Bongartz et al., 2018). In Sections 4.4.1–4.4.2 we detail how lower bounding and range reduction schemes available in MAiNGO are adapted to

subproblems from Section 4.2 to obtain the decomposable bounding schemes used in the processing subroutines. Since node processing comprises the main computational work, the respective subroutines are parallelized in our implementation. The main theoretical results we present in Section 4.5 do not depend on the presented bounding schemes, i.e., alternative ones may be employed analogously. Next, we discuss the branching of first- and second-stage variables (Subroutine 1) via standard bisection and the multisection from Section 4.3.1, and detail how the resulting nodes are respectively processed in "normal" and "sibling iterations" (Subroutines 2 and 3) in Section 4.4.3. Finally, we present the subroutines for the filtered multisection and orthant node generation in Section 4.4.4.

4.4.1. Lower and Upper Bounding

Our deterministic global solver MAiNGO (Bongartz et al., 2018) employs a general-purpose B&B algorithm with lower bounding problems obtained via McCormick-based relaxation techniques (McCormick, 1976; Tsoukalas and Mitsos, 2014; Villanueva, 2015; Chachuat et al., 2015; Najman and Mitsos, 2016a; Najman, Bongartz, and Mitsos, 2021). When solving TSP via equivalence to DE, we generate and solve such relaxations based on $\mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ for each node n. In the following, we abbreviate $\mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ as DE^n.

PBDAs like Cao and Zavala, 2019, on the other hand, only branch on the first-stage variables and solve N_s subproblems $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ (or variants thereof) in each node. To ensure convergence, the three reviewed algorithms (Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b) at least occasionally solve these subproblems to global optimality. This generally also requires branching on $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$, albeit not in the outer algorithm.

In MUSE-BB we also generate lower bounds based on SP_s, however, we partition both the $\mathcal{X}$ and $\mathcal{Y}$ domains in the same B&B tree and thus consider subproblems based on $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$ (abbreviated as SP_s^n in the following) instead of $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$. In contrast to PBDAs, the explicit partitioning of the $\mathcal{Y}$ domain, renders global solution of subproblems unnecessary for convergence. We therefore further relax the subproblems SP_s^n, resulting in cheaper lower bounding problems. In particular, we make use of the available relaxation techniques in MAiNGO to construct the following McCormick based convex relaxations of SP_s^n:

$$
\begin{aligned}
f_{\mathrm{MC},s}^n \coloneqq \min_{\substack{\boldsymbol{x}_s\in\mathcal{X}^n\\ \boldsymbol{y}_s\in\mathcal{Y}_s^n}} & f_s^{\mathrm{cv},n}(\boldsymbol{x}_s,\boldsymbol{y}_s) \\
\text{s.t. } & \boldsymbol{g}_{\mathrm{I}}^{\mathrm{cv},n}(\boldsymbol{x}_s) \leq \boldsymbol{0} \\
& \boldsymbol{g}_{\mathrm{II},s}^{\mathrm{cv},n}(\boldsymbol{x}_s,\boldsymbol{y}_s) \leq \boldsymbol{0},
\end{aligned}
\qquad \mathrm{MC}_s^n
$$

where $f_s^{\mathrm{cv},n}$, $\boldsymbol{g}_{\mathrm{I}}^{\mathrm{cv},n}$, and $\boldsymbol{g}_{\mathrm{II},s}^{\mathrm{cv},n}$ are the McCormick based convex relaxations of the functions $f_s, \boldsymbol{g}_{\mathrm{I}}$, and $\boldsymbol{g}_{\mathrm{II},s}$, on $\mathcal{X}^n \times \mathcal{Y}_s^n$, respectively (McCormick, 1976; Tsoukalas and Mitsos, 2014; Najman, Bongartz, and Mitsos, 2021). These problems are further linearized based on subtangents at one or more linearization points (cf. Najman and Mitsos, 2019). By default (and in all experiments in Section 4.6) we only linearize at the midpoint of the node domain.

The resulting lower bounding problems take the form:

$$\begin{aligned} f^n_{\mathrm{LP},s} \coloneqq \min_{\substack{\boldsymbol{x}_s \in \mathcal{X}^n \\ \boldsymbol{y}_s \in \mathcal{Y}^n_s \\ v \in \mathbb{R}}} \; & v \\ \text{s.t.} \quad & \mathrm{sub}^n_{f_s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq v \\ & \mathrm{sub}^n_{\boldsymbol{g}_{\mathrm{I}}}(\boldsymbol{x}_s) \leq \boldsymbol{0} \\ & \mathrm{sub}^n_{\boldsymbol{g}_{\mathrm{II},s}}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0} \end{aligned} \qquad \mathrm{LP}^n_s$$

Here, sub^n_ϕ are subtangents of the convex relaxation of the function ϕ at the center of the domain of node n, i.e.,

$$\mathrm{sub}^n_\phi(\bullet) \coloneqq \phi^{\mathrm{cv},n}(\boldsymbol{m}_\bullet) + \check{\nabla}\phi^{\mathrm{cv},n}(\boldsymbol{m}_\bullet)^\intercal(\bullet - \boldsymbol{m}_\bullet) \qquad \text{(subtangent)}$$

where the superscript 'cv,n' denotes the corresponding convex relaxation, $\boldsymbol{m}_\bullet$ denotes the midpoint of either $\mathcal{X}^n$ or $\mathcal{X}^n \times \mathcal{Y}^n_s$ (depending on the passed variables), and $\check{\nabla}$ denotes a subgradient, i.e., $\check{\nabla}\phi^{\mathrm{cv},n}(\boldsymbol{m}_\bullet) \in \partial\phi^{\mathrm{cv},n}(\boldsymbol{m}_\bullet)$, where $\partial\phi^{\mathrm{cv},n}(\boldsymbol{m}_\bullet)$ is the subdifferential of $\phi^{\mathrm{cv},n}$ at $\boldsymbol{m}_\bullet$. Since $f^n_{\mathrm{LP},s}$ are valid lower bounds on the globally optimal objective values $f^{\mathcal{X}^n,\mathcal{Y}^n_s}_{\mathrm{SP},s}$ of SP^n_s, they provide a valid lower bound for node n, i.e:

$$f^n_{\mathrm{LP}} \coloneqq \sum_{s \in \mathcal{S}} w_s \, f^n_{\mathrm{LP},s} \leq f^n_{\mathrm{SP}} \leq f^{\mathcal{X}^n,\mathcal{Y}^n}. \qquad (\mathrm{LPLB}^n)$$

Evidently this bound is generally weaker than the one obtained via global solution (see SPLB), but it is also much cheaper to compute.

For upper bounding, we solve instances of the form $\mathrm{RP}^{\mathcal{Y}^n_s}_s(\widetilde{\boldsymbol{x}}^n)$ (abbreviated as RP^n_s in the following), instead of $\mathrm{RP}^{\mathcal{Y}_s}_s(\widetilde{\boldsymbol{x}}^n)$ as in PBDAs. Furthermore, in contrast to Kannan, 2018 and Cao and Zavala, 2019 who solve their upper bounding problems globally, we again aim to reduce computational cost by solving RP^n_s locally. We obtain $\widetilde{\boldsymbol{x}}^n$ from the lower bounding solution corresponding to s^{rep} (see Section 4.2). If the corresponding local solutions of RP^n_s result in a feasible $\widetilde{\boldsymbol{y}}^n = (\widetilde{\boldsymbol{y}}_1, \cdots, \widetilde{\boldsymbol{y}}_{N_s})$, the corresponding objective values $f_{\mathrm{II},s}(\widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n_s) \geq f^{\mathcal{Y}^n_s}_{\mathrm{II},s}(\widetilde{\boldsymbol{x}}^n)$ can be aggregated to a globally valid upper bound $\overline{f}^n$, via the upper bounding function (UB), i.e:

$$\overline{f}^n \coloneqq \overline{f}(\widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n) \geq f^{\mathcal{X},\mathcal{Y}} \qquad (\mathrm{UB}^n)$$

If $\overline{f}^n$ is smaller than the previously best upper bound $\overline{f}$, the incumbent $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$ and $\overline{f}$ are updated with $(\widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n)$ and $\overline{f}^n$, respectively.

4.4.2. Range Reduction

In this section we discuss decomposable range reduction routines for tightening variable bounds in B&B algorithms for TSP. We first consider two general points, namely, how the NACs can enable fathoming by infeasibility after application of these routines, and how dominance rules give rise to scenario-specific objective cuts. We then present the specific routines employed in MUSE-BB. While range reduction is not necessary from a theoretical standpoint, it can improve the efficiency of the algorithm by reducing the search domain.

Based on decomposable bounding problems such as SP_s^n or LP_s^n, one can obtain decomposable range reduction routines by applying reduction techniques to the subproblems instead of the full problem DE^n. Standard techniques for feasibility-based reductions can be applied. As will be discussed in the following, however, optimality-based reductions require modified upper bounds instead of objective values of points, feasible in the subproblems. In both cases, the independence of subproblems allows for parallel updates of the bounds for the variables $(\boldsymbol{x}_s, \boldsymbol{y}_s)$ of each scenario s. After each round of range reduction, the NACs can be used to tighten the first-stage variable bounds. More explicitly, if $\mathcal{X}_s^n$ denotes the tightened first-stage variable domain for node n and scenario s after any of the presented decomposable range reduction routines, a valid reduction of the overall domain $\mathcal{X}^n$ is evidently given by the intersection $\mathcal{X}^{n\prime}$:

$$\mathcal{X}^{n\prime} := \bigcap_{s\in\mathcal{S}} \mathcal{X}_s^n \qquad (\mathcal{X}_s^n \text{ aggregation})$$

In particular, if $\mathcal{X}^{n\prime}$ is empty, node n can be fathomed by infeasibility.

If an upper bound $\overline{f}$ is known, dominance rules can be used to derive objective cuts for range reduction routines. Since in a decomposable bounding scheme objective values obtainable in any particular node n are limited from below by the local lower bound f_{SP}^n, all nodes for which $f_{\text{SP}}^n > \overline{f}$ holds can be *fathomed by dominance*. To derive a scenario-specific cutoff based on a given value of $\overline{f}$, we rewrite the dominance condition in terms of scenario-specific lower bounds. Using (SPLB), a node is dominated if

$$\sum_{s\in\mathcal{S}} w_s \, f_{\text{SP},s}^{\mathcal{X}^n,\mathcal{Y}_s^n} > \overline{f}.$$

Note that replacing any $f_{\text{SP},s}^{\mathcal{X}^n,\mathcal{Y}_s^n}$ by a smaller value (say $\underline{f}_{\text{SP},s}^n$) results in an even stronger condition, that implies the above. Thus for any particular scenario s, the node is dominated if

$$\underline{f}_{\text{SP},s}^n > \frac{\overline{f} - \sum_{s'\in\mathcal{S}\setminus\{s\}} w_{s'} \, \underline{f}_{\text{SP},s'}^n}{w_s} =: \overline{f}_s^n \qquad (s\text{-domination})$$

The above approach is a slight generalization to the scenario-specific upper bounds proposed by Li and Li, 2015 for the 'primal problems' in their decomposition method. In particular, any valid lower bounds $\underline{f}_{\text{SP},s}^n$ can be used. In MUSE-BB we use the maximum of $f_{\text{LP},s}^n$ and an interval arithmetic based lower bound based on the objective of SP_s^n.

MAiNGO implements three range reduction techniques: constraint propagation (CP, cf. e.g. Schichl and Neumaier, 2005), optimization-based bounds tightening (OBBT, cf. e.g. Gleixner et al., 2016), duality-based bounds tightening and probing (both referred to as DBBT in the following, cf. e.g. Ryoo and Sahinidis, 1995).

CP essentially refers to the inverse propagation of feasible intervals of the constraint values, i.e., $(-\infty, 0]$ in our case, to the variables (Schichl and Neumaier, 2005). This allows to determine conservative variable ranges for which the constraints can be fulfilled and thus enables domain reduction by intersecting the variable domains with these valid ranges. Thus, applying CP to the subproblems SP_s^n instead of DE^n directly gives a decomposable routine.

The OBBT procedure consists of minimizing or maximizing a selected variable v subject

to the (relaxed) constraints of the original problem (Gleixner et al., 2016). In our case, we consider scenario-specific OBBT-problems, based on the lower bounding subproblems LP_s^n, i.e., they take the form:

$$\begin{aligned} \underline{v} \backslash \overline{v} = \min_{\substack{\boldsymbol{x}_s \in \mathcal{X}^n \\ \boldsymbol{y}_s \in \mathcal{Y}_s^n}} \backslash \max\ & v \\ \text{s. t.} \quad & \mathrm{sub}_{f_s}^n(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \overline{f}_s^n \\ & \mathrm{sub}_{\boldsymbol{g}_\mathrm{I}}^n(\boldsymbol{x}_s) \leq \boldsymbol{0} \\ & \mathrm{sub}_{\boldsymbol{g}_{\mathrm{II},s}}^n(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0} \end{aligned} \qquad \mathrm{OBBT}_{s,v}^n$$

While no finite upper bound $\overline{f}$ is known, the first constraint is dropped. For each iteration, we initially consider all variables for OBBT, and apply a variant of the *trivial filtering heuristic* from Gleixner et al., 2016 after each pass. Similar OBBT based problems have been proposed, e.g., by Li and Li, 2016; Kannan, 2018; Cao and Zavala, 2019 for their respective algorithms.

DBBT uses objective bounds and duality information from the node subproblems that are typically solved in spatial B&B algorithms (Ryoo and Sahinidis, 1995) to tighten variable domains. In our case, if all subproblems LP_s^n are feasible, the solutions $(\widetilde{\boldsymbol{x}}_s, \widetilde{\boldsymbol{y}}_s)$, associated reduced cost multipliers $(\boldsymbol{r}_{x,s}, \boldsymbol{r}_{y,s})$, and lower bounds $f_{\mathrm{LP},s}^n$ are available. If in addition a finite upper bound $\overline{f}$ is known, we can compute scenario-specific $\overline{f}_s^n$ values from *s*-domination and perform DBBT. For variables v for which the solution value v^* corresponds to the respective lower or upper bound, the complementary bound may be tightened:

$$\begin{aligned} &\textbf{if } v^* = \underline{v}, \ \textbf{set } \overline{v} = \min\left(\overline{v}, \underline{v} + \frac{\overline{f}_s^n - f_{\mathrm{LP},s}^n}{r}\right) \\ &\textbf{if } v^* = \overline{v}, \ \textbf{set } \underline{v} = \max\left(\underline{v}, \overline{v} + \frac{\overline{f}_s^n - f_{\mathrm{LP},s}^n}{r}\right) \end{aligned} \qquad \text{(DBBT)}$$

where r is the corresponding entry in $\boldsymbol{r}_{x,s}$ or $\boldsymbol{r}_{y,s}$, which must be positive in the first case and negative in the second one. For variables for which the solution lies between the bounds, two probing variants of LP_s^n can be solved: in these probing LPs, the variable is temporarily fixed to one of its bounds and DBBT is applied based on the new reduced cost multipliers and optimal objective values. As probing is relatively expensive, it is deactivated by default (and in all experiments of Section 4.6).

Since each subproblem contains only part of the information of DE^n, the presented range-reduction routines will generally be less effective than their full space counterparts. Thus, the use of parallelized range reduction needs to result in sufficiently large reductions of wall time to warrant the looser variable bounds. In comparison to the solution time of a lower bounding problem, CP is computationally very cheap, which makes its decomposable variant less appealing. Nevertheless it must be used when processing sibling nodes obtained from multisection branching (cf. Section 4.3), as the resulting domains are needed for the generation of orthant nodes, also see Subroutine 5 in Section 4.4. OBBT on the other hand is a relatively expensive procedure. This typically causes OBBT to dominate the computational work done per iteration and thus makes a decomposable OBBT variant

Subroutine 1: `branchNode`(n)

```
1  v ← select a variable from (x, y) maximizing relative domain width × branching priority;
2  if v ∈ {x_i, | i ∈ {1, ··· , N_x}} then // v corresponds to some x_i
3  |  (d, u) ← bisect n along the domain of v;
4  |  N ← N ∪ {d, u};
5  else // v corresponds to y_{s',i} for some s'
6  |  i ← index for which v = y_{s',i};
7  |  d ← n; u ← n; // Initialize d and u as copies of n
8  |  foreach s ∈ S do // branch d/u down/up on all instances of v
9  |  |  v ← y_{s,i};
10 |  |  d ← lower half of bisecting d along the domain of v;
11 |  |  u ← upper half of bisecting u along the domain of v;
12 |  end
   |  // Since n, d, and u all share the same lower bound, the former
   |     is re-entered into N and the latter two are stored in M_sib
13 |  N ← N ∪ {n};
14 |  M_sib[n] ← (d, u);
15 end
```

more appealing. Finally, the use of decomposable lower bounding problems inherently requires the use of decomposable DBBT, as duality information necessary for a full space variant is not available.

4.4.3. Branching and Node Processing

In this section, we present the branching and processing routines of Algorithm 1. In Subroutine 1, we first present how processed nodes are branched, as this determines the kind of iteration that will be performed for the child nodes. Following this, we present the processing of nodes obtained from first- and second-stage branching in Subroutines 2 and 3.

Any processed node n that is not fathomed is branched on either a first-stage variable or on multiple second-stage variable instances, as outlined in Subroutine 1. For this, we select some first- or second-stage variable v, maximizing the product of *relative domain width* (i.e., current over original interval width) and branching priority (assumed to be nonzero), to ensure exhaustive partitioning. If v is an element of $\boldsymbol{x}$, i.e., $v = x_i$, we bisect the associated domain $\mathcal{X}_i^n = [\underline{x}_i, \overline{x}_i]$ at some branching point x_i^{b}, and add the two resulting nodes with the lower and upper part of the original domain (i.e., $[\underline{x}_i, x_i^{\mathrm{b}}]$ and $[x_i^{\mathrm{b}}, \overline{x}_i]$) to the list of open nodes (Subroutine 1 in Subroutine 1). In MUSE-BB, x_i^{b} always corresponds to the center of the interval, i.e., $0.5\,(\underline{x}_i + \overline{x}_i)$.

If instead, v is an element of $\boldsymbol{y}$, i.e., $v = y_{s',i}$, for some s', we perform the proposed multisection branching. As pointed out in Section 4.3, the child nodes of this multisection can subsequently be generated from the results of two complementary nodes. Therefore we only need to generate the lower and upper sibling node at this point. Taking the example from Fig. 4.1: multisecting a parent node p, corresponding to node **b)** in Fig. 4.1, results in sibling nodes d, and u, corresponding to nodes **d)**, and **e)**, respectively, which we create by branching all N_s instances of the selected second-stage variable $y_{s,i}$ down /

Subroutine 2: `processNode(`n`)`

1 do CP based on DE^n; fathom by dominance or infeasibility;
2 do OBBT based on LP^n_s; fathom by dominance; apply $\mathcal{X}^n_s$ aggregation, fathom by infeasibility ;
3 solve LP^n_s and set $\underline{f}^n_s \leftarrow f^n_{\mathrm{LP},s}\ \forall s \in \mathcal{S}$, use LPLB^n, set $\underline{f}^n \leftarrow f^n_{\mathrm{LP}}$, fathom by dominance or infeasibility ;
4 $\widetilde{\boldsymbol{x}}^n \leftarrow$ solution of LP^n_s with $s = s^{\mathrm{rep}}$;
5 $(\widetilde{\boldsymbol{y}}^n_s, \overline{f}^n_s) \leftarrow$ solution and objective value of $\mathrm{RP}^n_s\ \forall s \in \mathcal{S}$, update $(\widetilde{\boldsymbol{y}}^n, \overline{f}^n)$ via UB^n, fathom by dominance ;
6 do DBBT based on LP^n_s, fathom by dominance;
7 **return** $(\underline{f}^n, \overline{f}^n, \widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n)$

up (Subroutines 1–1).

For a practical algorithm, we need to limit the number of nodes that will be generated in the sibling iterations, as will be outlined in Section 4.4.4. This is done via a filtered multisection which requires domain and bound data from the parent node n as well as the sibling nodes. We therefore return the parent node to the list of open nodes and create the mapping $n \mapsto (d, u)$ in $\mathcal{M}_{\mathrm{sib}}$ (Subroutine 1). When the node n is selected a second time in Algorithm 1, this is detected via a lookup in $\mathcal{M}_{\mathrm{sib}}$ and we perform a sibling iteration instead.

For the root node and all nodes resulting from first-stage branching, we do a "normal iteration", i.e., the respective node is processed as specified in Subroutine 2, and either fathomed, or branched as specified in Subroutine 1. The only difference of Subroutine 2 with respect to a standard B&B algorithm is the possible use of decomposable bounding and range reduction routines from Section 4.4.1 and Section 4.4.2. In our implementation, we solve scenario subproblems for OBBT (Subroutine 2 of Subroutine 2), lower and upper bounding (Subroutine 2), and DBBT (Subroutine 2) in parallel, while the computationally cheap CP (Subroutine 2) is done using the full problem, DE^n. To generate a candidate solution $\widetilde{\boldsymbol{x}}^n$ for upper bounding (Subroutine 2), we use a representative scenario s^{rep} as outlined in Section 4.2.

With sibling nodes, obtained from second-stage branching, we do a "sibling iteration". Before we give the formal statement of the combined processing of siblings in Subroutine 3, we recall that child nodes from multisection can be generated by combining the results from different subproblems of both siblings (cf. Section 4.3). In contrast to Subroutine 2, the use of decomposable range reduction and bounding routines is thus mandatory in Subroutine 3. Moreover, we cannot perform $\mathcal{X}^n_s$ aggregation after doing range reduction routines on $n \in \{d, u\}$, because the resulting tightening would only be valid for the respective sibling node. However, we can first propagate results form range reduction of both siblings to the parent node p, whose multi section resulted in d and u, and then back to the siblings: let $\mathcal{X}^d_s$, $\mathcal{X}^u_s$ and $\mathcal{Y}^d_s$, $\mathcal{Y}^u_s$ denote the tightened variable domains obtained after applying some range reduction to the subproblems of d and u for scenario s. Then the unions of the first- and second-stage domains are a valid tightening of the corresponding domains from the parent node p, i.e:

$$\begin{aligned} \mathcal{X}^p_s &\leftarrow \mathrm{conv}\big(\mathcal{X}^d_s \cup \mathcal{X}^u_s\big) \\ \mathcal{Y}^p_s &\leftarrow \mathrm{conv}\big(\mathcal{Y}^d_s \cup \mathcal{Y}^u_s\big) \end{aligned} \qquad \text{(parent } s\text{-domain tightening)}$$

Subroutine 3: `processSiblings`(p, d, u)

foreach $s \in \mathcal{S}$ **do**
  foreach $n \in \{d, u\}$ **do**
    CP based on SP_s^n; fathom by dominance or infeasibility
  end
  apply parent s-domain tightening; fathom by infeasibility;
end
apply $\mathcal{X}_s^p$ aggregation, apply sibling s-domain tightening $\forall s \in \mathcal{S}$, fathom by infeasibility;
foreach $s \in \mathcal{S}$ **do**
  foreach $n \in \{d, u\}$ **do**
    OBBT based on LP_s^n; fathom by dominance
  end
  apply parent s-domain tightening; fathom by infeasibility;
end
apply $\mathcal{X}_s^p$ aggregation, apply sibling s-domain tightening $\forall s \in \mathcal{S}$, fathom by infeasibility;
foreach $s \in \mathcal{S}$ **do**
  foreach $n \in \{d, u\}$ **do**
    solve LP_s^n, set $\underline{f}_s^n \leftarrow f_{\mathrm{LP},s}^n$, and fathom by infeasibility
  end
  check for s-domination; fathom by dominance;
end
foreach $s \in \mathcal{S}$ **do**
  foreach $n \in \{d, u\}$ **do**
    do DBBT based on LP_s^n; fathom by dominance
  end
  apply parent s-domain tightening; fathom by infeasibility;
end
apply $\mathcal{X}_s^p$ aggregation, apply sibling s-domain tightening $\forall s \in \mathcal{S}$, fathom by infeasibility;
$\widetilde{\boldsymbol{x}}^p \leftarrow$ solution of LP_s^n with s from a variant of s^{rep} that considers all feasible scenarios for $n \in \{d, u\}$;
foreach $s \in \mathcal{S}$ **do**
  $(\widetilde{\boldsymbol{y}}_s^p, \overline{f}_s^p) \leftarrow$ solution and objective value of RP_s^p, check for s-domination; fathom by dominance;
end
update $(\widetilde{\boldsymbol{y}}^p, \overline{f}^p)$ via UB^p;
return $(\underline{f}^p, \overline{f}^p, \widetilde{\boldsymbol{x}}^p, \widetilde{\boldsymbol{y}}^p)$

Here, the use of the convex hull of the unions is purely for ease of implementation, as it ensures the resulting domains are representable as hyperrectangles. Once we have applied parent s-domain tightening for all scenarios, we can use the resulting $\mathcal{X}_s^p$ for $\mathcal{X}_s^p$ aggregation. Intersecting the resulting $\mathcal{X}^{p\prime}$ with $\mathcal{X}_s^d$ and $\mathcal{X}_s^u$ results in a valid tightening of the sibling domains:

$$\begin{aligned} \mathcal{X}_s^{d\prime} &\leftarrow \mathcal{X}_s^d \cap \mathcal{X}^{p\prime} \\ \mathcal{X}_s^{u\prime} &\leftarrow \mathcal{X}_s^u \cap \mathcal{X}^{p\prime} \end{aligned} \qquad \text{(sibling } s\text{-domain tightening)}$$

With this in place we can now review Subroutine 3. For each scenario, we execute the range reduction and lower bounding routines for the corresponding subproblem of both

siblings. Any of these routines may indicate that either d or u can be fathomed because the subproblem for some scenario s is dominated or infeasible. However, the results from the remaining subproblems of the fathomable sibling can still be combined with the results of the subproblem for s from the other sibling to generate child nodes. Thus we continue the sibling iteration as long as for each scenario there is at least one feasible, undominated subproblem from either sibling. For lower bounding (Subroutines 3–3 in Subroutine 3) we solve the subproblems LP_s^n, using the associated domains after CP (Subroutines 3–3) and OBBT (Subroutines 3–3). Following this, we perform DBBT (Subroutines 3–3). We perform all range reduction (Subroutines 3–3, Subroutines 3–3, and Subroutines 3–3), as well as bounding (Subroutines 3–3, and Subroutines 3–3) in parallel. Based on the final variable domains and objective bounds, we can generate processed orthant nodes as detailed in Section 4.4.4. In analogy to Subroutine 2, we could solve one upper bounding problem for each such orthant node, however, this would result in an exponential number of upper bounding problems. Instead, we choose to solve only a single set of upper bounding problems $\mathrm{RP}_s^{\mathcal{Y}_s^p}(\widetilde{\boldsymbol{x}}^p)$ (Subroutines 3–3 in Subroutine 3), using the $\mathcal{Y}_s^p$ domains, resulting from parent s-domain tightening after DBBT. We select $\widetilde{\boldsymbol{x}}^n$ to be one of the first-stage solutions of the feasible subproblems of both siblings, based on a representative scenario s^{rep}, that takes into account the subproblems of both siblings.

4.4.4. Filtered Multisection

In this section we present a filtered multisection that addresses issues pertaining to the inherently exponential number of child nodes resulting from multisection branching, as presented in Section 4.3. After motivating this filtered multisection we give a formal statement in Subroutine 4. Following this, we comment on the possibility of adapting a related approach used in Cao and Zavala, 2019, for branching on first-stage variables. Finally we present the generation of orthant nodes in Subroutine 5.

The ability to generate 2^{N_s} bounded child nodes by processing and recombining the results from just two sibling nodes may seem attractive, however, handling an exponential number of nodes for arbitrary N_s can quickly become an issue in practice. Consider for instance a simple problem with $N_x = N_y = 1$; simply storing the variable bounds of child nodes from a single second-stage branching as 8 byte double values requires $16\,(1+N_s)\,2^{N_s}$ bytes, e.g., more than two terabytes of memory for $N_s = 32$. At least in principle, we could avoid this memory issue by generating the nodes on demand in later iterations, however, doing this in an appropriate order, e.g., by increasing lower bound would require additional computations. More importantly, it is possible that for some of the bisections neither of the two subproblems improves the lower bound of the parent node significantly. This can result in a large number of nodes with weak objective bounds that all need to be processed separately, slowing down the algorithm.

To address this issue, we can select a subset of the N_s bisection candidates that allows for a significant increase of the lower bound or reduction of the overall domain size, compared to the parent node. We then revert the original multisection in favor of a second, filtered multisection, comprising only the variable instances corresponding to the selected bisection candidates. For this, we use Subroutine 4, which will be presented in the following. Note that each bisection candidate corresponds to a particular scenario s, a branched variable instance $y_{s,i}$, and two associated sibling subproblems with complementary domains for $y_{s,i}$. For each bisection candidate we get one of three results:

Subroutine 4: `filteredMultiSection`(p, d, u)

```
1  M ← empty map;                          // mapping s with single feasible subproblem
                                              to the corresponding sibling
2  L ← empty list;                         // containing s for which both sibling
                                              subproblems are feasible
3  foreach s ∈ S do
4      if f_s^d = ∞ then
5          M[s] = u;                       // variable corresponding to s will be branched
6      else if f_s^u = ∞ then
7          M[s] = d;                       // variable corresponding to s will be branched
8      else
9          append s to L;                  // variable corresponding to s might
                                              be branched (see
                                              Subroutines 4-4)
10     end
11 end
12 σ_max = max_{s∈L} σ_s;
13 if σ_max ≤ ε_σ^2 then
14     replace σ_s and σ_max with scores based on relative widths of variable domains
15 end
16 delete all s for which σ_s ≤ τ σ_max from L;
17 delete all but the k_max best entries from L;
18 return (M, L);
```

Case 1) Both subproblems are infeasible, this immediately implies infeasibility of the parent node p.

Case 2) Exactly one subproblem is infeasible, only the domain of the feasible subproblem can contribute to the generation of feasible orthant nodes, i.e., selecting this bisection candidate does not increase their number.

Case 3) Both subproblems are feasible, selecting this bisection candidate doubles the resulting number of orthant nodes.

Since Case 1) is already addressed by the fathoming rules in Subroutine 3, Subroutine 4 only needs to address Cases 2) and 3). We select all bisection candidates from Case 2) (Subroutines 4–4 in Subroutine 4), as they effectively result in a domain reduction, without affecting the number of generated nodes. The feasible subproblems associated with these bisection candidates are stored in the map $\mathcal{M}$. The remaining bisection candidates, corresponding to Case 3), are collected in $\mathcal{L}$ (Subroutine 4).

As the number $k \leq N_s$ of bisection candidates selected from Case 3) determines the resulting number of child nodes, we call k the *"effective number of bisections"*. To determine which bisection candidates should be selected, we use a heuristic based on strong-branching scores (Applegate et al., 1995; Achterberg, Koch, and Martin, 2005): given sibling nodes d and u obtained from the parent node p, each bisection candidate, i.e., each scenario s, is assigned a strong-branching score σ_s. For this, we employ the default scoring function of

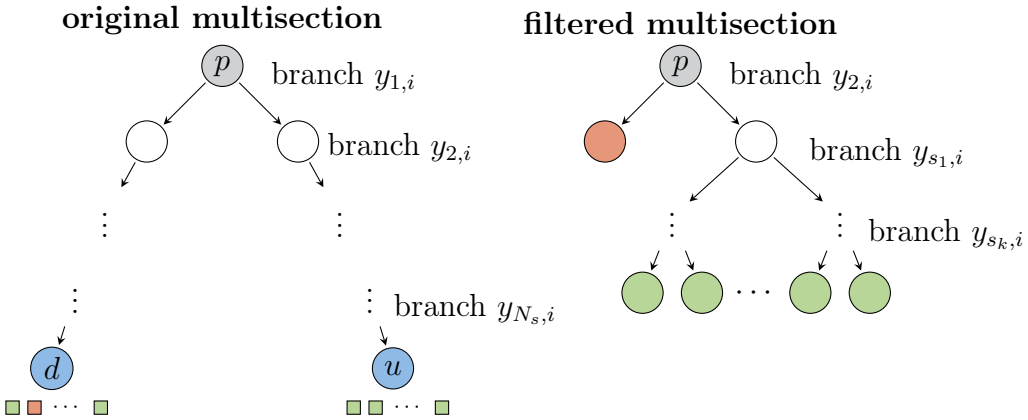


Fig. 4.2. Example for multisection branching and filtered multisection. In the original multisection (left) the parent node p is branched on all second stage variable instances $y_{s,i}$ for a given variable index i. Instead of generating all 2^{N_s} nodes, we only generate the leftmost and rightmost node, corresponding to branching all variable instances down (d) or up (u), respectively. We process these sibling nodes (blue) by solving the resulting subproblems (squares). In the example, the subproblem for $s = 2$ of d is infeasible (red) while all other subproblems are feasible (green). Right: based on the subproblem results, we perform a second, filtered multisection of p, involving a subset k of the original N_s bisection candidates (right). This can be interpreted as generating a new tree of k sequential bisections: we keep all bisections producing exactly one feasible subproblem (here only the bisection of $y_{2,i}$), as they do not increase the total number of child nodes. For the bisections resulting in two feasible subproblems, we consider the bound improvement w.r.t. the corresponding subproblems of p to compute the associated strong-branching scores σ_s. The bisection candidates are then filtered based on the values of σ_s and the algorithm parameters τ and $k_{\max}$. We reject bisections for which improvement is considered insufficient, i.e., those with $\sigma_s < \tau\sigma_{\max}$, for a threshold $\tau \in (0, 1]$. The remaining ones are sorted by descending strong-branching score, resulting in an ordering of the associated scenarios (i.e., $s_1, ..., s_k$). Of these bisections, we keep at most $k_{\max}$. Finally, we generate the corresponding 2^k orthant nodes (green) using appropriate combinations of domains and bounds from the feasible sibling subproblems.

SCIP, proposed in Achterberg, 2007, which is calculated as

$$\sigma_s := \max(\underline{f}_s^d - \underline{f}_s^p, \varepsilon_\sigma) \max(\underline{f}_s^u - \underline{f}_s^p, \varepsilon_\sigma) \qquad (\sigma_s)$$

Here the constant ε_σ ensures a nonnegative score for cases where only one sibling improves upon the parent bound.

We only keep scenarios from $\mathcal{L}$ with a score of at least $\tau\sigma_{\max}$, where $\tau \in (0, 1]$ and $\sigma_{\max}$ is the largest of the scores Subroutine 4. Additionally, a *maximum number of effective bisections* $k_{\max}$ is imposed to ensure that the filtered multisection produces at most $2^{k_{\max}}$ child nodes (Subroutine 4). If all scores are smaller than ε_σ^2, we instead rank and select bisection candidates based on relative widths of the associated variable domains (Subroutine 4). This ensures exhaustive partitioning in the limit, necessary for the convergence of MUSE-BB, also see Lemma 4.1 and Corollary 4.3 in Section 4.5. A visualization of the proposed multisection branching procedure is given in Fig. 4.2.

The use of strong-branching scores in Subroutine 4 suggests a relation between filtered

multisection and standard strong-branching, where *alternative* bisections of a set of N_v variables are considered. While standard strong-branching requires processing $2\,N_v$ *full nodes* to select a single bisection, i.e., generate 2 child nodes, we only process $2\,N_s$ *subproblems* (equivalent to 2 full nodes) and may generate an exponential number of nodes in each filtered multisection. Nevertheless, standard strong-branching might also be useful in MUSE-BB, as indicated by its use in the related algorithm of Cao and Zavala, 2019 for the selection of first-stage variables: in each iteration, the authors consider all elements of $\boldsymbol{x}$ via strong-branching, solving LP relaxations of the associated instances of DE^n for the $2\,N_x$ child nodes. For the two nodes of the selected bisection, they then perform the global solution of the subproblems SP_s, required for the convergence of their algorithm. While a similar approach could also be adopted in MUSE-BB, we do not require expensive global bounding routines for convergence; hence solving full-space LP relaxations based on DE^n is relatively expensive in our case. Alternatively we could solve the decomposable LP relaxations LP^n_s, and aggregate the strong-branching scores σ_s, e.g., via a w_s-weighted sum. As pointed out above, this would require to process $2\,N_x$ nodes instead of just 2. Due to the importance of first-stage branching for TSP (also see Section 4.6), this effort may in fact be warranted, however, we do not consider this idea further here, and instead branch only on individual first-stage variables as indicated in Subroutine 1.

The map $\mathcal{M}$, and list $\mathcal{L}$, returned by Subroutine 4 are used within Subroutine 5 for the generation of individual orthant nodes. For this, we collect the appropriate variable domains and subproblem objective values for each orthant from one of the siblings or the parent node (Subroutines 5–5 in Subroutine 5). For each scenario s, the respective node is determined, based on whether the associated bisection was selected ($s \in \mathcal{M}$ or $s \in \mathcal{L}$) or not (Subroutines 5–5). If s is in the map $\mathcal{M}$, we only use the data from the feasible subproblem of the associated sibling node (Subroutine 5). If instead, the scenario is in $\mathcal{L}$, appropriate subproblem data is taken based on the orthant index i (cf. Algorithm 1 of Algorithm 1) to determine the sibling node from which to use data (Subroutines 5–5 in Subroutine 5). Otherwise the bisection is rejected, i.e., we use the data from the parent (Subroutine 5). Note that the latter case does not imply that the solution of the associated subproblems was in vain, as it may still result in tightened variable bounds due to parent s-domain tightening (Subroutine 3 in Subroutine 3). Once data for all scenarios has been collected, we aggregate the overall second-stage domain and scenario-weighted lower bound (Subroutine 5). Finally we test whether the orthant node is infeasible or dominated and return it (Subroutine 5).

4.5. Theoretical Results

In this section we present convergence results for the lower bounding schemes used in MUSE-BB, and highlight the connection to the convergence of the algorithm itself. When applied to the domains of individual B&B nodes, the lower bounding problems presented in Section 4.4.1 give rise to different *lower bounding schemes* (LBSs). Their quality is determined by their underestimation of the true optimal value, and their capacity to quickly detect infeasible subdomains. In the following we analyze the asymptotic behavior of these two qualities for LBSs relevant to MUSE-BB, as the size of B&B nodes diminishes. In particular, we consider the LBSs based on: (i) dropping or dualizing the NACs, corresponding to the subproblems SP^n_s, or LSP^n_s, respectively, (ii) the McCormick relaxations

Subroutine 5: `generateOrthantNode`$(i, p, d, u, \mathcal{M}, \mathcal{L})$

$\boldsymbol{b} \leftarrow$ vector of $|\mathcal{L}|$ bits, representing i;
$\mathcal{X}^o \leftarrow \mathcal{X}^p$;
foreach $s \in \mathcal{S}$ **do**
  if $s \in \mathcal{M}$ **then**
    `// use data from the feasible subproblem of bisection` s
    $n \leftarrow \mathcal{M}[s]$;
  else if $s \in \mathcal{L}$ **then**
    `// use data from the sibling subproblem corresponding to orthant id` i
    $j \leftarrow$ position of s in $\mathcal{L}$;
    if $b_j = 0$ **then**
      $n \leftarrow d$;
    else
      $n \leftarrow u$
    end
  else
    `// use parent data (bisection` s `was filtered in Subroutines 4-4 of Subroutine 4)`
    $n \leftarrow p$;
  end
  $\mathcal{X}^o \leftarrow \mathcal{X}^o \cap \mathcal{X}^n_s$;
  $\mathcal{Y}^o_s \leftarrow \mathcal{Y}^n_s$;
  $\underline{f}^o_s \leftarrow \underline{f}^n_s$;
end
$\mathcal{Y}^o \leftarrow \bigtimes_{s \in \mathcal{S}} \mathcal{Y}^o_s$;
$\underline{f}^o \leftarrow \sum_{s \in \mathcal{S}} w_s \underline{f}^o_s$;
if $\mathcal{X}^o = \emptyset$ or $\underline{f}^o > \overline{f}$ **then** $\underline{f}^o = \infty$;
return o;

of subproblems from (i), and (iii) the linear programming relaxations, resulting from subtangent relaxation of subproblems from (ii). Formally, the asymptotic behavior of a LBS for a sequence of descendant nodes is quantified by the convergence order (Kannan and Barton, 2017a). We first introduce additional notation and definitions related to this convergence order in Section 4.5.1, and then present conditions under which different LBSs achieve first- and second-order convergence, respectively in Sections 4.5.2 and 4.5.3. As a result of the first-order convergence, we show that MUSE-BB guarantees finite termination with an ε_f-optimal solution in Section 4.5.2. In Section 4.5.3 we analyze an extension of MUSE-BB in which the NACs are dualized instead of dropped. We show that employing this dualization within MUSE-BB is equivalent to adding the terms $\boldsymbol{\lambda}_s^\intercal \boldsymbol{x}_s$, to the objective function relaxations in the subproblems LP^n_s, and performing dual iterations to update the multipliers $\boldsymbol{\lambda}_s$. Provided optimal multipliers $\boldsymbol{\lambda}^*_s$ are obtained, we show that this results in stronger convergence properties, with implications for the so-called cluster effect (Kearfott and Du, 1993; Du and Kearfott, 1994). In particular, while theoretical results of Kannan and Barton, 2017b indicate that the current implementation of MUSE-BB may mitigate clustering around typical constrained minimizers, mitigating clustering around typical unconstrained minimizers may require an extension such as the one presented in

Section 4.5.3.

Before we give the formal definition of a LBS and the associated convergence order, we highlight the impact of the quality of lower bounds via two examples. For this we define the width of an interval.

Definition 4.1 (width of a multidimensional interval)**.** A measure for the width of a multidimensional interval $\mathcal{V} = \bigtimes_{i\in\{1,\cdots,m\}} [\underline{v}_i, \overline{v}_i] \in \mathbb{IR}^m$ is given by:

$$\mathrm{W}(\mathcal{V}) \coloneqq \max_{i\in\{1,\cdots,m\}} (\overline{v}_i - \underline{v}_i)$$

As shown by Kannan and Barton, 2017b, the occurrence of clustering is related to the convergence order of the LBS, which in turn is defined in terms of the 'size of B&B nodes', i.e., the width of the domain of branched variables, measured by Definition 4.1 (also see Kannan and Barton, 2017a). In algorithms like PBDAs, this node size is given by $\mathrm{W}(\mathcal{X}^n)$, whereas in algorithms like MUSE-BB it is given by the width of the overall variable domain, i.e., $\mathrm{W}(\mathcal{Z}^n)$. While MUSE-BB will of course require more branching than PBDAs to reach a given node size, the LBSs used in MUSE-BB may achieve a higher convergence order than the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, used in PBDAs.

The following example illustrates this situation for LBSs based on SP_s, i.e., the simplest scenario relaxation, corresponding to dropping the NACs from $\mathrm{DE}_{\mathrm{NAC}}^{\mathcal{X},\mathcal{Y}}$: while the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, where only $\mathcal{X}$ is partitioned, results in an absolute optimality gap that diminishes with $\sqrt{\mathrm{W}(\mathcal{X}^n)}$, the gap produced by the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, which additionally partitions $\mathcal{Y}$, diminishes with $\mathrm{W}(\mathcal{Z}^n)$.

Example 4.1. Consider the following instance of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ with $N_x = N_y = 1, N_s = 2$ and an original domain with $\mathcal{X} = \mathcal{Y}_1 = \mathcal{Y}_2 = [0, 2]$. Take

$$\begin{aligned} w_1\, f_1(x_1, y_1) &= -y_1; \quad & \boldsymbol{g}_{\mathrm{II},1}(x_1, y_1) &= -x_1 + y_1^2 \\ w_2\, f_2(x_1, y_2) &= 2\, y_2; \quad & \boldsymbol{g}_{\mathrm{II},2}(x_2, y_2) &= x_2 - y_2^2 \end{aligned}$$

The objectives imply that at the optimum $\boldsymbol{z}^{\mathrm{DE}^n} = (x_1^{\mathrm{DE}^n}, y_1^{\mathrm{DE}^n}, y_2^{\mathrm{DE}^n})$ of DE^n, $y_1^{\mathrm{DE}^n}$ is maximized and $y_2^{\mathrm{DE}^n}$ is minimized. For any feasible node n with $\mathcal{Z}^n = [\underline{x}^n, \overline{x}^n] \times [\underline{y}_1^n, \overline{y}_1^n] \times [\underline{y}_2^n, \overline{y}_2^n]$, the bounds and constraints imply $y_1^{\mathrm{DE}^n} \leq \min\{\sqrt{x^{\mathrm{DE}^n}}, \overline{y}_1^n\}$ and $y_2^{\mathrm{DE}^n} \geq \max\{\sqrt{x^{\mathrm{DE}^n}}, \underline{y}_2^n\}$. We have $y_1^{\mathrm{DE}} = y_2^{\mathrm{DE}} = \sqrt{x^{\mathrm{DE}}}$ on the original domain, and thus $f(x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = \sqrt{x^{\mathrm{DE}}}$, which is minimized at $\boldsymbol{z}^{\mathrm{DE}} = (x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = (0, 0, 0)$, with objective value 0.

Now consider the lower bounds generated by lower bounding schemes based on SP_s on any nested sequence of nodes converging to the optimum $\boldsymbol{z}^{\mathrm{DE}}$. Since all nodes in such sequences satisfy $\underline{x}^n = \underline{y}_s^n = 0$, the optimal solutions of the associated instance of SP_s satisfy $y_1^{\mathrm{SP}^n} = \min\{\sqrt{\overline{x}^n}, \overline{y}_1^n\}$ and $y_2^{\mathrm{SP}^n} = \max\{\sqrt{\underline{x}^n}, \underline{y}_2^n\} = 0$, and thus from the constraint $\boldsymbol{g}_{\mathrm{II},2}$, we have $x_2^{\mathrm{SP}^n} = y_2^{\mathrm{SP}^n} = 0$.

In $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, only x is branched, and the width of a node n corresponds to $W^n = \mathrm{W}(\mathcal{X}^n) = \overline{x}^n$, while $\mathrm{W}(\mathcal{Y}_s^n) = \overline{y}_s^n = 2$ remains constant. Since $\overline{x}^n < 2$, we have: $y_1^{\mathrm{SP}^n} = \sqrt{W^n}$, and thus $f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = \sqrt{W^n}$.

In $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, both x and y_s are branched, and the width of a node n corresponds to $W^n = \mathrm{W}(\mathcal{Z}^n)$. For a given width W^n, the largest value for $f_{\mathrm{DE}}^{n'} - f_{\mathrm{SP}}^{n'}$ over all nodes n' with $\mathrm{W}(\mathcal{Z}^{n'}) = W^n$ will be produced by the node n with $\overline{x}^n = \overline{y}_s^n = W^n$. Once

$W^n < 1$, we have that $\sqrt{W^n} > W^n$, and thus $y_1^{\mathrm{SP}^n} = W^n$, and $f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = W^n$.

While Example 4.1 shows that for certain problems the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ will produce weaker bounds than $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$ for a given node width, the following example demonstrates that this is not always the case, i.e., both LBSs may produce absolute optimality gaps that diminish linearly (and not better) with the node width.

Example 4.2. Take Example 4.1, but change the constraints to

$$\boldsymbol{g}_{\mathrm{II},1}(x_1, y_1) = -x_1 + y_1; \qquad \boldsymbol{g}_{\mathrm{II},2}(x_2, y_2) = x_2 - y_2.$$

which implies that $y_1^{\mathrm{DE}} = y_2^{\mathrm{DE}} = x^{\mathrm{DE}}$ on the original domain, and thus $f(x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = x^{\mathrm{DE}}$. This is again minimized at $\boldsymbol{z}^{\mathrm{DE}} = (x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = (0, 0, 0)$, with objective value 0. Now for both $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, and $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, it is easy to see that $x_2^{\mathrm{SP}^n} = y_2^{\mathrm{SP}^n} = 0$, and $x_1^{\mathrm{SP}^n} = y_1^{\mathrm{SP}^n} = W^n$, resulting in $f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = W^n$, i.e., an optimality gap that decreases exactly linearly with the node width.

As we shall see in Section 4.5.1, β-order convergence of a LBS requires that the optimality gap decreases proportionally to $(W^n)^\beta$, with $\beta > 0$, i.e., a higher value of β is associated with a better quality of the LBS. Robertson, Cheng, and Scott, 2024 showed that the convergence orders below one of LBSs used in PBDAs are inherent to the projection resulting from running a B&B in the $\mathcal{X}$ space only. In particular, even LBSs based on the ideal relaxation, i.e., on convex envelopes of the scenario value functions $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ may have less than first-order convergence, unless $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ is Lipschitz, which is not guaranteed in general. In contrast, we show in Section 4.5.2 that the scheme $\mathrm{SP}_s^n = \mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, obtained by simply dropping the NACs, has at least first-order convergence under the much milder assumption that the objective and constraint functions of DE are Lipschitz. If additionally, the used convex relaxations are Lipschitz, subsequent convex and linear relaxations used in MUSE-BB preserve this first-order convergence.

As demonstrated by Examples 4.1 and 4.2, the convergence order may still be as low as one, despite branching on second-stage variables. In Section 4.5.3 we show that this limitation is inherent to dropping the NACs, and that dualizing them instead results in a LBS that is as least as strong as the presented one, but additionally guarantees second-order convergence at unconstrained minimizers.

Despite this promising outlook for MUSE-BB, we need to point out that the seemingly superior convergence order of LBSs for MUSE-BB compared to that of PBDAs may be relativized by the fact that the occurrence of clustering is not exclusively determined by convergence order, but also by the local growth order of objective and constraint functions, see Kannan and Barton, 2017b. Even if for a given problem, a LBS for MUSE-BB has a higher convergence order than a comparable scheme for a PBDA, the lower order might still be sufficient to mitigate clustering in PBDAs. This is because by operating in the projected space, the relevant growth order for PBDAs is that of of the scenario value functions $f_s^{\mathcal{X},\mathcal{Y}_s}$, which may also be reduced compared that of the original objective functions f_s. In Example 4.1, e.g., we have $f_1^{\mathcal{X},\mathcal{Y}_1}(x) = \sqrt{x}$, and thus a growth order of $1/2$, matching the convergence order of the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, indicating that clustering might still be avoided, despite the reduced convergence order. Conditions for which PBDAs or algorithms like MUSE-BB will show superior performance are thus not immediately clear from the present analysis.

4.5.1. Preliminaries

To avoid the so-called cluster effect (Kearfott and Du, 1993; Du and Kearfott, 1994; Wechsung, Schaber, and Barton, 2014) where a B&B algorithm visits a large number of nodes near approximate global minimizers, LBSs need to exhibit a sufficiently large convergence order. Early works on clustering (Kearfott and Du, 1993; Du and Kearfott, 1994; Wechsung, Schaber, and Barton, 2014) focused on clustering around unconstrained minimizers, where the convergence order of LBSs is equivalent to the convergence order of the relaxations used for the objective function. Around constrained minimizers, on the other hand, one additionally needs to consider the effect of relaxing the feasible set, leading to an extended notion of convergence order (Kannan and Barton, 2017b; Kannan and Barton, 2017a), which additionally depends on the convergence orders of the relaxations used for the constraint functions. In B&B for general nonlinear programming problems, relaxations of objective and constraints are typically generated by convex relaxation methods. In Bompadre and Mitsos, 2011 we therefore analyzed the convergence order of McCormick (McCormick, 1976), α-BB (Adjiman and Floudas, 2008), and convex hull relaxations. Convergence orders for (further) relaxation through polyhedral outer approximation were investigated by Rote, 1992; Tawarmalani and Sahinidis, 2004; Khan, 2018. While Kannan and Barton, 2017b consider a classical LBS for general nonlinear programming problems based on convex relaxation, their conclusions are not dependent on this type of LBS. Kannan and Barton, 2017a present a more general definition of a LBS, and give conditions under which convex and Lagrangian relaxations with appropriate convergence orders result in first- and second-order convergent LBS.

In preparation for Definition 4.4, where we use an extended notion of convergence order of a LBS in the sense of Kannan and Barton, 2017a, we introduce additional nomenclature and definitions. For each B&B node n and the corresponding *subproblem domains* $\mathcal{Z}_s^n := \mathcal{X}^n \times \mathcal{Y}_s^n$, we introduce the *scenario-specific feasible sets*

$$\mathcal{F}_s^n := \{(\boldsymbol{x}, \boldsymbol{y}_s) \in \mathcal{Z}_s^n : \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \leq \boldsymbol{0}\}.$$

Similarly, for the *overall domains* $\mathcal{Z}^n := \mathcal{X}^n \times \mathcal{Y}^n$, associated with each node n, we express the *feasible set* of $\mathrm{DE}^n = \mathrm{DE}^{\mathcal{X}^n, \mathcal{Y}^n}$ as

$$\mathcal{F}^n := \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{Z}^n : (\boldsymbol{x}, \boldsymbol{y}_s) \in \mathcal{F}_s^n \, \forall s \in \mathcal{S}\}.$$

Furthermore, since we branch on both $\boldsymbol{x}$ and $\boldsymbol{y}$, the distinction between them becomes irrelevant in many parts of the following analysis. For conciseness, we therefore aggregate the first- and second-stage variables, i.e., we introduce the notation $(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}, \boldsymbol{y}_1, \cdots, \boldsymbol{y}_{N_s}) =: \boldsymbol{z} \in \mathcal{Z}^n \in \mathbb{R}^{N_z}$, and $(\boldsymbol{x}, \boldsymbol{y}_s) =: \boldsymbol{z}_s \in \mathcal{Z}_s^n \in \mathbb{R}^{N_{z,s}}$, where, $N_z := N_x + N_s N_y$ and $N_{z,s} := N_x + N_y$.

Definition 4.2 (distance between two sets)**.** A measure for the distance of two sets $\mathcal{Z}_1, \mathcal{Z}_2, \subset \mathbb{R}^m$ is given by:

$$\mathrm{d}(\mathcal{Z}_1, \mathcal{Z}_2) := \inf_{\substack{\boldsymbol{z}_1 \in \mathcal{Z}_1 \\ \boldsymbol{z}_2 \in \mathcal{Z}_2}} \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|$$

Throughout this text, $\|\bullet\|$ denotes the Euclidian norm.

Definition 4.3 (violation measure)**.** A measure for the minimum constraint violation of some optimization problem $\widetilde{\mathrm{P}}(\mathcal{V})$ with variable domain $\mathcal{V} \in \mathbb{R}^{N_v}$ and constraints $\boldsymbol{g}_{\widetilde{\mathrm{P}}} : \mathbb{R}^{N_v} \to \mathbb{R}^{N_g^{\widetilde{\mathrm{P}}}}$, on some subdomain $\mathcal{V}^n \in \mathbb{V}$ is given by:

$$\begin{aligned} \mathrm{vio}_{\widetilde{P}}^n &:= \mathrm{d}\Big(\{\boldsymbol{g}_{\widetilde{\mathrm{P}}}(\boldsymbol{v}) : \boldsymbol{v} \in \mathcal{V}^n\}, \mathbb{R}_-^{N_g^{\widetilde{\mathrm{P}}}}\Big) \\ &= \min_{\boldsymbol{v} \in \mathcal{V}^n} \left(\sum_{j=1}^{N_g^{\widetilde{\mathrm{P}}}} \max\Big\{ g_{\widetilde{\mathrm{P}},j}(\boldsymbol{v}), 0 \Big\}^2 \right)^{1/2}, \end{aligned}$$

where $\mathbb{R}_-$ denotes the nonpositive orthant.

Alternative to Definition 4.3, one may also define the violation in terms of, e.g., the ∞-norm, which would yield $\min_{\boldsymbol{v} \in \mathcal{V}^n} \max_{j \in \{1, \cdots N_g^{\widetilde{\mathrm{P}}}\}} \max\{0; g_{\widetilde{\mathrm{P}},j}(\boldsymbol{v})\}$. We chose Definition 4.3, following Kannan and Barton, 2017b; Kannan and Barton, 2017a, who use it, within their definitions of convergence order of LBSs (Definition 8 and 14, respectively). For clarity, we separate the definition of violation from that of convergence order.

We adapt Definition 14 of Kannan and Barton, 2017a to scenario-based LBSs of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$. All such schemes effectively lift the deterministic equivalent formulation DE^n to the equivalent nonanticipativity formulation $\mathrm{DE}_{\mathrm{NAC}}^{\mathcal{X},\mathcal{Y}}$, which introduces separate first-stage variables and constraints for each scenario and couples them via the NACs. Following this, scenario-based LBSs obtain relaxations of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$, by dropping or dualizing the NACs from $\mathrm{DE}_{\mathrm{NAC}}^{\mathcal{X},\mathcal{Y}}$, potentially followed by further relaxations of the objective and constraints.

Definition 4.4 (Hausdorff convergence order of scenario-based LBSs)**.** Denote the optimal objective value of DE^n as f^n_{DE}, and let R^n be any relaxation of DE^n that decomposes into the N_s scenario relaxations of the form:

$$f^n_{\mathrm{R},s} \coloneqq \min_{\boldsymbol{z}_s \in \mathcal{F}^n_{\mathrm{R},s}} f_{\mathrm{R},s}(\boldsymbol{z}_s) \qquad \mathrm{R}^n_s$$

where for each s, the feasible set $\mathcal{F}^n_{\mathrm{R},s}$ contains $\mathcal{F}^n_s$, and the objective functions $f_{\mathrm{R},s}$ are such that the weighted sum of the optimal objective values $f^n_{\mathrm{R},s}$ underestimates f^n_{DE}, i.e.,

$$f^n_{\mathrm{R}} \coloneqq \sum_{s \in \mathcal{S}} w_s \, f^n_{\mathrm{R},s} \leq f^n_{\mathrm{DE}}.$$

We say that (the LBS based on) R^n_s has:

1. β_f-order (Hausdorff) convergence at a feasible point $\boldsymbol{z} \in \mathcal{Z}$ if there exists $c_f > 0$ such that for every $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ with $\boldsymbol{z} \in \mathcal{Z}^n$,

$$f^n_{\mathrm{DE}} - f^n_{\mathrm{R}} \leq c_f \, \mathrm{W}(\mathcal{Z}^n)^{\beta_f}$$

2. β_g-order (Hausdorff) convergence at an infeasible point $\boldsymbol{z} \in \mathcal{Z}$ if there exists $c_g > 0$ such that for every $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ with $\boldsymbol{z} \in \mathcal{Z}^n$,

$$\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{R}} \leq c_g \, \mathrm{W}(\mathcal{Z}^n)^{\beta_g}$$

We say that (the LBS based on) R^n_s has (Hausdorff) convergence of order β on $\mathcal{Z}$ if is has β-order (Hausdorff) convergence at each $\boldsymbol{z} \in \mathcal{Z}$.

The generic scenario-based relaxation R^n_s encompasses all LBSs we consider: LSP^n_s; SP^n_s; the additional relaxation of these problems, resulting from replacing all functions by their McCormick relaxations on $\mathcal{Z}^n$ (i.e., MC^n_s in the case of SP^n_s); and the linear outer approximation of MC^n_s through subtangents, LP^n_s. In all cases, the convergence order is with respect to DE^n, i.e., feasibility and infeasibility are always to be understood with respect to the original variables and constraints. As in Definition 14 of Kannan and Barton, 2017a, the convergence order at feasible [infeasible] points establishes an upper bound on the underestimation of the optimal objective value [minimal constraint violation] in terms of the node width. Thus, the theoretical results of Kannan and Barton, 2017b are directly applicable. In particular, assuming sufficiently small prefactors c_f and c_g, and that all minimizers are strict, the previous analyses indicate that second-order convergence at feasible points mitigates clustering around unconstrained minimizers located at points of differentiability (Wechsung, Schaber, and Barton, 2014; Kannan and Barton, 2017b), while first-order convergence suffices for unconstrained minimizers if they are located at points of nondifferentiability (Wechsung, 2014; Kannan and Barton, 2017b). At constrained minimizers, on the other hand, first-order convergence may mitigate clustering if the objective and active constraints grow linearly around the minimizer Kannan and Barton, 2017b.

Note that according to Definition 4.3, the constraint violations $\mathrm{vio}^n_{\mathrm{DE}}$ and $\mathrm{vio}^n_{\mathrm{R}}$ are defined relative to the overall constraints of the respective problems. In contrast to DE^n, all scenario relaxations R^n_s by definition have separate copies of the first-stage variables $\boldsymbol{x}$ and the first-stage constraints $\boldsymbol{g}_{\mathrm{I}}$ (or their relaxations) for each scenario s. Hence, the total

number of variables and constraints of the N_s subproblems R_s^n are $N_\xi \coloneqq N_s(N_x + N_y)$, and $N_g^{\mathrm{R}} \coloneqq N_s(N_{\mathrm{I}} + N_{\mathrm{II}})$, respectively. Similarly to $\boldsymbol{g}_{\mathrm{DE}}$, we define $\boldsymbol{g}_{\mathrm{R}}$ by aggregating the constraint functions of R_s^n for all s; i.e., $\boldsymbol{g}_{\mathrm{R}}$ is the vector-valued function $\boldsymbol{g}_{\mathrm{R}} : (\bigtimes_{s\in\mathcal{S}} \mathcal{Z}_s) \mapsto \mathbb{R}^{N_g^{\mathrm{R}}}$, such that for $\boldsymbol{\xi} = (\boldsymbol{x}_1, \boldsymbol{y}_1, \cdots, \boldsymbol{x}_{N_s}, \boldsymbol{y}_{N_s,N_y}) \in (\bigtimes_{s\in\mathcal{S}} \mathcal{Z}_s) \in \mathbb{IR}^{N_\xi}$ we have:

$$\boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\xi}) \coloneqq \begin{pmatrix} g_{\mathrm{R},1}(\boldsymbol{\xi}) \\ \vdots \\ g_{\mathrm{R},N_g^{\mathrm{R}}}(\boldsymbol{\xi}) \end{pmatrix},$$

e.g., when using LSP_s^n or SP_s^n for R_s^n, we define these entries as

$$\boldsymbol{g}_{\mathrm{LSP}}(\boldsymbol{\xi}) = \boldsymbol{g}_{\mathrm{SP}}(\boldsymbol{\xi}) = \begin{pmatrix} g_{\mathrm{I},1}(\boldsymbol{x}_1) \\ \vdots \\ g_{\mathrm{I},N_{\mathrm{I}}}(\boldsymbol{x}_{N_s}) \\ g_{\mathrm{II},1,1}(\boldsymbol{x}_1, \boldsymbol{y}_1) \\ \vdots \\ g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\boldsymbol{x}_{N_s}, \boldsymbol{y}_{N_s}) \end{pmatrix}.$$

Since the bounds in Definition 4.4 are relative to the width of the overall variable domain $\mathcal{Z}^n$, it is only meaningful for B&B algorithms for which this width diminishes to 0. MUSE-BB clearly satisfies this condition, as shown for completeness in the following result.

Lemma 4.1 (Exhaustive Subdivision)**.** *The branching scheme used in MUSE-BB is* exhaustive*, i.e., in the limit all infinite sequences of descendant nodes converge to some accumulation point.*

Proof. In Subroutine 1 of Subroutine 1 we eventually select the variable corresponding to the dimension of $\mathcal{Z}^n$ with largest relative domain width (since the effect of different branching priorities is canceled after a finite number of iterations). While the bisection of the selected variable can still be rejected during variable filtering (Subroutine 4 in Subroutine 4), this can only happen a finite number of times, as the strong-branching scores are based on lower bound *improvements* which inherently tend to zero. Thus, the width of all variable domains tends to zero. □

Note that since PBDAs only partition $\mathcal{X}$, $\mathrm{W}(\mathcal{Z}^n)$ would need to be substituted with $\mathrm{W}(\mathcal{X}^n)$ in the bounds of Definition 4.4 to obtain an appropriate alternative definition for PBDAs, also see the related Definition 14 and Section 5 of Kannan and Barton, 2017a.

4.5.2. First-Order Convergence

As we shall see in Lemma 4.2, branching on second-stage variables $\boldsymbol{y}$ in addition to first-stage variables $\boldsymbol{x}$, resolves the possibility of convergence orders below one, i.e., SP_s^n can be guaranteed to have (at least) first-order convergence under the weak assumption of Lipschitz continuity of the objective and constraint functions. Furthermore, Corollaries 4.1 and 4.2 show that the additional relaxations used in MUSE-BB preserve first-order convergence.

Assumption 4.1 (Lipschitz, factorable functions)**.** All constraint and objective functions are Lipschitz, i.e., there exist constants $L_{g,\mathrm{I},i} > 0;\ i = 1, \cdots, N_{\mathrm{I}}$, and for all $s \in \mathcal{S}$ there exist constants $L_{g,\mathrm{II},s,j} > 0,\ j = 1, \cdots, N_{\mathrm{II}};\ L_{f,s} > 0$, such that:

$$
\begin{aligned}
|g_{\mathrm{I},i}(\boldsymbol{x}) - g_{\mathrm{I},i}(\boldsymbol{x}')| &\leq L_{g,\mathrm{I},i} \|\boldsymbol{x} - \boldsymbol{x}'\| \quad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}, i = 1, \cdots, N_{\mathrm{I}}, \\
|g_{\mathrm{II},s,j}(\boldsymbol{z}_s) - g_{\mathrm{II},s,j}(\boldsymbol{z}'_s)| &\leq L_{g,\mathrm{II},s,j} \|\boldsymbol{z}_s - \boldsymbol{z}'_s\| \quad \forall \boldsymbol{z}_s, \boldsymbol{z}'_s \in \mathcal{Z}_s, j = 1, \cdots, N_{\mathrm{II}}, \\
|f_s(\boldsymbol{z}_s) - f_s(\boldsymbol{z}'_s)| &\leq L_{f,s} \|\boldsymbol{z}_s - \boldsymbol{z}'_s\| \quad \forall \boldsymbol{z}_s, \boldsymbol{z}'_s \in \mathcal{Z}_s.
\end{aligned}
$$

The following Lemma shows first-order convergence of the LBS based on SP_s^n. Its proof relies on the fact that in any given node n, points from the domains $\mathcal{Z}_s^n = \mathcal{X}^n \times \mathcal{Y}_s^n$ of scenario subproblems are at most $\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n)$ apart. Furthermore, the overall node domain is $\mathcal{Z}^n = \mathcal{X}^n \times \mathcal{Y}^n$ and thus $\mathrm{W}(\mathcal{Z}_s^n) \leq \mathrm{W}(\mathcal{Z}^n)$. We point out that all of the algebraic steps in the following proof would also hold when replacing SP_s^n with the LBS $\mathrm{SP}_s^{\mathcal{X}^n, \mathcal{Y}_s}$ used in PBDA. Thus in fact, both LBS have first-order convergence in the $\mathcal{Z}$ space. However, while for MUSE-BB $\mathrm{W}(\mathcal{Z}^n)$ tends to zero by Lemma 4.1, it does not for PBDAs, where $\mathcal{Y}^n = \mathcal{Y}$ for all nodes n, and hence only $\mathrm{W}(\mathcal{X}^n)$ tends to zero. A meaningful convergence order for $\mathrm{SP}_s^{\mathcal{X}^n, \mathcal{Y}_s}$ would therefore require bounds in terms of $\mathrm{W}(\mathcal{X}^n)$ instead of $\mathrm{W}(\mathcal{Z}^n)$, also see the note before Lemma 4.1 and the related Definition 14 and Section 5 of Kannan and Barton, 2017a.

Lemma 4.2 (first-order convergence of SP_s^n)**.** *Under Assumption 4.1, SP_s^n has a convergence order of $\beta \geq 1$.*

Proof. Recall that Definition 4.4 considers convergence orders at feasible and infeasible points with respect to DE^n, leading to a natural proof outline.

Convergence Order at Feasible Points: First consider some arbitrary point $\widetilde{\boldsymbol{z}} \in \mathcal{Z}$ that is *feasible* in DE. Note that both DE^n, and by extension, also its relaxation SP_s^n have optimal solutions for any subset $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ containing $\widetilde{\boldsymbol{z}}$. For any such subset, let $\boldsymbol{z}^{\mathrm{DE}^n} = (\boldsymbol{x}^{\mathrm{DE}^n}, \boldsymbol{y}^{\mathrm{DE}^n}) = (\boldsymbol{x}^{\mathrm{DE}^n}, \boldsymbol{y}_1^{\mathrm{DE}^n}, \cdots, \boldsymbol{y}_{N_s}^{\mathrm{DE}^n}) \in \mathcal{Z}^n$ be an optimal solution to DE^n, and define $\boldsymbol{z}_s^{\mathrm{DE}^n} = (\boldsymbol{x}^{\mathrm{DE}^n}, \boldsymbol{y}_s^{\mathrm{DE}^n}) \in \mathcal{Z}_s^n$. Similarly, let $\boldsymbol{z}_s^{\mathrm{SP}^n} = (\boldsymbol{x}_s^{\mathrm{SP}^n}, \boldsymbol{y}_s^{\mathrm{SP}^n}) \in \mathcal{Z}_s^n$ be an optimal solution to SP_s^n. Using the Lipschitz property of f_s, we can immediately express the difference in optimal objective values as:

$$
\begin{aligned}
f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n &= \sum_{s \in \mathcal{S}} w_s \big(f_s(\boldsymbol{z}_s^{\mathrm{DE}^n}) - f_s(\boldsymbol{z}_s^{\mathrm{SP}^n}) \big) \\
&\leq \sum_{s \in \mathcal{S}} w_s c_{f,s}^{\mathrm{SP}}\, \mathrm{W}(\mathcal{Z}^n)
\end{aligned}
$$

where $c_{f,s}^{\mathrm{SP}} \coloneqq L_{f,s}\sqrt{N_x + N_y}$. Since $\widetilde{\boldsymbol{z}}$ and $\mathcal{Z}^n$ were selected arbitrarily, SP_s^n has at least first-order convergence at all feasible points with $c_f = \sum_{s \in \mathcal{S}} w_s c_{f,s}^{\mathrm{SP}}$.

Convergence Order at Infeasible Points: Now consider a point $\widetilde{\boldsymbol{z}}$ that is *infeasible* in DE, i.e., $\widetilde{\boldsymbol{z}} \notin \mathcal{F}$. Any subset $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ containing $\widetilde{\boldsymbol{z}}$, either contains feasible points or it does not. By Definition 4.3, these two cases correspond to $\mathrm{vio}_{\mathrm{DE}}^n = 0$, and $\mathrm{vio}_{\mathrm{DE}}^n > 0$, respectively. In the former case we also have $\mathrm{vio}_{\mathrm{SP}}^n = 0$ by the fact that SP is a relaxation, i.e., the properties from Definition 4.4 hold trivially.

The remainder of the proof is concerned with the latter case, i.e., $\mathrm{vio}_{\mathrm{DE}}^n > 0$. Let $\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n} = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}) = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}_1^{\mathrm{DE}^n}, \cdots, \widetilde{\boldsymbol{y}}_{N_s}^{\mathrm{DE}^n}) \in \mathcal{Z}^n$ and $\boldsymbol{\zeta}^{\mathrm{DE}^n} \in \mathbb{R}_-^{N_g^{\mathrm{DE}}}$ be points at

which the minimum constraint violation $\mathrm{vio}^n_{\mathrm{DE}}$ is attained, i.e., $\mathrm{vio}^n_{\mathrm{DE}} = \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n}\right\|$. Similarly, let $\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n} = (\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}, \widetilde{\boldsymbol{y}}_1^{\mathrm{SP}^n}, \cdots, \widetilde{\boldsymbol{x}}_{N_s}^{\mathrm{SP}^n}, \widetilde{\boldsymbol{y}}_{N_s}^{\mathrm{SP}^n}) \in \bigtimes_{s\in\mathcal{S}} \mathcal{Z}_s^n$ and $\widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n} \in \mathbb{R}_-^{N_g^{\mathrm{SP}}}$ be points at which the minimum constraint violation $\mathrm{vio}^n_{\mathrm{SP}}$ is attained, i.e., $\mathrm{vio}^n_{\mathrm{SP}} = \left\|\boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n}\right\|$. Furthermore, define $\widetilde{\boldsymbol{z}}_s^{\mathrm{DE}^n} = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}_s^{\mathrm{DE}^n}) \in \mathcal{Z}_s^n$ and $\widetilde{\boldsymbol{z}}_s^{\mathrm{SP}^n} = (\widetilde{\boldsymbol{x}}_s^{\mathrm{SP}^n}, \widetilde{\boldsymbol{y}}_s^{\mathrm{SP}^n}) \in \mathcal{Z}_s^n$.

To derive an upper bound on $\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}}$, we first give a lower bound on the minimum constraint violation $\mathrm{vio}^n_{\mathrm{SP}}$. For this we drop positive terms in the definition of $\mathrm{vio}^n_{\mathrm{SP}}$, corresponding to the first-stage constraints of all but the first scenario:

$$
\begin{aligned}
\mathrm{vio}^n_{\mathrm{SP}} &= \left\|\boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n}\right\| \\
&= \left(\sum_{j}^{N_g^{\mathrm{SP}}} \left| g_{\mathrm{SP},j}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\zeta}_j^{\mathrm{SP}^n} \right|^2\right)^{1/2} \\
&= \Big(\left| g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}) - \widetilde{\zeta}_1^{\mathrm{SP}^n} \right|^2 + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_{\mathrm{I}}}^{\mathrm{SP}^n} \right|^2 \\
&\quad + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}_{N_s}^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_sN_{\mathrm{I}}}^{\mathrm{SP}^n} \right|^2 \\
&\quad + \left| g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_sN_{\mathrm{I}}+1}^{\mathrm{SP}^n} \right|^2 + \cdots + \left| g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_g^{\mathrm{SP}}}^{\mathrm{SP}^n} \right|^2\Big)^{1/2} \\
&\geq \Big(\left| g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}) - \widetilde{\zeta}_1^{\mathrm{SP}^n} \right|^2 + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_{\mathrm{I}}}^{\mathrm{SP}^n} \right|^2 \\
&\quad + \left| g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_sN_{\mathrm{I}}+1}^{\mathrm{SP}^n} \right|^2 + \cdots + \left| g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n}) - \widetilde{\zeta}_{N_g^{\mathrm{SP}}}^{\mathrm{SP}^n} \right|^2\Big)^{1/2} \\
&=: \left\|\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} - \boldsymbol{\zeta}^{\mathrm{SP}^n}\right\|
\end{aligned}
\tag{4.2}
$$

Note that this corresponds to a projection of the associated points from $\mathbb{R}^{N_g^{\mathrm{SP}}}$ onto $\mathbb{R}^{N_g^{\mathrm{DE}}}$, i.e., $\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n}, \boldsymbol{\zeta}^{\mathrm{SP}^n} \in \mathbb{R}^{N_g^{\mathrm{DE}}}$.

We can now derive the desired upper bound on $\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}}$. By Definition 4.3, we have

$$
\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} = \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n}\right\| - \left\|\boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n}\right\|,
$$

and underestimation of the subtracted part by the projection from 4.2 gives

$$
\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n}\right\| - \left\|\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} - \boldsymbol{\zeta}^{\mathrm{SP}^n}\right\|,
$$

By definition of the infimum in $\mathrm{vio}^n_{\mathrm{DE}}$, we have $\left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n}\right\| \leq \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}\right\|$ for all $\boldsymbol{\zeta} \in \mathbb{R}_-^{N_g^{\mathrm{DE}}}$, in particular, choosing $\boldsymbol{\zeta}^{\mathrm{SP}^n}$ results in

$$
\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{SP}^n}\right\| - \left\|\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} - \boldsymbol{\zeta}^{\mathrm{SP}^n}\right\|.
$$

Applying the reverse triangle inequality gives

$$
\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \widetilde{\boldsymbol{g}}^{\mathrm{SP}^n}\right\|,
$$

and thus by definition of the Euclidian norm and $\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n}$:

$$\begin{aligned}\mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{SP}}^n \leq \Big(&\left|g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n})\right|^2 + \cdots + \left|g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n})\right|^2 \\ &+ \left|g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{DE}^n}) - g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n})\right|^2 \\ &+ \cdots + \left|g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{DE}^n}) - g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n})\right|^2 \Big)^{1/2}.\end{aligned}$$

By Lipschitz continuity of each individual constraint function, all differences can be bounded by the respective Lipschitz constants

$$\begin{aligned}\mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{SP}}^n \leq \Bigg(&\left(L_{g_{\mathrm{I},1}}\left\|\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n} - \widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}\right\|\right)^2 + \cdots + \left(L_{g_{\mathrm{I},N_{\mathrm{I}}}}\left\|\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n} - \widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}\right\|\right)^2 \\ &+ \left(L_{g_{\mathrm{II},1,1}}\left\|\widetilde{\boldsymbol{z}}_1^{\mathrm{DE}^n} - \widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n}\right\|\right)^2 \\ &+ \cdots + \left(L_{g_{\mathrm{II},N_s,N_{\mathrm{II}}}}\left\|\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{DE}^n} - \widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n}\right\|\right)^2 \Bigg)^{1/2}.\end{aligned}$$

Finally, since the maximum distances of points in $\mathcal{X}^n$ and $\mathcal{Z}^n$ are $\sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n)$ and $\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n)$, respectively, and since both $\mathrm{W}(\mathcal{X}^n)$ and $\mathrm{W}(\mathcal{Z}_s^n)$ can be overestimated by $\mathrm{W}(\mathcal{Z}^n)$ we have:

$$\begin{aligned}\mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{SP}}^n \leq \Bigg(&\left(L_{g_{\mathrm{I},1}}\sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n)\right)^2 + \cdots + \left(L_{g_{\mathrm{I},N_{\mathrm{I}}}}\sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n)\right)^2 \\ &+ \left(L_{g_{\mathrm{II},1,1}}\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n)\right)^2 \\ &+ \cdots + \left(L_{g_{\mathrm{II},N_s,N_{\mathrm{II}}}}\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n)\right)^2 \Bigg)^{1/2} \leq c_g\,\mathrm{W}(\mathcal{Z}^n)\end{aligned}$$

Since $\widetilde{\boldsymbol{z}}$ and $\mathcal{Z}^n$ were selected arbitrarily, SP_s^n has at least first-order convergence at all infeasible points with

$$c_g = \sqrt{N_x \sum_{i=1}^{N_{\mathrm{I}}} L_{g,\mathrm{I},i}^2 + (N_x + N_y) \sum_{s\in\mathcal{S}} \sum_{j=1}^{N_{\mathrm{II}}} L_{g_{\mathrm{II},s,j}}^2}.$$

Conclusion: As the LBS based on SP_s^n has convergence orders of $\beta \geq 1$ at both feasible and infeasible points, it has convergence order of $\beta \geq 1$. □

Unsurprisingly, when the Assumption 4.1 is not satisfied, the convergence order of MUSE-BB can also be below 1. For instance, take Example 4.1 but use $w_1 f_1 = -\sqrt{y_1}$; this gives a convergence order of 0.5.

Next we show that both the McCormick based LBS, MC_s^n, as well as its linearization via subtangents, LP_s^n, inherit the first-order convergence of SP_s^n under mild additional assumptions. For both of these convergence results, we require the following assumption:

Assumption 4.2 (first-order pointwise convergent relaxations)**.** The objective function f_s and all elements of the constraint functions $\boldsymbol{g}_{\mathrm{I}}$ and $\boldsymbol{g}_{\mathrm{II},s}$ have first-order *pointwise convergent* relaxations, i.e., there exist constants $c^{\mathrm{MC}}_{f,s} > 0, s \in \mathcal{S}$, $c^{\mathrm{MC}}_{g,\mathrm{I},i} > 0, i = 1,\cdots,N_{\mathrm{I}}$, and $c^{\mathrm{MC}}_{g,\mathrm{II},s,j} > 0, s \in \mathcal{S}, j = 1,\cdots,N_{\mathrm{II}}$, such that for all $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ and any s, the convex relaxations $f^{\mathrm{cv},n}_s, \boldsymbol{g}^{\mathrm{cv},n}_{\mathrm{I}}$ and $\boldsymbol{g}^{\mathrm{cv},n}_{\mathrm{II},s}$ in MC^n_s satisfy

$$\begin{aligned} f_s(\boldsymbol{z}_s) - f^{\mathrm{cv},n}_s(\boldsymbol{z}_s) &\leq c^{\mathrm{MC}}_{f,s}\,\mathrm{W}(\mathcal{Z}^n_s), \quad \forall \boldsymbol{z}_s \in \mathcal{Z}^n_s, \\ g_{\mathrm{I},i}(\boldsymbol{x}) - g^{\mathrm{cv},n}_{\mathrm{I},i}(\boldsymbol{x}) &\leq c^{\mathrm{MC}}_{g,\mathrm{I},i}\,\mathrm{W}(\mathcal{X}^n), \quad \forall \boldsymbol{x} \in \mathcal{X}^n,\ i = 1,\cdots N_{\mathrm{I}}, \\ g_{\mathrm{II},s,j}(\boldsymbol{z}_s) - g^{\mathrm{cv},n}_{\mathrm{II},s,j}(\boldsymbol{z}_s) &\leq c^{\mathrm{MC}}_{g,II,s,j}\,\mathrm{W}(\mathcal{Z}^n_s), \quad \forall \boldsymbol{z}_s \in \mathcal{Z}^n_s,\ j = 1,\cdots N_{\mathrm{II}}. \end{aligned}$$

In fact, for many functions McCormick relaxations satisfying an even stronger variant of Assumption 4.2, with second- instead of just first-order pointwise convergence are known (also see Bompadre and Mitsos, 2011). For our purposes, however, Assumption 4.2 is sufficient.

Corollary 4.1 (first-order convergence of MC^n_s)**.** *Under Assumptions 4.1 and 4.2, MC^n_s has a convergence order of $\beta \geq 1$.*

Proof. By Lemma 4.2, the scheme SP^n_s has first-order convergence with respect to the original problem DE^n. Furthermore, under Assumption 4.2, the LBS MC^n_s has at least first-order convergence with respect to SP^n_s by Theorem 1 of Kannan and Barton, 2017a. Combining these results implies first-order convergence of MC^n_s with respect to DE^n. □

For the first-order convergence of LP^n_s, we additionally require the following assumption:

Assumption 4.3 (Lipschitz convex relaxations)**.** For any node n the convex relaxations $f^{\mathrm{cv},n}_s$, $\boldsymbol{g}^{\mathrm{cv},n}_{\mathrm{I}}$, and $\boldsymbol{g}^{\mathrm{cv},n}_{\mathrm{II},s}$ in MC^n_s are Lipschitz, i.e., there exist constants $L^{\mathrm{MC}}_{f,s} > 0$, $L^{\mathrm{MC}}_{g,\mathrm{I},i} > 0$; $i = 1,\cdots,N_{\mathrm{I}}$, and $L^{\mathrm{MC}}_{g,\mathrm{II},s,j}$, $j = 1,\cdots,N_{\mathrm{II}}$, that constitute upper bounds on the norm of the respective subgradients. In particular, this implies:

$$\begin{aligned} \|\check{\nabla} f^{\mathrm{cv},n}_s(\boldsymbol{z}_s)^{\intercal}(\boldsymbol{z}'_s - \boldsymbol{z}_s)\| &\leq L^{\mathrm{MC}}_{f,s}\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}^n_s), \quad \forall \boldsymbol{z}_s, \boldsymbol{z}'_s \in \mathcal{Z}^n_s \\ \|\check{\nabla} g^{\mathrm{cv},n}_{\mathrm{I},i}(\boldsymbol{x})^{\intercal}(\boldsymbol{x}' - \boldsymbol{x})\| &\leq L^{\mathrm{MC}}_{g,\mathrm{I},i}\sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n), \quad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}^n,\ i = 1,\cdots N_{\mathrm{I}}, \\ \|\check{\nabla} g^{\mathrm{cv},n}_{\mathrm{II},s,j}(\boldsymbol{z}_s)^{\intercal}(\boldsymbol{z}'_s - \boldsymbol{z}_s)\| &\leq L^{\mathrm{MC}}_{g,\mathrm{II},s,j}\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}^n_s), \quad \forall \boldsymbol{z}_s, \boldsymbol{z}'_s \in \mathcal{Z}^n_s,\ j = 1,\cdots N_{\mathrm{II}}. \end{aligned}$$

Assumption 4.3 is satisfied if the relaxations used for all intrinsic functions are Lipschitz (cf. Scott, Stuber, and Barton, 2011). This in turn is the case for standard relaxations of a wide class of functions, provided they are Lipschitz themselves.

Corollary 4.2 (first-order convergence of LP^n_s)**.** *Under Assumptions 4.1–4.3, LP^n_s has a convergence order of $\beta \geq 1$.*

Proof. We structure the proof as in Lemma 4.2.

Convergence Order at Feasible Points: First consider some arbitrary point $\widetilde{\boldsymbol{z}} \in \mathcal{Z}$ that is feasible in DE. For any subset $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ with $\widetilde{\boldsymbol{z}} \in \mathcal{Z}^n$, let $\boldsymbol{z}^{\mathrm{DE}^n}_s$ and $\boldsymbol{z}^{\mathrm{LP}^n}_s$ be solutions of DE^n, and LP^n_s, respectively, and note that

$$f^n_{\mathrm{LP},s} = \mathrm{sub}^n_{f_s}(\boldsymbol{z}^{\mathrm{LP}^n}_s) = f^{\mathrm{cv},n}_s(\boldsymbol{m}^n_{\boldsymbol{z}_s}) + \check{\nabla} f^{\mathrm{cv},n}_s(\boldsymbol{m}^n_{\boldsymbol{z}_s})^{\intercal}(\boldsymbol{z}^{\mathrm{LP}^n}_s - \boldsymbol{m}^n_{\boldsymbol{z}_s}),$$

where $\boldsymbol{m}^n_{\boldsymbol{z}_s}$ is the midpoint of $\mathcal{Z}^n_s$, see subtangent. We can bound the difference of optimal values of DE^n, and LP^n_s by subtracting and adding the terms $f_s(\boldsymbol{m}^n_{\boldsymbol{z}_s})$ and applying Assumptions 4.1–4.3:

$$\begin{aligned} f^n_{\mathrm{DE}} - f^n_{\mathrm{LP}} &= \sum_{s\in\mathcal{S}} w_s\big(f^n_{\mathrm{DE},s} - f^n_{\mathrm{LP},s}\big) = \sum_{s\in\mathcal{S}} w_s\big(f_s(\boldsymbol{z}_s^{\mathrm{DE}^n}) - \mathrm{sub}^n_{f_s}(\boldsymbol{z}_s^{\mathrm{LP}^n})\big) \\ &= \sum_{s\in\mathcal{S}} w_s\Big(f_s(\boldsymbol{z}_s^{\mathrm{DE}^n}) - f_s(\boldsymbol{m}^n_{\boldsymbol{z}_s}) + f_s(\boldsymbol{m}^n_{\boldsymbol{z}_s}) - f^{\mathrm{cv},n}_s(\boldsymbol{m}^n_{\boldsymbol{z}_s}) \\ &\qquad\qquad - \check{\nabla} f^{\mathrm{cv},n}_s(\boldsymbol{m}^n_{\boldsymbol{z}_s})^\intercal(\boldsymbol{z}_s^{\mathrm{LP}^n} - \boldsymbol{m}^n_{\boldsymbol{z}_s})\Big) \\ &\leq \sum_{s\in\mathcal{S}} w_s c^{\mathrm{LP}}_{f,s}\,\mathrm{W}(\mathcal{Z}^n) \end{aligned}$$

where $c^{\mathrm{LP}}_{f,s} \coloneqq \big((L_{f,s} + L^{\mathrm{MC}}_{f,s})\sqrt{N_x + N_y} + c^{\mathrm{MC}}_{f,s}\big)$. Thus LP^n_s has first-order convergence at feasible points with $c_f = \sum_{s\in\mathcal{S}} w_s c^{\mathrm{LP}}_{f,s}$.

Convergence Order at Infeasible Points: Now consider some arbitrary infeasible point $\widetilde{\boldsymbol{z}}$, and any subset $\mathcal{Z}^n \in \mathbb{I}\mathcal{Z}$ such that $\widetilde{\boldsymbol{z}} \in \mathcal{Z}^n$. As in the proof of Lemma 4.2, let $\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n} = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}) = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}_1^{\mathrm{DE}^n}, \cdots, \widetilde{\boldsymbol{y}}_{N_s}^{\mathrm{DE}^n}) \in \mathcal{Z}^n$, and $\boldsymbol{\zeta}^{\mathrm{DE}^n} \in \mathbb{R}_-^{N_g^{\mathrm{DE}}}$ be points at which the minimum constraint violation $\mathrm{vio}^n_{\mathrm{DE}}$ is attained, i.e., $\mathrm{vio}^n_{\mathrm{DE}} = \big\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n}\big\|$, and, let $\widetilde{\boldsymbol{\xi}}^{\mathrm{LP}^n} = (\widetilde{\boldsymbol{x}}_1^{\mathrm{LP}^n}, \widetilde{\boldsymbol{y}}_1^{\mathrm{LP}^n}, \cdots, \widetilde{\boldsymbol{x}}_{N_s}^{\mathrm{LP}^n}, \widetilde{\boldsymbol{y}}_{N_s}^{\mathrm{LP}^n}) \in \bigtimes_{s\in\mathcal{S}}(\mathcal{Z}^n_s)$ and $\widetilde{\boldsymbol{\zeta}}^{\mathrm{LP}^n} \in \mathbb{R}_-^{N_g^{\mathrm{LP}}}$ be points at which the minimum constraint violation $\mathrm{vio}^n_{\mathrm{LP}}$ is attained, i.e., $\mathrm{vio}^n_{\mathrm{LP}} = \Big\|\boldsymbol{g}_{\mathrm{LP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{LP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{LP}^n}\Big\|$, where $\boldsymbol{g}_{\mathrm{LP}}$ is the vector-valued function containing the constraints of all LP^n_s, i.e., the subtangents of the entries in $\boldsymbol{g}_{\mathrm{SP}}$, see subtangent.

Using the same arguments as in the proof of Lemma 4.2 with $\boldsymbol{g}_{\mathrm{LP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{LP}^n})$ instead of $\boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n})$ we can bound the difference in violation measures of DE^n and LP^n_s, resulting in:

$$\begin{aligned} \mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{LP}} \leq \Big(&\Big|g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - \mathrm{sub}^n_{g_{\mathrm{I},1}}(\widetilde{\boldsymbol{x}}_1^{\mathrm{LP}^n})\Big|^2 + \cdots + \Big|g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - \mathrm{sub}^n_{g_{\mathrm{I},N_{\mathrm{I}}}}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n})\Big|^2 \\ &+ \Big|g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{DE}^n}) - \mathrm{sub}^n_{g_{\mathrm{II},1,1}}(\widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n})\Big|^2 \\ &+ \cdots + \Big|g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{DE}^n}) - \mathrm{sub}^n_{g_{\mathrm{II},N_s,N_{\mathrm{II}}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n})\Big|^2\Big)^{1/2} \end{aligned}$$

as with the objective function, we can bound the differences between each constraint function and the respective subgradient, using Assumptions 4.1–4.3, which results in

$$\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{LP}} \leq c^{\mathrm{LP}}_g\,\mathrm{W}(\mathcal{Z}^n),$$

where

$$\begin{aligned} c^{\mathrm{LP}}_g \coloneqq \Bigg(&\sum_{i=1}^{N_{\mathrm{I}}}\Big((L_{g,\mathrm{I},i} + L^{\mathrm{MC}}_{g,\mathrm{I},i})\sqrt{N_x} + c^{\mathrm{MC}}_{g,\mathrm{I},i}\Big)^2 \\ &+ \sum_{s\in\mathcal{S}}\sum_{j=1}^{N_{\mathrm{II}}}\Big((L_{g,\mathrm{II},s,j} + L^{\mathrm{MC}}_{g,\mathrm{II},s,j})\sqrt{N_x + N_y} + c^{\mathrm{MC}}_{g,\mathrm{II},s,j}\Big)^2\Bigg)^{1/2}. \end{aligned}$$

Thus LP_s^n has first-order convergence at any infeasible point with $c_g = c_g^{\mathrm{LP}}$.

Conclusion: As the LBS based on LP_s^n has convergence orders of $\beta \geq 1$ at both feasible and infeasible points, it has convergence order of $\beta \geq 1$. □

We are now in the position to prove finite ε_f-convergence of MUSE-BB.

Corollary 4.3 (finite termination of MUSE-BB)**.** *Under Assumptions 4.1–4.3, MUSE-BB terminates finitely for any optimality tolerance $\varepsilon_f > 0$, either providing an ε_f-optimal solution or a certificate that the problem is infeasibile.*

Proof. By Lemma 4.1, each sequence of descendant nodes converges to some accumulation point $\widetilde{\boldsymbol{z}}$. We show that the use of any LBS R_s^n with convergence order of $\beta > 0$ implies that all such sequences finitely reach a node that can be fathomed by value dominance or infeasibility.

Convergence at Feasible Points: First consider sequences for which $\widetilde{\boldsymbol{z}}$ is feasible. After a finite number of iterations, any such sequence will produce a node n, for which $\mathrm{W}(Z^n) \leq \left(\frac{\varepsilon_f}{c_f}\right)^{1/\beta}$, which implies that $f_{\mathrm{DE}}^n - f_{\mathrm{R}}^n \leq \varepsilon_f$, i.e., that n is fathomed by value dominance.

Convergence at Infeasible Points: Next consider sequences for which $\widetilde{\boldsymbol{z}}$ is infeasible, and which are not terminated finitely because some descendant node can be fathomed by value dominance. By compactness of the feasible set, any such sequence will eventually produce a node $\widetilde{n}$ that contains no feasible point, and thus has a positive violation measure $\mathrm{vio}_{\mathrm{DE}}^{\widetilde{n}}$. Since the violation measure increases monotonically for descendants of node $\widetilde{n}$, the sequence is terminated when or before the descendant node n is produced, for which $\mathrm{W}(Z^n) \leq \left(\frac{\mathrm{vio}_{\mathrm{DE}}^{\widetilde{n}}}{c_g}\right)^{1/\beta}$, as this implies $0 \leq \mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{DE}}^{\widetilde{n}} \leq \mathrm{vio}_{\mathrm{R}}^n$, i.e., infeasibility is detected by the scheme R_s^n, and node n is fathomed by infeasibility.

Conclusion: In summary, each node sequence terminates finitely and since the original domain is compact, the total number of sequences must be finite. By Corollary 4.2, the assumptions imply that the LBS $\mathrm{R}_s^n = \mathrm{LP}_s^n$, used in MUSE-BB has a convergence order of $\beta > 1$, thus MUSE-BB terminates finitely, once all sequences of descendant nodes are terminated. □

After demonstrating first-order convergence of the LBS employed by MUSE-BB and the resulting ε_f-convergence, we now consider in which cases these convergence properties may be sufficient to mitigate clustering. As indicated by Kannan and Barton, 2017b, clustering may be mitigated around individual minimizers of DE, if the convergence order of the LBS is larger or equal to the order at which objective and constraint functions grow around this minimizer. While Example 4.2 demonstrates that SP_s^n (and by extension, also LP_s^n) may have a convergence order as low as one at *constrained minimizers*, objective and constraint functions often grow at a linear rate around such points (Kannan and Barton, 2017b). Therefore LP_s^n may mitigate clustering around certain constrained minimizers, provided the respective coefficients c_f and c_g are sufficiently small (Kannan and Barton, 2017b). On the other hand, at *partially or unconstrained minimizers*, where f is differentiable, f grows quadratically or faster in some of the feasible directions. As a result, a LBS needs to have at least second-order convergence at unconstrained minimizers to to mitigate clustering (Du and Kearfott, 1994; Wechsung, Schaber, and Barton, 2014; Kannan and Barton, 2017b). Unfortunately, the convergence order of SP_s^n may also be as low as one at unconstrained minimizers, as shown by the following example.

Example 4.3. Consider an instance of DE with $N_x = 1, N_y = 0, N_s = 2$ and an original domain $\mathcal{X} = [-1, 1]$. Take

$$w_1 f_1(x_1) = 0.5(x_1 - 1)^2; \qquad w_2 f_2(x_2) = 0.5(x_2 + 1)^2$$

such that $f(x) = x^2 + 1$, and thus the optimal solution and objective value are $x^{\mathrm{DE}} = 0$, and $f(x^{\mathrm{DE}}) = 1$, respectively. For any nested sequence of nodes converging to this optimum, the solutions x^{DE^n} of the node problem DE^n lie in $\mathcal{X}^n = [\underline{x}^n, \overline{x}^n]$, and thus $\underline{x}^n \leq 0, \overline{x}^n \geq 0$. For such nodes, the solutions of SP_s^n are $x_1^{\mathrm{SP}^n} = \overline{x}^n$, and $x_2^{\mathrm{SP}^n} = \underline{x}^n$, respectively. Hence the difference in objective values is:

$$\begin{aligned} f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n &= 1 - 0.5\Big((\overline{x}^n - 1)^2 + (\underline{x}^n + 1)^2\Big) \\ &= -0.5\,(\overline{x}^n)^2 + \overline{x}^n - \underline{x}^n - 0.5\,(\underline{x}^n)^2 \end{aligned}$$

Now consider a sequence for which $\overline{x}^n = W^n, \underline{x}^n = 0$; for this sequence the above expression simplifies to

$$f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = W^n - 0.5(W^n)^2.$$

Now for any $c_f > 0$ this expression becomes larger than $c_f (W^n)^2$ for the node n_0, for which

$$W^{n_0} < \frac{1}{c_f + 0.5},$$

i.e., SP_s^n is at best first-order convergent at the unconstrained minimizer x^{DE^n}.

In summary, the present implementation of MUSE-BB may suffer from clustering around unconstrained minimizers. To address this, an alternative LBS with at least quadratic convergence order is required. In the following section we analyze an extension of MUSE-BB whose LBS has this property.

4.5.3. Second-Order Convergence

In this section we show that using LSP_s^n instead of SP_s^n, i.e., dualizing the NACs instead of dropping them, enables at least second-order convergence at unconstrained minimizers. Additionally, we consider the resulting effect on the implementation, i.e., how the LBS LP_s^n needs to be adapted when using LSP_s^n.

A necessary condition for a LBS to have β-order convergence is that the relaxations used for its construction have β-order convergence, also see Kannan and Barton, 2017a. While this condition is generally not sufficient for β-order convergence of the resulting LBS, it is sufficient for β-order convergence around *Slater points, i.e., unconstrained feasible points* (Corollaries 2, 3 of Kannan and Barton, 2017a).

Corollary 6 of Robertson, Cheng, and Scott, 2024 shows that the optimal objective value $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*)$, obtained from the subproblems $\mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, where the NACs are dualized instead of dropped, is equivalent to minimizing the w_s-weighted sum of convex envelopes of $f_s^{\mathcal{X}^n,\mathcal{Y}}$. As a result, $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*)$ constitutes a (constant valued) relaxation of the objective function f on the domain $\mathcal{X}^n \times \mathcal{Y}$. Furthermore, they show that this relaxation is at least

second-order convergent with respect to $\mathrm{W}(\mathcal{X}^n)$, i.e., for some $c > 0$

$$\min_{\boldsymbol{x}\in\mathcal{X}^n} \sum_{s\in\mathcal{S}} f_s^{\mathcal{X}^n,\mathcal{Y}_s}(\boldsymbol{x}) - f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*) \leq c\,\mathrm{W}(\mathcal{X}^n)^\beta$$

with $\beta \geq 2$, provided the scenario value functions $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are $\mathcal{C}^2$, i.e., twice continuously differentiable. We point out that in fact, the slightly weaker assumption that $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ merely has bounded second-order directional derivatives, i.e., that it is $\mathcal{C}^{1,1}$, is already sufficient for second-order convergence of $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*)$, also see Zlobec, 2005. In the special case where the $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are convex, β above may take any positive value, i.e., the convergence is arbitrarily high. Note that β-order convergence of $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*)$ immediately implies β-order convergence of the LBS $\mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ at unconstrained feasible points (and in particular at unconstrained minimizers), because around such points f_{DE}^n, i.e., the optimal value of DE^n, is equivalent to $\min_{\boldsymbol{x}\in\mathcal{X}^n} \sum_{s\in\mathcal{S}} f_s^{\mathcal{X}^n,\mathcal{Y}_s}(\boldsymbol{x})$, also see Corollaries 2 and 3 of Kannan and Barton, 2017a. Furthermore, β-order convergence with respect to $\mathrm{W}(\mathcal{X}^n)$ implies β-order convergence with respect to $\mathrm{W}(\mathcal{Z}^n)$, since $\mathrm{W}(\mathcal{X}^n) \leq \mathrm{W}(\mathcal{Z}^n)$. As a result, the same line of argument naturally also holds for the scheme $\mathrm{LSP}_s^n := \mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, which for any $\mathcal{X}^n$ produces stronger bounds than $\mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$. Hence, the relaxations $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}^n}(\boldsymbol{\lambda}^*)$ are at least second-order convergent, and Corollaries 2 and 3 of Kannan and Barton, 2017a ensure second-order convergence of LSP_s^n at unconstrained feasible points. The following example demonstrates the improvement of convergence order of LSP_s^n over SP_s^n.

Example 4.4. Take the problem from Example 4.3. The optimal dual values for this problem are $\boldsymbol{\lambda}_s^* = (2, -2)$, such that the objectives of LSP_s^n are $f_s(x_s) + \lambda_s x_s = (x_s \mp 1)^2 \pm 2x_s = x_s^2 + 1$. Hence, both subproblems are solved at $x_1^{\mathrm{LSP}^n} = x_2^{\mathrm{LSP}^n} = x^{\mathrm{DE}^n} = 0$, and the difference in objective values is:

$$f_{\mathrm{DE}}^n - f_{\mathrm{LSP}}^n = 1 - 0.5\Big((0+1)^2 + (0+1)^2\Big) = 0,$$

i.e., LSP_s^n is exact and as such has arbitrarily high convergence order at the unconstrained minimizer x^{DE^n}.

Note that the arbitrarily high convergence order in Example 4.4 results from the fact that the scenario value functions $f_s^{\mathcal{X},\mathcal{Y}_s}$ are convex. If $f_s^{\mathcal{X},\mathcal{Y}_s}$ are not convex, at least second-order convergence is guaranteed by the previous arguments.

Several results from nonlinear parametric programming provide different regularity conditions under which $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are $\mathcal{C}^2$. In particular, if we assume f is $\mathcal{C}^2$, and that the second-order sufficient condition (SOSC):

$$\begin{aligned} \nabla f(\boldsymbol{z}^{\mathrm{DE}}) &= \mathbf{0} \\ \nabla^2 f(\boldsymbol{z}^{\mathrm{DE}}) &\succ 0 \end{aligned} \qquad (\mathrm{SOSC}(\boldsymbol{z}^{\mathrm{DE}}))$$

holds at an unconstrained minimizer $\boldsymbol{z}^{\mathrm{DE}} = (\boldsymbol{x}^{\mathrm{DE}}, \boldsymbol{y}^{\mathrm{DE}})$ of DE, the fact that $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are $\mathcal{C}^2$ follows from the Implicit Function Theorem (Fiacco, 1983, cf., e.g., Corollary 3.2.3).

Other variants of the Implicit Function Theorem provide similar results for unconstrained minimizers that do not satisfy $\mathrm{SOSC}(\boldsymbol{z}^{\mathrm{DE}})$, e.g., Theorem 3.3 of Ginchev, Torre, and Rocca, 2009, or even for constrained minimizers, satisfying certain regularity conditions, related to the growth of the Lagrangian of f, e.g., Fiacco, 1983 and Stechlinski, Khan, and Barton,

2018.

We next show how the stronger convergence properties of the LBS LSP_s^n can be incorporated into MUSE-BB via an adaption of the lower bounding problems subproblems LP_s^n. Recall that LP_s^n result from three subsequent levels of relaxation: after dropping the NACs from $\text{DE}_{\text{NAC}}^{\mathcal{X},\mathcal{Y}}$ (i), the resulting subproblems SP_s^n are further relaxed via McCormick's method (ii) and outer approximation (iii), resulting in the linear lower bounding problems LP_s^n. In this context, dualizing the NACs, corresponds to replacing the subproblems SP_s^n with LSP_s^n, and performing the subsequent relaxations. Note that the only difference between SP_s^n and LSP_s^n are the additional terms $\boldsymbol{\lambda}_s^\intercal \boldsymbol{x}_s$. The McCormick relaxation of the sum of the original, nonlinear objective $f_s(\boldsymbol{x}_s, \boldsymbol{y}_s)$, and the linear term $\boldsymbol{\lambda}_s^\intercal \boldsymbol{x}_s$ is simply $f_s^{\text{cv},n}(\boldsymbol{x}_s, \boldsymbol{y}_s) + \boldsymbol{\lambda}_s^\intercal \boldsymbol{x}_s$ (cf. Proposition 2 of Bompadre and Mitsos, 2011). Next we consider the subtangents of these terms: if $\check{\nabla} f_s^{\text{cv},n}$ is the subgradient of $f_s^{\text{cv},n}$, used in the original instance of LP_s^n, then $\check{\nabla} f_s^{\text{cv},n} + \boldsymbol{\lambda}_s$ is a valid subgradient of $f_s^{\text{cv},n}(\boldsymbol{x}_s, \boldsymbol{y}_s) + \boldsymbol{\lambda}_s^\intercal \boldsymbol{x}_s$ (cf. Proposition 2.3.3 of Clarke, 1990). As a result, replacing SP_s^n with LSP_s^n in MUSE-BB is equivalent to adding $\boldsymbol{\lambda}_s$ to the coefficients of $\boldsymbol{x}_s$ in the first set of constraints, of the subproblems LP_s^n.

While conceptually, the multipliers can be updated by performing dual iterations with these modified linear lower bounding subproblems, such updates will generally not converge to the optimal multipliers of the original problem LSP_s^n. Even though convergence over a sequence of nodes can be expected, as the node size diminishes and the McCormick relaxations, and linear relaxations converge towards the original functions, such a conversion in the limit may not be sufficient to yield second-order convergence of the resulting lower bounding scheme.

In summary, similar to PBDAs, the LBS used in MUSE-BB may be made second-order convergent at certain minimizers by dualizing the NACs instead of dropping them. However, the use of optimal dual multipliers $\boldsymbol{\lambda}^*$ appears to be a requirement for second-order convergence, and, as already pointed out in Section 4.4.1, obtaining such multipliers is generally very challenging. The fact that SP_s^n can be interpreted as an instance of LSP_s^n with the suboptimal multipliers $\boldsymbol{\lambda} = \boldsymbol{0}$, indicates that suboptimal multipliers may result in a first-order convergent LBS, also see the related result on a Lagrangian dual-based LBS for general nonlinear programming problems in Theorem 6 of Kannan and Barton, 2017a. While it may be sufficient to limit multiplier updates to small nodes suspected to contain the neighborhoods of critical minimizers, we leave the investigation of such approaches for future work.

4.6. Computational Results

We now present computational results obtained with the parallelized decomposition algorithm MUSE-BB, and outline how it compares against solving the deterministic equivalent formulation $\text{DE}^{\mathcal{X},\mathcal{Y}}$ with the standard version of MAiNGO. MUSE-BB performs upper bounding based on the subproblems SP_s^n, and OBBT, lower bounding, and DBBT, based on the separable subproblems LP_s^n. All scenario subproblems are solved simultaneously, using one thread per scenario. MAiNGO performs upper bounding based on DE^n and all other routines based on a linearization of DE^n, using a single thread.

We do not compare with other deterministic global solvers as these generally employ different routines for management of the B&B tree, generating relaxations of individual

functions, and solving individual lower and upper bounding problems, distorting the effect of the decomposition. Further, we focus our computational experiments on the effects of individual algorithm parameters, rather than conducting larger-scale computational studies, as the latter would require access to a library of two-stage test problems, which is currently unavailable. While previous works do consider some larger-scale problems, the implementations are either unpublished (e.g., the test library "GOSSIP" from Kannan, 2018) or a generic formulation is given while the concrete problem data is not (see, e.g., Section 7.1 in Li and Grossmann, 2019b).

We consider variants of a simple test problem with $N_x = N_y = 1$ and $N_s = 4$, 8, and 16, i.e., with different size based on the number of scenarios. The test problem is a simplified design and operation problem for a combined heat and power (CHP) system, based on stochastic heat and power demands. Scenarios for demand data are generated from a seeded pseudorandom sampling, ensuring identical instances upon repetition for a given N_s value. The problem involves nonlinearities related to economies of scale, thermal and electrical efficiencies, and the implementation of a minimal part-load constraint. A detailed description of the problem is given in Appendix B.

We focus on the performance difference of the lower bounding routines, hence all experiments are performed with initial points based on dense uniform sampling of 1000 values in each of the x and y_s domains, which always results in ε_f-optimal initial points, that are never improved during the course of the algorithm. We use the default settings of MAiNGO, including a relative optimality tolerance of 1%. All computational experiments are performed on the RWTH Compute Cluster "CLAIX-2018". Each compute node has 2 Intel Xeon Platinum 8160 Processors with 2.1 GHz, 24 cores each, i.e., there is a total of 48 cores per compute node, and 4 GB of main memory per core. In initial tests we observed significant variation of run times, both for MAiNGO and MUSE-BB. We attribute this variation to execution on particular – likely overloaded – compute nodes which consistently require longer solution times compared to other compute nodes. To reduce the effect of this variation, we repeat the solution of each considered instance 20 times and report median values of the resulting solution times and optimality gaps.

4.6.1. Importance of Branching Priority

Initially we will focus on the case $k_{\max} = 1$, i.e., we branch only on second-stage variable instances that either produce infeasible subproblems or produce the highest strong-branching score. This means each multisection of second stage variables results in at most 2 child nodes being created, i.e., as in a standard B&B algorithm like MAiNGO.

In problems like DE, exhibiting two-stage structure, the first-stage variables appear in all of the scenario subproblems, while the second-stage variable instances only appear in one, each. This suggests a higher importance of branching on first-stage vs. second-stage variables, especially with increasing N_s. In B&B algorithms, the priority with which variables are branched is typically controlled via branching priorities for individual variables, which are multiplied with the relative interval width before selecting a variable to branch on, also cf. the description of Subroutine 1. As a result, it seems intuitive that B&B algorithms solving DE may generally benefit from relatively high branching priorities for the first-stage variables compared to the second-stage variables, independent of whether decomposition is used or not. For this reason, we compare how MAiNGO and MUSE-BB

perform with different branching priority ratios

$$\rho = \frac{\text{first-stage branching priority}}{\text{second-stage branching priority}},$$

which in the present case ($N_x = N_y = 1$) correspond to the branching priority of x (the priority for y_s being 1).

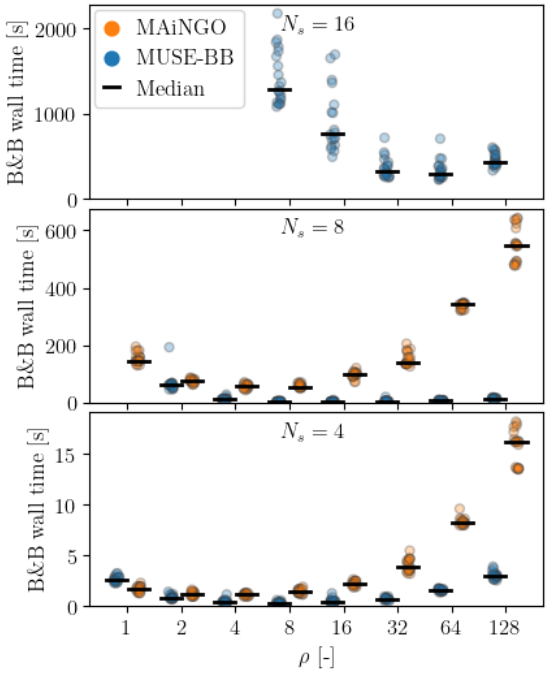


Fig. 4.3. Variation of solution time for deterministic equivalent (MAiNGO) and parallel decomposition (MUSE-BB) with ρ for $N_s \in \{4, 8, 16\}$ over 20 runs each. Parameter combinations without data points did not terminate within 3600s, also see Tab. 4.1.

Tab. 4.1. Median B&B wall times in seconds over 20 runs, or remaining relative optimality gaps in % (computed as 1 - ratio of lower to upper bound) after 3600 s for solving the CHP sizing model with different number of scenarios and branching priorities, using MAiNGO and MUSE-BB. Two out of the 20 runs for the instance $N_s = \rho = 16$, solved with MUSE-BB, timed out. The median is computed with respect to the remaining 18 runs. Minima for each column (highlighted in bold) indicate that the performance of MUSE-BB relatively to MAiNGO improves with an increase of scenarios, and thus problem size.

algor.	MAiNGO			MUSE-BB		
$\rho \backslash N_s$	4	8	16	4	8	16
1	1.7	145	17%	2.6	2.4%	20%
2	1.1	77	8.8%	0.82	62	7.6%
4	**1.1**	59	4.6%	0.42	12	2.8%
8	1.4	**55**	**3.7%**	**0.33**	4.3	1292
16	2.2	98	3.8%	0.43	**3.5**	765
32	3.9	142	4.3%	0.75	5.2	329
64	8.2	342	5.0%	1.6	7.8	**295**
128	16	550	5.7%	3.0	14	436

Fig. 4.3 shows the wall times spent in B&B when using MAiNGO and MUSE-BB on problem instances with $N_s \in \{4, 8, 16\}$. Both individual times (colored dots), as well as the median times (horizontal lines) are depicted. Tab. 4.1 lists the median wall times and relative gaps for instances which do not terminate within the time-limit of one hour. In general, the ρ values minimizing average wall time for each scenario are much lower for MAiNGO than for MUSE-BB. However, low ρ values lead to significantly worse performance for MUSE-BB than for MAiNGO, e.g., all runs for $(N_s, \rho) = (8, 1)$ time out after one hour with a median remaining gap of 2.4%. For $N_s = 16$, all instances solved with MAiNGO time out, while for MUSE-BB almost all instances with ρ values above 4 terminate (with the exception of two outliers for $\rho = 16$). This indicates the importance of appropriate branching priorities when solving stochastic problems in general, and when using MUSE-BB in particular. When comparing the best ρ values for each scenario (bold in Tab. 4.1), MUSE-BB outperforms MAiNGO in terms of wall time by a factor of 3.5 and 15 for $N_s = 4$ and $N_s = 8$, respectively. For $N_s = 16$ the value is expected to be

significantly larger than $3600/295.3 \approx 12$.

Even for these relatively small problem sizes, the performance improvement of MUSE-BB over MAiNGO is already comparable to, or even exceeds the number of scenarios and thus the number of used threads. This implies that MUSE-BB can be more favorable than more general parallelization approaches such as, e.g., the MPI parallelization of MAiNGO, where open nodes are processed by different CPUs (not used in this chapter). While such general parallelization approaches are more widely applicable, they do not exploit the special problem structure of DE. Consequently they may be used in conjunction with the parallel processing of individual B&B nodes presented in this chapter to optimally use computational infrastructure.

The results indicate that optimal branching priority ratios (i.e., ρ values resulting in minimal median wall time) may increase with the number of scenarios considered. Nevertheless, a projection-based approach, where only the first-stage variables are branched (corresponding to $\rho \to \infty$) appears unfavorable, as wall times increase significantly for large ρ-values.

4.6.2. Effect of Multisection

We next consider the effect of the multisection parameters $k_{\max}$, and τ. Recall that every time a second-stage variable is selected for branching, we solve the $2N_s$ independent subproblems, resulting from the multisection involving the corresponding N_s variable instances for different scenarios. We then use the results to compute strong-branching scores σ_s for each scenario, and create up to $2^{k_{\max}}$ child nodes, with the actual number being controlled by the value of the strong-branching threshold $\tau \in (0, 1]$, i.e., we reject scenarios with a strong-branching score below $\tau\sigma_s$, see Section 4.4.4.

For each N_s value, we take the three ρ values for which MUSE-BB performed best at $k_{\max} = 1$, and perform further experiments for $k_{\max} \in \{2, 4, 8\}$, and $\tau \in \{0.1, 0.2, 0.5, 0.8, 1\}$. Increasing values of $k_{\max}$, and decreasing values of τ allow a larger number of child nodes to be created from each multisection, i.e., the maximum is $2^8 = 256$ for $(k_{\max}, \tau) = (8, 1)$. We point out that multiple variables may achieve the maximum strong-branching score. Hence, even for $\tau = 1$, the settings $k_{\max} = 1$, and $k_{\max} > 1$, may produce different B&B trees (and thus wall times) for a given problem instance, as the latter setting allows creating more than 2 child nodes, while the former does not.

As before, we repeat the solution for each parameter combination 20 times. Since combinations with $N_s = 4$, and $N_s = 8$, show no clear trend for the effect of $k_{\max}$, or τ, we only focus on combinations with $N_s = 16$, the results of which are depicted in Fig. 4.4. The B&B wall times of all investigated combinations are visualized in Fig. C.1 in Chapter C. Only a small set of parameter combinations results in improvements over the best median wall time for $k_{\max} = 1$, (i.e., 295 s for $\rho = 64$). However, these improvements are mostly insignificant, with the best median wall time of 272 s (achieved for $(k_{\max}, \rho, \tau) = (4, 32, 1)$) corresponding to an improvement of less than 8%. For the remaining parameter combinations median wall times remain the same or increase. While combinations with $(N_s, k_{\max}) = (16, 2)$ show no clear trend for the effect of τ, for $(N_s, k_{\max}) = (16, 4)$, and $(16, 8)$, an increase of τ results in reductions of wall time. We point out that setting the strong-branching threshold τ to a value of 1 produces very similar results as setting $k_{\max}$ to 1, since only bisections that produce the highest strong-branching score may be selected. In fact, for the considered parameter combinations, the total number of iterations for $k_{\max} > 1$ only depends on τ,

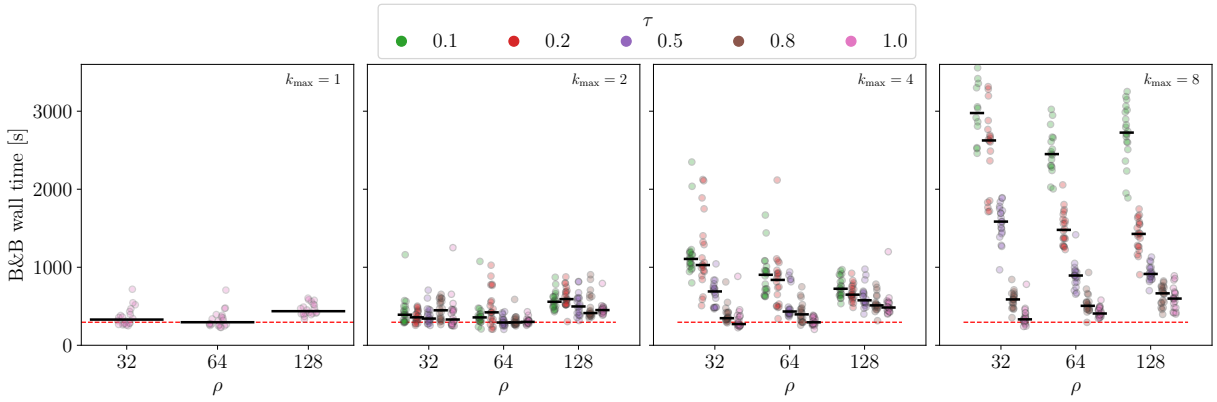


Fig. 4.4. Variation of solution times for solving the CHP sizing problem using MUSE-BB with $k_{\max} \in \{1, 2, 4, 8\}$, and $\tau \in \{0.1, 0.2, 0.5, 0.8, 1\}$ for the three ρ values resulting in the lowest median wall times for $(k_{\max}, N_s) = (1, 16)$. For each parameter combination, a set of 20 runs is performed. For $(k_{\max}, \rho, \tau) = (8, 32, 0.1)$, $(8, 32, 0.2)$, and $(8, 64, 0.1)$, 8, 2, and 3 runs timed out after 3600 s, respectively. Medians with respect to the remaining runs are depicted as horizontal black lines and the lowest median time for $k_{\max} = 1$ is depicted as a dashed red line for reference. Increasing $k_{\max}$ generally results in larger wall times, and for $k_{\max} = 4$ and 8, increasing τ results in smaller wall times.

with the corresponding values being around 0.06% lower than those for $k_{\max} = 1$.

For the considered problem, increasing $k_{\max}$ and reducing τ tends to result in increases of median wall times, however, the generality of this finding needs to be investigated with a larger group of problems. In particular, it is conceivable that for problems in which multiple variable instances have a comparatively strong effect on the objective or feasible set, values of $k_{\max} > 1$ and $\tau < 1$ may be preferable.

For $k_{\max} = 1$ or $\tau = 1$, the behavior of MUSE-BB is very similar to that of a standard B&B algorithm solving the deterministic equivalent with a strong-branching heuristic. While this is not commonly done, range reduction similar to that of MUSE-BB, i.e., using the intersections of variable domains from rejected bisections for the selected one, may also be done on the basis of full-space bounding problems within classical strong-branching. Whereas MUSE-BB solves smaller, independent subproblems, the bounds obtained from such an adapted strong-branching routine in a standard B&B are naturally stronger. This trade-off appears to be worth further study in future work.

4.6.3. Scaling with N_s

As we pointed out in Section 4.1, the fact that MUSE-BB employs a B&B search in the full variable space implies that the number of nodes visited, and thus computational effort, scales exponentially with N_s in the worst-case. This is despite the fact that the proposed multisection branching allows processing an exponential number of nodes with an effort that is linear in N_s, since each of the resulting nodes may need to be further branched and processed.

The computational results from previous sections confirm this expected superlinear scaling with N_s, but also highlight the superiority of MUSE-BB over the solution of the deterministic equivalent via MAiNGO. For MAiNGO the computational time of the best

parameter combinations increases by a factor of 50 (55 s / 1.1 s) when going from $N_s = 4$ to $N_s = 8$.(cf. bold times in Tab. 4.1). In contrast, for MUSE-BB, the corresponding factor is only 10.6 (3.5 s / 0.33 s).

The optimal value of ρ appears to scale approximately linearly with N_s. Thus it may appear that for large numbers of scenarios, MUSE-BB will behave somewhat like a PBDA, in the sense that branching is done primarily on $\boldsymbol{x}$. However, recall that each time a particular second-stage variable instance is selected for branching, the current implementation chooses all other instances of that variable for multisection. Since the number of second-stage variable instances increases linearly with N_s, a fixed value of ρ would thus result in more frequent branching on a given second-stage variable instance. Therefore the observed increase of ρ does not necessarily imply a more frequent branching of $\boldsymbol{x}$, but can rather be seen as a compensation for the above-mentioned behavior. Furthermore, unlike PBDAs, MUSE-BB avoids the global solution of subproblems. This difference in computational effort complicates direct comparisons based on individual iterations of MUSE-BB and PBDAs. Again, a more comprehensive comparison with a larger set of test problems will be needed to determine which class of algorithms is best suited to different types of problems.

4.7. Conclusion

In this chapter, we presented MUSE-BB, a multisection B&B-based decomposition algorithm for the deterministic global optimization of general nonconvex nonlinear two-stage problems. We prove finite ε_f-convergence, show favorable convergence order of our lower bounding scheme, compared to existing algorithms, and provide initial computational results indicating good scalability of MUSE-BB with the number of scenarios.

Existing decomposition algorithms for two-stage nonconvex MINLP problems (Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b) have been classified as PBDAs (Robertson, Cheng, and Scott, 2020; Robertson, Cheng, and Scott, 2024), since they all employ spatial B&B in the first-stage variables. PBDAs achieve this by solving decomposable subproblems of both first- and second-stage variables in each node. To obtain good lower bounds, these subproblems are solved globally via a nested spatial B&B. Instead, we propose to branch on both first- and second-stage variables within a single B&B tree, and to further relax subproblems, avoiding duplicate branching on first-stage variables, and the nesting of spatial B&B procedures. We either branch normally on a single first-stage variable, or we simultaneously branch on multiple second-stage variables from different scenarios. While such multisection produces an exponential number of child nodes, the total number of distinct subproblems is linear in the number of bisected variables, by virtue of the decomposition. Thus, we only need to process the distinct subproblems and can generate child nodes by appropriately combining the subproblem results. To avoid an excessive number of child nodes with poor lower bounds, we only use a subset of bisections (reverting the remaining ones). We select the bisections based on their associated strong-branching scores, which are readily available after processing. This allows to only generate child nodes corresponding to the most promising bisections with highest strong-branching scores.

Our theoretical results show that by branching on all variables, the lower bounding scheme of MUSE-BB generally has a convergence order of one, if all functions are Lipschitz.

This is in contrast to lower bounding schemes of existing decomposition algorithms, which may have convergence orders below one, in general (Robertson, Cheng, and Scott, 2024). Whether or not this improved convergence order actually translates into an advantage with respect to the occurrence of clustering is however not clear at this point, and requires further investigation.

We perform initial computational experiments with a small test problem, which despite its size still incorporates relevant nonlinearities found in applications. Our results highlight the importance of choosing appropriate branching priorities for both general B&B and decomposition algorithms. Moreover, the results show that even for this small problem and small numbers of scenarios, MUSE-BB can significantly outperform the standard version of our open-source deterministic global solver MAiNGO, applied to the deterministic equivalent formulation. For the considered problem instances, the best wall times of MUSE-BB are achieved when essentially limiting the number of child nodes resulting from multisection to two. In this case, MUSE-BB behaves very similar to a B&B algorithm solving the deterministic equivalent using a strong-branching heuristic that employs certain range reduction with the rejected bisections. Naturally a comparison of MUSE-BB with such a heuristic constitutes promising future work.

5. Conclusion and Outlook

5.1. Conclusion

Designing energy systems for operation in uncertain or variable conditions requires accurate modeling, tractable problem formulations, and efficient solution algorithms of the resulting optimization problems. This thesis addresses open challenges and provides software for all of these areas.

In Chapter 2, we motivated and presented a new framework for structured energy system modeling called COMANDO. By combining desirable features from algebraic modeling languages (AMLs) and differential-algebraic modeling frameworks, COMANDO offers advanced modeling capabilities, not available in existing energy system modeling frameworks. The implementation of COMANDO as an open-source Python package allows users to incorporate customized routines for model creation, problem formulation and transformation, and solution. Furthermore the existing set of interfaces to AMLs and solvers can be easily extended by domain experts to interface with additional software. We demonstrated the broad applicability of COMANDO via four case studies: the greenfield design and operation of an industrial energy system, demand response of a building energy system, waste-heat integration into a district heating network, and globally optimal operation of an organic Rankine cycle (ORC).

In Chapter 3, we showcased how COMANDO can be applied to create detailed component and system models for an air-cooled geothermal ORC. Due to the variability of ambient temperature, the ORC design needs to account for a wide range of operating conditions. We build a detailed model of the overall process by aggregating models for the pump, heat exchangers, turbine and air-cooled condenser. The component models incorporate correlations for accurate prediction of operational behavior and the resulting costs, which account for changes in design quantities, as well as varying ambient conditions. We formulate an optimization problem of the form TSP, with the aim of maximizing expected total annualized revenue by simultaneously optimizing the design and operation under the consideration of multiple ambient temperatures. Whereas previous works frequently treat the design and operation of ORCs sequentially, consider only on individual system components in detail, or use nondeterministic approaches for optimization, we demonstrate that the problem we formulate can be optimized globally despite the high level of detail. We show that systems designed for individual operational scenarios are generally infeasible, but also, that the results of such scenario-analysis provide guarantees on the quality of the solution to the overall problem, where multiple operating conditions are considered simultaneously.

In Chapter 4, we propose a new decomposition algorithm termed MUSE-BB, which addresses recent observations of potential issues with state-of-the-art decomposition algorithms for the general, nonconvex form of TSP. While these algorithms aim to achieve scalability by projecting the original problem into the space of first-stage variables (corre-

sponding to design decisions in our context), this approach may suffer from both practical and theoretical drawbacks. Instead, we propose an alternative partitioning procedure that considers both stages (i.e., design and operation) explicitly. To do this efficiently, we propose to perform multisection, i.e., simultaneous branching, on all scenario instances of a particular operational variable. We show that the convergence order of the resulting lower bounding scheme is generally higher than in existing methods, which indicates that MUSE-BB may be less likely to suffer from clustering. Numerical results show that even for a small test problem, using MUSE-BB is generally faster than solving the deterministic equivalent using the standard version of our state-of-the-art B&B solver MAiNGO, both in CPU as well as wall time. Furthermore, the constant size of subproblems and parallel nature of MUSE-BB resulted in more significant improvements with an increasing number of considered scenarios. While further experiments with larger problems are warranted, a similar trend we expect a similar trend as computational complexity increases superlinearly with problem size.

The contributions of this thesis improve several parts of the workflow for modeling and optimizing the design and operation of of energy systems under uncertainty. As we have shown, some realistic energy system design and operation problems can already be addressed with the tools presented in this thesis, however, their use can require a significant level of expertise. Further research to sharpen our understanding of good modeling practices and the impact of algorithmic parameters may help guide users, or even allow for automation of certain parts of the workflow, thus lowering the barrier to entry.

5.2. Outlook

In the following, we propose directions for future work. In particular, we suggest extensions and improvements to COMANDO, emphasize the need to better understand how different modeling choices affect computational tractability, suggest several enhancements for MUSE-BB, and highlight the particular importance of branching priorities in design and operation problems. Finally we point out the need for further research on the implications of low convergence orders in projection-based decomposition algorithms (PBDAs), and outline, how some of the other potential issues with existing PBDAs could be addressed.

Via its MAiNGO interface, COMANDO provides access to MUSE-BB, and thus, to a decomposition method for generic energy system design and operation problems of the form TSP. Furthermore, MAiNGO offers an MPI parallelization of its B&B algorithm. This tree-parallelism allows for distributed solution of deterministic equivalent problems and can even be employed in conjunction with the node-parallelism of MUSE-BB. Naturally, additional interfaces to other state-of-the-art methods such as those of Kannan, 2018; Cao and Zavala, 2019; Li and Grossmann, 2019b, or to existing software packages for MILP decomposition methods (Gamrath and Lübbecke, 2010; Kim and Zavala, 2017) or parallelized solution of the deterministic equivalent (Munguia, Oxberry, and Rajan, 2016) would be desirable. Being able to use multiple solution approaches from COMANDO would enable computational comparisons for different problem classes; in particular, offering insights into which approach is preferable in different settings, e.g., depending on the number of design and operational variables, and scenarios. To this end, the collection of relevant test problems from the literature seems desirable. While the current focus of COMANDO is on combined design and operation, its infrastructure is already capable of supporting al-

ternative uses, such as system analysis, or detailed operational optimization (e.g. Caspari et al., 2019), and can in principle be further extended to address multiperiod problems such as capacity expansion, or other approaches to long term planning (e.g. Baumgärtner et al., 2019b).

Modeling a complete energy system such as the ORC from Chapter 3 at a sufficient level of detail requires a lot of attention to ensure the resulting optimization problems remain tractable. While this issue is evidently not exclusive to optimal design and operation under uncertainty, the latter setting is especially susceptible to problem tractability. Apart from the possibility of applying dedicated solution algorithms like MUSE-BB to improve tractability, it will thus also be crucial to further develop modeling practices that explicitly take into account the impact of modeling choices on tractability. In particular, component-oriented modeling approaches may deteriorate tractability compared to monolithic approaches when the coupling of component models introduces many additional variables and constraints. One approach we already employed to address this in Chapter 3 is to combine component-orient modeling with reduced-space formulations. This hides intermediate expressions from the optimizer instead of introducing additional variables that need to be accounted for by the optimizer. Previous work highlighted the benefits of reduced-, compared to full-space formulations, including the reduced dimensionality of the space that needs to be partitioned in B&B search, a reduced size of bounding subproblems, a reduced number of bound tightening subproblems, effects resembling constraint propagation (Bongartz and Mitsos, 2017). Nevertheless, reduced-space formulations commonly result in reoccurring nonlinear subexpressions which may result in weaker relaxations compared to full-space methods (Tsoukalas and Mitsos, 2014; Najman, Bongartz, and Mitsos, 2021). In particular, this issue may arise when using ANNs for the prediction of fluid properties and process states (Huster, Schweidtmann, and Mitsos, 2019; Schweidtmann et al., 2019), as done in Chapter 3: The predicted quantities are expressed as nonlinear functions of the ANN inputs, and are used for the calculation of several other quantities. This can be particularly critical when the quality of relaxations is not considered during training, as two ANNs with a given accuracy may result in vastly different relaxations. Naturally, similar issues also exist with more classical data-driven models, such as, e.g., polynomial fits. To this end, data driven models for the use in global optimization of energy systems need to explicitly account for relaxation quality. While, the introduction of auxiliary variables allows for a trade-off between the relaxation quality and problem size, this process can be extremely time-consuming and error prone. While the introduction of auxiliary variables can be automated to some extent, much more research is needed to identify clear guidelines for this process.

Several improvements and generalizations are also conceivable for MUSE-BB. As with existing decomposition algorithms, our lower bounding scheme may be improved by dualizing the coupling (nonanticipativity) constraints instead of dropping them. As we showed in Chapter 4, the resulting lower bounding scheme would have second-order convergence at minimizers satisfying certain regularity conditions. However, this extension requires optimal dual multipliers, which are expensive to compute in general. The details of when such an extension is beneficial and how it can be implemented efficiently still need to be clarified. Furthermore, it may be interesting to generalize the current implementation to problems with different numbers of second-stage variables and constraints. A related extension would allow branching on arbitrary combinations of second-stage variables from different scenarios, instead of limiting multisection to scenario instances of a particular

second-stage variable. Of particular interest is the proper selection of branching priorities in more general cases. Our computational results confirm our intuition that first-stage priorities become more important with an increase in the number of scenarios. However, further research is needed to clarify how the branching priority ratio between first- and second-stage variables generalizes to more realistic problems and more generally, how branching priorities should be chosen in design and operation problems. Finally, the decomposable bounding routines of MUSE-BB may also enable efficient strong-branching in problems that do not fall into the category of two-stage programming problems, but still exhibit block structures, coupled by complicating constraints.

An important topic that warrants further investigation is the effectiveness of PBDAs. Similar to the use of reduced-space formulations in modeling, their use of projection significantly reduces the domain to be searched. This is advantageous for scalability but may potentially incur a reduced convergence order of the lower bounding scheme (Robertson, Cheng, and Scott, 2020). While current research does not yet permit a clear answer on the practical implications of this reduced convergence order, one concern is that it may result in clustering (Du and Kearfott, 1994; Wechsung, Schaber, and Barton, 2014; Kannan and Barton, 2017b). Even if this concern turns out to be unfounded, the duplicate consideration of first-stage variables in the inner and outer algorithms of current PBDAs seems wasteful. It is conceivable for a PBDA to globally solve the scenario subproblems in a procedural manner instead of repeating this process for each node of an outer B&B tree. This is achievable by maintaining separate B&B trees for each scenario subproblem while using the respective least lower bounds for bounding. Such an algorithm would still enjoy the benefits of projection, while avoiding the duplicate search in the first-stage variable domain and the nesting of two B&B methods.

Appendix A.

Details on geothermal ORC design and operation

A.1. Training of ANNs

For this work, we trained ANN surrogate models for the three quantities:

1. $r_h(\Delta h_{\mathrm{rel}})$: off-design turbine efficiency reduction due to enthalpy drop according to Ghasemi et al., 2013b

2. $r_{\dot{V}}(\dot{V}_{\mathrm{rel}})$: off-design turbine efficiency reduction due to volume flow rate Ghasemi et al., 2013b[1]

3. $F^{-1}_{T,\mathrm{DES}}(e_{\mathrm{h}}, e_{\mathrm{c}})$: inverse of mean temperature correction factor for desuperheater, based on the correlation of Schedwill, 1968 for a single row of finned tubes, also see Kuppan, 2013.

We provide a comparison of training approaches using Keras (Chollet et al., 2015) and nonlinear least squares regression. The ANNs used for the computational study were trained using the latter approach.

A.1.1. Data Generation

For the training of both $r_h(\Delta h_{\mathrm{rel}})$, and $r_{\dot{V}}(\dot{V}_{\mathrm{rel}})$, we used 1000 equidistant data points in the domain [0.2, 1.2]. For $F^{-1}_{T,\mathrm{DES}}$, we filter from a grid of 45×45 equidistant data points in the domain $[0, 1]^2$ all points that had a not-a-number (NaN) value or that lie above the line defining the 90% limit, i.e., points $(e_{\mathrm{h}}, e_{\mathrm{c}})$ for which

$$e_{\mathrm{h}} > 0.9\, e_{\mathrm{c}}\ \mathrm{W}\left(-\frac{0.9\exp(-0.9/e_{\mathrm{c}})}{e_{\mathrm{c}}}\right), \tag{A.1}$$

where W is the principal solution of the Lambert W function, i.e., $x = \mathrm{W}(x)\, e^{\mathrm{W}(x)}$. Note that Eq. (A.1) corresponds to the inverse of Eq. (3.66) in Section 3.1.4.2. We filtered using Eq. (A.1) instead of Eq. (3.66), as the former is numerically more stable due to the vertical slope of the limit for $e_{\mathrm{h}} \to 0.9$. Along the limiting line we added an additional 100 data points for a total of 1124 used data points to ensure a good resolution of this region of

[1] Note that the journal publication is missing a 0 in one of the coefficients, the correct correlation can be found in the preprint (Ghasemi et al., 2013a)

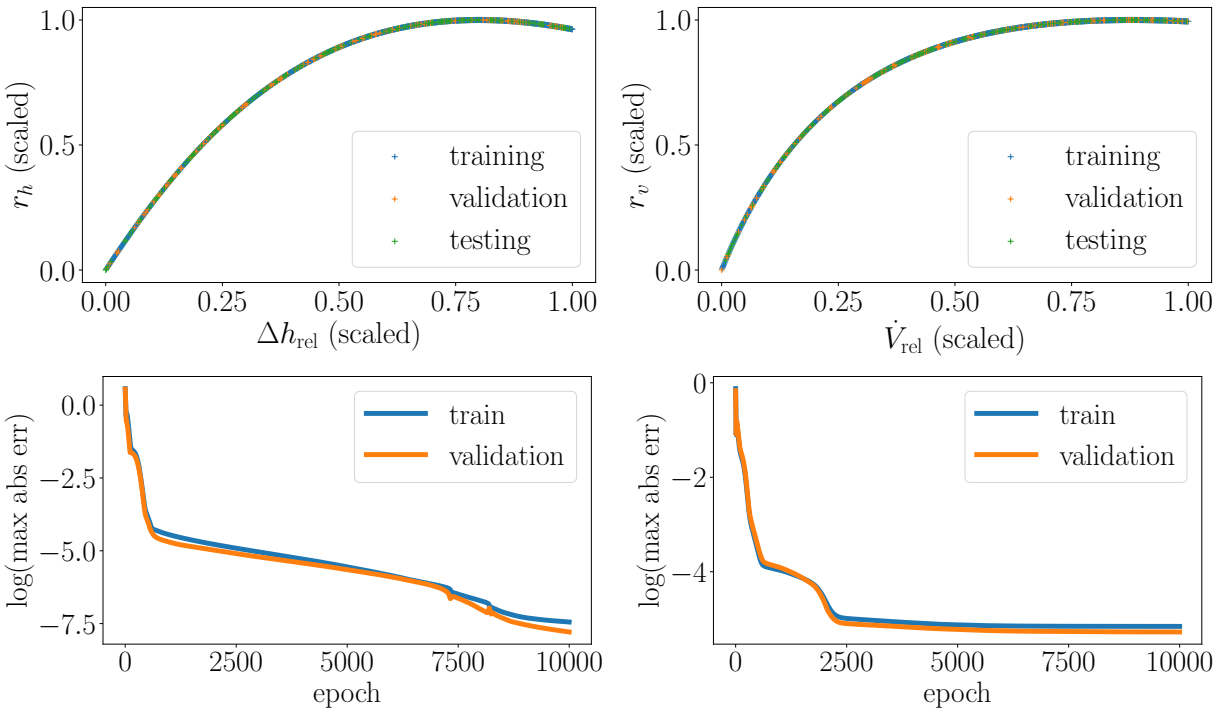


Fig. A.1. Top: Training, validation and test data points used for the quantities $r_h(\Delta h_{\mathrm{rel}})$ and $r_{\dot{V}}(\dot{V}_{\mathrm{rel}})$. Bottom: Training history for training Keras models with MSE as loss function using Adam over 10000 epochs.

high slopes. In all cases, the resulting data sets are shuffled and split into 70% training, 15% validation and 15% test points. The resulting data points after normalizing inputs and outputs to the domain [0, 1] are shown in the top part of Fig. A.1, and in the left part of Fig. A.2, respectively.

A.1.2. Training with Keras

After preliminary tests, we decided to use a single hidden layer with two tanh-activated neurons for both $r_h(\Delta h_{\mathrm{rel}})$, and $r_{\dot{V}}(\dot{V}_{\mathrm{rel}})$, and four tanh-activated neurons for $F^{-1}_{T,\mathrm{DES}}$. For comparison with regression training we used sequential Keras models with mean squared error as loss function, trained using the Adam optimizer (Kingma and Ba, 2014). For each quantity, we trained ten ANNs. The training histories for the best performing ANNs (i.e., the ones with the lowest maximum absolute errors over the test data sets) are shown in the bottom of Fig. A.1, and the right of Fig. A.2, respectively.

A.1.3. Training via Nonlinear Regression

The training of ANNs can also be formulated as the nonlinear regression problem

$$\min_{\boldsymbol{w},\boldsymbol{b}} \sum_{x \in \mathcal{X}_{\mathrm{train}}} \left(f(x) - ANN_f(\boldsymbol{w}, \boldsymbol{b}, x)\right)^2 / |\mathcal{X}_{\mathrm{train}}| \tag{A.2}$$

where $\boldsymbol{w}$ and $\boldsymbol{b}$ are the vectors of weights and biases, respectively, f is the function to be approximated, $\mathcal{X}_{\mathrm{train}}$ is the training data, and ANN_f is the ANN to be trained.

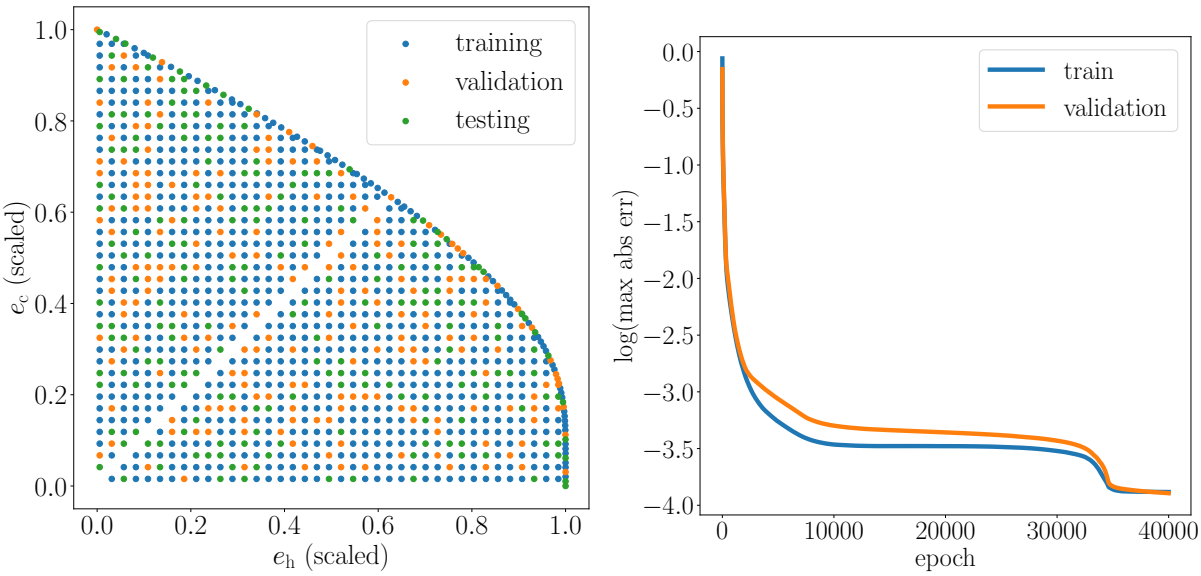


Fig. A.2. Left: Training, validation and test data points used for the quantity $F^{-1}_{T,\mathrm{DES}}(e_\mathrm{h}, e_\mathrm{c})$. Right: Training history for training Keras models with MSE as loss function using Adam over 40000 epochs.

We formulated the respective problems for each of the three quantities, and performed multiple local searches using IPOPT (Kawajir, Laird, and Wächter, 2011) via the multistart option of BARON (Sahinidis, 2020). For this we allowed the same time as was used for the training of the best-performing ANN using Keras. The results are given in Tab. A.1. It can be seen that the ANNs obtained from regression outperform the Keras training on average. Furthermore, on the test set regression-based ANNs outperform the best Keras-based ANNs for $r_{\dot{V}}$ in terms of MSE, and those for $F_{T,\mathrm{DES}}$ in terms of both MSE and MAE.

$\boldsymbol{r_h}$	MSE train	MSE validation	MSE test	MAE train	MAE validation	MAE test
Keras mean	2.565e-05	2.452e-05	2.658e-05	0.01549	0.0136	0.01501
Keras best	6.875e-08	5.957e-08	6.518e-08	0.0006913	0.0005441	0.0006532
regression	7.389e-08	5.301e-08	7.77e-08	0.001394	0.001122	0.001323
$\boldsymbol{r_{\dot{V}}}$	MSE train	MSE validation	MSE test	MAE train	MAE validation	MAE test
Keras mean	1.968e-05	2.318e-05	1.999e-05	0.01802	0.01833	0.01637
Keras best	5.171e-06	5.19e-06	4.706e-06	0.007552	0.006692	0.007442
regression	4.439e-06	4.537e-06	4.13e-06	0.008073	0.007215	0.007964
$\boldsymbol{F^{-1}_{T,\mathrm{DES}}}$	MSE train	MSE validation	MSE test	MAE train	MAE validation	MAE test
Keras mean	0.0001038	0.0001236	0.0001413	0.04251	0.04568	0.04523
Keras best	5.009e-05	5.937e-05	5.767e-05	0.02809	0.02594	0.02587
regression	2.511e-05	1.993e-05	2.243e-05	0.01607	0.0153	0.01559

Tab. A.1. Comparison of the mean squared error (MSE), and maximum absolute error (MAE) over test, validation, and training data resulting from the ANNs obtained via training with Keras, and the regression approach, for the three considered quantities.

A.2. Resulting Processes

The diagrams of temperature over specific entropy for the average-temperature design (ATD) and the multiple-temperature design (MTD) are shown in Fig. A.3. The ATD is optimized considering only the average ambient temperature. The resulting design is optimized for this ambient temperature to such a degree that it constrains operation in other ambient temperatures to use the same operational states for the working fluid. For temperatures above approximately 17.49 °C this operational rigidity renders the problem infeasible under the original assumptions. Allowing for a partial use of the available brine mass flow and tolerating a violation of the assumed limit of 20 m/s for the shell velocity of the superheater (maximum velocity about 20.1 m/s), operation for ambient temperatures up to about 22.4 °C becomes possible. However, higher temperatures remain infeasible as the ACC fans already operate at their peak load. For the MTD such an issue does not occur: The optimization considers operation at 11 ambient temperatures from −10–40 °C and allows for the working fluid states to be adjusted to the ambient temperature.

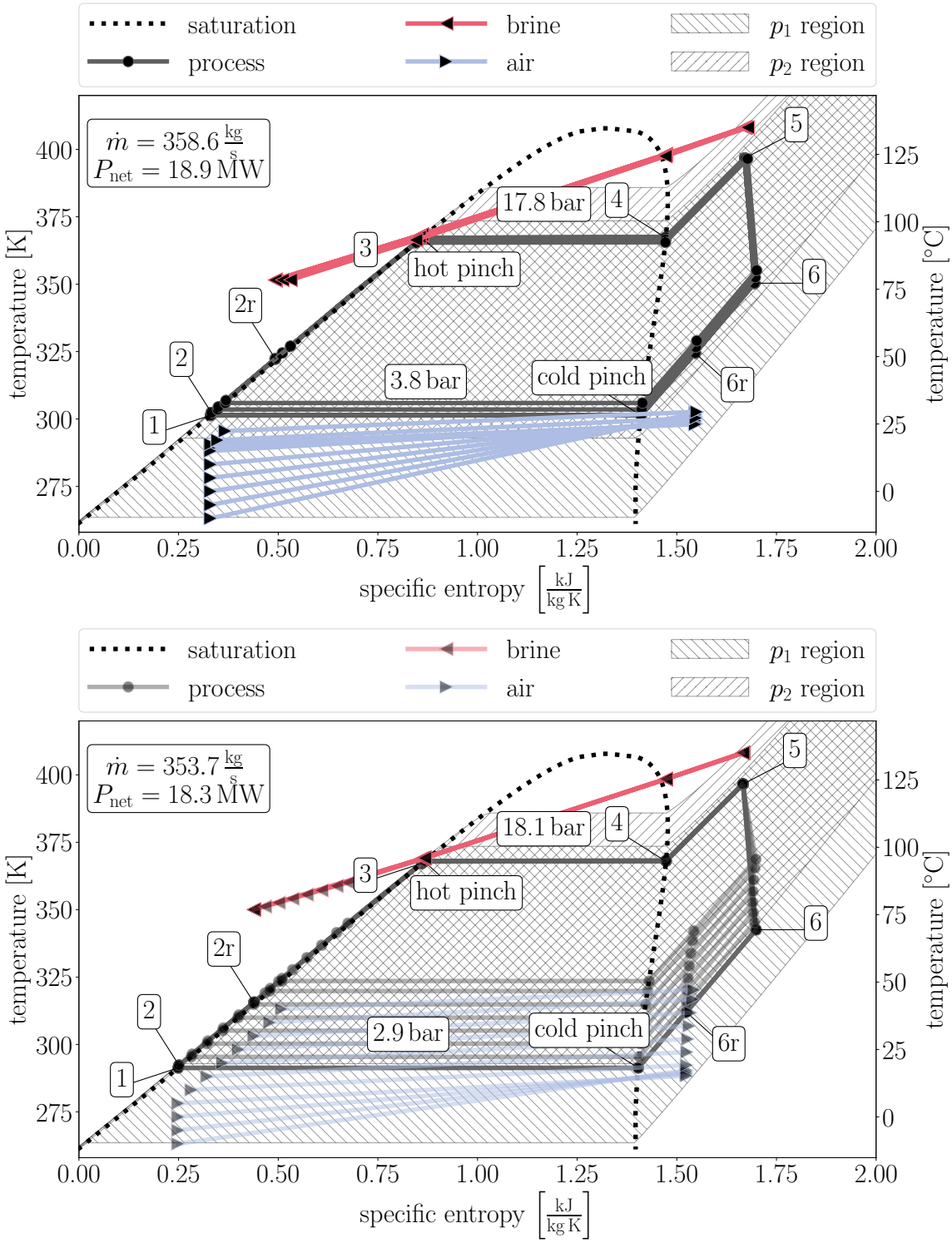


Fig. A.3. Temperature-entropy diagrams for operation at feasible ambient temperatures for the average-temperature design (ATD, top), and for all considered ambient temperatures for the multiple-temperature design (MTD, bottom). In the ATD case, process states are identical for all ambient temperatures while in the MTD case, operation can be adjusted to the ambient temperature. The values for $\dot{m}$, P_{net}, pressures and state labels shown correspond to the average and the lowest ambient temperature (15.85 °C and −10 °C) for the ATD and MTD case, respectively.

Appendix B.

Test Problem: CHP Sizing

We consider the design of a combined heat and power (CHP) unit, i.e., an equipment sizing problem whose aim is to satisfy given heat and power demands at minimum cost, see Fig. B.1. The size of the CHP is expressed as a nominal heat output $\dot{Q}_{\text{nom}}$, which

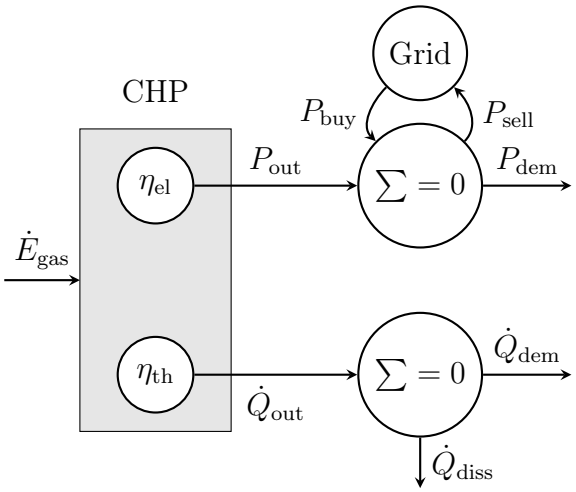


Fig. B.1. Conceptual CHP operation

corresponds to the maximum thermal output $\dot{Q}_{\text{out}}$. The actual output at any given point is determined by a relative heat output $\dot{Q}_{\text{rel}}$:

$$\dot{Q}_{\text{out}} \coloneqq \dot{Q}_{\text{nom}}\, \dot{Q}_{\text{rel}} \tag{B.1}$$

The energy input to the CHP in terms of lower heating value of natural gas, $\dot{E}_{\text{gas}}$, can be calculated via the thermal efficiency η_{th}, which is a function of $\dot{Q}_{\text{nom}}$ and $\dot{Q}_{\text{rel}}$:

$$\dot{E}_{\text{gas}} \coloneqq \frac{\dot{Q}_{\text{out}}}{\eta_{\text{th}}(\dot{Q}_{\text{nom}}, \dot{Q}_{\text{rel}})} \tag{B.2}$$

Following this the power output P_{out} can be computed via the electrical efficiency η_{th}, which is also a function of $\dot{Q}_{\text{nom}}$ and $\dot{Q}_{\text{rel}}$:

$$P_{\text{out}} \coloneqq \dot{E}_{\text{gas}}\, \eta_{\text{el}}(\dot{Q}_{\text{nom}}, \dot{Q}_{\text{rel}}) \tag{B.3}$$

The functional form of the efficiencies η_{th} and η_{el} is given by:

$$\eta_{\text{th}}(\dot{Q}_{\text{nom}}, \dot{Q}_{\text{rel}}) \coloneqq \eta_{\text{th,nom}}(\dot{Q}_{\text{nom}})\, \eta_{\text{th,rel}}(\dot{Q}_{\text{rel}}) \tag{B.4}$$

$$\eta_{\text{th,nom}}(\dot{Q}_{\text{nom}}) \coloneqq 0.498 - \frac{\dot{Q}_{\text{nom}}}{21.17\,\text{MW}} \tag{B.5}$$

$$\eta_{\text{th,rel}}(\dot{Q}_{\text{rel}}) \coloneqq 1.10 - 0.0768\,(\dot{Q}_{\text{rel}} + 0.130)^2 \tag{B.6}$$

$$\eta_{\text{el}}(\dot{Q}_{\text{nom}}\, \dot{Q}_{\text{rel}}) \coloneqq \eta_{\text{el,nom}}(\dot{Q}_{\text{nom}})\, \eta_{\text{el,rel}}(\dot{Q}_{\text{rel}}) \tag{B.7}$$

$$\eta_{\text{el,nom}}(\dot{Q}_{\text{nom}}) \coloneqq 0.372 + \frac{\dot{Q}_{\text{nom}}}{21.17\,\text{MW}} \tag{B.8}$$

$$\eta_{\text{el,rel}}(\dot{Q}_{\text{rel}}) \coloneqq 1.02 - 0.435\,(0.774\,\dot{Q}_{\text{rel}} - 1)^2 \tag{B.9}$$

A heat shortage, defined as

$$\dot{Q}_{\text{short}} \coloneqq \dot{Q}_{\text{dem}} - \dot{Q}_{\text{out}} \tag{B.10}$$

must be avoided (i.e., $\dot{Q}_{\text{short}}$ must be negative). Correspondingly, the power shortage can be defined as:

$$P_{\text{short}} \coloneqq P_{\text{dem}} - P_{\text{out}} \tag{B.11}$$

A power shortage can be addressed by purchasing power from the grid, i.e.:

$$P_{\text{buy}} \coloneqq \max(0, P_{\text{short}}) \tag{B.12}$$

If P_{short} or $\dot{Q}_{\text{short}}$ are negative, the excess power can be sold to the grid at a reduced price, while the excess heat can be dissipated into the environment:

$$\dot{P}_{\text{sell}} \coloneqq \max(0, -P_{\text{short}}) \tag{B.13}$$

$$\dot{Q}_{\text{diss}} \coloneqq \max(0, -\dot{Q}_{\text{short}}). \tag{B.14}$$

With these definitions we can formulate a reduced-space problem that contains the nominal heat output as the only first-stage variable, i.e: $\boldsymbol{x} = (\dot{Q}_{\text{nom}}) \in [1.4\,\text{MW}, 2.3\,\text{MW}]$, and the part-load in each scenario as the only second-stage variable, i.e: $\boldsymbol{y}_s = (\dot{Q}_{\text{rel},s}) \in [0, 1]$.

We choose total annualized costs (TAC) in million € as the objective function. The first-stage objective function describes the annualized investment costs (according to an economy of scales approach) and the second-stage objectives correspond to the annual operating costs in each scenario:

$$f_{\text{I}}(\boldsymbol{x}) = 149\,567\,€/\text{a} \left(\frac{\dot{Q}_{\text{nom}}}{1\,\text{MW}} \right)^{0.9} \times 10^{-6} \tag{B.15}$$

$$\begin{aligned} f_{\text{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) = T_{\text{op}}(\quad & p_{\text{gas}}\, \dot{E}_{\text{gas},s} \\ & + p_{\text{el,buy}}\, P_{\text{buy},s} \\ & - p_{\text{el,sell}}\, P_{\text{sell},s}) \times 10^{-6} \end{aligned} \tag{B.16}$$

Where $T_{\text{op}} = 6\,000\,\text{h/a}$, $p_{\text{gas}} = 80\,€/(\text{MW h})$, $p_{\text{el,buy}} = 250\,€/(\text{MW h})$, $p_{\text{el,sell}} = 100\,€/(\text{MW h})$

We approximate the requirement that the CHP unit must either be inactive or operate

above a minimal part-load threshold of 50% with quadratic second-stage constraints of the form

$$0.0619263 - (\dot{Q}_{\mathrm{rel},s} - 0.25115)^2 \leq 0 \tag{B.17}$$

which restrict the relative outputs $\dot{Q}_{\mathrm{rel},s}$ to less than 0.1%, or more than 50% part-load. An additional constraint is that

$$\dot{Q}_{\mathrm{short},s} \leq 0, \tag{B.18}$$

also see Eq. (B.10). Note that Eq. (B.17) implies that heat demands corresponding to part-loads between 0.1% and 50% cannot be satisfied. To ensure the considered instances have a feasible solution, the randomly generated heat demands are set to 0 if they fall into this range. Similarly, the generated power and heat demands are capped to the highest possible production.

Appendix C.

Effect of effective partition limit $k_{\max}$ and strong-branching threshold τ

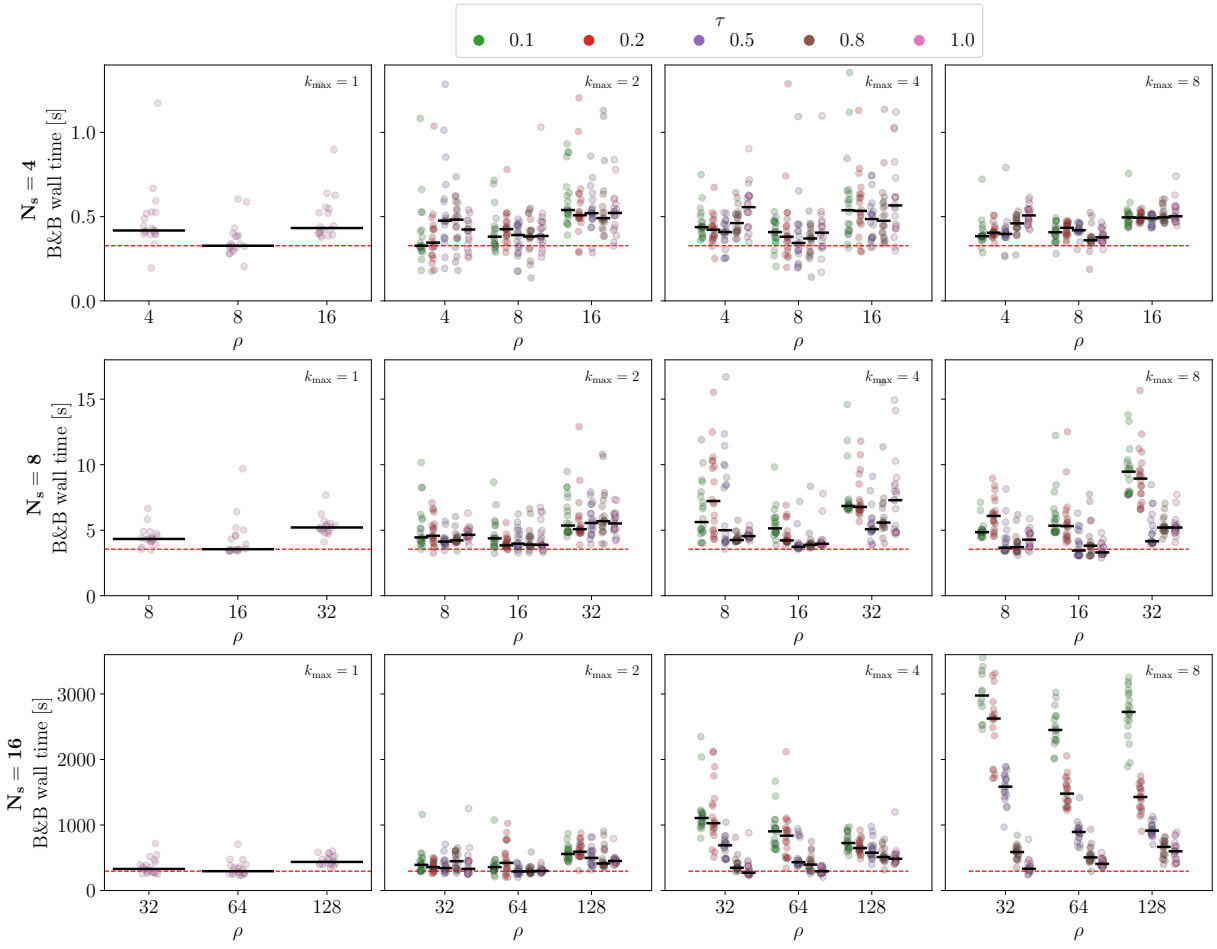


Fig. C.1. Variation of solution times for solving the CHP sizing problem using MUSE-BB with $k_{\max} \in \{1, 2, 4, 8\}$, and $\tau \in \{0.1, 0.2, 0.5, 0.8, 1\}$ for the three ρ values resulting in the lowest median wall times for $N_s = 4, 8$, and 16, with $k_{\max} = 1$. For each parameter combination, a set of 20 runs is performed. For $(k_{\max}, \rho, \tau) = (8, 32, 0.1)$, $(8, 32, 0.2)$, and $(8, 64, 0.1)$, 8, 2, and 3 runs timed out after 3600 s, respectively. Medians with respect to the remaining runs are depicted as horizontal black lines and the lowest median time for $k_{\max} = 1$ is depicted as a dashed red line for reference. Whereas for $N_s = 4$, and 8, no clear trend is discernible, for $N_s = 16$, increasing $k_{\max}$ generally results in larger wall times, and for $k_{\max} = 4$, and 8, increasing τ results in smaller wall times.

Bibliography

Achterberg (2007). "Constraint Integer Programming". PhD thesis. TU Berlin.

Achterberg, Koch, and Martin (2005). "Branching rules revisited". In: *Oper. Res. Lett.* 33.1, pp. 42–54. DOI: 10.1016/j.orl.2004.04.002.

Adjiman and Floudas (2008). "alphaBB Algorithm". In: *Encyclopedia of Optimization*. New York: Springer, pp. 61–73. DOI: 10.1007/978-0-387-74759-0_11.

Ahmad, Linnhoff, and Smith (1988). "Design of Multipass Heat Exchangers: An Alternative Approach". In: *J. Heat Transfer* 110.2, pp. 304–309. DOI: 10.1115/1.3250484.

Aien, Hajebrahimi, and Fotuhi-Firuzabad (2016). "A comprehensive review on uncertainty modeling techniques in power system studies". In: *Renew. Sustain. Energy Rev.* 57, pp. 1077–1089. DOI: 10.1016/j.rser.2015.12.070.

Åkesson et al. (2010). "Modeling and optimization with Optimica and JModelica.org—Languages and tools for solving large-scale dynamic optimization problems". In: *Comput. Chem. Eng.* 34.11, pp. 1737–1749. DOI: 10.1016/j.compchemeng.2009.11.011.

Andiappan (2017). "State-Of-The-Art Review of Mathematical Optimisation Approaches for Synthesis of Energy Systems". In: *Process Integr. Optim. Sustain.* 1.3, pp. 165–188. DOI: 10.1007/s41660-017-0013-2.

Androulakis, Maranas, and Floudas (1995). "alphaBB: A global optimization method for general constrained nonconvex problems". In: *J. Glob. Optim.* 7.4, pp. 337–363. DOI: 10.1007/bf01099647.

Angulo, Ahmed, and Dey (2016). "Improving the integer L-shaped method". In: *INFORMS J. Comput.* 28.3, pp. 483–499. DOI: https://doi.org/10.1287/ijoc.2016.0695.

Anstreicher (2001). "Linear Programming: Interior Point Methods". In: *Encyclopedia of Optimization*. Springer US, pp. 1279–1281. DOI: 10.1007/0-306-48332-7_262.

Applegate et al. (1995). *Finding Cuts in the TSP (A Preliminary Report)*. Tech. rep. AT&T Bell Labs.

Astolfi et al. (2014a). "Binary ORC (organic Rankine cycles) power plants for the exploitation of medium–low temperature geothermal sources – Part A: Thermodynamic optimization". In: *Energy* 66, pp. 423–434. DOI: 10.1016/j.energy.2013.11.056.

Astolfi et al. (2014b). "Binary ORC (Organic Rankine Cycles) power plants for the exploitation of medium–low temperature geothermal sources – Part B: Techno-economic optimization". In: *Energy* 66, pp. 435–446. DOI: 10.1016/j.energy.2013.11.057.

Astolfi et al. (2017). "Application of the Novel "Emeritus" Air Cooled Condenser in Geothermal ORC". in: *Energy Procedia* 129, pp. 479–486. DOI: 10.1016/j.egypro.2017.09.164.

Atabay (2017). "An open-source model for optimal design and operation of industrial energy systems". In: *Energy* 121, pp. 803–821. DOI: 10.1016/j.energy.2017.01.030.

Bahl et al. (2018). "Typical Periods for Two-Stage Synthesis by Time-Series Aggregation with Bounded Error in Objective Function". In: *Front. Energy Res.* 5, p. 35. DOI: 10.3389/fenrg.2017.00035.

Bakken, Skjelbred, and Wolfgang (2007). "eTransport: Investment planning in energy supply systems with multiple energy carriers". In: *Energy* 32.9, pp. 1676–1689. DOI: 10.1016/j.energy.2007.01.003.

Bansal et al. (2000). "Simultaneous design and control optimisation under uncertainty". In: *Comput. Chem. Eng.* 24.2-7, pp. 261–266. DOI: 10.1016/s0098-1354(00)00475-0.

Baumgärtner et al. (2019a). "RiSES3: Rigorous Synthesis of Energy Supply and Storage Systems via time-series relaxation and aggregation". In: *Comput. Chem. Eng.* 127, pp. 127–139. DOI: 10.1016/j.compchemeng.2019.02.006.

Baumgärtner et al. (2019b). "RiSES4: Rigorous Synthesis of Energy Supply Systems with Seasonal Storage by relaxation and time-series aggregation to typical periods". In: *Proc. ECOS 2019* (June 23–28, 2019). Wrocław, Poland, pp. 263–274.

Bundesinstitut für Bau-, Stadt- und Raumforschung (BBSR) (2014). *Kosten energierelevanter Bau- und technischer Anlagenteile bei der energetischen Sanierung von Nichtwohngebäuden/ Bundesliegenschaften.* URL: https://www.bbsr.bund.de/BBSR/DE/veroeffentlichungen/bbsr-online/2014/ON062014.html (visited on July 28, 2024).

Bell et al. (2014). "Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp". In: *Ind. Eng. Chem. Res.* 53.6, pp. 2498–2508. DOI: 10.1021/ie4033999.

Beller (1976). *Reference energy system methodology.* Tech. rep. Brookhaven National Lab., Upton, NY (USA).

Bellman and Zadeh (1970). "Decision-Making in a Fuzzy Environment". In: *Manage. Sci.* 17.4, B-141-B–164. DOI: 10.1287/mnsc.17.4.b141.

Belotti (2019). *Couenne: a user's manual.* URL: https://www.coin-or.org/Couenne/couenne-user-manual.pdf (visited on May 13, 2024).

Ben-Tal and Nemirovski (2002). "Robust optimization - methodology and applications". In: *Math. Programm.* 92.3, pp. 453–480. DOI: 10.1007/s101070100286.

Benders (1962). "Partitioning procedures for solving mixed-variables programming problems". In: *Numer. Math.* 4.1, pp. 238–252. DOI: 10.1007/BF01386316.

Beuzekom, Gibescu, and Slootweg (2015). "A review of multi-energy system planning and optimization tools for sustainable urban development". In: *IEEE PowerTech* (June 29–July 2, 2015). Eindhoven, The Netherlands: IEEE, pp. 1–7. DOI: 10.1109/ptc.2015.7232360.

Biel and Johansson (2022). "Efficient Stochastic Programming in Julia". In: *INFORMS J. Comput.* 34.4, pp. 1885–1902. DOI: 10.1287/ijoc.2022.1158.

Birge (1985). "Decomposition and partitioning methods for multistage stochastic linear programs". In: *Oper. Res.* 33.5, pp. 989–1007.

Birge and Louveaux (2011). *Introduction to Stochastic Programming.* New York: Springer. DOI: 10.1007/978-1-4614-0237-4.

Bisschop (2006). *AIMMS optimization modeling.* Lulu.com.

Bompadre and Mitsos (2011). "Convergence rate of McCormick relaxations". In: *J. Glob. Optim.* 52.1, pp. 1–28. DOI: 10.1007/s10898-011-9685-2.

Bongartz and Mitsos (2017). "Deterministic global optimization of process flowsheets in a reduced space using McCormick relaxations". In: *J. Glob. Optim.* 69.4, pp. 761–796. DOI: 10.1007/s10898-017-0547-4.

Bongartz et al. (2018). *MAiNGO: McCormick based Algorithm for mixed integer Nonlinear Global Optimization*. Tech. rep. http://permalink.avt.rwth-aachen.de/?id=729717. The open-source version is available at https://git.rwth-aachen.de/avt-svt/public/maingo. The corresponding Python package `maingopy` is available at https://pypi.org/project/maingopy. Process Systems Engineering (AVT.SVT), RWTH Aachen University. (Visited on July 28, 2024).

Borunda et al. (2016). "Bayesian networks in renewable energy systems: A bibliographical survey". In: *Renew. Sustain. Energy Rev.* 62, pp. 32–45. DOI: 10.1016/j.rser.2016.04.030.

Brown, Hörsch, and Schlachtberger (2018). "PyPSA: Python for Power System Analysis". In: *J. Open Res. Softw.* 6.1, p. 4. DOI: 10.5334/jors.188.

Bundesministerium für Verkehr, Bau und Stadtentwicklung (BMVBS) (2012). *Ermittlung von spezifischen Kosten energiesparender Bauteil-, Beleuchtungs-, Heizungs- und Klimatechnikausführungen bei Nichtwohngebäuden für die Wirtschaftlichkeitsuntersuchungen zur EnEV 2012*. URL: https://www.bbsr.bund.de/BBSR/DE/veroeffentlichungen/ministerien/bmvbs/bmvbs-online/2012/ON082012.html (visited on July 28, 2024).

Bussieck and Meeraus (2004). "General algebraic modeling system (GAMS)". in: *Modeling Languages in Mathematical Optimization*. Springer US, pp. 137–157. DOI: 10.1007/978-1-4613-0215-5_8.

Calise et al. (2014). "Thermoeconomic analysis and off-design performance of an organic Rankine cycle powered by medium-temperature heat sources". In: *Solar Energy* 103, pp. 595–609. DOI: 10.1016/j.solener.2013.09.031.

Cao, Song, and Khan (2019). "Convergence of Subtangent-Based Relaxations of Nonlinear Programs". In: *Processes* 7.4, p. 221. DOI: 10.3390/pr7040221.

Cao and Zavala (2019). "A scalable global optimization algorithm for stochastic nonlinear programs". In: *J. Glob. Optim.* 75.2, pp. 393–416. DOI: 10.1007/s10898-019-00769-y.

Capra and Martelli (2015). "Numerical Optimization of Combined Heat and Power Organic Rankine Cycles – Part B: Simultaneous Design & Part-load Optimization". In: *Energy* 90, pp. 329–343. DOI: 10.1016/j.energy.2015.06.113.

Carøe and Schultz (1999). "Dual decomposition in stochastic integer programming". In: *Oper. Res. Lett.* 24.1-2, pp. 37–45. DOI: 10.1016/S0167-6377(98)00050-9.

Casartelli et al. (2015). "Power Block Off-design Control Strategies for Indirect Solar ORC Cycles". In: *Energy Procedia* 69, pp. 1220–1230. DOI: 10.1016/j.egypro.2015.03.166.

Caspari et al. (2019). "DyOS - A Framework for Optimization of Large-Scale Differential Algebraic Equation Systems". In: *29th ESCAPE*. ed. by A. A. Kiss et al. Vol. 46. Comput. Aided Chem. Eng. Elsevier, pp. 619–624. DOI: 10.1016/b978-0-12-818634-3.50104-1.

Čertík et al. (2019). *symengine 0.4.0*. URL: https://github.com/symengine/symengine (visited on July 28, 2024).

Chachuat et al. (2015). "Set-Theoretic Approaches in Analysis, Estimation and Control of Nonlinear Systems". In: *IFAC-PapersOnLine* 48.8, pp. 981–995. DOI: 10.1016/j.ifacol.2015.09.097.

Charnes and Cooper (1959). "Chance-constrained programming". In: *Manage. Sci.* 6.1, pp. 73–79.

Chen et al. (2011). "Decomposition strategy for the global optimization of flexible energy polygeneration systems". In: *AIChE J.* 58.10, pp. 3080–3095. DOI: 10.1002/aic.13708.

Chollet et al. (2015). *Keras*. URL: https://keras.io (visited on July 28, 2024).

Frangopoulos, Von Spakovsky, and Sciubba (2002). "A Brief Review of Methods for the Design and Synthesis Optimization of Energy Systems". In: *Int. J. Thermodyn.* 5 (4), pp. 151–160.

Clarke (1990). *Optimization and Nonsmooth Analysis*. SIAM. DOI: 10.1137/1.9781611971309.

COMANDO Documentation (2021). URL: https://comando.readthedocs.io (visited on July 28, 2024).

COMANDO Repository (2021). URL: https://jugit.fz-juelich.de/iek-10/public/optimization/comando (visited on July 28, 2024).

Connolly et al. (2010). "A review of computer tools for analysing the integration of renewable energy into various energy systems". In: *Appl. Energy* 87.4, pp. 1059–1082. DOI: 10.1016/j.apenergy.2009.09.026.

Cooke (1984). "On Prediction of Off-Design Multistage Turbine Pressures by Stodola's Ellipse". In: *1984 Joint Power Gener. Conf.: GT Papers* (Sept. 30–Oct. 4, 1984). Toronto, Canada: ASME. DOI: 10.1115/84-jpgc-gt-14.

Csallner, Csendes, and Markót (2000). "Multisection in Interval Branch and Bound Methods for Global Optimization – I. Theoretical Results". In: *J. Glob. Optim.* 16.4, pp. 371–392. DOI: 10.1023/a:1008354711345.

Cuthrell and Biegler (1987). "On the optimization of differential-algebraic process systems". In: *AIChE J.* 33.8, pp. 1257–1270. DOI: 10.1002/aic.690330804.

Dantzig (1955). "Linear Programming under Uncertainty". In: *Manage. Sci.* 1.3-4, pp. 197–206.

Dantzig and Infanger (2010). "A Probabilistic Lower Bound for Two-Stage Stochastic Programs". In: *Stochastic Programming*. Springer New York, pp. 13–35. DOI: 10.1007/978-1-4419-1642-6_2.

Dantzig and Wolfe (1960). "Decomposition Principle for Linear Programs". In: *Oper. Res.* 8.1, pp. 101–111. DOI: 10.1287/opre.8.1.101.

Dembo (1991). "Scenario optimization". In: *Ann. Oper. Res.* 30.1, pp. 63–80. DOI: 10.1007/bf02204809.

Demirhan et al. (2019). "Energy systems engineering - a guided tour". In: *BMC Chem. Eng.* 1.1, p. 11. DOI: 10.1186/s42480-019-0009-5.

Dorfner (2016). "Open source modelling and optimisation of energy infrastructure at urban scale". PhD thesis. Technical University of Munich.

Du and Kearfott (1994). "The Cluster Problem in Multivariate Global Optimization". In: *J. Glob. Optim.* 5.3, pp. 253–265. DOI: 10.1007/bf01096455.

Dunning, Huchette, and Lubin (2017). "JuMP: A Modeling Language for Mathematical Optimization". In: *SIAM Review* 59.2, pp. 295–320. DOI: 10.1137/15m1020575.

Dür and Horst (1997). "Lagrange Duality and Partitioning Techniques in Nonconvex Global Optimization". In: *J. Optim. Theory Appl.* 95.2, pp. 347–369. DOI: 10.1023/a:1022687222060.

Elmqvist and Mattsson (1997). "MODELICA-the next generation modeling language-an international design effort". In: *Proc. 1st World Congr. Syst. Simul.* (Sept. 1–3, 1997). Singapore, pp. 1–5.

El-Emam and Dincer (2013). "Exergy and exergoeconomic analyses and optimization of geothermal organic Rankine cycle". In: *Appl. Therm. Eng.* 59.1-2, pp. 435–444. DOI: 10.1016/j.applthermaleng.2013.06.005.

Erdogan, Colpan, and Cakici (2017). "Thermal design and analysis of a shell and tube heat exchanger integrating a geothermal based organic Rankine cycle and parabolic trough solar collectors". In: *Renew. Energy* 109, pp. 372–391. DOI: 10.1016/j.renene.2017.03.037.

Ferris et al. (2009). "An extended mathematical programming framework". In: *Comput. Chem. Eng.* 33.12, pp. 1973–1982. DOI: 10.1016/j.compchemeng.2009.06.013.

Fiacco, ed. (1983). *Introduction to sensitivity and stability analysis in nonlinear programming*. Elsevier. DOI: 10.1016/s0076-5392(08)x6041-2.

Fishbone and Abilock (1981). "Markal, a linear-programming model for energy systems analysis: Technical description of the bnl version". In: *Int. J. Energy Res.* 5.4, pp. 353–375. DOI: 10.1002/er.4440050406.

Fisher (1981). "The Lagrangian Relaxation Method for Solving Integer Programming Problems". In: *Manage. Sci.* 27.1, pp. 1–18. DOI: 10.1287/mnsc.27.1.1.

Fourer, Gay, and Kernighan (1990). "A Modeling Language for Mathematical Programming". In: *Manage. Sci.* 36.5, pp. 519–554. DOI: 10.1287/mnsc.36.5.519.

Frangopoulos (2018). "Recent developments and trends in optimization of energy systems". In: *Energy* 164, pp. 1011–1020. DOI: 10.1016/j.energy.2018.08.218.

Friedman et al. (2013). "Block-oriented modeling of superstructure optimization problems". In: *Comput. Chem. Eng.* 57, pp. 10–23. DOI: 10.1016/j.compchemeng.2013.04.008.

Füllner, Kirst, and Stein (2020). "Convergent upper bounds in global minimization with nonlinear equality constraints". In: *Math. Programm.* 187.1–2, pp. 617–651. DOI: 10.1007/s10107-020-01493-2.

Gabrielli et al. (2018). "Optimal design of multi-energy systems with seasonal storage". In: *Appl. Energy* 219, pp. 408–424. DOI: 10.1016/j.apenergy.2017.07.142.

Gamrath and Lübbecke (2010). "Experiments with a generic Dantzig-Wolfe decomposition for integer programs". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 239–252. DOI: 10.1007/978-3-642-13193-6_21.

Ganguli, Tung, and Taborek (1985). "Parametric Study of Air-cooled Heat Exchanger Finned Tube Geometry". In: *AIChE symp. ser.* (Aug. 4, 1985). Vol. 81. 245. Denver, CO, pp. 122–128.

Geoffrion (1970a). "Elements of Large Scale Mathematical Programming Part II: Synthesis of Algorithms and Bibliography". In: *Manage. Sci.* 16.11, pp. 676–691. DOI: 10.1287/mnsc.16.11.676.

Geoffrion (1970b). "Elements of Large-Scale Mathematical Programming Part I: Concepts". In: *Manage. Sci.* 16.11, pp. 652–675. DOI: 10.1287/mnsc.16.11.652.

Geoffrion (1972). "Generalized Benders Decomposition". In: *J. Optim. Theory Appl.* 10.4, pp. 237–260. DOI: 10.1007/BF00934810.

Geoffrion (2009). *Lagrangean Relaxation and Its Uses in Integer Programming*. Tech. rep. California University, pp. 243–281. DOI: 10.1007/978-3-540-68279-0_9.

Ghasemi et al. (2013a). "Modeling and optimization of a binary geothermal power plant". Preprint, available at http://hdl.handle.net/1721.1/104031. (Visited on July 28, 2024).

Ghasemi et al. (2013b). "Modeling and optimization of a binary geothermal power plant". In: *Energy* 50, pp. 412–428. DOI: 10.1016/j.energy.2012.10.039.

Ghasemi et al. (2013c). "Optimization of Binary Geothermal Power Systems". In: *23rd ESCAPE*. vol. 32. Comput. Aided Chem. Eng. Elsevier, pp. 391–396. DOI: 10.1016/b978-0-444-63234-0.50066-x.

Gilman, Lambert, and Lilienthal (2006). "Chapter 15: Micropower System Modeling with Homer". In: *Integration of Alternative Sources of Energy*. Vol. 1. 1. John Wiley & Sons New York, NY, USA, pp. 379–418. DOI: 10.1002/0471755621.ch15.

Ginchev, Torre, and Rocca (2009). "$\mathcal{C}^{k,1}$ Functions, Characterization, Taylor's Formula and Optimization: A Survey". In: *Real Anal. Exchange* 35.2, pp. 311–342.

Ginocchi, Ponci, and Monti (2021). "Sensitivity Analysis and Power Systems: Can We Bridge the Gap? A Review and a Guide to Getting Started". In: *Energies* 14.24, p. 8274. DOI: 10.3390/en14248274.

Gleixner et al. (2016). "Three enhancements for optimization-based bound tightening". In: *J. Glob. Optim.* 67.4, pp. 731–757. DOI: 10.1007/s10898-016-0450-4.

Glücker et al. (2022). "Incorporating AC Power Flow into the Multi-Energy System Optimization Framework COMANDO". in: *2022 Open Source Modelling and Simulation of Energy Systems (OSMSES)* (Apr. 4–5, 2022). IEEE. DOI: 10.1109/osmses54027.2022.9769138.

Gnielinski (1976). "New Equations for Heat and Mass Transfer in Turbulent Pipe and Channel Flow". In: *Int. Chem. Eng.* 16.2, pp. 359–368.

Gnielinski (1983). "Heat exchanger design handbook". In: *Heat exchanger design handbook*. Vol. 1. Hemisphere Washington, DC. chap. Forced Convection in Ducts, p. 5.

Goderbauer et al. (2016). "An adaptive discretization MINLP algorithm for optimal synthesis of decentralized energy supply systems". In: *Comput. Chem. Eng.* 95, pp. 38–48. DOI: 10.1016/j.compchemeng.2016.09.008.

Gómez-Aláez et al. (2017). "Off-design study of a waste heat recovery ORC module in gas pipelines recompression station". In: *Energy Procedia* 129, pp. 567–574. DOI: 10.1016/j.egypro.2017.09.205.

Grimstad and Andersson (2019). "ReLU networks as surrogate models in mixed-integer linear programs". In: *Comput. Chem. Eng.* 131, p. 106580. DOI: 10.1016/j.compchemeng.2019.106580.

Grossmann et al. (2016). "Recent Advances in Mathematical Programming Techniques for the Optimization of Process Systems under Uncertainty". In: *Comput. Chem. Eng.* 91, pp. 3–14. DOI: 10.1016/j.compchemeng.2016.03.002.

Guignard and Kim (1987). "Lagrangean decomposition: A model yielding stronger Lagrangean bounds". In: *Math. Programm.* 39.2, pp. 215–228. DOI: 10.1007/bf02592954.

Gurobi Optimization, LLC (2020). *Gurobi Optimizer Reference Manual*. URL: http://www.gurobi.com (visited on July 28, 2024).

Hart, Watson, and Woodruff (2011). "Pyomo: modeling and solving mathematical programs in Python". In: *Math. Prog. Comp.* 3.3, pp. 219–260. DOI: 10.1007/s12532-011-0026-8.

Hering, Xhonneux, and Müller (2021). "Design optimization of a heating network with multiple heat pumps using mixed integer quadratically constrained programming". In: *Energy* 226, p. 120384. DOI: 10.1016/j.energy.2021.120384.

Hernandez-Galan and Plauchu (1989). "Determination of fouling factors for shell-and-tube type heat exchangers exposed to Los Azufres geothermal fluids". In: *Geothermics* 18.1-2, pp. 121–128. DOI: 10.1016/0375-6505(89)90018-7.

Higle and Sen (1991). "Stochastic decomposition: An algorithm for two-stage linear programs with recourse". In: *Math. Oper. Res.* 16.3, pp. 650–669. DOI: 10.1287/moor.16.3.650.

Hilpert et al. (2018). "The Open Energy Modelling Framework (oemof) - A new approach to facilitate open science in energy system modelling". In: *Energy Strategy Rev.* 22, pp. 16–25. DOI: 10.1016/j.esr.2018.07.001.

Horst and Tuy (1996). *Global Optimization*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-662-03199-5.

Howells et al. (2011). "OSeMOSYS: the open source energy modeling system: an introduction to its ethos, structure and development". In: *Energy Policy* 39.10, pp. 5850–5870. DOI: 10.1016/j.enpol.2011.06.033.

Huchette, Lubin, and Petra (2014). "Parallel Algebraic Modeling for Stochastic Optimization". In: *1st Workshop High Perform. Tech. Comput. Dyn. Lang.* (New Orleans Convention Center, Nov. 17, 2014). New Orleans, LA: IEEE, pp. 29–35. DOI: 10.1109/hptcdl.2014.6.

Hunter, Sreepathi, and DeCarolis (2013). "Modeling for insight using tools for energy model optimization and analysis (Temoa)". In: *Energy Econ.* 40, pp. 339–349. DOI: 10.1016/j.eneco.2013.07.014.

Huster, Schweidtmann, and Mitsos (2019). "Impact of Accurate Working Fluid Properties on the Globally Optimal Design of an Organic Rankine Cycle". In: *9th FOCAPD*. ed. by S. G. Muñoz, C. D. Laird, and M. J. Realff. Vol. 47. Comput. Aided Chem. Eng. Elsevier, pp. 427–432. DOI: 10.1016/b978-0-12-818597-1.50068-0.

Huster, Schweidtmann, and Mitsos (2020a). "Globally optimal working fluid mixture composition for geothermal power cycles". In: *Energy* 212, p. 118731. DOI: 10.1016/j.energy.2020.118731.

Huster, Schweidtmann, and Mitsos (2020b). "Hybrid Mechanistic Data-Driven Modeling for the Deterministic Global Optimization of a Transcritical Organic Rankine Cycle". In: *30th ESCAPE*. vol. 48. Comput. Aided Chem. Eng. Elsevier, pp. 1765–1770. DOI: 10.1016/b978-0-12-823377-1.50295-0.

Huster et al. (2020). "Deterministic global superstructure-based optimization of an organic Rankine cycle". In: *Comput. Chem. Eng.* 141, p. 106996. DOI: 10.1016/j.compchemeng.2020.106996.

(2024). URL: https://idaes.org/ (visited on June 29, 2024).

Jalving, Cao, and Zavala (2019). "Graph-based modeling and simulation of complex systems". In: *Comput. Chem. Eng.* 125, pp. 134–154. DOI: 10.1016/j.compchemeng.2019.03.009.

Jalving et al. (2017). "A graph-based computational framework for simulation and optimisation of coupled infrastructure networks". In: *IET Generation, Transmission & Distribution* 11.12 (12), pp. 3163–3176. DOI: 10.1049/iet-gtd.2016.1582.

Jentsch et al. (2008). *Handbuch zur Entscheidungsunterstützung - Fernwärme in der Fläche: Leitungsgebundene Wärmeversorgung im ländlichen Raum*. Tech. rep. Fernwärmeversorgung Niederrhein GmbH, Dinslaken et al. URL: https://publica.fraunhofer.de/entities/publication/5e9c382e-5f87-4e32-a56a-f8bc02c24e0f/details (visited on July 28, 2024).

Johnston et al. (2019). "Switch 2.0: A Modern Platform for Planning High-Renewable Power Systems". In: *SoftwareX* 10, p. 100251. DOI: 10.1016/j.softx.2019.100251.

Kalikatzarakis and Frangopoulos (2016). "Thermo-economic optimization of synthesis, design and operation of a marine organic Rankine cycle system". In: *Proc. Institution of Mechanical Engineers, Part M: J. Engineering for the Maritime Environment* 231.1, pp. 137–152. DOI: 10.1177/1475090215627179.

Kall and Wallace (1994). *Stochastic Programming*. John Wiley and Sons Ltd.

Kannan (2018). "Algorithms, analysis and software for the global optimization of two-stage stochastic programs". PhD thesis. Massachusetts Institute of Technology. URL: https://dspace.mit.edu/handle/1721.1/117326 (visited on May 13, 2024).

Kannan and Barton (2017a). "Convergence-order analysis of branch-and-bound algorithms for constrained problems". In: *J. Glob. Optim.* 71.4, pp. 753–813. DOI: 10.1007/s10898-017-0532-y.

Kannan and Barton (2017b). "The cluster problem in constrained global optimization". In: *J. Glob. Optim.* 69.3, pp. 629–676. DOI: 10.1007/s10898-017-0531-z.

Kaplan, Sfar, and Shilon (1999). "Small scale geothermal power plants with less than 5.0 MW capacity". In: *Bull. Hydrogéol.* 17, pp. 433–440.

Karmakar, Mahato, and Bhunia (2009). "Interval oriented multi-section techniques for global optimization". In: *J. Comput. Appl. Math.* 224.2, pp. 476–491. DOI: 10.1016/j.cam.2008.05.025.

Karuppiah and Grossmann (2007). "A Lagrangean based branch-and-cut algorithm for global optimization of nonconvex mixed-integer nonlinear programs with decomposable structures". In: *J. Glob. Optim.* 41.2, pp. 163–186. DOI: 10.1007/s10898-007-9203-8.

Kawajir, Laird, and Wächter (2011). *Introduction to Ipopt*.

Kazazakis (2017). "Parallel computing, interval derivative methods, heuristic algorithms, and their implementation in a numerical solver, for deterministic global optimization". PhD thesis. Imperial College London.

Kearfott and Du (1993). "The Cluster Problem in Global Optimization: the Univariate Case". In: *Validation Numerics*. Computing Supplementum. Vienna: Springer, pp. 117–127. DOI: 10.1007/978-3-7091-6918-6_10.

Kern and Kraus (1972). *Extended Surface Heat Transfer*. New York: McGraw-Hill.

Khajavirad and Michalek (2009). "A Deterministic Lagrangian-Based Global Optimization Approach for Quasiseparable Nonconvex Mixed-Integer Nonlinear Programs". In: *J. Mech. Design* 131.5. DOI: 10.1115/1.3087559.

Khan (2018). "Subtangent-based approaches for dynamic set propagation". In: *2018 IEEE Conference on Decision and Control (CDC)* (Miami, FL, USA, Sept. 17–19, 2018), pp. 3050–3055. DOI: 10.1109/cdc.2018.8618872.

Kim et al. (2018). "Efficient design optimization of complex system through an integrated interface using symbolic computation". In: *Adv. Eng. Software* 126, pp. 34–45. DOI: 10.1016/j.advengsoft.2018.09.006.

Kim and Zavala (2017). "Algorithmic innovations and software for the dual decomposition method applied to stochastic mixed-integer programs". In: *Math. Programm. Comput.* 10.2, pp. 225–266. DOI: 10.1007/s12532-017-0128-z.

King and Wallace (2012). *Modeling with Stochastic Programming*. Springer New York. DOI: 10.1007/978-0-387-87817-1.

Kingma and Ba (2014). "Adam: A Method for Stochastic Optimization". In: *http://arxiv.org/abs/1412.6980*. arXiv: 1412.6980 [cs.LG].

Kirst, Stein, and Steuermann (2015). "Deterministic upper bounds for spatial branch-and-bound methods in global minimization with nonconvex constraints". In: *TOP* 23.2, pp. 591–616. DOI: 10.1007/s11750-015-0387-7.

Kleywegt, Shapiro, and Homem-de-Mello (2002). "The sample average approximation method for stochastic discrete optimization". In: *SIAM J. Optim.* 12.2, pp. 479–502. DOI: 10.1137/s1052623499363220.

Kuppan (2013). *Heat Exchanger Design Handbook*. CRC Press. DOI: 10.1201/b14877.

Kydes (1978). *Brookhaven Energy System Optimization Model: its variants and uses. [BESOM; TESOM; MARKAL]*. tech. rep. OSTI. DOI: 10.2172/6146529.

Kyriakarakos, Ntavou, and Manolakos (2020). "Investigation of the Use of Low Temperature Geothermal Organic Rankine Cycle Engine in an Autonomous Polygeneration Microgrid". In: *Sustainability* 12.24, p. 10475. DOI: 10.3390/su122410475.

La Seta et al. (2016). "Combined Turbine and Cycle Optimization for Organic Rankine Cycle Power Systems—Part B: Application on a Case Study". In: *Energies* 9.6, p. 393. DOI: 10.3390/en9060393.

Lampe et al. (2014). "Simultaneous Optimization of Working Fluid and Process for Organic Rankine Cycles Using PC-SAFT". in: *Ind. Eng. Chem. Res.* 53.21, pp. 8821–8830. DOI: 10.1021/ie5006542.

Langiu, Dahmen, and Mitsos (2022). "Simultaneous optimization of design and operation of an air-cooled geothermal ORC under consideration of multiple operating points". In: *Comput. Chem. Eng.* 161, p. 107745. DOI: 10.1016/j.compchemeng.2022.107745.

Langiu et al. (2021). "COMANDO: A Next-Generation Open-Source Framework for Energy Systems Optimization". In: *Comput. Chem. Eng.*, p. 107366. DOI: 10.1016/j.compchemeng.2021.107366.

Langiu et al. (2024). "MUSE-BB: A Decomposition Algorithm for Nonconvex Two-Stage Problems using Strong Multisection Branching". Submitted to *J. Glob. Optim.*, preprint available at https://optimization-online.org/?p=26862.

Laporte and Louveaux (1993). "The integer L-shaped method for stochastic integer programs with complete recourse". In: *Oper. Res. Lett.* 13.3, pp. 133–142. DOI: 10.1016/0167-6377(93)90002-X.

Lazzaretto and Manente (2014). "A New Criterion to Optimize ORC Design Performance Using Efficiency Correlations for Axial and Radial Turbines". In: *Int. J. Thermodyn.* 17.3, p. 192. DOI: 10.5541/ijot.562.

Lee, Tester, and You (2019). "Systems analysis, design, and optimization of geothermal energy systems for power production and polygeneration: State-of-the-art and future challenges". In: *Renew. Sustain. Energy Rev.* 109, pp. 551–577. DOI: 10.1016/j.rser.2019.04.058.

Li and Grossmann (2018). "An improved L-shaped method for two-stage convex 0–1 mixed integer nonlinear stochastic programs". In: *Comput. Chem. Eng.* 112, pp. 165–179. DOI: 10.1016/j.compchemeng.2018.01.017.

Li and Grossmann (2019a). "A finite ϵ-convergence algorithm for two-stage stochastic convex nonlinear programs with mixed-binary first and second-stage variables". In: *J. Glob. Optim.* 75.4, pp. 921–947. DOI: 10.1007/s10898-019-00820-y.

Li and Grossmann (2019b). "A generalized Benders decomposition-based branch and cut algorithm for two-stage stochastic programs with nonconvex constraints and mixed-binary first and second stage variables". In: *J. Glob. Optim.* 75.2, pp. 247–272. DOI: 10.1007/s10898-019-00816-8.

Li and Li (2015). "Global optimization of an industrial natural gas production network". In: *IFAC-PapersOnLine* 48.8, pp. 337–342. DOI: 10.1016/j.ifacol.2015.08.204.

Li and Li (2016). "Domain reduction for Benders decomposition based global optimization". In: *Comput. Chem. Eng.* 93, pp. 248–265. DOI: 10.1016/j.compchemeng.2016.06.009.

Li and Cui (2024). "A Decomposition Algorithm for Two-Stage Stochastic Programs with Nonconvex Recourse Functions". In: *SIAM J. Optim.* 34.1, pp. 306–335. DOI: 10.1137/22M1488533.

Li and Barton (2015). "Optimal design and operation of energy systems under uncertainty". In: *J. Process Control* 30, pp. 1–9. DOI: 10.1016/j.jprocont.2014.11.004.

Li, Sundaramoorthy, and Barton (2014). "Nonconvex Generalized Benders Decomposition". In: *Optimization in Science and Engineering*. New York: Springer, pp. 307–331. DOI: 10.1007/978-1-4939-0808-0_16.

Li, Tomasgard, and Barton (2011). "Nonconvex Generalized Benders Decomposition for Stochastic Separable Mixed-Integer Nonlinear Programs". In: *J. Optim. Theory Appl.* 151.3, pp. 425–454. DOI: 10.1007/s10957-011-9888-1.

Li et al. (2011). "Stochastic pooling problem for natural gas production network design and operation under uncertainty". In: *AIChE J.* 57.8, pp. 2120–2135. DOI: 10.1002/aic.12419.

Lio, Manente, and Lazzaretto (2016). "Predicting the Optimum Design of Single Stage Axial Expanders in ORC Systems: Is There a Single Efficiency Map for Different Working Fluids?" In: *Appl. Energy* 167, pp. 44–58. DOI: 10.1016/j.apenergy.2016.01.020.

Liu et al. (2018). "A Review of Modeling Approaches and Tools for the Off-design Simulation of Organic Rankine Cycle". In: *J. Therm. Sci.* 27.4, pp. 305–320. DOI: 10.1007/s11630-018-1023-2.

Liu, Georgiadis, and Pistikopoulos (2010). "Advances in energy systems engineering". In: *Ind. Eng. Chem. Res.* 50.9, pp. 4915–4926. DOI: 10.1021/ie101383h.

Loulou and Labriet (2007). "ETSAP-TIAM: the TIMES integrated assessment model Part I: Model structure". In: *Comput. Manag. Sci.* 5.1-2, pp. 7–40. DOI: 10.1007/s10287-007-0046-z.

Lueg et al. (2021). *reluMIP: Open Source Tool for MILP Optimization of ReLU Neural Networks*. Version 1.0.0. DOI: https://doi.org/10.5281/zenodo.5601907. URL: https://github.com/ChemEngAI/ReLU_ANN_MILP.

Macchi and Perdichizzi (1981). "Efficiency Prediction for Axial-flow Turbines Operating with Nonconventional Fluids". In: *Journal of Engineering for Power* 103.4, pp. 718–724. DOI: 10.1115/1.3230794.

Macchi (2013). "The Choice of Working Fluid: The Most Important Step for a Successful Organic Rankine Cycle (and an Efficient Turbine)". In: *2nd Int. Seminar ORC Power Syst.* (Oct. 7–8, 2013). Rotterdam, The Netherlands, pp. 7–8.

Macchi and Astolfi (2017a). *Organic Rankine Cycle (ORC) Power Systems: Technologies and Applications*. Elsevier. DOI: 10.1016/c2014-0-04239-6.

Macchi and Astolfi (2017b). "Organic Rankine Cycle (ORC) Power Systems: Technologies and Applications". In: *Organic Rankine Cycle (ORC) Power Systems: Technologies and Applications*. Elsevier. Chap. Axial Flow Turbines for Organic Rankine Cycle Applications, pp. 299–319. DOI: 10.1016/b978-0-08-100510-1.00009-0.

Madansky (1960). "Inequalities for Stochastic Linear Programming Problems". In: *Manage. Sci.* 6.2, pp. 197–204. DOI: 10.1287/mnsc.6.2.197.

Magnusson and Åkesson (2015). "Dynamic Optimization in JModelica.org". In: *Processes* 3.2, pp. 471–496. DOI: 10.3390/pr3020471.

Mago et al. (2008). "An examination of regenerative organic Rankine cycles using dry fluids". In: *Appl. Therm. Eng.* 28.8-9, pp. 998–1007. DOI: 10.1016/j.applthermaleng.2007.06.025.

Manente et al. (2013). "An Organic Rankine Cycle Off-design Model for the Search of the Optimal Control Strategy". In: *Energy* 58, pp. 97–106. DOI: 10.1016/j.energy.2012.12.035.

Markót, Csendes, and Csallner (2000). "Multisection in Interval Branch and Bound Methods for Global Optimization – II. Numerical Tests". In: *J. Glob. Optim.* 16.3, pp. 219–228. DOI: 10.1023/a:1008359223042.

Martelli and Amaldi (2014). "PGS-COM: A hybrid method for constrained non-smooth black-box optimization problems: Brief review, novel algorithm and comparative evaluation". In: *Comput. Chem. Eng.* 63, pp. 108–139. DOI: 10.1016/j.compchemeng.2013.12.014.

Martelli, Capra, and Consonni (2015). "Numerical optimization of Combined Heat and Power Organic Rankine Cycles – Part A: Design optimization". In: *Energy* 90, pp. 310–328. DOI: 10.1016/j.energy.2015.06.111.

Martinsen and Krey (2008). "Compromises in energy policy—Using fuzzy optimization in an energy systems model". In: *Energy Policy* 36.8, pp. 2983–2994. DOI: 10.1016/j.enpol.2008.04.005.

Massé (1946). *Les Réserves et la Régulation de l'Avenir dans la vie Économique*. Ed. by P. Hermann & Cie.

Mavromatidis, Orehounig, and Carmeliet (2018). "Uncertainty and global sensitivity analysis for the optimal design of distributed energy systems". In: *Appl. Energy* 214, pp. 219–238. DOI: 10.1016/j.apenergy.2018.01.062.

Mavrotas (2009). "Effective implementation of the ϵ-constraint method in Multi-Objective Mathematical Programming problems". In: *Appl. Math. Comput.* 213.2, pp. 455–465. DOI: 10.1016/j.amc.2009.03.037.

Mazzi, Rech, and Lazzaretto (2015). "Off-design Dynamic Model of a Real Organic Rankine Cycle System Fuelled by Exhaust Gases from Industrial Processes". In: *Energy* 90, pp. 537–551. DOI: 10.1016/j.energy.2015.07.083.

McCormick (1976). "Computability of global solutions to factorable nonconvex programs: Part I - Convex underestimating problems". In: *Math. Programm.* 10.1, pp. 147–175. DOI: 10.1007/bf01580665.

Meroni et al. (2016). "Combined Turbine and Cycle Optimization for Organic Rankine Cycle Power Systems—Part A: Turbine Model". In: *Energies* 9.5, p. 313. DOI: 10.3390/en9050313.

Meurer et al. (2017). "SymPy: symbolic computing in Python". In: *PeerJ Comput. Sci.* 3, e103. DOI: 10.7717/peerj-cs.103.

Miller et al. (2018). "Next Generation Multi-Scale Process Systems Engineering Framework". In: *13th PSE* (Manchester Grand Hyatt, July 1, 2018). San Diego, CA: Elsevier, pp. 2209–2214. DOI: 10.1016/b978-0-444-64241-7.50363-3.

Mines and Wendt (2013). *Air-Cooled Condensers for Next Generation Geothermal Power Plants: Final Report*. Tech. rep. Idaho National Laboratory (INL).

Mistry and Misener (2016). "Optimising heat exchanger network synthesis using convexity properties of the logarithmic mean temperature difference". In: *Comput. Chem. Eng.* 94, pp. 1–17. DOI: 10.1016/j.compchemeng.2016.07.001.

Mitsos, Chachuat, and Barton (2009). "McCormick-Based Relaxations of Algorithms". In: *SIAM J. Optim.* 20.2, pp. 573–601. DOI: 10.1137/080717341.

Mitsos et al. (2018). "Challenges in process optimization for new feedstocks and energy sources". In: *Comput. Chem. Eng.* 113, pp. 209–221. DOI: 10.1016/j.compchemeng.2018.03.013.

Munguia, Oxberry, and Rajan (2016). "PIPS-SBB: A Parallel Distributed-Memory Branch-and-Bound Algorithm for Stochastic Mixed-Integer Programs". In: *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE. DOI: 10.1109/ipdpsw.2016.159.

Najman, Bongartz, and Mitsos (2021). "Linearization of McCormick relaxations and hybridization with the auxiliary variable method". In: *J. Glob. Optim.* DOI: 10.1007/s10898-020-00977-x.

Najman and Mitsos (2016a). "Convergence analysis of multivariate McCormick relaxations". In: *J. Glob. Optim.* 66.4, pp. 597–628. DOI: 10.1007/s10898-016-0408-6.

Najman and Mitsos (2016b). "Convergence Order of McCormick Relaxations of LMTD Function in Heat Exchanger Networks". In: *26th ESCAPE*. vol. 38. Comput. Aided Chem. Eng. Elsevier, pp. 1605–1610. DOI: 10.1016/b978-0-444-63428-3.50272-1.

Najman and Mitsos (2019). "Tighter McCormick Relaxations through Subgradient Propagation". In: *J. Glob. Optim.* 75.3, pp. 565–593. DOI: 10.1007/s10898-019-00791-0.

Navarro and Vassiliadis (2014). "Computer algebra systems coming of age: Dynamic simulation and optimization of DAE systems in Mathematica™". in: *Comput. Chem. Eng.* 62, pp. 125–138. DOI: 10.1016/j.compchemeng.2013.11.004.

Nazif (2011). "Feasibility of Developing Binary Power Plants in the Existing Geothermal Production Areas in Indonesia". In: *United Nations University Geothermal Training Programme, Reykjavik, Iceland*, pp. 709–735.

Nicholson et al. (2018). "pyomo.dae: a modeling and automatic discretization framework for optimization with differential and algebraic equations". In: *Math. Prog. Comp.* 10.2, pp. 187–223. DOI: 10.1007/s12532-017-0127-0.

Nikolić (2016). "DAE Tools: equation-based object-oriented modelling, simulation and optimisation software". In: *PeerJ Comput. Sci.* 2, e54. DOI: 10.7717/peerj-cs.54.

Norkin, Pflug, and Ruszczyński (1998). "A branch and bound method for stochastic global optimization". In: *Math. Programm.* 83.1–3, pp. 425–450. DOI: 10.1007/bf02680569.

Nusiaputra, Wiemer, and Kuhn (2014). "Thermal-economic Modularization of Small, Organic Rankine Cycle Power Plants for Mid-enthalpy Geothermal Fields". In: *Energies* 7.7, pp. 4221–4240. DOI: 10.3390/en7074221.

Ogbe and Li (2019). "A joint decomposition method for global optimization of multiscenario nonconvex mixed-integer nonlinear programs". In: *J. Glob. Optim.* 75.3, pp. 595–629. DOI: 10.1007/s10898-019-00786-x.

Oliveira et al. (2013). "A Lagrangean decomposition approach for oil supply chain investment planning under uncertainty with risk considerations". In: *Comput. Chem. Eng.* 50, pp. 184–195. DOI: 10.1016/j.compchemeng.2012.10.012.

Papoulias and Grossmann (1983). "A structural optimization approach in process synthesis—I". in: *Comput. Chem. Eng.* 7.6, pp. 695–706. DOI: 10.1016/0098-1354(83)85022-4.

Pedregosa et al. (2011). "Scikit-learn: Machine Learning in Python". In: *J. Mach. Learn. Res.* 12.85, pp. 2825–2830. URL: http://jmlr.org/papers/v12/pedregosa11a.html.

Pereira and Pinto (1991). "Multi-stage stochastic optimization applied to energy planning". In: *Math. Programm.* 52.1-3, pp. 359–375. DOI: 10.1007/bf01582895.

Perera and Kamalaruban (2021). "Applications of reinforcement learning in energy systems". In: *Renew. Sustain. Energy Rev.* 137, p. 110618. DOI: 10.1016/j.rser.2020.110618.

Pfeiffer (2012). "Optimization Library for Interactive Multi-Criteria Optimization Tasks". In: *Proc. 9th Int. MODELICA Conf.* (Sept. 3, 2012). MODELLICA. Munich, Germany: Linköping University Electronic Press, pp. 669–680. DOI: 10.3384/ecp12076669.

Pfenninger, Hawkes, and Keirstead (2014). "Energy systems modeling for twenty-first century energy challenges". In: *Renew. Sustain. Energy Rev.* 33, pp. 74–86. DOI: 10.1016/j.rser.2014.02.003.

Pfenninger and Keirstead (2015). "Renewables, nuclear, or fossil fuels? Scenarios for Great Britain's power system considering costs, emissions and energy security". In: *Appl. Energy* 152, pp. 83–93. DOI: 10.1016/j.apenergy.2015.04.102.

Pierobon et al. (2013). "Multi-objective Optimization of Organic Rankine Cycles for Waste Heat Recovery: Application in an Offshore Platform". In: *Energy* 58, pp. 538–549. DOI: 10.1016/j.energy.2013.05.039.

Pili, Spliethoff, and Wieland (2019). "Effect of cold source conditions on the design and control of organic rankinecycles for waste heat recovery from industrial processes". In: *32nd ECOS* (June 23–28, 2019). Wrocław, Poland.

Pili et al. (2017). "Techno-economic Analysis of Waste Heat Recovery with ORC from Fluctuating Industrial Sources". In: *Energy Procedia* 129, pp. 503–510. DOI: 10.1016/j.egypro.2017.09.170.

Pili et al. (2019a). "Efficiency Correlations for Off-design Performance Prediction of Orc Axial-flow Turbines". In: *5th Int. Seminar ORC Power Syst.* NTUA. Athens, Greece.

Pili et al. (2019b). "Simulation of Organic Rankine Cycle – Quasi-steady state vs dynamic approach for optimal economic performance". In: *Energy* 167, pp. 619–640. DOI: 10.1016/j.energy.2018.10.166.

Pistikopoulos (1995). "Uncertainty in process design and operations". In: *Comput. Chem. Eng.* 19, pp. 553–563. DOI: 10.1016/0098-1354(95)87094-6.

Pistikopoulos and Diangelakis (2016). "Towards the integration of process design, control and scheduling: Are we getting closer?" In: *Comput. Chem. Eng.* 91, pp. 85–92. DOI: 10.1016/j.compchemeng.2015.11.002.

Powell (2021). "From Reinforcement Learning to Optimal Control: A Unified Framework for Sequential Decisions". In: *Studies in Systems, Decision and Control*. Springer International Publishing, pp. 29–74. DOI: 10.1007/978-3-030-60990-0_3.

Process Systems Enterprise (1997–2023). *gPROMS*. URL: http://web.archive.org/web/20231001180933/https://www.psenterprise.com/sectors/academic/info-faq (visited on July 28, 2024).

Puterman (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons. DOI: 10.1002/9780470316887.

Robertson, Cheng, and Scott (2024). "On the convergence order of value function relaxations used in decomposition-based global optimization of nonconvex stochastic programs". In: *J. Glob. Optim.* DOI: 10.1007/s10898-024-01458-1.

Robertson, Cheng, and Scott (2020). "Convergence Rate Analysis for Schemes of Relaxations in Decomposition Methods for Global Nonconvex Stochastic Optimization" (virtual, Nov. 17, 2020). CAST plenary session of AIChE Annual Meeting 2020.

Rockafellar and Wets (1991). "Scenarios and policy aggregation in optimization under uncertainty". In: *Math. Oper. Res.* 16.1, pp. 119–147. DOI: 10.1287/moor.16.1.119.

Rote (1992). "The convergence rate of the sandwich algorithm for approximating convex functions". In: *Computing* 48.3–4, pp. 337–361. DOI: 10.1007/bf02238642.

Ryoo and Sahinidis (1995). "Global optimization of nonconvex NLPs and MINLPs with applications in process design". In: *Comput. Chem. Eng.* 19.5, pp. 551–566. DOI: 10.1016/0098-1354(94)00097-2.

Saelens et al. (2019). "Towards a DESTEST: a District Energy Simulation Test Developed in IBPSA Project 1". In: *Proc. Build. Simul. 2019: 16th Conference of IBPSA*. ed. by V. Corrado et al. Building Simulation Conference proceedings. Rome, Italy: IBPSA, pp. 3569–3577. DOI: 10.26868/25222708.2019.210806.

Sahinidis (2004). "Optimization under uncertainty: state-of-the-art and opportunities". In: *Comput. Chem. Eng.* 28.6-7, pp. 971–983. DOI: 10.1016/j.compchemeng.2003.09.017.

Sahinidis (2020). *BARON user manual: v. 20.10.16*. URL: http://www.minlp.com/downloads/docs/baron%20manual.pdf (visited on Feb. 2, 2021).

Sahinidis (2024). *BARON user manual: v. 24.05.08*. URL: http://www.minlp.com/downloads/docs/baron%20manual.pdf (visited on May 8, 2024).

Saleh et al. (2007). "Working fluids for low-temperature organic Rankine cycles". In: *Energy* 32.7, pp. 1210–1221. DOI: 10.1016/j.energy.2006.07.001.

Sass and Mitsos (2019). "Optimal operation of dynamic (energy) systems: When are quasi-steady models adequate?" In: *Comput. Chem. Eng.* 124, pp. 133–139. DOI: 10.1016/j.compchemeng.2019.02.011.

Sass et al. (2020). "Model Compendium, Data, and Optimization Benchmarks for Sector-Coupled Energy Systems". In: *Comput. Chem. Eng.* 135, p. 106760. DOI: 10.1016/j.compchemeng.2020.106760.

Schäfer et al. (2019a). "Economic nonlinear model predictive control using hybrid mechanistic data-driven models for optimal operation in real-time electricity markets: In-silico application to air separation processes". In: *J. Process Control* 84, pp. 171–181. DOI: 10.1016/j.jprocont.2019.10.008.

Schäfer et al. (2019b). "Reduced dynamic modeling approach for rectification columns based on compartmentalization and artificial neural networks". In: *AIChE J.* 65.5, e16568. DOI: 10.1002/aic.16568.

Schäfer et al. (2020). "Wavelet-based grid-adaptation for nonlinear scheduling subject to time-variable electricity prices". In: *Comput. Chem. Eng.* 132, p. 106598. DOI: 10.1016/j.compchemeng.2019.106598.

Schedwill (1968). "Thermische Auslegung von Kreuzstromwärmeaustauschern. Fortschr-Ber." In: *VDI Zeitschrift* 6.19.

Schichl and Neumaier (2005). "Interval Analysis on Directed Acyclic Graphs for Global Optimization". In: *J. Glob. Optim.* 33.4, pp. 541–562. DOI: 10.1007/s10898-005-0937-x.

Schilling et al. (2015). "Working Fluid Selection for Organic Rankine Cycles Based on Continuous-molecular Targets". In: *ASME ORC 2015 - 3rd Int. Semin. ORC Power Syst.* Vol. 1. Brussels, Belgium, pp. 732–741.

Schilling et al. (2017). "1-stage CoMT-CAMD: An Approach for Integrated Design of ORC Process and Working Fluid Using PC-SAFT". in: *Chem. Eng. Sci.* 159, pp. 217–230. DOI: 10.1016/j.ces.2016.04.048.

Schrattenholzer (1981). *The energy supply model MESSAGE*. tech. rep. International Institute for Applied Systems Analysis.

Schütz et al. (2018). "Comparison of clustering algorithms for the selection of typical demand days for energy system synthesis". In: *Renew. Energy* 129, pp. 570–582. DOI: 10.1016/j.renene.2018.06.028.

Schweidtmann and Mitsos (2018). "Deterministic Global Optimization with Artificial Neural Networks Embedded". In: *J. Optim. Theory Appl.* 189, pp. 925–948. DOI: 10.1007/s10957-018-1396-0.

Schweidtmann et al. (2019). "Deterministic Global Process Optimization: Accurate (single-species) Properties Via Artificial Neural Networks". In: *Comput. Chem. Eng.* 121, pp. 67–74. DOI: 10.1016/j.compchemeng.2018.10.007.

Scott and Barton (2015). "Reachability Analysis and Deterministic Global Optimization of DAE Models". In: *Differential-Algebraic Equations Forum*. Springer International Publishing, pp. 61–116. DOI: 10.1007/978-3-319-22428-2_2.

Scott, Stuber, and Barton (2011). "Generalized McCormick relaxations". In: *J. Glob. Optim.* 51.4, pp. 569–606. DOI: 10.1007/s10898-011-9664-7.

Seeling-Hochmuth (1997). "A combined optimisation concept for the design and operation strategy of hybrid-PV energy systems". In: *Sol. Energy* 61.2, pp. 77–87. DOI: 10.1016/s0038-092x(97)00028-5.

Serth (2007). *Process Heat Transfer: Principles and Applications*. Elsevier. DOI: 10.1016/b978-0-12-373588-1.x5000-1.

Shah (1979). "A general correlation for heat transfer during film condensation inside pipes". In: *Int. J. Heat Mass Transf.* 22.4, pp. 547–556. DOI: 10.1016/0017-9310(79)90058-9.

Shah (2009). "An Improved and Extended General Correlation for Heat Transfer during Condensation in Plain Tubes". In: *HVAC&R Res.* 15.5, pp. 889–913. DOI: 10.1080/10789669.2009.10390871.

Shao and Scott (2018). "Convex relaxations for global optimization under uncertainty described by continuous random variables". In: *AIChE J.* 64.8, pp. 3023–3033. DOI: 10.1002/aic.16064.

Shu et al. (2019). "Optimal operation of energy systems with long-term constraints by time-series aggregation in receding horizon optimization". In: *Proc. ECOS 2019* (June 23–28, 2019). Ed. by L. Czarnowska, W. Stanek, and P. Gładysz. Wrocław, Poland: Institute of Thermal Technology, pp. 1981–1990.

Smith (1997). "On the optimal design of continuous processes." PhD thesis. Imperial College London (University of London).

Smith and Pantelides (1997). "Global optimisation of nonconvex MINLPs". In: *Comput. Chem. Eng.* 21, S791–S796. DOI: 10.1016/s0098-1354(97)87599-0.

Smith (2005). *Chemical Process: Design and Integration*. John Wiley & Sons.

Song et al. (2020). "Thermo-economic Optimization of Organic Rankine Cycle (ORC) Systems for Geothermal Power Generation: A Comparative Study of System Configurations". In: *Front. Energy Res.* 8. DOI: 10.3389/fenrg.2020.00006.

Soroudi and Amraee (2013). "Decision making under uncertainty in energy systems: State of the art". In: *Renew. Sustain. Energy Rev.* 28, pp. 376–384. DOI: 10.1016/j.rser.2013.08.039.

Sorourifar, Choksi, and Paulson (2021). "Computationally efficient integrated design and predictive control of flexible energy systems using multi-fidelity simulation-based Bayesian optimization". In: *Optimal Control Applications and Methods* 44.2, pp. 549–576. DOI: 10.1002/oca.2817.

Stechlinski, Khan, and Barton (2018). "Generalized Sensitivity Analysis of Nonlinear Programs". In: *SIAM J. Optim.* 28.1, pp. 272–301. DOI: 10.1137/17m1120385.

Subramanian, Gundersen, and Adams (2018). "Modeling and Simulation of Energy Systems: A Review". In: *Processes* 6.12, p. 238. DOI: 10.3390/pr6120238.

Tartière and Astolfi (2017). "A World Overview of the Organic Rankine Cycle Market". In: *Energy Procedia* 129, pp. 2–9. DOI: 10.1016/j.egypro.2017.09.159.

Tawarmalani and Sahinidis (2004). "Global optimization of mixed-integer nonlinear programs: A theoretical and computational study". In: *Math. Programm.* 99.3, pp. 563–591. DOI: 10.1007/s10107-003-0467-6.

Thieriot et al. (2011). "Towards design optimization with OpenModelica emphasizing parameter optimization with genetic algorithms". In: *Proc. 8th Int. MODELICA Conf.* (Technical Univeristy Dresden). Linköping University Electronic Press. Dresden, Germany, pp. 756–762. DOI: 10.3384/ecp11063756.

Tsoukalas and Mitsos (2014). "Multivariate McCormick relaxations". In: *J. Glob. Optim.* 59.2-3, pp. 633–662. DOI: 10.1007/s10898-014-0176-0.

Turton et al. (2018). *Analysis, Synthesis and Design of Chemical Processes*. Prentice Hall.

Usher et al. (2023). "Global sensitivity analysis to enhance the transparency and rigour of energy system optimisation modelling". In: *Open Res. Europe* 3, p. 30. DOI: 10.12688/openreseurope.15461.1.

Valente et al. (2009). "Extending Algebraic Modelling Languages for Stochastic Programming". In: *INFORMS J. Comput.* 21.1, pp. 107–122. DOI: 10.1287/ijoc.1080.0282.

Van Slyke and Wets (1969). "L-shaped linear programs with applications to optimal control and stochastic programming". In: *SIAM J. Appl. Math.* 17.4, pp. 638–663.

Vielma, Ahmed, and Nemhauser (2010). "Mixed-Integer Models for Nonseparable Piecewise-Linear Optimization: Unifying Framework and Extensions". In: *Oper. Res.* 58.2, pp. 303–315. DOI: 10.1287/opre.1090.0721.

Vigerske and Gleixner (2017). "SCIP: Global optimization of mixed-integer nonlinear programs in a branch-and-cut framework". In: *Optim. Methods Softw.* 33.3, pp. 1–31. DOI: 10.1080/10556788.2017.1335312.

Villanueva (2015). "Set-theoretic methods for analysis estimation and control of nonlinear systems". PhD thesis. Imperial College London.

Voll et al. (2012). "Superstructure-free synthesis and optimization of distributed industrial energy supply systems". In: *Energy* 45.1, pp. 424–435. DOI: 10.1016/j.energy.2012.01.041.

Voll et al. (2013). "Automated superstructure-based synthesis and optimization of distributed energy supply systems". In: *Energy* 50, pp. 374–388. DOI: 10.1016/j.energy.2012.10.045.

Wallace (2000). "Decision Making Under Uncertainty: Is Sensitivity Analysis of Any Use?" In: *Oper. Res.* 48.1, pp. 20–25. DOI: 10.1287/opre.48.1.20.12441.

Wallace and Fleten (2003). "Stochastic Programming Models in Energy". In: *Handbooks in Operations Research and Management Science*. Elsevier, pp. 637–677. DOI: 10.1016/s0927-0507(03)10010-2.

Watson, Woodruff, and Hart (2012). "PySP: modeling and solving stochastic programs in Python". In: *Math. Prog. Comp.* 4.2, pp. 109–149. DOI: 10.1007/s12532-012-0036-1.

Wechsung (2014). "Global optimization in reduced space". PhD thesis. Massachusetts Institute of Technology.

Wechsung, Schaber, and Barton (2014). "The cluster problem revisited". In: *J. Glob. Optim.* 58.3, pp. 429–438. DOI: 10.1007/s10898-013-0059-9.

Wets (1996). "Challenges in stochastic programming". In: *Math. Programm.* 75.2, pp. 115–135. DOI: 10.1007/bf02592149.

Yunt et al. (2008). "Designing man-portable power generation systems for varying power demand". In: *AIChE J.* 54.5, pp. 1254–1269. DOI: 10.1002/aic.11442.

Zadeh (1965). "Fuzzy sets". In: *Inform. Control* 8.3, pp. 338–353. DOI: 10.1016/s0019-9958(65)90241-x.

Zlobec (2005). "On the Liu-Floudas Convexification of Smooth Programs". In: *J. Glob. Optim.* 32.3, pp. 401–407. DOI: 10.1007/s10898-004-3134-4.

DOI: 10.18154/RWTH-2025-01116

Aachener
Verfahrenstechnik

RWTH AACHEN
UNIVERSITY