Deterministic Global and Hierarchical Optimization for Experimental Design

Deterministische globale und hierarchische Optimierung für Versuchsplanung

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

vorgelegt von

Daniel Jungen

Berichter: Universitätsprofessor Alexander Mitsos, Ph.D.
Associate Professor Dr.-Ing. Diego Alejandro Muñoz

Tag der mündlichen Prüfung: 03.12.2025

Diese Dissertation ist auf den Internetseiten
der Universitätsbibliothek online verfügbar.

Titel: Deterministic Global and Hierarchical Optimization for Experimental Design

Autor: Daniel Jungen

Reihe: Aachener Verfahrenstechnik Series
AVT.SVT – Process Systems Engineering
Band 44 (2026)

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
https://www.avt.rwth-aachen.de

Volltext verfügbar: 10.18154/RWTH-2025-10903

Preface

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Systemverfahrenstechnik (AVT.SVT). Zunächst möchte ich meinem Doktorvater, Prof. Alexander Mitsos, für seine Förderung und sein Vertrauen während meiner Zeit an der AVT.SVT danken. Besonders schätze ich die vielen Freiräume, die er mir in der Forschung gewährt hat, sowie seine familienfreundliche Haltung, die es mir ermöglicht hat, Familie und Forschung miteinander zu vereinbaren. Mein Dank gilt außerdem Assoc. Prof. Diego Alejandro Muñoz für die Übernahme des Korreferats sowie Prof. Anna Mechler, die den Vorsitz der Prüfungskommission übernommen hat.

Ich bin Hatim Djelassi und Kaan Rehwasser herzlich dankbar, dass sie mich in vielerlei Hinsicht bereits während meiner Masterarbeit auf das Verfassen dieser Arbeit vorbereitet haben. Zudem bedanke ich mich bei all meinen Mitautorinnen und Mitautoren für die angenehme und produktive Zusammenarbeit. Mein Dank gilt auch allen Kolleginnen und Kollegen, die an der AVT.SVT zu dem außergewöhnlich offenen und einladenden Arbeitsumfeld in der Forschung, in der Lehre und auch abseits der Arbeit beitragen. Insbesondere dankbar bin ich in dieser Hinsicht Anita Ziegler, Aron Zingler, Jan Schulze, Jan Rittig und Philipp Ackermann sowie meinen Bürokolleginnen und -kollegen Eike Cramer, Luise Kaven, Kaan Rehwasser und Laura Lang. Ein herzlicher Dank für die schönen gemeinsamen Pausen gilt zudem dem ehemaligen „Mathematiker-Büro“ bestehend aus Dominik Bongartz, Hatim Djelassi, Jaromił Najman und Susanne Saß. Außerdem bin ich Wanda Frohn, Petra Eissa, Didem Uslu, Jutta Friedrich, Sascha Gerhards, Fabio Zuraszek und ihren Teams für die Unterstützung im Hinblick auf IT, Verwaltung und bürokratischer Prozesse dankbar.

Ich möchte auch Eva, Thomas, Heidrun, Frank sowie Philipp und Jonas und ihren Familien für ihre Unterstützung vor, während und nach der Promotion danken. Abschließend bedanke ich mich bei Marlene für ihre Unterstützung und ihren unermüdlichen Rückhalt in allen Lebenslagen.

Aachen, im Februar 2026 *Daniel Jungen*

Contents

Abbreviations and Mathematical Symbols **IX**

Kurzfassung **XIII**

Summary **XV**

Publications and Copyrights **XVII**

1 Introduction and Overview **1**

2 Merit of Deterministic Global Optimization of the Acquisition Function in Bayesian Optimization **5**
2.1 Introduction . . . 5
2.2 Methods . . . 7
2.2.1 Acquisition Functions . . . 7
2.2.2 Optimizers . . . 8
2.2.3 Bayesian Optimization Termination Criterion . . . 9
2.2.4 Case Studies . . . 10
2.3 Results . . . 12
2.3.1 Convergence to a Globally Near-Optimal Minimum . . . 13
2.3.2 Iterations to Convergence . . . 18
2.3.3 Additional Case Studies . . . 19
2.4 Discussion . . . 22
2.5 Recommendations . . . 23
2.6 Future Work . . . 25

3 Discretization-Based Semi-Infinite and Bilevel Programming Solvers **27**
3.1 Background and Motivation . . . 27
3.2 The Problem Classes SIP, GSIP, MINMAX, and BLP . . . 30
3.3 Adaptive Discretization-Based Algorithms . . . 32
3.3.1 Approaches for SIPs . . . 32
3.3.2 Approaches for GSIPs . . . 37
3.3.3 Approaches for BLPs . . . 39
3.4 The Open-Source Software libDIPS . . . 40
3.4.1 User-Oriented Introduction and Overview . . . 41
3.4.2 Implementation of the Framework . . . 44
3.5 Library of Test Problems . . . 46
3.6 Numerical Experiments . . . 51
3.6.1 Definition of Feasibility and Numerical Challenges . . . 53
3.6.2 Performance Results of SIP Solvers . . . 54
3.6.3 Performance Results of GSIP Solvers . . . 61

3.6.4 Performance Results of BLP Solvers . . . 63
3.6.5 Testing of Two Exemplary Scientific Hypotheses . . . 63
3.7 Conclusion and Outlook . . . 65

4 An Improved Oracle Adaptation for Bilevel Programs 69
4.1 Background . . . 69
4.2 Algorithm Description and Subproblem Formulations . . . 70
4.3 Outline of Proof of Convergence . . . 71
4.4 Numerical Experiments . . . 73
4.5 Conclusion . . . 74

5 Optimal Experimental Design for Guaranteed Parameter Estimation with Rigorous Calculation of Liquid-Liquid Equilibrium Embedded 75
5.1 Introduction and Literature Review . . . 75
5.1.1 Approaches for Modeling and Fitting Liquid-Liquid Equilibria . . . 78
5.1.2 Optimal Experimental Design for Liquid-Liquid Equilibria Characterization . . . 80
5.2 Thermodynamic Background . . . 80
5.2.1 Modeling Liquid-Liquid Equilbria . . . 81
5.2.2 Conditions for the Exclusion of Problematic Parameter Values . . . 82
5.3 Optimal Experimental Design for Guaranteed Parameter Estimation . . . 85
5.3.1 Optimal Experimental Design for Liquid-Liquid Equilibrium Systems . . . 87
5.4 Computational Setup . . . 91
5.5 Case Study . . . 94
5.6 Conclusion and Outlook . . . 96

6 Adaptive Discretization-Based Algorithms for Semi-Infinite Programs with Unbounded Variables 99
6.1 Preliminaries . . . 100
6.1.1 Notation, Formulation, and Subproblems . . . 101
6.1.2 Assumptions . . . 102
6.1.3 Proof of Convergence . . . 103
6.2 Illustrative Examples of SIPs With Unbounded Host Sets . . . 106
6.2.1 Illustrative Example for $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded . . . 106
6.2.2 Illustrative Example for $\mathcal{X}$ Unbounded and $\mathcal{Y}$ Compact . . . 108
6.3 Examples of Assumptions That Are Easier to Prove . . . 108
6.3.1 $\mathcal{X}$ Unbounded $\mathcal{Y}$ Compact . . . 109
6.3.2 $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded . . . 112
6.3.3 $\mathcal{X}$ Unbounded and $\mathcal{Y}$ Unbounded . . . 113
6.4 Numerical Case Studies . . . 113
6.4.1 $\mathcal{X}$ Unbounded and $\mathcal{Y}$ Compact . . . 114
6.4.2 $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded . . . 115
6.5 Conclusion and Outlook . . . 115

7 Conclusion and Outlook 119

A Supplemental Material Regarding Merit of Deterministic Global Optimization of the Acquisition Function in Bayesian Optimization 123
A.1 Additional Test Cases . . . 123
A.1.1 Six-Hump Camelback . . . 123
A.1.2 Ackley . . . 123
A.1.3 Hartmann . . . 123
A.2 Set Up Details . . . 124
A.3 BO Frameworks . . . 125
A.4 Simple Regret . . . 126
A.4.1 Considering Only Runs That Reached a Globally Near-Optimal Solution . . . 126
A.4.2 Considering All Runs, Despite Where They Converged . . . 128
A.5 Iterations to Convergence for All Runs . . . 130

B Supplemental Definitions 133
B.1 $(\varepsilon^{\mathrm{a}})$-Feasibility . . . 133
B.2 SIP-Slater and GSIP-LLP-Slater Point . . . 133
B.3 ε^{f}-optimal (G)SIP-Slater Point . . . 134

C Supplemental Material Regarding Discretization-Based Semi-Infinite and Bilevel Programming Solvers 135
C.1 Blankenship and Falk Solution Algorithm for SIPs . . . 135
C.2 Further Subproblem Formulations . . . 135
C.2.1 SIP Solvers . . . 135
C.2.2 MINMAX Solver . . . 136
C.2.3 BLP Solvers . . . 136
C.3 Details on UBP-Guard . . . 136
C.4 Note on Integer Capabilities of the Solvers . . . 138
C.5 Example Subproblem Formulation for *RRHS* . . . 138
C.6 Changes in the Implemented Solvers . . . 140
C.7 Settings and Tolerances Used in the Numerical Experiments . . . 142
C.8 Additional Performance Profiles . . . 145
C.8.1 Additional SIP Performance Profiles . . . 145
C.8.2 Additional GSIP Performance Profiles . . . 151
C.8.3 Additional BLP Performance Profiles . . . 154
C.8.4 Additional Performance Profiles for the Scientific Hypothesis . . . 156
C.9 Implemented Problem Instances . . . 157

D Supplemental Material Regarding Optimal Experimental Design for Guaranteed Parameter Estimation with Rigorous Calculation of Liquid-Liquid Equilibrium Embedded 191
D.1 Liquid-liquid Equilibirum Model . . . 191
D.1.1 Reduction of the Model Complexity . . . 191
D.1.2 Thermodynamic Relations . . . 192
D.2 Conditions for the Exclusion of Problematic Parameter Values . . . 192
D.2.1 Convexity of $\bar{G}^{\mathrm{M}}$ Outside the Phase Split . . . 192
D.2.2 Gibbs Tangent Plane Below $\bar{G}^{\mathrm{M}}$ at the Boundaries . . . 193

D.2.3 Secant Strictly Below $\bar{G}^{\mathrm{M}}$ in the Interior of the Phase Split 194
D.3 Computational Setup 195

E Supplemental Material Regarding Adaptive Discretization-Based Algorithms for Semi-Infinite Programs with Unbounded Variables 197
E.1 Lower-Bounding Procedure 197
E.1.1 Lower Semi-Continuous Constraint Function g 197
E.1.2 Additional Illustrative Examples 198
E.2 Upper-Bounding Procedure 199
E.2.1 Definitions and Assumptions of the Upper-Bounding Procedure . . 199
E.2.2 $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded 200
E.2.3 Proof of Convergence 201

Bibliography 205

Abbreviations and Mathematical Symbols

The following listings introduce abbreviations and symbols used throughout this dissertation. Please note that the list of symbols does not form an exhaustive enumeration of all symbols used; aditional symbols that are only locally used, along with specific subscripts and superscripts, are declared close to their respective usage.

Abbreviations

BLP	bilevel program
BO	Bayesian optimization
BOARPET	Bilevel Optimization Algorithm for Rigorous & Robust Parameter Estimation in Thermodynamics
cMLE	conditional Maximum Likelihood Estimation
EAR	equal area rule
ESIP	existence-constrained semi-infinite program
FIR	finite impulse response
GP	Gaussian process
GPE	guaranteed parameter estimation
GSIP	generalized semi-infinite program
ILS	informed local solver
IMS	informed multi-start
KKT	Karush-Kuhn-Tucker
L-BFGS-B	limited-memory Broyden-Fletcher-Goldfarb-Shanno solver with box constraints
LBD	lower bound
LBP	lower-bounding problem
LHS	Latin hypercube sampling
LLE	liquid-liquid equilibrium
LLP	lower-level problem
LP/(MI)LP	(mixed-integer) linear program
MINMAX	minmax program
MPEC	mathematical program with equilibrium constraints
NLP/(MI)NLP	(mixed-integer) nonlinear program
OED	optimal experimental design
PK-MIQP	Piecewise-linear Kernel Mixed Integer Quadratic Programming
QCQP/(MI)QCQP	(mixed-integer) quadratically constrained quadratic program
QP/(MI)QP	(mixed-integer) quadratic program
SCP	set of consistent parameters

SIP	semi-infinite program
TP	tangent plane
UBD	upper bound
UBP	upper-bounding problem
ULP	upper-level problem

Symbols

Bayesian Optimization

ϵ	termination tolerance
f	objective function
κ	hyperparameter of LCB
LCB	lower confidence bound acquisition function
μ	mean
N	number of initial data points
σ	standard deviation
U	Müller-Brown potential

Hierarchical Optimization

α	algorithmic parameter of *GSIP-RRHS*
d	algorithmic parameter of *BLP-Box*
ε	restriction parameter of *RRHS* or tolerance
f	objective function
f^{t}	target objective value
$\boldsymbol{g}$	coupling inequality constraint function
h	lower-level objective function in a bilevel program
LBD	lower bound
UBD	upper bound
$\boldsymbol{v}$	constraint function
$\boldsymbol{x}$	upper-level variables
$\boldsymbol{x}^{\mathrm{S}}$	Slater point
$\mathcal{X}$	feasible set of the upper-level
$\bar{\mathcal{X}}$	host set of the upper-level variables
$\boldsymbol{y}$	lower-level variables
$\mathcal{Y}$	feasible set of the lower level
$\bar{\mathcal{Y}}$	host set of the lower-level variables
$\mathcal{Y}^{\mathrm{X}}$	set of discretization points, used with different superscripts X

Optimal Experimental Design for Guaranteed Parameter Estimation

α	algorithmic parameter
δ_x	back-off value
$\mathcal{D}^{\mathrm{X}}$	set of discretization points, used with different superscripts X
ε	tolerance
f^{X}	objective function value, used with different superscripts X
γ	activity coefficient
$\bar{G}^{\mathrm{M}}$	molar Gibbs free energy of mixing
$\boldsymbol{p}$	parameters
$\hat{\boldsymbol{p}}$	nominal parameter
$\mathcal{P}$	parameter set
T	temperature
$\mathcal{T}$	set of admissible temperatures
$\boldsymbol{x}$	mole fractions
φ	phase ratio
y	mole fraction (dummy symbol)

Mathematical Operators

$\mathbb{N}$	natural numbers
$\{1,\ldots,n\}$	the set $\{i \in \mathbb{N} \mid 1 \leq i \leq n\}$
$\mathbb{Z}$	integers
$\mathbb{R}$	real numbers
$\lvert\cdot\rvert$	cardinality of a set or absolute value of a scalar
$\lvert\lvert \cdot \rvert\rvert$	vector or matrix norm
$\lceil\cdot\rceil$	ceiling function (rounding up to nearest integer)
$\frac{\partial \cdot}{\partial x}\big\rvert_T$	partial derivative w.r.t. x, evaluated at T, where x and T are placeholder variables
$\cup$	set union
$\cap$	set intersection
$\setminus$	set difference
$\mathrm{cl}(\cdot)$	closure
$\vee$	logical disjunction (OR)
$\wedge$	logical conjunction (AND)
s.t.	subject to
$\{\boldsymbol{x}\}_{k=0}^{m}$	sequence of points, i.e., $\{\boldsymbol{x}^0, \boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m\}$

Kurzfassung

In dieser Dissertation wird hierarchische Optimierung in optimaler Versuchsplanung für garantierte Parameterschätzung angewendet und dabei der Einsatz sowie die Zugänglichkeit von hierarchischen Optimierungsmethoden verbessert. Ebenso werden die Vor- und Nachteile von globaler Optimierung in der Bayes'schen Optimierung analysiert.

Bayes'sche Optimierung, welche Gauß-Prozesse als Ersatzmodelle verwendet, hat sich als ein effektives Verfahren zur Optimierung von Black-Box-Funktionen hervorgetan. In diesem Kontext werden die Performance Vor- und Nachteile an vier Beispielen untersucht, wenn die Akquisitionsfunktion mit einem globalen deterministischen Löser anstatt mit den üblichen Lösern optimiert wird.

Komplementär dazu erfordert die optimale Versuchsplanung für garantierte Parameterschätzung die Kenntnis des zugrunde liegenden Modells in Form eines Gleichungssystems. Dieses beinhaltet für gewöhnlich mit Unsicherheiten behaftete Parameter, die zuvor geschätzt werden müssen. In dieser Dissertation wird die Formulierung der optimalen Versuchsplanung für garantierte Parameterschätzung dahingehend erweitert, dass auch Modelle betrachtet werden können, die durch ein eingebettetes Optimierungsproblem definiert sind. Das hier betrachtete eingebettete Optimierungsproblem entspricht der thermodynamisch rigorosen Berechnung von flüssig-flüssig Gleichgewichten. Das resultierende Optimierungsproblem gehört zur Klasse der hierarchischen Optimierungsprobleme.

Obwohl in vergangenen Jahren mehrere adaptive diskretisierungsbasierte Methoden zur Lösung dieser Probleme veröffentlich wurden, mangelt es an numerischen Vergleichen. In dieser Dissertation wird eine open source Software zur Lösung dieser Probleme vorgestellt, die zudem über eine umfangreiche Bibliothek von Testproblemen verfügt. Die Software sowie die Testprobleme werden verwendet, um verschiedene adaptive diskretisierungsbasierte Methoden miteinander zu vergleichen und bessere algorithmischen Parameter der verschiedenen Methoden zu ermitteln.

Aufbauend darauf wird eine Anpassung einer bestehenden adaptiven diskretisierungsbasierten Methode zur Lösung von Bilevel-Optimierungsproblemen vorgeschlagen, und diese mit der ursprünglichen und anderen in der Software implementierten Methoden verglichen.

Für die oben genannte erweiterte Formulierung zur optimale Versuchsplanung für garantierte Parameterschätzung wird ein spezialisierter Lösungsalgorithmus vorgestellt und die Machbarkeit der Methode gezeigt.

In vielen Anwendungen der hierarchischen Optimierung, einschließlich optimaler Versuchsplanung für garantierte Parameterschätzung, können die notwendigen Annahmen der verwendeten Lösungsalgorithmen verletzt sein. Deshalb werden die notwendigen Annahmen eines bestehenden Lösungsalgorithmus untersucht und es kann gezeigt werden, dass schwächere Annahmen möglich sind. Da viele adaptive diskretisierungsbasierte Lösungsalgorithmen für hierarchische Optimierung auf den algorithmischen Prinzipien des untersuchten Lösungsalgorithmus basieren, ist davon auszugehen, dass die gezeigten Konvergenzgarantien unter den präziseren, etwas schwächeren Annahmen direkt von dem untersuchten Algorithmus auf die anderen Algorithmen ähnlichen Prinzips übertragbar sind.

Summary

In this dissertation, hierarchical optimization is applied in optimal experimental design for guaranteed parameter estimation, and the applicability and accessibility of hierarchical optimization methods are improved. Furthermore, we analyze the advantages and disadvantages of using global optimization in Bayesian optimization.

Bayesian optimization using Gaussian processes as a surrogate model has been effectively used to optimize expensive to evaluate black-box-functions. In this context, we investigate the advantages and disadvantages of using deterministic global optimization for optimizing the acquisition function, by comparing the Bayesian optimization performance when using a deterministic global solver with two conventional solvers across four different test functions.

In contrast to Bayesian optimization, optimal experimental design for guaranteed parameter estimation requires knowledge of the underlying model as an equation system. These models usually include parameters that must be estimated, and naturally, the parameter values are inherently uncertain. We extend the formulation of optimal experimental design for guaranteed parameter estimation to systems whose input-output relation is implicitly defined through an optimization problem, which corresponds in our application to the rigorous computation of liquid-liquid equilibria.

The resulting optimization problem of the latter approach is a hierarchical program that is generally challenging to solve. Although multiple adaptive discretization-based methods for their solution have been proposed in recent decades, their numerical comparison is lacking. We remedy this by presenting an open-source software for solving multiple hierarchical optimization programs. Our software includes an extensive library of test problems, which we compiled from existing benchmark libraries and other problems in the literature. Utilizing our software and library of test problems, we compare multiple adaptive discretization-based solvers and conduct parameter tuning for the algorithmic parameters associated with each solver.

On this basis, we propose an adaptation of an existing adaptive discretization-based method for solving bilevel programs, implement it in our software, and compare it to existing solvers, including the original approach. For the above-mentioned extended formulation for optimal experimental design for guaranteed parameter estimation, we present a specialized solution algorithm and provide a proof-of-concept, demonstrating the feasibility of our method.

In many applications, including optimal experimental design for guaranteed parameter estimation, the necessary assumptions of the employed hierarchical optimization solvers may be violated. Therefore, we examine the necessary assumptions of an existing semi-infinite optimization method, and show that slightly weaker assumptions are possible. As many adaptive discretization-based solution algorithms have the examined solution algorithm as their predecessor, it can be presumed that the convergence guarantees shown with the more precise, slightly weaker assumptions are directly transferable from one algorithm to the other.

Publications and Copyrights

This dissertation is a result of the research performed by the author during his time at the Chair of Process Systems Engineering (AVT.SVT) between 2019 and 2025. Parts of this dissertation have already been published as preprints, in peer-reviewed journals as conference articles, or have been submitted for publication. The publications are partially integrated into the dissertation chapters as described below. These publications partly originated from collaborative research, and thus the contributions of each (co-)author are stated below the corresponding publications.

- *A. Georgiou, D. Jungen, L. Kaven, V. Hunstig, C. Frangakis, I. Kevrekidis, and A. Mitsos. Deterministic global optimization of the acquisition function in bayesian optimization: To do or not to do?* submitted to Journal of Global Optimization*, 2025a.*

  Preprint: Georgiou et al. (2025b); reprinted (adapted) with permission under the Creative Commons Attribution 4.0 International License (CC BY 4.0). To view a copy of this licence, visit `https://creativecommons.org/licenses/by/4.0/`. This article forms the basis of Chapter 2 and Appendix A. Parts of the publication are further used in Chapter 1.

  Content of this publication has also been used in Georgiou (2025).

  Author's contribution: All authors contributed to the study's conception and/or design. A. Georgiou and **D. Jungen** contributed equally to this work. Methodology development, software implementation, execution of experiments, and formal analysis were led by A. Georgiou, supported by **D. Jungen**. A. Georgiou and **D. Jungen** drafted the initial manuscript, with all authors contributing to writing and editing. V. Hunstig validated the numerical results and software functionality. C. Frangakis developed the statistical framework and conducted conditional logistic regression and minimax testing. L. Kaven contributed to the conceptual framework. I. Kevrekidis and A. Mitsos provided supervision, conceptual guidance, and secured funding.

- *D. Jungen, A. Zingler, H. Djelassi, and A. Mitsos. libDIPS — Discretization-based semi-Infinite and bilevel Programming Solvers.* submitted to Mathematical Programming and Computation*, 2023b.*

  Preprint: Jungen et al. (2023a), © Jungen, Zingler, Djelassi & Mitsos. This article forms the basis of Chapter 3 and Appendices B and C. The corresponding supplementary information is incorporated in Appendix C. Parts of the publication are further used in Chapter 1.

  Content of this publication has also been used in Zingler (2025) and has been presented at scientific conferences (Jungen et al., 2023c,d).

Author's contribution: All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by A. Zingler and **D. Jungen**. A. Zingler and **D. Jungen** contributed equally to this work. The first draft of the manuscript was written by A. Zingler and **D. Jungen** and all authors commented on previous versions of the manuscript. H. Djelassi implemented the first version of the software. A. Zingler and **D. Jungen** continued the development of the software under the supervision of A. Mitsos All authors read and approved the final manuscript (Jungen et al., 2023a).

- *D. Jungen and A. Mitsos. An improved oracle adaption for bilevel programs. In F. Manenti and G. V. Reklaitis, editors,* Proceedings of the 34th European Symposium on Computer Aided Process Engineering / 15th International Symposium on Process Systems Engineering (ESCAPE-34/PSE-24)*, volume 53 of* Computer Aided Chemical Engineering*, pages 3301–3306. Elsevier, 2024a. doi:10.1016/B978-0-443-28824-1.50551-2.*

  This article was published in Computer Aided Chemical Engineering, Vol 53, D. Jungen & A. Mitsos An Improved Oracle Adaption for Bilevel Programs, Page 3301–3306, © Elsevier B.V. 2024. Reprinted (adapted) with permission. This article forms the basis of Chapter 4. Parts of the publication are further used in Chapter 1.

  Content of this publication has also been presented at a scientific conference (Jungen and Mitsos, 2024b).

  Author's contribution: D. Jungen: Conceptualization, Methodology, Software, Validation, Investigation, Writing, Visualization. A. Mitsos: Conceptualization, Resources, Writing - Review & Editing, Supervision, Project administration, Funding acquisition.

- *D. Jungen, J.-F. Laub, and A. Mitsos. Optimal experimental design for guaranteed parameter estimation with rigorous formulation of the embedded liquid-liquid equilibrium.* Chemical Engineering Science*, page 123437, 2026. doi:10.1016/j.ces.2026.123437.*

  Reprinted (adapted) with permission under the Creative Commons Attribution 4.0 International License (CC BY 4.0). To view a copy of this licence, visit `https://creativecommons.org/licenses/by/4.0/`. © The Author(s) 2026. Published by Elsevier Ltd. This article forms the basis of Chapter 5 and Appendix D. Parts of the publication are further used in Chapter 1 and Appendix C.1.

  Content of this publication has also been presented at a scientific conference (Jungen et al., 2025).

  Author's contribution: D. Jungen: Conceptualization, Methodology, Software, Validation, Investigation, Writing - Original Draft, Visualization. J.-F. Laub: Software (first version), Validation, Preliminary Investigation, Writing - Review & Editing. A. Mitsos: Conceptualization, Resources, Writing - Review & Editing, Supervision, Project administration, Funding acquisition.

- *D. Jungen, H. Djelassi, and A. Mitsos. Adaptive discretization-based algorithms for semi-infinite programs with unbounded variables.* Mathematical Methods of Operations Research*, 96(1):83–112, 2022. doi:10.1007/s00186-022-00792-y.*

  Reprinted (adapted) with permission under the Creative Commons Attribution 4.0 International License (CC BY 4.0). To view a copy of this licence, visit `https://creativecommons.org/licenses/by/4.0/`. © The Author(s) 2022. This article forms the basis of Chapter 6 and Appendix E. Parts of the publication are further used in Chapter 1.

  Author's contribution: D. Jungen: Conceptualization, Methodology, Software, Validation, Investigation, Writing - Original Draft, Visualization. H. Djelassi: Methodology, Validation, Investigation, Writing - Review & Editing. A. Mitsos: Conceptualization, Resources, Writing - Review & Editing, Supervision, Project administration, Funding acquisition.

Acknowledgements and Declarations

While at AVT.SVT, the author supervised or co-supervised the (project-)theses of Dennis Grasmik, Vera Ubbenjans, Phong Vu Nguyen, Inken Maria Michael, Malte Frank, Yufei Hu, Jan-Frederic Laub, Yi-Ju Chen, Finja Backhaus, Jan Eifert, Julia Jasovski, Jason Klein, Thanh My Pham, Jannik Dohmen, and Verena Hunstig. In addition, the author worked with the student and graduate research assistants Anirudh Malhotra, Phong Vu Nguyen, Laurens Lueg, Nina Müller-Späth, Jan-Frederic Laub, Dominic Wittner, Benita Wolpers, Marc-Lennard Morisse, and, Elvis Sim. The work of all students is sincerely acknowledged. Many colleagues and students contributed in the development of the software framework libDIPS, which was started during the dissertation of Hatim Djelassi (Djelassi, 2020). A comprehensive list of contributors can be found in the project's author file under `https://git.rwth-aachen.de/avt-svt/public/libdips`. Jan-Frederic Laub, Yi-Ju Chen, Inken Maria Michael, Finja Backhaus, Jan Eifert, Julia Jasovski, Jason Klein, and Thanh My Pham helped with the collection of test problems from literature.

We acknowledge the financial support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – Cluster of Excellence 2186 „The Fuel Science Center“ – ID: 390919832. Computations were performed with computing resources granted by RWTH Aachen University. Further, we acknowledge the use of the GPT-3.5 and 4.0 language model developed by OpenAI and Perplexity AI as an implementation aid, specifically as a code snippet generator, for creating shell scripts to automate the execution of the benchmark tests on the RWTH High Performance Computing cluster and for generating ideas for converting data formats to create the plots presented in this work. During the preparation of this work, the author utilized Grammarly's writing assistant tool, DeepL, Perplexity AI, and GPT to enhance the manuscript's readability and language by generating ideas for improving previously drafted text. After using these tools/services, the author reviewed and edited the content as needed and takes full responsibility for the content.

This dissertation was prepared using LaTeX. Data analysis, modeling, and visualization were conducted primarily with Python. The computational work was performed on systems running Microsoft Windows and Ubuntu. Additional tools used include CMake, Visual Studio, Visual Studio Code, Git, Citavi, Overleaf, and standard office applications.

1 Introduction and Overview

Optimal experimental design (OED) is an approach that can help guide experimental efforts, maximizing knowledge gain while minimizing costs (Franceschini and Macchietto, 2008). The optimization procedures employed for that purpose often rely on mechanistic models to simulate the underlying process system.

If no mechanistic model exists, and there is no intention of developing a mechanistic model, Bayesian optimization (BO) has emerged as a valuable tool for optimizing process systems by treating the process system as an expensive-to-evaluate black-box function (Jones et al., 1998, Shahriari et al., 2016, Wang et al., 2023). BO employs the construction of a cheap-to-evaluate surrogate model, which also provides uncertainty estimates. Often, as in this dissertation, a Gaussian process (GP) (Rasmussen and Williams, 2006) is used as the surrogate model. The surrogate model determines the next best sampling point of the black-box function via optimization of a selected acquisition function, typically balancing *exploitation* and *exploration*. The surrogate model is then improved upon with the information obtained through the newly evaluated point; repetition of this process locates a (hopefully *global*ly or near-optimal) minimum of the black-box function, i.e., of the process system.

On the contrary, if the development of a mechanistic model for, e.g., process optimization and/or simulation, is desired, their development usually involves three key steps. First, the model structure is defined, which often relies on semi-empirical models. The semi-empirical models inherently include parameters that need to be estimated based on time-consuming and costly experiments (Kontogeorgis et al., 2021). Therefore, it is essential to design experiments with great care, which are then conducted in the second step to collect measurement data. Finally, in the third step, the measurement data is utilized for parameter estimation; eventually, yielding a model that can be used for process system optimization. In the first step, the model building, OED can be applied for model discrimination (Box and Hill, 1967). In the second step, after the model structure has been identified, OED can be used for parameter estimation to increase parameter precision (Atkinson et al., 2007, Franceschini and Macchietto, 2008). This dissertation is, apart from BO, concerned with the second step, and we assume there is no structural system-model mismatch.

OED for parameter estimation can be categorized into two main approaches: statistical OED (e.g., refer to Aster and Thurber (2013), Pronzato (2008) reviewing theory, and Franceschini and Macchietto (2008) reviewing applications) and bounded-error OED methods (Pronzato and Walter, 1990). Statistical OED methods are often based on maximum likelihood methods and aim to minimize the estimated parameter variances by considering the stochastic characteristic of the measurement uncertainties and their impact on the parameter uncertainty (Franceschini and Macchietto, 2008). These statistical methods are justifiable if the stochastic characteristics of measurement uncertainties are known, as is often the case (Dong et al., 2005). On the other hand, the necessary assumptions for the statistical methods are sometimes not met or challenging to verify (Milanese and Belforte, 1982). For example, many data sets in the literature are reported without un-

certainty information (Dai et al., 2019a), and for biological systems, in pharmacokinetics, or clinical studies, often only a limited number of data points exist, repeated measurements are not possible (Belforte et al., 1987), or extremely costly and/or time-consuming (Marvel and Williams, 2012). In these situations, reliable statistical analysis and statistical assumptions regarding the probability density function of the measurement errors are prohibited. In these cases, (bounded-error) OED for guaranteed parameter estimation (GPE) can be applied because it does not rely on statistical assumptions and does not propose replicate experiments (Pronzato and Walter, 1989). OED for GPE is an approach for calculating an experimental design (to be carried out) that minimizes the size of the so-called feasible parameter set, resulting in high parameter reliability. In OED for GPE, we encounter concepts and notions of bilevel and (generalized) semi-infinite programming because the definition of the objective and/or constraints of the OED problem involves solving a lower-level optimization problem.

Semi-infinite programs (SIPs), generalized semi-infinite programs (GSIPs), minmax programs (MINMAX), and bilevel programs (BLPs) are hierarchical optimization programs that occur, apart from OED for GPE applications, in many other applications, e.g., gemstone cutting (Winterfeld, 2008), thermodynamics (Clark and Westerberg, 1990, Gümüş, 1997), when considering worst-case uncertainties as in robust optimization and control (Ben-Tal and Nemirovski, 2002, Djelassi et al., 2021, Goerigk et al., 2025, Muñoz and Marquardt, 2013), or flexibility analysis (Swaney and Grossmann, 1985). A reliable and fast solution to such optimization problems is paramount but challenging. Problems belonging to the class of hierarchical optimization programs define their objective or constraints through a lower-level optimization problem. Therefore, this lower-level optimization problem must be solved globally, even to determine the feasibility of a given candidate solution point. If the lower-level problems (LLPs) of the (G)SIP, MINMAX, or BLP are convex and satisfy certain regularity conditions, the hierarchical program can be reformulated as a single-level problem. However, in many applications, the LLPs are nonconvex.

Multiple theoretical approaches have been developed over the past few decades for the global solution of hierarchical programs without convexity assumptions, with substantial recent advances. These include discretization methods, adaptive discretization-based approaches, and adaptations thereof, overestimation methods using interval methods and relaxation methods, optimal value function approaches, and relaxation-based Branch & Bound methods (Bhattacharjee et al., 2005a,b, Blankenship and Falk, 1976, Djelassi and Mitsos, 2017, Falk and Hoffman, 1977, Floudas and Stein, 2008, Lo Bianco and Piazzi, 2001, Marendet et al., 2020, Mitsos, 2011, Mitsos and Tsoukalas, 2015, Seidel and Küfer, 2022, Stein and Steuermann, 2012, Tsoukalas and Rustem, 2011). See Djelassi et al. (2021) for more details. However, rigorous numerical performance comparisons between these algorithms are lacking. Indeed, if numerical comparisons are provided at all, they typically compare a small selection of algorithms on small test sets, across different platforms, and with various subsolvers required during the solution. Additionally, some algorithms have parameters that impede a fair comparison.

In this dissertation, we investigate the complementary approaches described above: BO, and OED for GPE. Specifically, we investigate how global optimization can be leveraged for the solution of BO, and hierarchical optimization methods in the context of OED for GPE problems. Furthermore, we advance hierarchical optimization methods, which we need for OED for GPE.

The main contributions encompass four aspects:

- We investigate the advantages and disadvantages of using a *deterministic global* solver compared to conventional *local* and *stochastic global* solvers in the context of BO using a GP as the surrogate model.
- We present an open-source software that implements multiple adaptive discretization-based solvers, comes with an extensive library of test problems, and we use this software to compare solvers numerically and to identify tradeoffs in the parameter tuning.
- We extend the formulation and solution procedure of OED for GPE to systems whose input-output relation is implicitly given by an equation system with additional semi-infinite constraints.
- We examine the underlying assumptions of an adaptive discretization-based algorithm for SIPs and give sharper, slightly relaxed, assumptions, widening the applicability of the investigated, and possibly conceptually related algorithms.

We are thus advancing BO, OED for GPE, and hierarchical optimization. In the following outline of the dissertation, we will detail these advancements and highlight the research gaps addressed in each respective chapter.

In Chapter 2, we investigate in the context of BO the advantages and disadvantages of using a *deterministic global* solver compared to conventional *local* and *stochastic global* solvers for the optimization of the acquisition function. Although the acquisition function often exhibits local minima and its global optimization has been stated to be important, local solvers are commonly used for optimization in the literature (Kim and Choi, 2021, Wilson et al., 2018). For our investigation, we perform repeated numerical experiments using four benchmark function, utilizing the lower confidence bound acquisition function.

In Chapter 3, we introduce the hierarchical optimization programs (G)SIPs, MINMAX, and BLPs. Recall that while multiple adaptive discretization-based algorithms have been published for these problem classes in recent decades, rigorous numerical performance comparisons between these algorithms are insufficient. We remedy this by our open-source software libDIPS, which enables users to flexibly switch between (i) the implemented solvers within a single problem class and (ii) between the employed subsolver. Note that libDIPS was already used in a preliminary form in Djelassi (2020). We advance comparability by compiling a benchmark library of over 600 problem instances. Our set includes eight merged test sets and additional problem instances from over 80 literature sources, which we use to compare the implemented solvers numerically and to identify tradeoffs in the parameter tuning.

In Chapter 4, we present an adaptation of the adaptive discretization-based approach of Tsoukalas et al. (2009) for BLPs. We utilize the infrastructure of libDIPS to implement our adaptation and the original approach, and show that our approach outperforms our implementation of the original approach. This showcases that (i) libDIPS serves as a development environment for implementing new and existing discretization-based algorithms and (ii) there is likely some hidden potential in the other algorithms.

In Chapter 5, we consider OED for GPE and extend the method so that it can be applied to liquid-liquid equilibrium (LLE) characterization using excess models and rigorous

thermodynamics for the computation of the LLE. For computing LLEs, often excess Gibbs free energy models are used because they are highly flexible. However, spurious phase splits can be predicted using these models due to (i) using non-rigorous computations, or (ii) parameter values of the excess model that are *problematic* in the sense that they lead to erroneous model behavior. To prevent these problems from affecting the experimental design, we (i) employ rigorous calculations of the LLE by using the Baker's criterion (Baker et al., 1982), and (ii) enforce additional requirements, based on Mitsos et al. (2009b), to avoid *problematic* parameter values that lead to erroneous model behavior. We solve the resulting OED problem using (generalized) semi-infinite programming techniques and provide a proof-of-concept using the Redlich-Kister model to plan the OED to reduce parameter uncertainty. Our approach successfully avoids wrong predictions due to *problematic* parameter values and computes an experimental design that significantly reduces the predicted parameter uncertainty.

In Chapter 6, we examine the lower-bounding procedure of the discretization-based algorithm by Mitsos (2011) for SIPs. However, unlike Mitsos (2011) and other earlier works, which rely on the assumption of compact host sets, we investigate whether we can relax the assumption of compact host sets. This is of particular interest because in our OED for GPE application in Chapter 5 and in other applicadtions, the assumption of compact host sets is sometimes not given. We show that indeed convergence problems can arise when considering SIPs with non-compact host sets. To mitigate these convergence problems, we give sharper, slightly relaxed assumptions with which we achieve the same convergence guarantees. However, these sharpened assumptions may be difficult to prove *a priori*. For these cases, we propose additional, stricter assumptions that might be easier to prove and that imply our sharpened assumptions. Finally, we review exemplarily which applications are tractable with the proposed additional assumptions. It is noteworthy that the considered procedure and assumptions are essentially the same in the broad class of adaptive discretization-based algorithms which we reviewed in Chapter 3. Hence, the results are likely transferable to other discretization-based algorithms for hierarchical programs.

In Chapter 7, we conclude the dissertation with a summary of the main findings and an outline of potential directions for future research.

2 Merit of Deterministic Global Optimization of the Acquisition Function in Bayesian Optimization

In this chapter, we explore the advantages and disadvantages of using a deterministic global solver in the context of BO. We use an initial small dataset, simulating a real scenario where experiments are costly and use GPs as the surrogate model. We compare BO using a deterministic global solver against two conventional optimization solvers.

The chapter is organized as follows. In Section 2.1, we provide a brief introduction to the topic, and in Section 2.2, we compactly describe the employed solvers, BO termination criterion, and case study formulations. Section 2.3 presents our findings, detailing the performance of the various solvers for our baseline case study, the Müller-Brown potential, and three additional benchmark functions. Section 2.4 discusses these results, while Section 2.5 offers insights and recommendations on using global deterministic solvers in BO. Finally, we outline future research directions in Section 2.6.

2.1 Introduction

With the growing utility of BO, an array of software packages have been developed, c.f., Table A.2 in Appendix A.3. The limited-memory Broyden-Fletcher-Goldfarb-Shanno solver with box constraints (L-BFGS-B) (Zhu et al., 1997), a *local*, gradient-based optimization algorithm, and multi-start L-BFGS-B, a *stochastic global* optimization approach that initializes L-BFGS-B from multiple starting points, are among the most frequently used solvers to optimize the acquisition function. Many acquisition functions are inherently prone to *local minima*: if the original objective function is multimodal, this characteristic is reflected in the exploitative part of the acquisition function after sufficient samples; moreover, irrespectively of the original objective function, if the acquisition function is explorative, uncertain regions result in additional local/global minima. Thus, the global optimization of the acquisition function has been stated to be important for improving BO performance, i.e., locating a globally near-optimal solution of the black-box function in few iterations (Kim and Choi, 2021, Wilson et al., 2018). However, local solvers, such as L-BFGS-B or trust-region optimization methods (Cartis et al., 2022, Wu et al., 2023), are not guaranteed to converge to a global minimum. Other approaches that try to alleviate this issue include using more sophisticated stochastic global optimization methods, such as evolutionary optimization algorithms (Bradford et al., 2018, Hansen, 2006, Song et al., 2024); exploiting the potential additive structure of objective functions to partition and simplify optimization (Oh et al., 2018, Wang et al., 2018); maintaining multiple, simpler surrogate models and utilizing a multi-armed bandit strategy at each iteration (TuRBO) (Eriksson et al., 2019); and hierarchical partitioning of the domain before local optimization (Kawaguchi

et al., 2015). However, all these approaches do not guarantee convergence to a global minimum — or only guarantee it in an infinite amount of time (Rinnooy Kan and Timmer, 1987).

Using a *deterministic global* solver, as e.g., Branch & Bound algorithms (Land and Doig, 1960), BARON (Khajavirad and Sahinidis, 2018), SCIP (Bolusani et al., 2024), Gurobi (Gurobi Optimization, 2025), or MAiNGO (Bongartz et al., 2018), directly guarantees the avoidance of local minima. Among the available alternatives, DIRECT (Jones et al., 1993) is the most widely used deterministic global solver in BO frameworks. Still, its use in BO has been limited (Li et al., 2017), and its adoption remains constrained as it becomes computationally prohibitive in high-dimensional settings (Kim and Choi, 2021). Similarly, mixed-integer programming is rarely employed in BO with GP surrogates but has found application with non-Gaussian models, such as neural networks (Papalexopoulos et al., 2022) and decision trees (Thebelt et al., 2022).

Recent advances, such as piecewise-linear approximations for GP kernels, offer, in theory, global optimality guarantees in the BO context while significantly improving computational efficiency (Xie et al., 2025). The piecewise linear approximation of the GP kernel by Xie et al. (2025) leads to a corresponding mixed integer quadratic program (Piecewise-linear Kernel Mixed Integer Quadratic Programming (PK-MIQP)) for the optimization of the (approximated) acquisition function. Xie et al. (2025) prove that under suitable assumptions, (a) their approximated kernel mean (and variance) converges in the limit to the true mean (and variance) and (b) give probabilistic bounds on regret for the case where the approximation error is finite. Xie et al. (2025) demonstrate the benefit of their PK-MIQP formulation on synthetic functions, constrained benchmarks, and a hyperparameter tuning task and compare the results to various other gradient- and sampling-based methods. However, their theoretical guarantees do not hold for all kernels, as the kernel must fulfill a smoothness assumption.

This chapter explores the advantages and disadvantages of using a deterministic global solver in the context of BO without any aforementioned approximations. We use an initial small dataset, simulating a real scenario where experiments are costly. We compare BO using a deterministic global solver to the conventional approaches of using an informed local solver (ILS) and informed multi-start (IMS) to optimize the acquisition function. The term "*informed*" is added to signify that the solver's initialization is strategically selected, as detailed in Sections 2.2.2.1 and 2.2.2.2. For ILS, we use the L-BFGS-B solver, and for IMS, we use the same L-BFGS-B solver but introduce multiple restarts. For the deterministic global solver, we use the open-source solver MAiNGO (Bongartz et al., 2018), which employs MeLOn (Schweidtmann and Mitsos, 2019) as a submodule to integrate machine learning models (in our case, GPs) into optimization problems. The parallel computing capabilities of MAiNGO, in conjunction with its reduced space formulation of GPs and custom relaxations for various functions, make it a promising solver, shown to outperform alternatives (Schweidtmann et al., 2021) and, more importantly, make global optimization with GPs embedded tractable in more realistic settings. However, computational costs still remain a major challenge. To maintain an overall tractable computational time of our numerical experiments, we terminate each optimization run in MAiNGO, i.e., the current optimization of the acquisition function, after a predefined time limit, *assuming* that the current solution is indeed a globally near-optimal solution of the acquisition function (corroboration, to follow, Section 2.2.2.3).

For the illustrative examples provided in this chapter, we use `GPYTorch` GPs (as currently required for MeLOn) with the standard `BOTorch` v0.9.5 (Balandat et al., 2020) framework, which allows us to switch between various optimizers easily. We compare the points visited by BO when optimizing the acquisition function using ILS, IMS, and MAiNGO and aim to identify whether the number of black-box function evaluations before BO convergence is affected. The main contributions of this chapter are as follows:

- We explore the use of a deterministic global solver (MAiNGO) for optimizing the acquisition function in BO, extending the one-shot global optimization of GPs in Schweidtmann et al. (2021). Given sufficient (finite) time, MAiNGO guarantees finding an (approximate) global solution, but for CPU efficiency, we impose a time limit, *assuming* optimality from stabilized bounds; manual checks support our assumption (Section 2.2.2.3). Such (strict) time limits would be too restrictive in practice.
- We provide an in-depth analysis of (a) the statistical probability of convergence to a globally near-optimal solution of the black-box function and (b) the statistical distribution of iterations until BO convergence when using ILS, IMS, and MAiNGO, applied to the Müller-Brown potential, which we use as our baseline case study.
- We demonstrate how statistical methods can quantify variance in solver performance and analyze the trade-off between selecting a high-quality initial dataset and choosing the solver.
- We further validate our findings on BO using ILS, IMS, and MAiNGO with three alternative benchmark functions characterized by higher problem dimensionality and/or multiple global minima.

2.2 Methods

We omit a detailed introduction to BO with GPs and refer the reader to readily available literature, e.g., Frazier (2018), Garnett (2023), Mockus (1989), Rasmussen and Williams (2006) and the references therein. Instead, we focus on the most essential aspects of BO with GPs for our investigation: the acquisition function, the solvers employed, and the termination criterion used.

2.2.1 Acquisition Functions

Common acquisition functions in BO include the following: probability of improvement, expected improvement, upper / lower confidence bound, entropy-based, Monte-Carlo-based, and feasibility-based functions (Gan et al., 2021, Wilson et al., 2018). We limit our study, to the popular (Gan et al., 2021) lower confidence bound acquisition function, defined as

$$LCB = \mu - \kappa\sigma \tag{LCB}$$

The hyperparameter κ controls the amount of exploitation vs. exploration: a low value exploits the model by decreasing the relative weight on the mean μ, whereas a high value explores by increasing the weight on the standard deviation σ. Prior works (Kandasamy

et al., 2015, Srinivas et al., 2012) have established cumulative regret bounds by dynamically adjusting κ over iterations, ensuring a balance between exploration and exploitation. However, in this study, we mainly study a fixed κ, the default in many BO frameworks (e.g., `BOTorch`). We aim to minimize (LCB) with each BO iteration.

2.2.2 Optimizers

In the following subsections, we review the optimizers which we will us in our case studies in Section 2.2.4.

2.2.2.1 Informed Local Solver

We utilize the standard L-BFGS-B solver, available in `scipy.optimize`, for our ILS. The term "informed" is added to signify that the solver's initialization is strategically selected. Solver initialization plays an important role for local optimization solvers. For nonconvex problems, the initial point significantly impacts the solver's ability to find a global optimum. Commercial local solvers and optimization experts typically implement an initialization strategy. We opt for the standard selection process for solver initialization used in `BOTorch` v0.9.5 (Balandat et al., 2020), c.f., `botorch/optim/initializers.py` on GitHub: First, 20 candidates are drawn using a Sobol sequence (Sobol', 1967) within the search space. From these 20 candidates, a single one is drawn with a probability proportional to its acquisition function value, normalized by the mean and standard deviation of the acquisition values across all candidates. Then, the selected candidate is used as the initial point for the L-BFGS-B solver. This approach prioritizes initialization from a candidate with a better (lower) acquisition function value.

2.2.2.2 Stochastic Global Solver

For stochastic global optimization, we use ILS, as described in Section 2.2.2.1, but introduce multiple restarts to further explore the search space. We perform five independent restarts of ILS, where, within each restart, the previously described selection process is used for the initialization of the employed L-BFGS-B solver. This IMS procedure aims to avoid local minima, though it does not guarantee that convergence to a global minimum will be achieved in finite time.

2.2.2.3 Deterministic Global Solver

We employ the open-source deterministic global solver MAiNGO v0.8.1 (Bongartz et al., 2018), which leverages reduced-space formulations of GPs through MeLOn (Schweidtmann and Mitsos, 2019) and custom relaxations of the acquisition function (implemented in MC++ (Chachuat et al., 2015)). Recall that this configuration offers drastic speed-ups of the optimization process compared to full-space formulations with standard deterministic global solvers (Schweidtmann et al., 2021), making BO in realistic settings tractable. Note that for the investigated acquisition function, i.e., (LCB), no custom relaxation is implemented in MAiNGO.

We configure MAiNGO to use CPLEX version 22.1.1 (International Business Machines Corporation, 2022) as the lower bounding solver, SLSQP as the upper bounding solver, and IPOPT (Wächter and Biegler, 2006) for multi-start. To reduce the CPU time, we terminate

each optimization run in MAiNGO, i.e., the current iterate, after 30 min, *assuming* that the current solution is indeed a globally near-optimal solution of the acquisition function. This assumption indicates that we have obtained a globally near-optimal solution through the upper-bounding procedure, but the lower-bounding procedure has not yet (fully) confirmed the solution. The observation that in these cases, the upper bound (UBD) remains unchanged while the lower bound (LBD) continues to increase, further corroborates the assumption. We validate and confirm our assumption by random manual inspection of the acquisition function. If we terminate an iterate and that iterate has a remaining relative optimal gap greater than 1 order of magnitude higher than the set relative optimality tolerance `epsilonR` in MAiNGO (e.g., the true gap is ≥ 0.1 when `epsilonR` $= 0.01$), we loosen `epsilonR` by one order of magnitude for subsequent iterates (e.g., now `epsilonR` $= 0.1$). Within each BO run, we start with `epsilonR` $= 0.01$. All other MAiNGO settings have been left at the default values.

2.2.3 Bayesian Optimization Termination Criterion

An important consideration, but still an open question, is when to appropriately terminate the outer-loop optimization process. The most common approach is a resource-based termination, where a fixed budget — typically defined by the total number of iterations or the total cost of experiments — is prespecified. While this approach does not guarantee convergence to a minimum, it is straightforward to implement and practical in many applications.

Another category of termination criteria evaluates the progress of the BO. A commonly used metric, especially used in benchmarking, is a simple regret, defined as $|f^* - f^t| < \varepsilon$, with f^* being the globally optimal objective value of the black-box function f, and f^t being the objective value for the current iterate $\boldsymbol{x}^t$. However, this method requires prior knowledge of the globally optimal objective value: we refrain from using such knowledge-dependent criteria to ensure the study remains both practical and realistic (though we provide such simple regret plots in Appendix A.4).

A more general alternative to simple regret is the expected minimum simple regret (Ishibashi et al., 2023), which removes the need for prior knowledge of the globally optimal objective value. Other progress-based approaches include monitoring local regret within a convex region (Li et al., 2023), detecting stagnation in the best-observed minimum over a specified number of iterations, or terminating when the expected improvement or probability of improvement falls below a threshold (Nguyen et al., 2017). Probabilistic, confidence-based measures have also been proposed in the literature (Bolton et al., 2004, Dai et al., 2019b, Makarova et al., 2022, Wilson, 2024).

Herein, we adapt a simple and practical progress-based termination criterion. Specifically, BO is terminated when either the distance between the current candidate $\boldsymbol{x}^t$ and any prior candidate $\boldsymbol{x}^k$ falls below the predefined threshold $\varepsilon^{x,1}$; or again when $\boldsymbol{x}^t$ and any prior candidate $\boldsymbol{x}^k$, $k = \{1, ..., t-1\}$ falls below the predefined threshold $\varepsilon^{x,2}$ in conjunction with the objective function value of the current candidate f^t being close to the best-found

solution $f^{*,t-1} = \min_{k=\{1,..,t-1\}} f^k$:

$$\underbrace{\left(\min_{k=\{1,..,t-1\}} \left\|\boldsymbol{x}^t - \boldsymbol{x}^k\right\| < \varepsilon^{x,1}\right)}_{\text{TC-1}} \vee \left(\underbrace{\min_{k=\{1,..,t-1\}} \left\|\boldsymbol{x}^t - \boldsymbol{x}^k\right\| < \varepsilon^{x,2}}_{\text{TC-2.1}} \wedge \underbrace{\left(\left|f^t - f^{*,t-1}\right| < \varepsilon^{\mathrm{f,r}} \cdot f^{*,t-1} \vee \left|f^t - f^{*,t-1}\right| < \varepsilon^{\mathrm{f,a}}\right)}_{\text{TC-2.2}}\right), \tag{TC}$$

with $\varepsilon^{x,1} < \varepsilon^{x,2}$ and $\varepsilon^{\mathrm{f,r}},\ \varepsilon^{\mathrm{f,a}} > 0$. The key idea behind this easy-to-implement termination criterion is that for systems without any (measurement) noise, as considered in our investigation, we would not repeat the same experiment multiple times (the candidate is in close proximity to another one; TC-1 in (TC)). Furthermore, we would terminate once the candidate is in proximity to another one, and the current iterate is comparable (in terms of objective value) to the best-found objective function value (TC-2.1 and TC-2.2 in (TC), respectively). Future work should explore the effects of alternative stopping criteria.

2.2.4 Case Studies

In the following section, we briefly review the benchmark functions that we use in our investigation. We choose the Müller-Brown potential function (Section 2.2.4.1) as our baseline benchmark problem to evaluate the performance of ILS, IMS, and MAiNGO within BO. Using this function as a baseline benchmark problem, we aim to answer the following questions:

1. How does the choice of solver affect successful convergence to a globally near-optimal solution in the BO setting, and to what extent is that success dependent on the initial dataset?

2. Given that we chose our initial dataset well enough (discussed below), which solver converges to a globally near-optimal solution in the BO setting in the fewest iterations?

In Section 2.2.4.2, we introduce three additional benchmark functions characterized by higher dimensions or multiple global minima, which we use to validate our findings on BO using ILS, IMS, and MAiNGO.

2.2.4.1 Müller-Brown Potential

The canonical Müller-Brown potential (Müller and Brown, 1979) is defined as

$$U(\boldsymbol{x}) = \sum_{i=1}^{4} A_i \exp\Big(a_i\big(x_1 - w_{0i}^1\big)^2 + b_i\big(x_1 - w_{0i}^1\big)\big(x_2 - w_{0i}^2\big) + c_i\big(x_2 - w_{0i}^2\big)^2\Big), \tag{2.1}$$

with the coefficients defined as in Table 2.1. In our considered domain of $x_1 \in [-1.5, 1], x_2 \in [-0.5, 2]$, U has two local minima and one global minimum. The surrogate model used in BO is a GP with Matérn kernel ($\nu = 5/2$) and marginal log-likelihood. The lengthscale

hyperparameters are locally optimized with each BO iteration, and we assume the data points have no associated noise. Input features are scaled to be between 0 and 1, and the output is scaled to have (approximately) zero mean and unit variance. We optimize (LCB) with each BO iteration.

Table 2.1: Coefficients of the Müller-Brown potential.

i	A_i	a_i	b_i	c_i	w_{0i}^1	w_{0i}^2
1	-200	-1	0	-10	1	0
2	-100	-1	0	-10	0	0.5
3	-170	-6.5	11	-6.5	-0.5	1.5
4	15.0	0.7	0.6	0.7	-1	1

In the baseline case study of the Müller-Brown potential, $\kappa = 2$ is selected for the acquisition function to balance exploitation and exploration. This value falls between two proposed scheduling approaches. Srinivas et al. (2012) proposes a schedule that provides cumulative regret guarantees. The schedule is defined as:

$$\kappa_{t,\mathrm{S}} = \sqrt{2 \log \left(\frac{M\, t^2\, \pi^2}{6\, \delta} \right)}, \tag{$\kappa_{t,\mathrm{S}}$}$$

with M being the (finite) number of points the search domain is discretized into and δ a small positive number. Srinivas et al. (2012) note that the schedule may be too conservative (exploratory) in practice and, in their experiments, scale κ down by a factor of $\sqrt{5}$ for improved performance. For $\delta = 0.1$, $M = 1 \times 10^6$ (near infinite), and after scaling down by $\sqrt{5}$, this would start around $\kappa_{1,\mathrm{S}} = 2.6$ and increase to $\kappa_{30,\mathrm{S}} = 3.1$ over 30 iterations: we call this schedule $\kappa_{t,\mathrm{S}}$. Kandasamy et al. (2015) also argue this is still too conservative (exploratory) in practice and offer a schedule starting at $\kappa_{1,\mathrm{K}} = 0.3$ and increasing to $\kappa_{30,\mathrm{K}} = 1.6$, following

$$\kappa_{t,\mathrm{K}} = \sqrt{0.2D \log (2t)}, \tag{$\kappa_{t,\mathrm{K}}$}$$

with D being the dimension of the search domain. Therefore, selecting $\kappa = 2$ for our baseline case study provides a reasonable initial estimate, particularly given that the optimal balance is typically unknown *a priori*.

Each experiment begins with three randomly chosen data points ($N = 3$), sampled using Latin hypercube sampling (LHS) (McKay et al., 1979). This approach mimics data collection following a real experimental setup, where only limited (but hopefully) well-distributed data are available at the start. We terminate BO using the previously introduced termination criterion (TC) with $\varepsilon^{x,1} = 0.001$, $\varepsilon^{x,2} = 0.05$, $\varepsilon^{\mathrm{f,r}} = 0.01$, and $\varepsilon^{\mathrm{f,a}} = 0.5$.

2.2.4.2 Additional Case Studies

We further evaluate the performance of BO using ILS, IMS, and MAiNGO on the 2D Camelback benchmark function, selected for its two global minima, along with the 3D Ackley and 4D Hartmann functions, both characterized by a single global minimum and increasing problem dimensionality. Exact formulations for each function are provided in Appendix A.1, with termination criterion thresholds detailed in Appendix A.2. We use the same GP setup as in our base study (Matérn kernel, marginal log-likelihood, no noise).

2.3 Results

A key distinction for BO across the different solvers lies in the reproducibility of a solution trajectory: BO with a deterministic global solver consistently produces a single, reproducible value for the number of iterations to convergence under fixed inputs ((a) initial dataset; (b) acquisition function; and (c) termination criterion; and no noise). In contrast, BO with ILS or with IMS exhibits variability in the number of iterations to convergence due to the stochastic initialization of the solvers (described in Sections 2.2.2.1 and 2.2.2.2), resulting in a distribution of iteration counts for repeated runs under identical input conditions. Therefore, we define a *case study* as characterized by the same underlying black-box function, acquisition function, and termination criterion. Within each *case study*, we conduct multiple *experiments*, each defined by a unique initial dataset (all datasets of the same size). A *run* refers to a single complete execution of the BO loop for a given *experiment*: we conduct multiple *runs* of each *experiment* for each *case study*.

To address our first research question of Section 2.2.4 (regarding *where* the "outer" BO trajectories converge), we use the following approach:

- In Section 2.3.1.1, we compare the trajectories of the BO iterates using MAiNGO, ILS, and IMS in a single run of our baseline case study (characterized by the Müller-Brown function (2.1), using (LCB) with $\kappa = 2$ as the acquisition function, and $N = 3$ initial data points). This representative single-experiment, *single-run* example allows us to highlight how the specific solvers directly influence the convergence behavior of the BO.

- In Section 2.3.1.2, we extend this to multiple runs of the *same experiment* to investigate solver variability under identical initial conditions, given the stochastic nature of ILS and IMS.

- We then proceed to investigate multiple runs of *multiple experiments*, still of the same baseline case study, in Section 2.3.1.3 to explore how the initial dataset influences the probability of convergence to a globally near-optimal solution.

- In Section 2.3.1.4, we perform statistical testing to quantify the influence of solver choice *and* initial dataset using our previously computed runs.

- Finally, in Section 2.3.1.5, we synthesize our findings to answer our first research question of Section 2.2.4.

We address our second research question of Section 2.2.4 (regarding *how fast* the "outer" BO trajectories reach a globally near-optimal solution) in Section 2.3.2, where we compare the BO runs (across all experiments) that successfully converged to a globally near-optimal solution for all three solvers. As our performance measure, we chose the number of iterations required to satisfy the defined termination criterion. For additional insight, particularly for readers working with a fixed iteration budget, we include simple regret plots in Appendix A.4. These plots illustrate the rate at which "outer" BO trajectories approach a global minimum, offering a complementary perspective on solver performance. Section 2.3.3 briefly presents additional case studies that explore different black-box functions to assess the generalizability of our findings.

2.3.1 Convergence to a Globally Near-Optimal Minimum

2.3.1.1 Demonstrative Single-Run Example: Analyzing Solver Influence in a Single Experiment

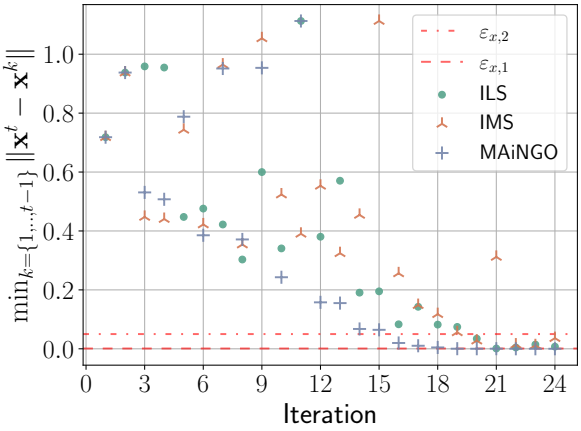
This section presents a representative single-run example to illustrate *how* and *why* different solvers influence which minimum BO converges to. Figure 2.1 visualizes the trajectories of the "outer" BO iterates using MAiNGO, ILS, and IMS in a single run of our baseline case study (characterized by the Müller-Brown case study (2.1) and using (LCB) with $\kappa = 2$ and $N = 3$ initial data points) for a single experiment. We reiterate that in this baseline case study, BO convergence is defined by (TC) with $\varepsilon^{x,1} = 0.001$, $\varepsilon^{x,2} = 0.05$, $\varepsilon^{\mathrm{f,r}} = 0.01$, and $\varepsilon^{\mathrm{f,a}} = 0.5$.

Figure 2.1a tracks the distance between each candidate point and all previously sampled points. Dashed red lines indicate the predefined threshold for the *distance-based* portions of our termination criterion. That is, our termination criterion is satisfied when either the bottom dashed line is reached (representing $\varepsilon^{x,1}$ in TC-1), or both the top dashed line is reached (representing $\varepsilon^{x,1}$ in TC-2.1) and the current candidate is close to the prior best point in Figure 2.1b (representing TC-2.2). In this run, all three BO trajectories converge to different minima: BO with IMS reaches a globally near-optimal solution of (2.1), while BO with ILS and BO with MAiNGO terminate at local minima, as shown in Figure 2.1b (thick gray bars denote the three (local and global) minima of (2.1)).

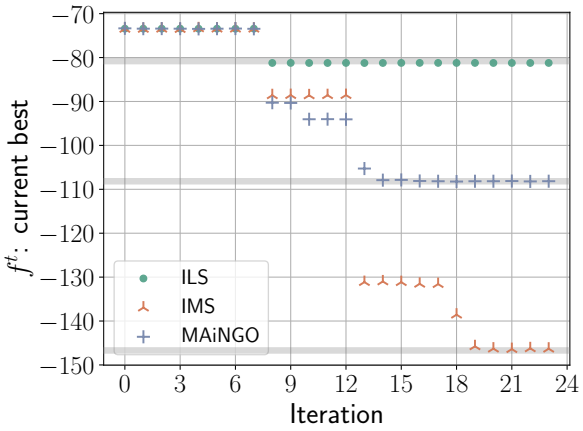
Unlike optimization *of the acquisition function* with a deterministic global solver, its optimization with ILS or IMS can result in locating one of its local minima. Locating a local minimum of the acquisition function could result in continued exploration of a previously uncharted region of the black-box function (this is also possible *after* the termination criterion is satisfied; c.f., Figure 2.1a). While this additional exploration increases the likelihood of reaching a globally near-optimal solution (of the "outer" BO loop), it comes at the cost of additional BO iterations (and potentially past our termination criteria). Moreover, repeated convergence to similar local minima can lead to indefinite iteration when relying solely on TC-2.1 and TC-2.2 in (TC). To mitigate indefinite iterations, safeguards such as TC-1 in (TC) enforce timely termination, though they risk premature stopping if the new candidate is very close to a previously sampled suboptimal point (c.f., Section 2.2.3).

2.3.1.2 Multiple Runs of the Same Experiment: Analyzing Solver Variability

We reiterate that, due to the stochastic initialization of IMS and ILS, repeating a single experiment can result in different BO trajectories when using these solvers. This section considers multiple runs of the same experiment of our baseline case study to demonstrate solver variability under identical initial conditions. Across 80 runs using (LCB) with $\kappa = 2$ and the same $N = 3$ initial data points (i.e., consistent with the setup in Section 2.3.1.1), BO utilizing MAiNGO never(!) finds a globally near-optimal solution of (2.1), while utilizing IMS and ILS will find it with a probability of 19 % and 29 %, respectively. The mean iterations to termination are 16.0 ± 0, 19.2 ± 5.1, and 21.2 ± 6.5 for BO with MAiNGO, IMS, and ILS, respectively ($\pm$ one standard deviation). Among the runs that converged to a globally near-optimal solution, the mean iterations increased to 25.8 ± 5.3 for BO with IMS and 29.3 ± 5.7 for BO with ILS.



(a) For this run, BO utilizing MAiNGO fulfills (TC) after 16 iterations, BO with ILS after 20, and BO with IMS also after 20. If left to run past fulfilling (TC), BO utilizing ILS continues to explore previously uncharted regions. Dashed red lines indicate the predefined threshold for the distance termination criteria.



(b) BO with ILS and with MAiNGO terminate at a local minimum, but BO with IMS finds a globally near-optimal solution. If left to run past fulfilling (TC), BO utilizing ILS also finds a globally near-optimal solution. Three thick gray horizontal lines mark the three minima.

Figure 2.1: BO trajectories for a single run in the baseline case study, with the black-box function defined by (2.1), an acquisition function using (LCB) with $\kappa = 2$, and an initial dataset of $N = 3$ points. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

2.3.1.3 Multiple Runs for Multiple Experiments: Analyzing Initial Dataset Influence

In Section 2.3.1.2, we observed that BO with IMS and BO with ILS, due to their stochastic nature, exhibit different convergence trajectories across runs of a single experiment, while BO with MAiNGO consistently failed to find a globally near-optimal solution. However, these findings are based on a fixed initial dataset, raising the question: *How does the probability of convergence to a globally near-optimal solution change when the initial dataset varies?*

To answer this question, we now investigate solver robustness across multiple experiments for our baseline case study using (LCB) with $\kappa = 2$ and $N = 3$ initial data points. All initial datasets were generated via LHS. Table 2.2 and Figure 2.2 summarize the probabilities of convergence to a globally near-optimal solution of (2.1). As shown in Figure 2.2, all solvers exhibit comparable convergence probabilities when the iteration count is capped at 30 iterations. However, when no iteration cap is imposed, BO with IMS and ILS demonstrate a higher probability of reaching a globally near-optimal solution than BO with MAiNGO. This finding suggests that BO with IMS and ILS often require more iterations than BO with MAiNGO to achieve convergence — a trend further explored in Section 2.3.2. The statistical significance of these probability differences is examined in Section 2.3.1.4.

Beyond solver selection, we found two key factors that influence the probability of convergence to a globally near-optimal solution of the black-box function, c.f., Table 2.2:

- Increasing the size of the initial dataset provides a more representative surrogate model, i.e., GPs, of the black-box function right from the start. This increases the likelihood of reaching a globally near-optimal solution of the black-box function.

- Increasing the exploration parameter κ in (LCB) also boosts the probability of convergence to a globally near-optimal solution across all solvers.

Table 2.2: Probability of convergence to a globally near-optimal solution of the Müller-Brown case study (2.1), aggregated over multiple experiments and runs. The κ schedule $\kappa_{t,\mathrm{S}}$ is more exploratory than the baseline of $\kappa = 2$; $\kappa_{t,\mathrm{K}}$ is more exploitative. Details on the number of experiments and runs, and termination criteria are provided in Appendix A.2.

κ	N	ILS	IMS	MAiNGO
2	3	0.79	0.80	0.70
3	3	0.94	0.97	0.96
2	10	0.83	0.83	0.81
2	20	0.92	0.94	0.90
$\kappa_{t,\mathrm{S}}$	3	0.92	0.98	1.0
$\kappa_{t,\mathrm{K}}$	3	0.53	0.53	0.51

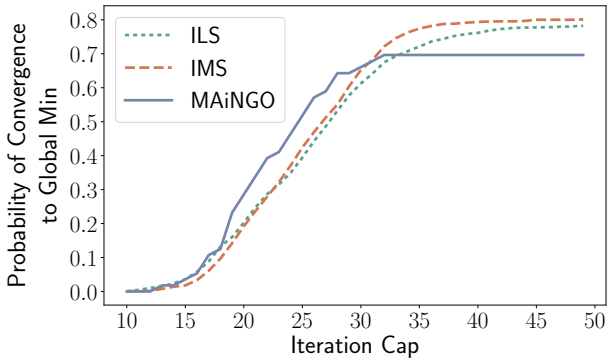


Figure 2.2: The probability of convergence to a globally near-optimal solution of the Müller-Brown case study (2.1) utilizing the specified solver is comparable among all solvers if the maximum number of iterations is limited to 30; beyond that, IMS dominates. The probability is computed as the fraction of runs that successfully converged (*and* converged to a globally near-optimal solution) within the given iteration cap (limit), over the total number of runs for each solver. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

2.3.1.4 A Statistical Look: Quantifying the Influence of the Solver and Initial Dataset

Here, we address statistically (a) whether the differences seen in Figure 2.2 between solvers are systematic or could be due to chance alone; and (b) how the influence of the solver compares to that of the initial dataset. Note that the statements are for a given case study (all characterized by the same black-box and acquisition functions), as opposed to across families of functions. We use the data of Section 2.3.1.3, which consists of multiple runs across multiple experiments (characterized by different initial datasets). To meet our objective, we must account for the statistical structure of the design of the data. Let $y^s_{dat,r}$ denote the indicator variable for whether solver s ultimately converges to a globally

near-optimal solution of the black-box function in run r of an experiment characterized by initial dataset *dat*. The design of the data is then represented statistically by the following structure:

1. Conditionally on a given dataset *dat*, the convergence behaviors are independent among different solvers;

2. Conditionally on a given dataset, indicators of $y^{\text{ILS}}_{dat,r=1}, ..., y^{\text{ILS}}_{dat,r=31}$, are independent Bernoulli given *dat*, with probability $p^{\text{ILS}}_{dat} = \mathbb{E}_{\text{over } r}\{y^{\text{ILS}}_{dat,r} \mid dat\}$; and similarly for using BO with IMS. For BO with MAiNGO, given *dat*, $y^{\text{MAiNGO}}_{dat,r=1}$ also identifies the probability p^{MAiNGO}_{dat}, which is, due to MAiNGO being a deterministic solver, 0 or 1.

3. For solver s, the convergence probabilities $p^{\text{s}}_{dat}, dat = 1, ...$ are independent and identically distributed; their average is denoted below as $p^{\text{s}} := \mathbb{E}_{\text{over } dat}\{y^{\text{s}}_{dat}\}$.

Do Different Solvers Converge Equally Often, Averaging Over Datasets and Runs?

We assessed whether p^s differs across our three solvers (s = ILS, IMS, MAiNGO), by testing the hypothesis

$$p^{\text{ILS}} = p^{\text{MAiNGO}} = p^{\text{IMS}}. \tag{H}$$

To do this, we estimated the conditional logistic regression of the model that assumes

$$\begin{aligned} \text{odds } p^{\text{ILS}}_{dat}/\text{odds } p^{\text{MAiNGO}}_{dat} &= \exp(\alpha) \\ \text{odds } p^{\text{IMS}}_{dat}/\text{odds } p^{\text{MAiNGO}}_{dat} &= \exp(\beta), \end{aligned} \tag{2.2}$$

using conditional Maximum Likelihood Estimation (cMLE) (Andersen, 1970); where odds $(p) := p/(1-p)$. This model is often used to estimate odd ratios of probabilities between groups when these ratios are assumed to be common across the matching factor (initial datasets). Estimation of this model is also useful in our case because (a) it addresses the matching of solvers' runs on experiments (datasets); (b) if solvers are overall equivalent, i.e., (H) holds, then the cMLE estimates of α and β are known to center at zero (a proof is given in Georgiou et al. (2025a)) even if the *true* odds ratios in the left-hand sides of (2.2) differ arbitrarily across datasets.

The estimated average probabilities p^s are 80 % for BO using IMS, 79 % for ILS, and 70 % for MAiNGO ($p-test$ value = 0.036 for the comparison that all three are statistically equivalent). We answer our question — do different solvers converge equally often? — by pairwise comparisons, which suggest that the probability of convergence of BO with IMS or ILS are statistically different from BO using MAiNGO, whereas BO with IMS or ILS are not statistically significantly different from each other.

Comparing the Importance of the Initial Dataset with that of the Solver

To compare the influence of the initial dataset with that of the solver, we use minimax tests and refer the reader to the structure of the design described in Section 2.3.1.4. Consider a list with all possible subsets *ds*, of half (28) of the 56 datasets *dat*. For each specific such *ds*, and for each solver s, we calculate the empirical proportion $\hat{p}^s_{ds}$ of convergence over the runs and the datasets in *ds*. We ask the following questions.

1. **Question:** If we choose a solver s, and then "arbitrarily choose a subset ds from the above list," what is the worst that can happen, i.e., what is the subset ds that minimizes the convergence probability $\hat{p}^s_{ds}$? From an optimization perspective, this question can be formulated as a discrete minmax problem that reads $\max_s \min_{ds} \hat{p}^s_{ds}$.
   Answer: We calculated these minima as 68 % (ILS), 54 % (MAiNGO), and 69 % (IMS). So, if we can only choose the solver to be used in BO, we should choose IMS to maximize (over solvers) the worst possible convergence (over subsets given solver) and assure convergence is $\geq 69\,\%$.

2. **Question:** If we choose a subset ds, and "arbitrarily choose a solver," what is the worst that can happen, i.e., what is the solver that minimizes the convergence probability? Note that this questions corresponds to the discrete minmax problem $\max_{ds} \min_s \hat{p}^s_{ds}$.
   Answer: The minimum convergence probability over solvers for a given subset ds ranges from 54 % – 86 % over ds, and their maximum 86 % occurs at a particular dataset, say ds^*, when using, as it happens, again IMS. So, if we can only choose a subset ds, we should choose the subset ds^* that maximizes (over subsets ds) the worst possible convergence (over solvers given subset ds) and assure that we have convergence $\geq 86\,\%$.

This analysis suggests that the choice of the dataset is at least as important as the choice of the solver to achieve a high probability of convergence.

2.3.1.5 Reconciling Solver Influence and Dataset Dependence

We return to our first question from Section 2.3: How does the choice of solver affect successful convergence to a globally near-optimal solution, and to what extent is that success dependent on the initial dataset? Our analysis shows BO utilizing MAiNGO does not always converge to a globally near-optimal solution of the black-box function and can terminate at a local minimum. Among the solvers tested, BO using IMS or ILS demonstrates statistically significantly higher probabilities of finding a globally near-optimal solution than BO utilizing MAiNGO (c.f., Table 2.2 and cMLE results). Furthermore, the observed heterogeneity highlights that, while BO utilizing MAiNGO and IMS differ statistically in their performance (as compared to each other), the initial dataset plays a critical role in determining to which minimum (local or global) BO converges towards, as we have shown for this specific case study using minimax tests. It seems reasonable to postulate that convergence to a local minimum of the black-box function occurs when the initial dataset is not representative of the true objective function, leading to a poorly constructed initial GP surrogate, combined with an acquisition function that insufficiently favors exploration. BO with IMS or ILS can compensate for insufficient exploration and a poor initial surrogate through stochasticity; BO with MAiNGO cannot. In other words, suboptimal optimization of poorly chosen acquisition functions can be preferable to their optimal solution.

Incorporating stochastic elements in the optimization process parallels using a stochastic gradient descent solver in the training phase of, e.g., deep-learning (Zhou et al., 2019) and physics-informed neural networks (Markidis, 2021). In machine learning, stochastic gradient descent solvers are widely used to avoid or escape local minima of the loss function during training (Kleinberg et al., 2018). Therein, if only a local solver, such as L-BFGS-B,

is used in the training phase, the training is prone to termination at a local minimum (Markidis, 2021). Therefore, a hybrid approach is typically applied in the training phase of, e.g., physical-informed neural networks: First, a stochastic gradient descent solver is used to avoid (premature) termination at local minima as stochastic gradient descent solver are capable of "escaping" local minima (Kleinberg et al., 2018). Because stochastic gradient descent solvers converge fast to the vicinity of a global optimum, but the iterates then will stay around this global optimum with constant probability (Kleinberg et al., 2018), subsequently: Second, the solution is refined using a local solver, e.g., L-BFGS-B (Markidis, 2021).

It is important to note that these findings are based on our single baseline case study: various datasets consisting of three initial points and (LCB) with $\kappa = 2$ as the acquisition function. Significantly increasing the number of initial data points (making an exploitative acquisition function more appropriate) or increasing the exploration factor κ in (LCB) improves the probability of convergence to a globally near-optimal solution of the black-box function (Table 2.2). Unfortunately, selecting an appropriate exploration factor for (LCB) is challenging without prior domain knowledge, and increasing the number of initial samples is often constrained by physical, time, or cost limitations. These findings emphasize the importance of developing strategies for selecting (a) initial sampling points to construct a high-quality initial dataset *before* entering the BO loop (see, e.g., (Bull, 2011, Ren and Sweet, 2024, Wang et al., 2024)), (b) more suitable acquisition functions, and (c) methods to detect poorly performing acquisition functions early in the optimization process or "self-adaptive" acquisition functions (see, e.g., (Chen et al., 2023, Maitra, 2024, Yan et al., 2022)).

2.3.2 Iterations to Convergence

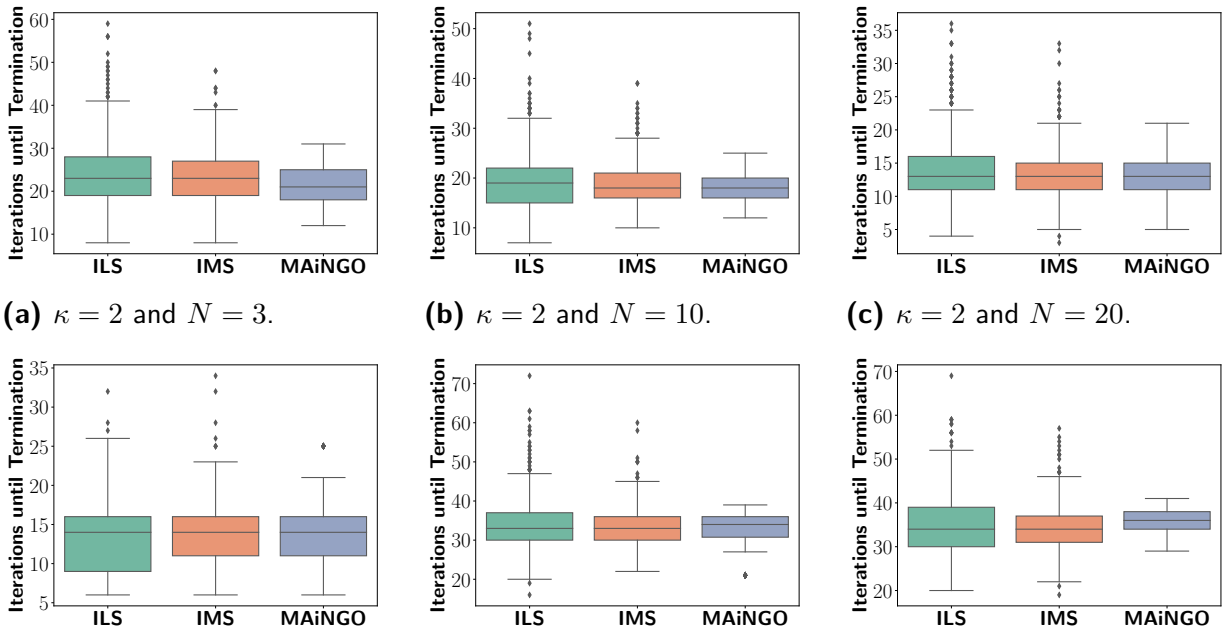
Having established how solver choice and initial dataset impact the probability of convergence to a globally near-optimal solution, we now investigate efficiency (question two of Section 2.3): Given a well-chosen initial dataset, which solver within BO converges, as defined by (TC), in the fewest iterations?

The following analysis focuses exclusively on runs that successfully reached a globally near-optimal solution, grouped by experiment (i.e., initial dataset). Experiments where any solver failed to achieve at least one successful run are excluded. This approach allows for a controlled comparison of iteration counts until convergence, assuming *a priori* knowledge of success while accounting for dataset effects. Including all runs — regardless of which minimum they reached — i.e., using a broader dataset, would introduce a bias: solvers with lower convergence probabilities tend to terminate earlier, requiring fewer iterations, while those more likely to reach a globally near-optimal solution explore further and take longer. For completeness, statistics with this broader dataset are provided in Appendix A.5.

In this restricted analysis, conducted for our baseline case study (using (LCB) with $\kappa = 2$ and $N = 3$ initial data points), BO utilizing MAiNGO required fewer iterations on average to satisfy (TC) compared to both BO utilizing IMS and BO utilizing ILS (Figure 2.3a). Statistical comparisons using one-sided t-tests (paired by experiments) confirm that this difference is significant, demonstrating MAiNGO's advantage in terms of mean BO iterations to convergence (c.f., Table 2.3). The same trends hold when the number of initial points in the dataset is increased to 10 or 20 (when a more exploitative acquisition function is appropriate), but not when κ is increased to 3 or $\kappa_{t,\mathrm{S}}$ is used (more

exploratory acquisition functions), c.f., Figures 2.3b and 2.3e and Table 2.3. These results are consistent with Xie et al. (2025), who demonstrate the effectiveness of PK-MIQP in exploitative scenarios with $N \geq 10$. Beyond achieving a lower mean number of iterations as compared to BO with ILS or IMS, BO utilizing MAiNGO also exhibits reduced variability by eliminating the stochasticity associated with the solvers. This finding is consistent across all case studies (all tested κ and initial dataset sizes).

For an alternative perspective, we present simple regret plots, which illustrate how quickly BO trajectories approach a globally near-optimal solution for each solver, in Appendix A.4: these plots reinforce the aforementioned findings.



(a) $\kappa = 2$ and $N = 3$. **(b)** $\kappa = 2$ and $N = 10$. **(c)** $\kappa = 2$ and $N = 20$.

(d) $\kappa = \kappa_{t,\mathrm{K}}$ and $N = 3$. **(e)** $\kappa = 3$ and $N = 3$. **(f)** $\kappa = \kappa_{t,\mathrm{S}}$ and $N = 3$.

Figure 2.3: Distribution of iteration counts for runs that successfully converged to a globally near-optimal solution, considering only experiments where all three solvers have at least one successful run. Results are shown for a series of Müller-Brown case studies. Details on the number of experiments, runs, and termination criteria are provided in Appendix A.2. For the κ schedules $\kappa_{t,\mathrm{S}}$ and $\kappa_{t,\mathrm{K}}$ refer to Section 2.2.4.1. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

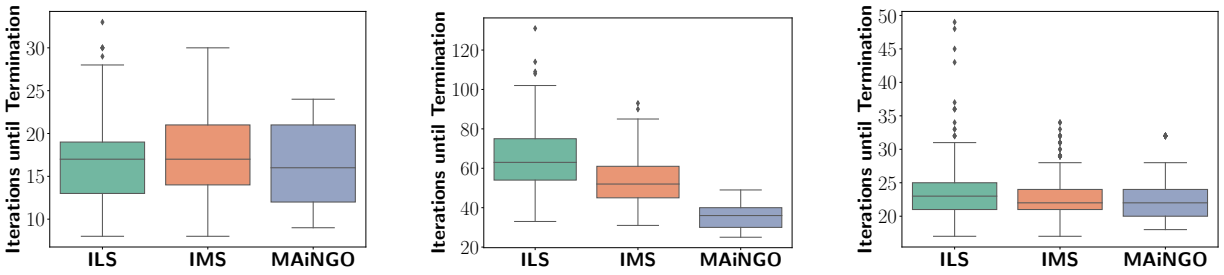
2.3.3 Additional Case Studies

We tested BO with ILS, IMS, and MAiNGO on three additional benchmark functions, i.e., 2D Six-Hump Camelback, 3D Ackley, and 4D Hartmann to evaluate generalizability. As in our baseline case, we use (LCB) as the acquisition function with $\kappa = 2$. Appendix A.1 provides the formulations for these functions, while Appendix A.2 outlines the termination thresholds, number of experiments, and runs per experiment. Figure 2.4 and Table 2.4 present the iterations to convergence distributions and statistical comparisons using one-sided t-tests, paired by experiment.

Table 2.3: Summary of convergence statistics for each case study, considering only runs that successfully reached a globally near-optimal solution. Experiments are included only if at least one run converged to a globally near-optimal solution for all three solvers. Number of runs (*count*), mean, median, and standard deviation of iterations to convergence are reported for this subset of data. One-sided t-tests, *paired by experiment*, evaluate whether MAiNGO converges in fewer iterations than IMS or ILS. MAiNGO converges in fewer iterations than ILS for all case studies and significantly fewer iterations than IMS for all but the case study using the exploratory acquisition function ($\kappa = 3$, $\kappa = \kappa_{t,\mathrm{S}}$). Details on the total number of experiments, runs, and termination criteria for each case study are provided in Appendix A.2. For the κ schedules $\kappa_{t,\mathrm{S}}$ and $\kappa_{t,\mathrm{K}}$ refer to Section 2.2.4.1.

κ	N	Solver	Count	Mean	Median	Std	t-test	p-value
2	3	ILS	1085	23.8	23.0	7.24	4.11	1.0×10^{-4}
2	3	IMS	1140	23.3	23.0	5.47	3.84	2.3×10^{-4}
2	3	MAiNGO	1209	21.3	21.0	4.50	-	-
3	3	ILS	703	34.3	33.0	6.43	1.22	1.2×10^{-1}
3	3	IMS	722	33.4	33.0	4.55	-0.56	7.1×10^{-1}
3	3	MAiNGO	744	33.2	34.0	4.14	-	-
2	10	ILS	738	19.3	19.0	5.78	2.48	1.0×10^{-2}
2	10	IMS	760	18.8	18.0	4.26	1.54	6.8×10^{-2}
2	10	MAiNGO	775	18.1	18.0	3.06	-	-
2	20	ILS	1093	13.8	13.0	4.91	2.89	3.2×10^{-3}
2	20	IMS	1138	13.1	13.0	3.71	0.47	3.2×10^{-1}
2	20	MAiNGO	1147	13.0	13.0	3.37	-	-
$\kappa_{t,\mathrm{S}}$	3	ILS	404	35.3	34.0	7.28	-0.73	7.6×10^{-1}
$\kappa_{t,\mathrm{S}}$	3	IMS	431	33.9	34.0	5.34	-2.21	9.8×10^{-1}
$\kappa_{t,\mathrm{S}}$	3	MAiNGO	441	35.4	36.0	3.46	-	-
$\kappa_{t,\mathrm{K}}$	3	ILS	496	13.3	14.0	4.57	0.28	3.9×10^{-1}
$\kappa_{t,\mathrm{K}}$	3	IMS	529	13.7	14.0	4.55	2.20	1.9×10^{-2}
$\kappa_{t,\mathrm{K}}$	3	MAiNGO	546	13.3	14.0	4.56	-	-

The 2D Six-Hump Camelback function has two global minima, though we did not differentiate between them. In the BO context, MAiNGO performed comparably to ILS on this function, while IMS required significantly more iterations to reach (TC). For higher-dimension functions such as the 3D Ackley and 4D Hartmann, BO *with MAiNGO consistently converged in significantly fewer iterations* than BO with ILS or IMS, *suggesting* BO *utilizing* ILS *and* IMS *are more prone to getting stuck in local optima as dimensionality increases*. Results for the 4D Hartmann function again highlight the importance of initial dataset quality; BO utilizing MAiNGO, as compared to BO utilizing ILS or IMS, was the least likely to converge to a globally near-optimal solution if the initial dataset (or acquisition function and κ value) was poorly selected.



(a) 2D Six-Hump Camelback **(b)** 3D Ackley **(c)** 4D Hartmann

Figure 2.4: Distribution of iteration counts for runs that successfully converged to a globally near-optimal solution, considering only experiments where all three solvers have at least one successful run. Results are shown for a series of case studies growing in dimensions. Details on the number of experiments, runs, termination criteria, κ values, and number of points in the initial dataset are provided in Appendix A.2. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

Table 2.4: Summary of convergence statistics for various objective function case studies. The probability of convergence to a globally near-optimal solution (*prob*) is computed over all experiments and runs. The remaining statistics—the number of successful runs (*count*), and for those runs, the mean, median, and standard deviation of iterations to convergence—are reported only for experiments where at least one run successfully converged to a globally near-optimal solution for all three solvers. This approach controls for dataset effects. One-sided t-tests, *paired by experiment*, evaluate whether BO with MAiNGO converges in fewer iterations than BO with IMS or ILS. BO with MAiNGO converges in fewer iterations than with IMS for all case studies, and significantly fewer iterations than with ILS for all but the 2D Camelback. Details on the total number of experiments, runs, and termination criteria for each case study are provided in Appendix A.2.

Test Function	Solver	Prob	Count	Mean	Median	Std	t-test	p-value
2D Camelback	ILS	1	1271	16.5	17.0	3.99	-0.05	5.20×10^{-1}
	IMS	1	1271	17.6	17.0	4.65	2.23	1.60×10^{-2}
	MAiNGO	1	1271	16.5	16.0	4.60	-	-
3D Ackley	ILS	1	465	65.0	63.0	15.33	38.6	6.68×10^{-19}
	IMS	1	465	53.5	52.0	11.1	30.8	4.34×10^{-17}
	MAiNGO	1	465	35.3	36.0	6.31	-	-
4D Hartmann	ILS	0.80	334	23.5	23.0	4.17	2.31	1.60×10^{-2}
	IMS	0.74	350	23.1	22.0	3.50	2.95	3.90×10^{-3}
	MAiNGO	0.73	352	22.5	22.0	3.34	-	-

2.4 Discussion

In this chapter, we investigated the advantages and limitations of employing the *deterministic global* solver MAiNGO for optimizing the acquisition function in BO, comparing its performance to that of the more conventional stochastic solvers used: IMS, i.e., a *stochastic global* solver, and ILS. For our investigation, we used the Müller-Brown potential function as our baseline case study with the lower confidence bound (LCB) acquisition function under varying κ.

The most important consideration in BO is arguably the probability of convergence to a *global*ly near-optimal solution of the black-box function. We analyzed this critereon using (LCB) and varying initial datasets for optimizing the Müller-Brown potential. Our findings reveal that, in *exploitative* settings, BO using MAiNGO (compared to BO using ILS or IMS) is more prone to terminating at a *local* minimum of the black-box function when the acquisition function is insufficiently exploratory and especially when the initial dataset lacks diversity. In contrast, BO using IMS and ILS have a higher probability of identifying a globally near-optimal solution under these conditions than BO using MAiNGO. Increasing κ (the exploration parameter of (LCB)) or the size and diversity of the initial dataset boosts convergence to a globally near-optimal solution across all solvers. We also demonstrated the utility of statistical methods (e.g., conditional logistic regression, minimax testing, and paired t-tests) for quantifying the trade-offs between selecting a high-quality initial dataset and choosing an appropriate solver while accounting for the stochasticity introduced by the solver initialization.

The second most important consideration is the number of iterations needed to reach the termination criterion (TC), i.e., the number of experiments to be performed, especially when experiments are costly. Our findings reveal that, in *exploitative* settings and given that BO converges to a globally near-optimal solution, BO using MAiNGO generally converges in fewer iterations than BO using ILS or IMS. We found that when the acquisition function is more *exploratory*, BO using MAiNGO often requires a similar, if not greater, number of iterations to satisfy (TC) than BO with ILS or IMS.

We extended our analysis (of the probability of converging to a globally near-optimal solution and the number of iterations to reach the termination criterion (TC)) to three additional benchmark functions, suggesting that these trends are generalizable. Additionally, our results indicate that MAiNGO may be rather advantageous in the BO setting with *increasing dimensionality of the black-box function*. Arguably, this apparent advantage in higher-dimensional problems is the most suggestive finding.

Beyond the aforementioned considerations, the computational cost per BO iteration is also a key consideration in real-world applications. In our investigation, MAiNGO generally required more time per BO iteration than IMS or ILS, particularly when the number of optimization variables was large or the problem relaxations were weak (resulting in weak LBDs and, as such, very slow convergence). Some BO iterations in MAiNGO took seconds, while other iterations exceeded the prespecified time limit for MAiNGO, necessitating an adjustment of optimality tolerances. As a test, we removed the prespecified time limit in MAiNGO. Removing this time limit led to some iterations taking as much as 24 h. This highlights a trade-off: using deterministic global solvers in the BO context can reduce the number of iterations needed for convergence but may require more computation time per iteration. Notably, the benefits (fewer iterations) and drawbacks (longer per-iteration times) scale with the dimensionality of the black-box function. In applications involving

costly experiments, allowing a single MAiNGO iteration to run overnight (or even substantially longer) is warranted, given its advantage of minimizing the total number of BO iterations compared to BO with ILS or IMS. Significant progress has already been made in reducing computation times for optimization with GPs embedded through advancements in reduced-space formulations (as opposed to conventional full-spaced formulations) (Schweidtmann et al., 2021) and specialized Branch & Bound algorithms (Tang et al., 2024). These developments have made deterministic global optimization for BO increasingly computationally feasible, positioning this contribution among the first to explore its potential in BO. However, further reductions in MAiNGO's computation time are still needed to fully realize this potential (see Section 2.6).

2.5 Recommendations

Based on the four functions considered, we discuss some practical considerations. Note that our following recommendations depend on the perspective of the decision maker. A practitioner seeking a good solution may prioritize efficiency over rigor and be satisfied with finding a local minimum, while another may instead value rigor and finding a globally near-optimal solution. For those seeking rigor, it is important to note that BO inherently lacks complete precision due to potential non-representative initial data and suboptimal acquisition function hyperparameters. Considering these perspectives and the findings of Section 2.4, we provide the following heuristic recommendations:

- **Prioritize initial dataset quality, especially when using deterministic global solvers in** BO**.** A poorly representative initial dataset can produce a misleading initial surrogate model, reducing the effectiveness of BO in finding a globally near-optimal solution of the black-box function. While this issue affects all solvers, it is particularly problematic for deterministic global solvers like MAiNGO when paired with insufficiently exploratory acquisition functions. Our statistical analysis on our baseline case study indicates that initial dataset selection *is at least as* important as solver choice with regard to the probability of convergence to a globally near-optimal solution. A poor initial dataset may also prolong convergence.

- **Consider leveraging deterministic global solvers like MAiNGO when well-tuned acquisition functions are available; otherwise, if tuning is not possible, favor stochastic solvers.** BO using MAiNGO excels when the acquisition function is well-calibrated for the task (e.g., exploitative or exploratory) and when the initial surrogate GP is representative of the black-box function. BO with MAiNGO reliably follows the behavior dictated by the acquisition function. For example, if the decision maker envisions highly exploitative behavior (inscribed into the (LCB) acquisition function with a low value of κ), BO using MAiNGO is indeed very exploitative. This is not the case for BO using ILS or IMS, as these solvers might only report a local solution of the acquisition function, deviating from the intended (highly exploitative) behavior. This stochastic damping, caused by ILS or IMS, reduces the direct impact of acquisition function hyperparameters, which can be beneficial when the initial dataset is suboptimal or the acquisition function is insufficiently exploratory. Stochasticity can increase the likelihood of finding a globally near-optimal solution by facilitating the exploration of previously uncharted regions: Bull (2011), de Ath

et al. (2021) introduce further stochasticity by incorporating ϵ-greedy strategies into the BO framework. Parallels exist in machine learning, where stochastic gradient descent solvers are used to "escape" local minima (Markidis, 2021) to further explore previously uncharted regions.

- **If the goal is to find *any* local minimum of the black-box function, consider pairing an exploitative acquisition function with a deterministic global solver, even in the absence of well-tuned (hyperparameters of) acquisition functions or diverse initial datasets.** The strict adherence of deterministic global solvers like MAiNGO to the acquisition function enables rapid exploitation of known regions, minimizing the total number of iterations when paired with exploitative acquisition functions. This makes it an efficient choice when locating a globally near-optimal solution is not necessary.

- **Use exploratory acquisition functions with deterministic global solvers judiciously.** In exploratory settings, BO with MAiNGO achieves comparable or better probabilities of convergence to a globally near-optimal solution but often requires more iterations compared to ILS or IMS. We postulate this is because MAiNGO strictly follows the acquisition function, while IMS and ILS may get trapped in local minima of the acquisition function, unintentionally exhibiting (different) exploratory behavior.

- **Utilize hybrid approaches.** A practical strategy may be to begin with a stochastic solver or exploratory acquisition function when the representativeness of the initial dataset is uncertain. As the dataset becomes sufficiently rich (i.e., when the associated GPs encompasses enough of the the black-box function), transitioning to an exploitative function with MAiNGO can capitalize on its efficiency in converging to a globally near-optimal solution. This approach contrasts with theoretical approaches in (Kandasamy et al., 2015, Srinivas et al., 2012), which advocate increased exploration over time. We hypothesize that the proposed approach could perform well in practice and leave its rigorous validation to future research. Our hypothesis is supported by the successful application of similar hybrid approaches in machine learning (Markidis, 2021), where in the training phase, first, a stochastic gradient descent solver is used to efficiently navigate the search space and obtain a globally near-optimal solution, and subsequently a local solver is employed to refine this solution.

Many of these recommendations require knowledge of what constitutes an exploratory versus exploitative acquisition function (and how to balance them in the absence of knowledge) and when an initial dataset is representative enough. These factors are challenging to determine *a priori* and often require informed judgment. Several works are actively addressing these challenges. Bull (2011), de Ath et al. (2021) balance exploitation and exploration by incorporating ϵ-greedy strategies into the BO framework; Chen et al. (2023), Maitra (2024), Yan et al. (2022) work on "self-adaptive" acquisition functions; and Bull (2011), Ren and Sweet (2024), Wang et al. (2024) provide insights on selecting the initial dataset.

2.6 Future Work

Our investigation showed that deterministic global solvers are effective in BO when the acquisition function is well-tuned and the initial dataset is diverse. Consequently, a promising direction for future research is the development of acquisition functions that can be dynamically tuned on-the-fly or tuned without requiring prior knowledge (Chen et al., 2023, Maitra, 2024, Yan et al., 2022). Additionally, determining whether an initial dataset is sufficiently diverse *before* beginning BO and devising sampling strategies to produce such diverse datasets (e.g., Bull (2011), Ren and Sweet (2024), Wang et al. (2024)) remain critical challenges: these challenges require a deeper understanding of what constitutes a "diverse-enough" dataset for initial reliable surrogate modeling. Advancing these areas can increase the probability of BO converging to a globally near-optimal solution, broadening the scenarios where deterministic global solvers are effective. The complementary strategy of starting with a stochastic solver and transitioning to MAiNGO also warrants further investigation.

In exploitative settings with a well-tuned acquisition function and diverse initial dataset, BO with MAiNGO can converge in significantly fewer iterations than stochastic solvers. However, we observed that computation times with MAiNGO could be considerable, necessitating setting time limits. Specifically, while the UBD in MAiNGO converges relatively quickly, the LBD does not, c.f., Section 2.2.2.3. This observation aligns with findings of Tang et al. (2024), who have shown that McCormick relaxations (McCormick, 1976), which are also used in MAiNGO for generating a LBD, and also αBB underestimators (Adjiman et al., 1996) may be weak when optimizing the posterior mean function for GPs (due to a similar structure of (LCB), its relaxation may also be weak). To address this limitation, Tang et al. (2024) proposed a spatial Branch & Bound algorithm for optimizing GPs posterior mean functions, achieving significant speed-ups. Future work should extend this approach to acquisition functions or develop stronger lower-bounding heuristics to optimize acquisition functions more efficiently. Improving LBD convergence would enhance the computational efficiency of MAiNGO, making it even more competitive for BO when appropriate.

To further validate deterministic global solvers in BO, more extensive benchmark testing and real-world experiments are required, particularly under noisy conditions, which were not considered in this study. Additionally, exploring deterministic global optimization for GPs hyperparameter optimization (e.g., during training), instead of the local solver currently used, could enhance BO performance. These research directions will collectively clarify the scenarios in which deterministic global solvers are most effective, advancing BO's capability to solve complex optimization problems in science and engineering.

In this chapter, we examined BO, an approach that is suitable for process optimization when no mechanistic model exists. In contrast, when mechanistic knowledge exists, model-based approaches are usually more advantageous for process optimization. However, the effectiveness of the process optimization in these cases naturally depends on the quality of the underlying mechanistic model. To reduce parametric uncertainty in the mechanistic model and thereby increase its reliability, we will investigate OED for GPE in Chapter 5. As we will encounter concepts and notions of (G)SIPs, MINMAX, and BLPs in the formulation and solution of the OED problem, we will first examine these programs and their solution approaches in the following chapters.

3 Discretization-Based Semi-Infinite and Bilevel Programming Solvers

In this chapter, we lay the foundation for Chapters 4 to 6 by introducing the notions and concepts of (G)SIPs, MINMAX, and BLPs. After providing background information in Section 3.1 on the program classes (G)SIPs, MINMAX, and BLPs and existing solution approaches, we briefly review in Section 3.2 the formulation of the considered program classes. In Section 3.3, we recapitulate the main ideas of adaptive discretization-based algorithms for solving these programs. We then introduce in Section 3.4 an open-source C++ library, called libDIPS, which contains implementations of the algorithms reviewed in Section 3.3. In Section 3.5, we compile a library of test problems consisting of unified existing test sets and other test problems found in the literature. Section 3.6 covers performance tests. We use our library of test problems to evaluate the overall performance of the implemented solvers under "best-case" parameter settings. In addition, we analyze how sensitive each solver is to changes in its parameters. We summarize our work of this chapter in Section 3.7 and propose future work.

3.1 Background and Motivation

In this chapter, we focus on deterministic hierarchical programs, namely (G)SIPs, MINMAX, and BLPs, excluding probabilistic problem formulations. We limit ourselves to adaptive discretization-based approaches for the solution of these programs. A key benefit of these adaptive discretization-based approaches is that they can utilize well-established optimization solvers, primarily as a black box for solving subproblems. Moreover, (adaptive) discretization-based algorithms are relatively general: they support integer variables at both levels (Djelassi et al., 2019, Mitsos, 2010) and, apart from any specific requirements imposed by the subsolvers, require mainly compact domains and continuous functions. Their practicality is further demonstrated by successful applications in both research and industry (Djelassi et al., 2018, Küfer et al., 2008). In recent decades, numerous adaptive discretization-based algorithms have been published to solve different program classes, as shown in Table 3.2.

Evaluating the relative performance of these algorithms is difficult because computational performance is usually assessed on different platforms and implementations. In addition, many algorithms use inherent algorithmic tuning parameters (in the following simlpy called parameters) that complicate direct comparison due to their substantial influence on performance. Finally, no unified set of test problems is consistently used across publications, so performance evaluation is typically based on an individual and small set of test problems. To remedy this, we implement multiple adaptive discretization-based algorithms within a coherent software framework and compile an extensive library of test problems for comparison.

While we focus on comparing the implemented adaptive discretization-based algorithms, these algorithms are already used as a point of comparison in numerical evaluations of other algorithms. Specifically, Cerulli et al. (2022), Kleniati and Adjiman (2014), Marendet et al. (2020) compared the numerical performance of their approaches against the algorithms proposed by Djelassi and Mitsos (2017), Mitsos (2011), Mitsos et al. (2008a). For their comparisons, Cerulli et al. (2022) used their own implementations of Mitsos (2011), Marendet et al. (2020) used the GAMS implementation provided by Djelassi and Mitsos (2017), and Kleniati and Adjiman (2014) used the data reported by Mitsos et al. (2008a). The additional effort needed to reimplement the respective adaptive discretization-based algorithm, especially without access to the commercially distributed subsolvers or the ability to conduct comparisons based solely on the original publication, may have deterred researchers from using it as a reference point. An alternative to reimplementation would be to use one of the publicly available solvers for hierarchical programs. Table 3.1 provides an overview, focusing on those solvers that support nonlinearity and nonconvexity at one of the levels. However, potential reasons preventing their use in a comparison are: limitations placed on the problem type at the upper or lower level, the lack of a deterministic global solution, or that the relevant problem class, i.e., (G)SIP or BLP, is not supported. Additionally, some solvers are commercially distributed or utilize proprietary dependencies.

We remedy this by implementing multiple adaptive discretization-based algorithms in a coherent open-source software called libDIPS – Discretization-based semi-Infinite and bilevel Programming Solvers. libDIPS provides solvers for the deterministic global solution of (G)SIPs, MINMAX, and BLPs with upper- and lower-levels of (MI)NLP type. Our software is not only freely available and user-friendly, but it also serves as an ideal framework for implementing both existing and new algorithms, particularly making the implementation of discretization-based algorithms straightforward (also see Chapters 4 and 5). Additionally, we compile an extensive library of test problems to compare the algorithms. The combination of libDIPS and the library of test problems arguably makes further comparisons between algorithms easier for other researchers and makes the solution of hierarchical programs more accessible. Finally, we compare the solvers numerically using our library of test problems and identify tradeoffs in the parameter tuning. In summary, we provide (i) a software framework for the solution of hierarchical programs, (ii) compile a library of problem instances, and use the former two to (iii) run comparative tests of adaptive discretization-based algorithms to identify tradeoffs in the parameter tuning.

Table 3.1: Overview of solvers with publicly available implementations for the (deterministic) hierarchical programs supported in libDIPS. We exclude solvers for the purely discrete case and focus on those that allow for nonlinearity or nonconvexity. The solvers by Auslender et al. (2015), Beykal et al. (2020) are published but not publicly available. *: Type describes the problem type ((MI)(N)LP, (MI)QP) of the upper-level problem (ULP) and LLP, respectively. †: Continuous upper-level variables should not appear in the lower level.

	SIP	GSIP	BLP	Det. Global	Type (upper-lower)*	Algorithm Type
EAGO (Wilhelm and Stuber, 2022)	✓			✓	MINLP-MINLP	Adaptive discretization
NSIPS (Vaz et al., 2004a,b)	✓				NLP-unconstrained NLP	Discretization, transcription to integral
SIP with QP-LL (Cerulli et al., 2022)	✓			✓	convex NLP-QP	Adaptive discretization and Dualization
IbexSIP (Marendet et al., 2020)	✓			✓	NLP-NLP	Interval Method with Branch & Bound
MibS (Tahernejad et al., 2020)			✓	✓	MILP-MILP†	Branch & Cut
Fischetti (Fischetti et al., 2017)			✓	✓	MILP-MILP†	Branch & Cut
BASBL (Kleniati and Adjiman, 2015, Paulavičius and Adjiman, 2020)			✓	✓	MINLP-MINLP	Branch & Sandwich
B-POP (Avraamidou and Pistikopoulos, 2019)			✓	✓	MIQP-MIQP	Multi-parametric
BiOpt (Zhou and Zemkoho, 2021)			✓		NLP-NLP	Semismooth Newton
Mixed-Integer Bilevel solver (Bomze et al., 2025)			✓		MIQP-QP†	No-good Cuts & Refinement
GAMS EMP (GAMS Development Corporation, 2019)			✓	✓	NLP-convex NLP	KKT-based MPEC
libDIPS (this work)	✓	✓	✓	✓	MINLP-MINLP	Adaptive discretization

3.2 The Problem Classes SIP, GSIP, MINMAX, and BLP

In the following, we state the problem formulations and the notation used for the problem classes (G)SIP, MINMAX, and BLP. Throughout this and the following chapters, we use bold font for vector-valued symbols and calligraphic font for sets. While we highlight important aspects of each problem class, we discuss only the problem classes and their formulations that are directly tractable with libDIPS. For example, the algorithm for *pessimistic* bilevel problems from Wiesemann et al. (2013) can be easily implemented in the discussed software framework; however, we do not discuss this problem class because the algorithm is currently not implemented. Furthermore, we focus on features related to the algorithmic ideas introduced in Section 3.3. Most problem classes are well-established; we thus provide only a brief description of the algorithms to lay the foundation for the subsequent discussion and refer to surveys for a more thorough discussion (Djelassi et al., 2021, Guerra-Vázquez et al., 2008, Reemtsen and Rückmann, 1998, Sinha et al., 2018).

SIPs are formulated as

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \forall\ \boldsymbol{y}\in\mathcal{Y}, \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y} \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}, \end{aligned} \tag{SIP}$$

the objective function $f : \mathbb{R}^{n_{\mathrm{x}}} \to \mathbb{R}$, the semi-infinite constraint function $\boldsymbol{g}^{\mathrm{u}} : \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{y}}} \to \mathbb{R}^{n_{\mathrm{gu}}}$, non-coupling upper-level equality and inequality constraints $\boldsymbol{v}^{\mathrm{iu}} : \mathbb{R}^{n_{\mathrm{x}}} \to \mathbb{R}^{n_{\mathrm{viu}}}$ and $\boldsymbol{v}^{\mathrm{eu}} : \mathbb{R}^{n_{\mathrm{x}}} \to \mathbb{R}^{n_{\mathrm{veu}}}$, non-coupling lower-level equality and inequality constraints $\boldsymbol{v}^{\mathrm{il}} : \mathbb{R}^{n_{\mathrm{y}}} \to \mathbb{R}^{n_{\mathrm{vil}}}$ and $\boldsymbol{v}^{\mathrm{el}} : \mathbb{R}^{n_{\mathrm{y}}} \to \mathbb{R}^{n_{\mathrm{vel}}}$. We denote the n_{x} upper-level variables by $\boldsymbol{x}$, which consist of $n_{\mathrm{x,c}}$ continuous and $n_{\mathrm{x,i}}$ integer variables. Similarly, the n_{y} lower-level variables $\boldsymbol{y}$ consist of $n_{\mathrm{y,c}}$ continuous and $n_{\mathrm{y,i}}$ integer variables. We define the host sets

$$\bar{\mathcal{X}} \coloneqq \left\{\left[\boldsymbol{x}^{\mathrm{lb}}, \boldsymbol{x}^{\mathrm{ub}}\right] \cap \left(\mathbb{R}^{\mathrm{n_{x,c}}} \times \mathbb{Z}^{\mathrm{n_{x,i}}}\right)\right\}, \quad \bar{\mathcal{Y}} \coloneqq \left\{\left[\boldsymbol{y}^{\mathrm{lb}}, \boldsymbol{y}^{\mathrm{ub}}\right] \cap \left(\mathbb{R}^{\mathrm{n_{y,c}}} \times \mathbb{Z}^{\mathrm{n_{y,i}}}\right)\right\}, \tag{3.1}$$

where the superscripts "lb" and "ub" denote finite UBDs and LBDs of the upper- and lower-level variables, respectively. These host sets are used in the definitions of the upper- and lower-level feasible sets, $\mathcal{X}$ and $\mathcal{Y}$, in (SIP); as well as in (to follow) (MINMAX), (GSIP), and (BLP). If continuous variables are present, the lower-level feasible set $\mathcal{Y}$ is a set of infinite cardinality.

While non-compact host sets can theoretically be handled with adaptive discretization-based algorithms under certain assumptions (see Chapter 6) — provided they are supported by the subsolvers — we specifically restrict ourselves to host sets defined by finite bounds on both the continuous and integer variables and assume that all functions involved are continuous within these bounds.

We consider MINMAX of the form

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}} \quad & \max_{y\in\mathcal{Y}} f(\boldsymbol{x},\boldsymbol{y}), \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y} \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}, \end{aligned} \tag{MINMAX}$$

the objective function $f : \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{y}} \to \mathbb{R}$ and all other functions and sets as defined for (SIP). (MINMAX) is a specialization of (SIP), as can be seen from the reformulation

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X},t\in\mathbb{R}} \quad & t \\ \text{s.t.} \quad & f(\boldsymbol{x},\boldsymbol{y}) - t \leq 0,\ \forall\, \boldsymbol{y} \in \mathcal{Y}, \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x} \in \bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y} \coloneqq \left\{\boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}. \end{aligned} \tag{MINMAX-REF}$$

GSIPs are an extension to SIPs that allow dependency of the lower-level feasible set on the upper-level variables. We consider GSIPs of the form

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \forall\, \boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x}), \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x} \in \bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y}(\boldsymbol{x}) \coloneqq \left\{\boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}, \end{aligned} \tag{GSIP}$$

the coupling lower-level inequality constraint function $\boldsymbol{g}^{\mathrm{l}} : \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{y}} \to \mathbb{R}^{n_{\mathrm{gl}}}$, and all other functions and sets as defined for (SIP).

The optimistic formulation of a BLP reads

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\bar{\mathcal{Y}}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{y} \in \arg\min_{\boldsymbol{z}\in\mathcal{Y}(\boldsymbol{x})} h(\boldsymbol{x},\boldsymbol{z}), \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x} \in \bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y}(\boldsymbol{x}) \coloneqq \left\{\boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}, \end{aligned} \tag{BLP}$$

the upper-level objective function $f : \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{y}} \to \mathbb{R}$, the lower-level objective function $h : \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{y}} \to \mathbb{R}$, and all other functions and sets defined as for (SIP). Note that (BLP) contains variables $\boldsymbol{y}$ which are constrained to be in the set of global optimizers of an embedded optimization problem.

The reviewed problem classes are closely related and, thus, share some common traits. For example, due to their hierarchical structure, establishing the feasibility of a candidate solution point requires solving an embedded optimization problem globally for all classes. Thus, global optimization techniques are (implicitly or explicitly) needed in general for the solution process, and consequently, all the problem classes mentioned are computationally demanding. Since GSIPs are generalizations of SIPs and MINMAX are a specialization of SIPs, one can, under certain assumptions, use any algorithm for solving GSIPs to solve the other two problem classes. However, given the expected high computational cost of solving these problems, implementing specialized algorithms is likely beneficial over an implementation that only addresses the most general problem class. BLPs and GSIPs are closely related: if $\nexists \bar{\boldsymbol{x}} \in \mathcal{X}$ such that $\mathcal{Y}(\bar{\boldsymbol{x}}) = \emptyset$, (BLP) and (GSIP) are equivalent (Stein and Still, 2002). However, there is a distinctive difference if a point $\bar{\boldsymbol{x}}$ exists which leads to an empty lower-level feasible set $\mathcal{Y}(\bar{\boldsymbol{x}}) = \emptyset$; the point $\bar{\boldsymbol{x}}$ is infeasible in (BLP), while it is feasible in (GSIP). The close relation between the problem classes is also reflected in the close connection between the algorithmic approaches discussed in the next section.

3.3 Adaptive Discretization-Based Algorithms

In this section, we introduce multiple adaptive discretization-based algorithms for the solution of the hierarchical optimization programs covered in Section 3.2, which are implemented in our open-source software libDIPS.

We highlight similarities and connections between algorithms to illustrate the benefits of collecting the implementations of these adaptive discretization-based algorithms in a single software. Note that we only give a basic description of the algorithms, focusing on the essential algorithmic ideas and the algorithm's parameters. We refer to the original publications for a detailed description.

A significant benefit of adaptive discretization-based algorithms is that they can utilize well-established optimization solvers, mainly as a black box. The main challenge in solving (SIP), (GSIP), (MINMAX), and (BLP) compared to standard NLPs is the infinite cardinality of the set $\mathcal{Y}$. This challenge can be addressed through various generalizations and adaptations of the adaptive discretization-based algorithm for SIPs proposed by Blankenship and Falk (1976), which is inspired by Remez (1962). Conceptually, the approach of Blankenship and Falk (1976) replaces the infinite index set $\mathcal{Y}$ by a finite discretized set $\mathcal{Y}^{\mathrm{d}} \subsetneq \mathcal{Y}$. This discretized problem gives an approximation of (SIP). Through an adaptive refinement scheme, points are added to $\mathcal{Y}^{\mathrm{d}}$, and the approximation of (SIP) is improved.

The implemented solvers are listed in Table 3.2. Note that apart from the solver *BLP-Box*, all implemented solvers are capable of handling integer variables. The interested reader may refer to Appendix C.4, where we motivate that although some of the publications on which we base our implementation do not explicitly cover the integer case, the implemented solvers support integer variables.

All the algorithms implemented as solvers in libDIPS are conceptually closely related to the approach of Blankenship and Falk (1976). Therefore, we will provide a conceptual description of the algorithms within the context of adapting the central algorithmic idea presented in Blankenship and Falk (1976). We refer the interested reader to the original publications listed in Table 3.2 for a detailed description of the algorithm underlying the respective solver.

3.3.1 Approaches for SIPs

Algorithmic Approach Underlying *B&F*

The SIP algorithm of Blankenship and Falk (1976) proposes to relax (SIP) by replacing the infinite index set with a finite one, i.e., $\mathcal{Y}^{\mathrm{LBP}} \subsetneq \mathcal{Y}$. We implement this algorithm in libDIPS as the solver called *B&F*. For a simplified pseudo algorithm of *B&F* refer to Algorithm C.1 in Appendix C.1. The resulting upper-level problem (ULP)

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathcal{X}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}, \boldsymbol{y}^k\big) \leq \boldsymbol{0},\ \forall\ \boldsymbol{y}^k \in \mathcal{Y}^{\mathrm{LBP}}, \\ \text{with} \quad & \mathcal{X} \coloneqq \big\{\boldsymbol{x} \in \bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\big\}, \end{aligned} \tag{LBP}$$

Table 3.2: Overview of the adaptive discretization-based algorithms implemented in libDIPS. † Existence-constrained semi-infinite programs (ESIPs) are not considered in this dissertation and, hence, we omit a describtion thereof. For further information on existence-constrained semi-infinite programs (ESIPs), please refer to Djelassi and Mitsos (2021) and the publication that serves as the basis of this chapter (Jungen et al., 2023a).

Problem Class	Solver Name	Original Publication	Comment
SIP	*B&F*	Blankenship and Falk (1976)	Finite termination with feasible point not guaranteed
	RRHS	Mitsos and Tsoukalas (2015)	
	Oracle	Tsoukalas and Rustem (2011)	
	Hybrid	Djelassi and Mitsos (2017)	
MINMAX	*Minmax*	Falk and Hoffman (1977)	
GSIP	*GSIP-RRHS*	Mitsos and Tsoukalas (2015)	Finite termination not guaranteed Harwood (2019)
	*	Djelassi et al. (2019)	Relax GSIP to derive an SIP; solve resulting SIP with any SIP solver *, c.f., Section 3.3.2
ESIP†	*B&F*	Djelassi and Mitsos (2021)	Upper-bounding procedure of Djelassi and Mitsos (2021) not implemented
BLP	*BLP-Box*	Mitsos et al. (2008a)	Implementation does not support discrete variables; extension presented in Mitsos (2010) supports discrete variables
	BLP-noBox	Djelassi et al. (2019)	

yields a candidate point $\bar{\boldsymbol{x}}$; if (LBP) is solved globally, a LBD on (SIP) is obtained. Feasibility of point $\bar{\boldsymbol{x}}$ is checked through solving the LLP

$$\begin{aligned} &\max_{\boldsymbol{y}\in\mathcal{Y}} \quad \max_{j\in\{1\ldots n_{\mathrm{gu}}\}} g_j^{\mathrm{u}}(\bar{\boldsymbol{x}},\boldsymbol{y}), \\ &\text{with} \quad \mathcal{Y} \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}, \end{aligned} \tag{LLP}$$

and upper-level variables fixed to the corresponding values.

If the optimal objective value of (LLP) is less than or equal a predefined tolerance $\varepsilon^{\mathrm{a}} \geq 0$, $\bar{\boldsymbol{x}}$ is ε^{a}-SIP-feasible (c.f., Definition 2 in Appendix B.1) and the algorithm terminates. If the optimal objective value is strictly greater than ε^{a}, the solution point $\boldsymbol{y}^*$ of (LLP) is added to the discretization, i.e., $\mathcal{Y}^{\mathrm{LBP}} \leftarrow \mathcal{Y}^{\mathrm{LBP}} \cup \boldsymbol{y}^*$; then, (LBP) is solved again.

Remark 1 Instead of solving (LLP), where we maximize over all entries of $\boldsymbol{g}^{\mathrm{u}}$, it is also possible to solve a separate LLP for each entry. Note that if we solve a separate LLP for each entry of $\boldsymbol{g}^{\mathrm{u}}$, we must solve n_{gu} optimization problems, and up to n_{gu} points are added to the discretization in each iteration.

Another possibility is to introduce for each entry of $\boldsymbol{g}^{\mathrm{u}}$ a separate discretization $\mathcal{Y}^{\mathrm{d},i}$ with $i = 1, ..., n_{\mathrm{gu}}$.

libDIPS supports both alternatives, but neither is used in the numerical experiments in Section 3.6.

It has been proven numerous times in literature (Blankenship and Falk, 1976, Djelassi and Mitsos, 2017, Mitsos, 2011) that if (SIP) is feasible and $\varepsilon^{\mathrm{a}} = 0$, the accumulation points of this algorithm are optimal SIP-feasible points (c.f., Definition 4 in Appendix B.1). Additionally, if (SIP) is infeasible, (LBP) will be infeasible after finitely many iterations. However, there can generally be no finite termination guarantee for $\varepsilon^{\mathrm{a}} = 0$. Finite termination can only be guaranteed for $\varepsilon^{\mathrm{a}} > 0$. However, in this case, an SIP-feasible point is usually not generated. The proof of finite convergence usually relies on relatively mild assumptions, i.e., compactness of host sets, continuity of f and $\boldsymbol{g}^{\mathrm{u}}$ on their respective host sets, and approximate global solution of subproblems (Blankenship and Falk, 1976, Djelassi and Mitsos, 2017, Harwood et al., 2021, Mitsos, 2011). For a detailed discussion of the necessary assumptions and a proof of convergence refer to Chapter 6.

Algorithmic Approaches Underlying *RRHS*, *Oracle*, and *Hybrid*

Since the approach of *B&F* only generates LBDs and generally does not terminate finitely with an SIP-feasible point, several adaptations of the underlying algorithm have been proposed to generate improving sequences of UBDs. The following algorithms share the commonality that, akin to Blankenship and Falk (1976), they iteratively solve a sequence of subproblems: a discretized ULP and, subsequently, a LLP for fixed upper-level variables. The solution points of the LLP are then used to discretize the ULP.

The upper-bounding approaches of the implemented solvers can be summarized as follows: The algorithm proposed by Mitsos (2011), implemented as *RRHS*, achieves finite termination through an upper-bounding scheme in combination with the lower-bounding scheme from *B&F*. The algorithm proposed by Tsoukalas and Rustem (2011), implemented as *Oracle*, utilizes an oracle problem to conduct a bisection search in the objective space.

The algorithm proposed by Djelassi and Mitsos (2017), implemented as *Hybrid*, attempts to combine the best of both, *RRHS* and *Oracle*.

As mentioned in the introduction, other upper-bounding approaches exist; however, since they are not currently implemented in libDIPS, we will not review them here and instead focus on the mentioned algorithms in the following:

RRHS generates UBDs through an upper-bounding procedure using a so-called upper-bounding problem (UBP). This UBP is constructed by relaxing (SIP) by replacing the set $\mathcal{Y}$ with a set of finite cardinality, i.e., $\mathcal{Y}^{\mathrm{UBP}} \subsetneq \mathcal{Y}$ (equivalent to *B&F*). At the same time, the discretized (semi-infinite) constraint is tightened by introducing a restriction of the right-hand side $\varepsilon^{\mathrm{r}} > 0$, which is initialized with the parameter $\varepsilon^{\mathrm{r},0} > 0$, resulting in

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}, \boldsymbol{y}^k\big) \leq -\varepsilon^{\mathrm{r}} \cdot \mathbf{1},\ \forall\ \boldsymbol{y}^k \in \mathcal{Y}^{\mathrm{UBP}}, \\ \text{with} \quad & \mathcal{X} \coloneqq \big\{\boldsymbol{x} \in \bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \mathbf{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \mathbf{0}\big\}. \end{aligned} \tag{UBP}$$

Note that neither a relaxation nor a restriction of (SIP) is generally attained by (UBP) because of the discretization (relaxation) combined with the restriction of the right-hand side. Therefore, it is not guaranteed that the solution of (UBP) immediately generates an upper-bound. However, the generation of (improving) upper-bounds is guaranteed through a successive finer discretization of $\mathcal{Y}^{\mathrm{UBP}}$ and a decreasing rule for ε^{r}, which is applied under certain conditions. The employed decreasing rule is $\varepsilon^{\mathrm{r}} \leftarrow \frac{\varepsilon^{\mathrm{r}}}{\varepsilon^{\mathrm{red}}}$, with a fixed reduction rate $\varepsilon^{\mathrm{red}}$.

The algorithm alternates between iterations that solve (LBP), as in *B&F*, and iterations that solve (UBP). In both cases, (LLP) is subsequently solved to determine SIP-feasibility of the candidate points generated by either (UBP) or (LBP). The solution points of (LLP) are used to populate $\mathcal{Y}^{\mathrm{LBP}}$ or $\mathcal{Y}^{\mathrm{UBP}}$, respectively.

Finite termination to an ε^{f}-optimal SIP-Slater point (see Definition 8 in Appendix B.3) is guaranteed through the converging lower-bounding procedure, which is equivalent to *B&F*, and the converging upper-bounding procedure using (UBP) with the decreasing rule for ε^{r}. As *RRHS* uses the idea of Blankenship and Falk (1976) as the lower bounding procedure, the same assumptions are necessary. The employed upper-bounding procedure additionally relies on the existence of an ε^{f}-optimal SIP-Slater point and a decreasing rule of ε^{r}. For a detailed discussion of the necessary assumptions and a proof of convergence, refer to Appendix E.2.3.

Informally, the primary objective behind the upper-bounding procedure using (UBP) is to identify an appropriate combination of two key factors: a discretization $\mathcal{Y}^{\mathrm{UBP}}$ (which corresponds to a relaxation of the set $\mathcal{Y}$) and a restriction ε^{r} applied to the right-hand side of the semi-infinite constraint. With a suitable combination, SIP-feasible points are generated that also provide an UBD to (SIP). In short, if (UBP) is feasible, but the corresponding solution is SIP-infeasible, we need to reduce the relaxation by enhancing the discretization. Otherwise, we reduce the restriction ε^{r} since it was either too restrictive and thus led to the infeasibility of (UBP), even with the relaxation inherent in the discretization, or a SIP-feasible point was found, signaling the need to reduce the restriction.

Note that the performance of the upper-bounding procedure is strongly influenced by the initial restriction parameter $\varepsilon^{\mathrm{r},0}$, the rate of reduction $\varepsilon^{\mathrm{red}}$, and the initial discretization

$\mathcal{Y}^{\mathrm{UBP}}$. For example, reducing the restriction too quickly, for instance, by using a large reduction rate $\varepsilon^{\mathrm{red}}$, can lead to many iterations (Djelassi and Mitsos, 2017).

Oracle uses an oracle adaption of Blankenship and Falk (1976) to determine if a target objective value f^{t} is attainable. Given a preexisting UBD and LBD on the optimal objective value of (SIP), the target objective value f^{t} is updated through a bisection search in the objective space. The adapted, discretized ULP is given as

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}} \quad & \max\left\{ f(\boldsymbol{x}) - f^{\mathrm{t}},\ \max_{\substack{\boldsymbol{y}^k\in\mathcal{Y}^{\mathrm{ORA}},\\ j\in\{1\ldots n_{\mathrm{gu}}\}}} g_j^{\mathrm{u}}\left(\boldsymbol{x},\boldsymbol{y}^k\right) \right\}, \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{ \boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0} \right\}, \end{aligned} \tag{ORA}$$

and $\mathcal{Y}^{\mathrm{ORA}} \subsetneq \mathcal{Y}$. This problem can be reformulated to avoid the max-operator, c.f., (ORA-REF) in Appendix C.2.1.

Analogous to *B&F*, (ORA) is solved with a subsequent solution of (LLP) with fixed upper-level variables $\bar{\boldsymbol{x}}$. If the optimal objective value of (ORA) is greater than 0, f^{t} is not attainable. If (LLP) proves the current candidate point $\bar{\boldsymbol{x}}$ to be SIP-infeasible, the optimal solution point of (LLP) is used for the discretization of $\mathcal{Y}^{\mathrm{ORA}}$. Else $\bar{\boldsymbol{x}}$ is SIP-feasible, and f^{t} is attainable.

Although one might expect the method to inherit strong robustness from the bisection procedure, *Oracle* relies on stricter assumptions for finite convergence compared to *RRHS*. These stricter assumptions pertain to the local minima of $\boldsymbol{g}^{\mathrm{u}}$ (Lemma 4.2 in Tsoukalas and Rustem (2011)).

Hybrid aims to combine the ideas of the two previous approaches. Recall that the performance of *RRHS* depends on the parameters $\varepsilon^{\mathrm{r},0}$ and $\varepsilon^{\mathrm{red}}$. *Hybrid* essentially uses the same iterative approach for the upper- and lower-bounding as *RRHS*. Additionally, the subproblem

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X},\eta} \quad & -\eta \\ \text{s.t.} \quad & f(\boldsymbol{x}) - f^{\mathrm{RES}} \leq 0 \\ & \boldsymbol{g}^{\mathrm{u}}\left(\boldsymbol{x},\boldsymbol{y}^k\right) \leq -\eta\cdot\boldsymbol{1},\ \forall\ \boldsymbol{y}^k\in\mathcal{Y}^{\mathrm{RES}}, \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{ \boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0} \right\}, \end{aligned} \tag{RES}$$

is solved which employs a seperate discretization $\mathcal{Y}^{\mathrm{RES}}$ and target objective value f^{RES}, similarly to f^{t} in (ORA). In this way, some iterations can generate an adaptive optimal update to the restriction parameter ε^{r} by using the solution of (RES) to update ε^{r}. Incorporating the updated restriction parameter in (UBP), *Hybrid* aims to reduce the performance dependence on the parameters $\varepsilon^{\mathrm{r},0}$ and $\varepsilon^{\mathrm{red}}$. Finite convergence of *Hybrid* is guaranteed under the same assumptions necessary for *RRHS* (Djelassi and Mitsos, 2017).

Table 3.3 summarizes the subproblems of the SIP algorithms above. The user must provide these subproblems to use the respective solver in libDIPS. The subproblems of the respective solvers of the problem classes GSIPs, MINMAX, and BLPs, covered in the following sections, are also listed in Table 3.3.

Table 3.3: List of subproblems used in the solvers. Note that some of the solvers automatically generate auxiliary problems. Integer variables in the upper and lower levels are supported by all solvers except for the current version of *BLP-Box*. $^{+}$ Subproblem definition see Appendix C.2.2. † Subproblem definition see Appendix C.2.3.

Problem Class	Solver Name	Subproblems Provided by the User
SIP	*B&F*	(LBP), (LLP)
	RRHS	(LBP), (UBP), (LLP)
	Oracle	(LBP), (ORA), (LLP)
	Hybrid	(LBP), (UBP), (RES), (LLP)
MINMAX	*Minmax*	(MINMAX-LBP)$^{+}$, (MINMAX-LLP)
GSIP	*GSIP-RRHS*	(LBP), (UBP), (GSIP-LLP), (GSIP-AUX)
	*	see SIP solvers
BLP	*BLP-noBox*	(BLP-LBP), (BLP-UBP), (BLP-LLP)†, (BLP-AUX)†
	BLP-Box	(BLP-LBP), (BLP-UBP), (BLP-LLP)†, (BLP-AUX)†, (BLP-AUX-V)†

Algorithmic Approach Underlying *Minmax*

In Falk and Hoffman (1977), a specialization of the approach from Blankenship and Falk (1976) is presented for (MINMAX). This specialization is based on the fact that the best UBD for a given upper-level point $\bar{\boldsymbol{x}}$ is provided by the LLP

$$\begin{aligned} &\max_{y \in \mathcal{Y}} \quad f(\bar{\boldsymbol{x}}, \boldsymbol{y}), \\ &\text{with} \quad \mathcal{Y} \coloneqq \left\{ \boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0} \right\}. \end{aligned} \tag{MINMAX-LLP}$$

As a result, an UBD is readily obtained without the upper-bounding procedures discussed above. A solver using this unique structure is implemented as ***Minmax*** in libDIPS. As *Minmax* is a specialization of *B&F*, it relies on the same underlying assumptions.

3.3.2 Approaches for GSIPs

There are two main approaches for the global solution of (GSIP) in the context of adaptive discretization-based methods in the literature: (i) relaxation of (GSIP) to an SIP and its solution with an SIP approach and (ii) adaptation of the underlying approach used in *RRHS* to GSIPs.

GSIP to SIP Reformulation and Relaxation

Commonly, (GSIP) is reformulated and relaxed to an SIP. First, the semi-infinite constraint in (GSIP) is reformulated to

$$\boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0},\ \forall\ \boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x}) = \{\boldsymbol{y} \in \hat{\mathcal{Y}} : \boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0}\}, \tag{3.2}$$

where

$$\hat{\mathcal{Y}} \coloneqq \left\{ \boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0} \right\}. \tag{3.3}$$

Then, (3.2) is further reformulated to

$$\boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \vee \boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x}, \boldsymbol{y}) > \boldsymbol{0},\ \forall\ \boldsymbol{y} \in \hat{\mathcal{Y}}, \tag{3.4}$$

which is subsequently relaxed to

$$\boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \vee \boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x}, \boldsymbol{y}) \geq \boldsymbol{0},\ \forall\ \boldsymbol{y} \in \hat{\mathcal{Y}}. \tag{3.5}$$

We can now define

$$g_i^{\mathrm{u\ relax}} \coloneqq \min \left\{ g_i^{\mathrm{u}}(\boldsymbol{x}, \boldsymbol{y}), \min_{j \in \{1 \dots n_{\mathrm{gl}}\}} - g_j^{\mathrm{l}}(\boldsymbol{x}, \boldsymbol{y}) \right\}, \tag{GSIP-REF}$$

with $i \in \{1 \dots n_{\mathrm{gu}}\}$, which corresponds to a min-reformulation of the conjunction in (3.5). Finally, an SIP relaxation of (GSIP) is obtained, by replacing the left-hand-side of the semi-infinite constraint in (SIP), i.e., $\boldsymbol{g}^{\mathrm{u}}$, with $\boldsymbol{g}^{\mathrm{u\ relax}}$.

The relaxation stems from relaxing the strict inequality in $\boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x}, \boldsymbol{y}) > 0$ to $\boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x}, \boldsymbol{y}) \geq 0$, c.f., (3.4) and (3.5). As shown in Günzel et al. (2007), this relaxation is, under certain assumptions, equivalent to relaxing the feasible set of (GSIP) to its closure. If this property is not present, the relaxation can be strict. In the following, we assume that the relaxation is not strict, i.e., that the minimum of the relaxed problem is equal to the infimum of the original problem (Djelassi, 2020, Mitsos and Tsoukalas, 2015). For further details on the GSIP to SIP reformulation and relaxation, the interested reader may refer to Djelassi (2020), Günzel et al. (2007).

State-of-the-art SIP solvers can be applied, as described in Section 3.3, for the solution of the derived SIP. As it should be apparent whether an SIP or a derived SIP from a GSIP is considered, we use the same solver names as in Section 3.3 whenever a derived SIP is solved with an SIP solver.

Adaption of *RRHS* for GSIPs

The previous approach, i.e., solving the SIP derived from a GSIP with SIP solvers, has the inherent potential disadvantage of ignoring the original GSIP structure. On the contrary, the algorithm published in Mitsos and Tsoukalas (2015), implemented as *GSIP-RRHS*, takes this structure into account. Similar to *RRHS*, Mitsos and Tsoukalas (2015) employ a restriction of the right-hand side approach to generate UBDs and an adaptation of *B&F* to generate LBDs. They use the aforementioned GSIP to SIP relaxation and reformulation, but then account for the GSIP structure when checking the feasibility of a candidate point and when generating discretization points. The main change is that after the GSIP LLP

$$\begin{aligned} g^{\mathrm{u},*} = \max_{\boldsymbol{y} \in \mathcal{Y}} \quad & \max_{j \in \{1 \dots n_{\mathrm{gu}}\}} g_j^{\mathrm{u}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{l}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \leq \boldsymbol{0}, \\ \text{with} \quad & \mathcal{Y} \coloneqq \left\{ \boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0} \right\} \end{aligned} \tag{GSIP-LLP}$$

is solved to check the feasibility of the candidate point $\bar{\boldsymbol{x}}$, the auxiliary problem

$$\begin{aligned} \min_{\boldsymbol{y}\in\mathcal{Y}} \quad & \max_{j\in\{1\ldots n_{\mathrm{gl}}\}} g^{\mathrm{l}}(\bar{\boldsymbol{x}},\boldsymbol{y}) \\ \text{s.t.} \quad & \max_{j\in\{1\ldots n_{\mathrm{gu}}\}} g_j^{\mathrm{u}}(\bar{\boldsymbol{x}},\boldsymbol{y}) \geq \alpha\cdot g^{\mathrm{u},*}, \\ \text{with} \quad & \mathcal{Y} \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\} \end{aligned} \tag{GSIP-AUX}$$

and $\alpha > 0$ is solved to find a GSIP-LLP-Slater point, c.f., Definition 7 in Appendix B.2. This GSIP-LLP-Slater point is needed because only a discretization point that strictly fulfills the coupling inequality constraints $\boldsymbol{g}^{\mathrm{l}}(\bar{\boldsymbol{x}},\boldsymbol{y}) \leq \boldsymbol{0}$ will, if added to the discretization, provide a restriction in the derived subproblem (LBP). The solution point of (GSIP-LLP) will not necessarily provide a restriction if used as a discretization point.

The assumptions used by Mitsos and Tsoukalas (2015) are similar to those of *RRHS*, but have been adapted for the GSIP case (compactness of host sets, continuity of all functions on their respective host sets, global solution of subproblems, infimum of the original (GSIP) is equal to the minimum of used SIP relaxation, and existence of an ε^{f}-optimal GSIP-Slater point, c.f, Definition 9 in Appendix B.3).

The original publication searched for a GSIP-LLP-Slater point whose relative suboptimality in the lower level was bounded over all iterations by a fixed factor $\alpha > 0$. However, depending on the a priori chosen α, generating a GSIP-LLP-Slater point might fail (Harwood, 2019). Our implementation of this approach, i.e., *GSIP-RRHS*, tries to address the issues raised by Harwood (2019) by adaptively choosing α whenever a GSIP-LLP-Slater point is not attained by (GSIP-AUX), c.f., Section 3.4.2.1. However, in some instances, numerical issues related to identifying GSIP-LLP Slater points persist for *GSIP-RRHS*, as discussed in Section 3.6.3.

3.3.3 Approaches for BLPs

For BLPs we implement two solvers, namely ***BLP-Box*** based on Mitsos et al. (2008a) and ***BLP-noBox*** based on Djelassi et al. (2019). The current implementation of *BLP-Box*, based on Mitsos et al. (2008a), does not support discrete variables. However, the extension thereof, presented in Mitsos (2010), supports discrete variables. *BLP-Box* and *BLP-noBox* are closely connected to the approach of Blankenship and Falk (1976). This is due to the reformulation of the BLP to a GSIP using the value function reformulation

$$\boldsymbol{y} \in \arg\min_{z\in\mathcal{Y}(\boldsymbol{x})} h(\boldsymbol{x},\boldsymbol{z}) \iff \boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x}) \wedge [h(\boldsymbol{x},\boldsymbol{y}) - h(\boldsymbol{x},\boldsymbol{z}) \leq 0,\ \forall\ \boldsymbol{z}\in\mathcal{Y}(\boldsymbol{x})].$$

Accordingly, both approaches utilize

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x})} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\mathrm{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{x}\in\tilde{\mathcal{X}}\left(\boldsymbol{z}^k\right) \implies h(\boldsymbol{x},\boldsymbol{y}) - h\left(\boldsymbol{x},\boldsymbol{z}^k\right) \leq 0,\ \forall\ \boldsymbol{z}^k\in\mathcal{Y}^{\mathrm{LBP}}, \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y}(\boldsymbol{x}) \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{g}^{\mathrm{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\mathrm{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}, \end{aligned} \tag{BLP-LBP}$$

to generate a LBD based on the discretization $\mathcal{Y}^{\mathrm{LBP}}$, and a parametric set $\tilde{\mathcal{X}}$. The primary difference between the two approaches lies in how the set $\tilde{\mathcal{X}}$ is selected. In Mitsos et al.

(2008a), the authors compute boxes in each iteration k around the current iterate for the upper-level variables $\bar{\boldsymbol{x}}$. The initial size of the box is proportional to the difference between the UBD and LBD of the lower-level variable, scaled by a scalar d. This scalar is initialized to d^0 and iteratively reduced by the factor d^{red} until it can be assured that the discretization point $\boldsymbol{z}^k$ stays feasible concerning the lower-level constraints when the upper-level variable $\boldsymbol{x}$ is inside the box. This approach to compute the parametric set $\tilde{\mathcal{X}}$ is implemented in our solver *BLP-Box*. The approach in Djelassi et al. (2019) generates the set $\tilde{\mathcal{X}}$ in a parametric way by setting $\tilde{\mathcal{X}}(\boldsymbol{z}) \coloneqq \{\boldsymbol{x} \in \mathcal{X} : \boldsymbol{z} \in \mathcal{Y}(\boldsymbol{x})\}$. To generate discretization points, a Slater point of the lower level with respect to the coupling constraint $\boldsymbol{g}^{\text{l}}$ is searched, similarly to the procedure in *GSIP-RRHS* but with an absolute tolerance instead of the relative tolerance α in (GSIP-AUX).

Both approaches utilize a probing problem specifically catered toward BLPs to generate UBDs. The probing problem is given by

$$\begin{aligned} \min_{\boldsymbol{y} \in \mathcal{Y}(\bar{\boldsymbol{x}})} \quad & f(\bar{\boldsymbol{x}}, \boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\text{u}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \leq \boldsymbol{0} \\ & h(\bar{\boldsymbol{x}}, \boldsymbol{y}) \leq h^*(\bar{\boldsymbol{x}}) + \varepsilon^{\text{l,UBP}} \\ & LBD \leq f(\bar{\boldsymbol{x}}, \boldsymbol{y}), \end{aligned} \tag{BLP-UBP}$$

$$\text{with} \quad \mathcal{Y}(\bar{\boldsymbol{x}}) \coloneqq \left\{\boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{g}^{\text{l}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\},$$

the estimate $h^*(\bar{\boldsymbol{x}})$ for the optimal value of the lower level at the current value for the upper-level variables $\bar{\boldsymbol{x}}$, a small positive tolerance $\varepsilon^{\text{l,UBP}}$, and a known LBD on the upper-level objective LBD. The LLP of *BLP-noBox*, *BLP-Box*, and *BLP-Oracle* (to follow in Chapter 4) to compute $h^*(\bar{\boldsymbol{x}})$ is formulated as

$$\begin{aligned} h^*(\bar{\boldsymbol{x}}) = \min_{\boldsymbol{y} \in \mathcal{Y}(\bar{\boldsymbol{x}})} \quad & h(\bar{\boldsymbol{x}}, \boldsymbol{y}), \\ \text{with} \quad & \mathcal{Y}(\bar{\boldsymbol{x}}) \coloneqq \left\{\boldsymbol{y} \in \bar{\mathcal{Y}} : \boldsymbol{g}^{\text{l}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}. \end{aligned} \tag{BLP-LLP}$$

Both approaches rely on similar assumptions: compactness of host sets, continuity of all functions on their respective host sets, (approximate) global solution of subproblems, and a Slater-point assumption on the lower-level to be able to construct the discretization, c.f., Assumption 3 in Mitsos et al. (2008a) and Assumption 7 in Djelassi et al. (2019). The assumptions in Djelassi et al. (2019) differ slightly from those in Mitsos et al. (2008a), allowing for an approximate solution of the subproblems, discrete variables, and coupling equality constraints that couple the upper and lower level.

3.4 The Open-Source Software libDIPS

As outlined in the introduction and highlighted in Section 3.3, the reviewed algorithms and implemented solvers are closely related and, hence, can benefit from a shared infrastructure. By combining all these solvers into a single software, we can compare them more fairly because they use the same programming language, share a common codebase, and access subsolvers through a unified interface. Therefore, we implemented all solvers in the open-source C++ software libDIPS – Discretization-based semi-Infinite and bilevel Programming Solvers, which was already used in a preliminary form in Djelassi (2020).

The latest version of libDIPS, including the text-based parser, support for the MPI parallelized version of MAiNGO (Bongartz et al., 2018), and interfaces to the supported subsolvers, is open-source under the Eclipse Public License v2.0, accessible under `https://git.rwth-aachen.de/avt-svt/public/libdips`. Additional information on downloading and compiling libDIPS and its dependencies, as well as a list of tested platforms for Microsoft Windows and Linux, is available on the documentation page of the libDIPS repository.

3.4.1 User-Oriented Introduction and Overview

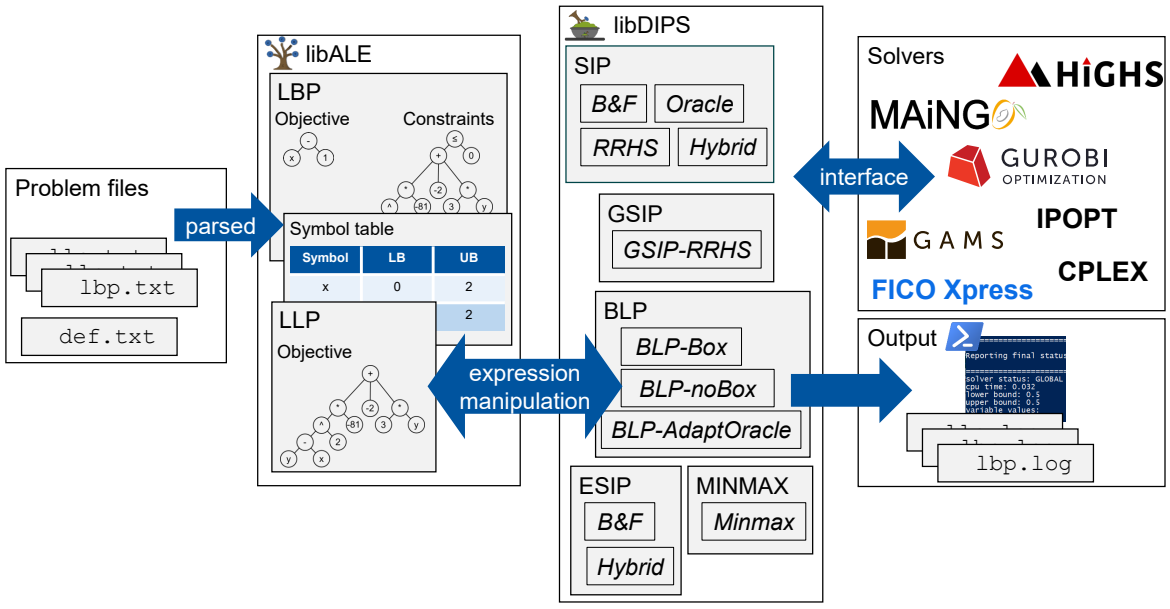


Figure 3.1: High-level user interface-oriented software structure overview. *BLP-AdaptOracle* see Chapter 4. Existence-constrained semi-infinite programs (ESIPs) are not considered in this dissertation and, hence, we omit a description thereof. Note: Adapted from Jungen et al. (2023a).

We present the software structure at a high level in Figure 3.1, focusing on the user's perspective. Based on this structure, we summarize the high-level workflow of the software:

- The respective subproblems are provided by the user in an easily editable and human-readable form using the domain-specific language provided by libALE (Zingler et al., 2022).
- The user input is parsed using libALE, which stores the objective and constraints of each subproblem as logical expression trees and the parameters and variables in a symbol table.
- One of the solvers implemented in libDIPS, c.f., Table 3.2, which the user has selected, is used to solve the problem.
- One of the several interfaced state-of-the-art optimization solvers is used to solve the subproblem. Currently, CPLEX (International Business Machines Corporation, 2022), GUROBI (Gurobi Optimization, 2025), IPOPT (Wächter and Biegler, 2006),

FICO Xpress (Fair Isaac Corporation, 2023), and MAiNGO (Bongartz et al., 2018) as well as the GAMS C++ API (GAMS Development Corporation, 2019) are available. The user can switch between them easily. Note that, with a suitable license, GAMS provides access to over 30 solvers, including, e.g., BARON (Khajavirad and Sahinidis, 2018) and ANTIGONE (Misener and Floudas, 2014)

- Output is printed onto the command line, and additional log files are written.

There are two options for formulating a problem for use in libDIPS. A significant design decision was to allow the user to manually and individually formulate the subproblems (in an opinionated way). Alternatively, we also provide templated input files that allow for automatically deriving the necessary subproblems from a high-level problem description. In the following Section 3.4.1.1, we elaborate on the benefits of individually formulating the subproblems with a small example.

3.4.1.1 Example Solution of an SIP via *B&F*

Suppose we want to solve the following SIP using *B&F*.

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}} \quad & f(\boldsymbol{x}) \coloneqq 2x_1^2 + x_2^2 \\ \text{s.t.} \quad & g^{\mathrm{u}}(\boldsymbol{x}, y) \coloneqq \min\{x_1 + y_1, x_2 + y_2\} \leq 0,\ \forall\ y \in \mathcal{Y}, \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x} \in \bar{\mathcal{X}} \coloneqq [-3,3]^2 : v^{\mathrm{eu}}(\boldsymbol{x}) \coloneqq x_1 + x_2 - 1 = 0\right\} \\ & \mathcal{Y} \coloneqq [-1,1]^2 \end{aligned} \tag{Ex}$$

Using the domain-specific language provided by libALE, we can write the lower-bounding subproblem as a direct transcription of (LBP), as shown in listing 3.1 for the `lbp.txt` file. The syntax for the subproblems should be intuitive. For a detailed introduction to the domain-specific language provided by libALE and the syntax for defining an optimization problem, the reader may refer to the libALE documentation (Zingler et al., 2022) and MAiNGO documentation (Bongartz et al., 2018).

If we were to naively or automatically generate the subproblem (LLP) from the problem description (i.e., from $f, g^{\mathrm{u}}, v^{\mathrm{eu}}$ and $\mathcal{Y}$), the resulting objective would read $\max_{\boldsymbol{y}\in\mathcal{Y}} \min\{x_1 + y_1, x_2 + y_2\}$. Because the user can define the subproblems independently from each other in libDIPS, we can use a different formulation in (LLP). Indeed we should use the epigraph formulation $\max_{\boldsymbol{y}\in\mathcal{Y}} t$ s.t. $t \leq x_1 + y_1,\ t \leq x_2 + y_2$. This formulation enables us to solve a linear optimization problem rather than a nonlinear one. From our experience, such an opinionated reformulation is paramount for an efficient solution in bigger optimization problems, especially when binary variables are involved. The opinionated subproblem formulation of (LLP), transcribed in the `llp.txt` file, is shown in listing 3.2.

In addition to the subproblem files, the user has to provide a definitions file, called `def.txt`, which documents the meanings of some of the symbols used in the subproblems. For *B&F*, the user must provide the algorithm with the lower-level variable name, in this case, `y`, and the discretized semi-infinite set of the lower-bounding problem (LBP), `lbp_y_disc`, and the index set `lbp_k_disc` of the LBP.

The interested reader may refer to Appendix C.5 for the necessary subproblem files and extended `def.txt` file for the solution of (Ex) with *RRHS*. Further example input files for the other solvers are provided in the libDIPS repository.

```
definitions:
real[2] x in [-3, 3];

set{index} lbp_k_disc := {};
real[0,2] lbp_y_disc := 0;

objective:
2*x[1]^2+x[2]^2;

constraints:
x[1]+x[2] = 1.0;
forall k in lbp_k_disc :
    min(x[1]+lbp_y_disc[k,1],x[2]+lbp_y_disc[k,2]) <= 0;
```

Listing 3.1: Listing of the file `lbp.txt` for the solution of the exemplary problem (Ex). $\mathcal{Y}^{\mathrm{LBP}}$ is represented by `lbp_y_disc` and `lbp_k_disc`, where the former contains the elements of the set and the latter contains the indices. Both are internally updated by the used solver. Note that in this example, $\mathcal{Y}^{\mathrm{LBP}}$ is initialized as an empty set; thus, the size of the first dimension of `lbp_y_disc` is zero.

```
definitions:
real t in [-4,4];
real[2] y in [-1, 1];

objective:
-t;

constraints:
x[1]+y[1] >= t;
x[2]+y[2] >= t;
```

Listing 3.2: Listing of the file `llp.txt` for the solution of (Ex). Note that all subproblems are written as minimization problems.

```
programdefinitions("lbp"):
set_disc(1) = lbp_k_disc;
set_disc_parameter_src(1, lbp_y_disc) = y;
```

Listing 3.3: Listing of the file `def.txt` for the solution of (Ex) using *B&F*.

3.4.1.2 Supplied Solver Instantiations

As the name suggests, libDIPS constitutes a *library* of adaptive discretization-based solvers that can be used to solve hierarchical optimization problems in a C++ application. For convenience, we include a demo executable for each implemented solver listed in Table 3.2. The inputs to these demo executables are paths to the subproblem files discussed in Section 3.4.1.1 and a dedicated output folder. Other parameters, such as the naming scheme of the input files, are set to default values but can be easily changed in the demo source files. By default, MAiNGO is used as the subsolver. This behavior can be changed by adjusting the corresponding options in the build configuration file.

We also provide the necessary executable to reproduce the computational experiments using the library of test problems, to be introduced in Section 3.5. Additionally, we provide a command-line tool that facilitates similar computational experiments by running the appropriate implemented algorithms for a user-defined list of problem instances.

Please refer to the continuously updated libDIPS documentation for more details on how to set up libDIPS, change the default settings or subsolver, and solve a problem or (a subset of) all test problems using the demo executables of the respective solvers.

3.4.2 Implementation of the Framework

Following the workflow described in Section 3.4.1, the requirements for the framework and structure of libDIPS are

(i) Provision of an intuitive and expressive way for users to input the problem.

(ii) Representation of the problems in a way that facilitates the efficient implementation of the specialized operations needed by the adaptive discretization-based algorithms.

(iii) Interfacing the representation of the subproblems with several subsolvers.

(iv) Abstraction of frequent operations and further auxiliary functionalities, such as logging.

Concerning (i), we provide a simple parser for the metadata written in the `def.txt` file, which *connects* the different subproblems. For the definitions of the subproblems themselves, we decided to use the human-readable domain-specific language provided and parsed by libALE. libALE is also being developed at our chair and was created to be used as a dependency in libDIPS and the deterministic global optimizer MAiNGO. Defining our own domain-specific language enables us to introduce specialized language constructs that aid in formulating hierarchical optimization problems, leading to a more intuitive user experience and supporting features of subsolvers. In terms of intuitive user experience, libALE directly supports universal quantifications over finite sets such as $\forall \boldsymbol{y} \in \mathcal{Y}^{\mathrm{LBP}}$ as a first-class concept. These universal quantifications appear frequently in the subproblems. In the future, we plan to support quantifiers over sets with infinite cardinality, allowing for the natural formulation of hierarchical optimization problems without explicitly formulating the subproblems, thereby eliminating the need for the templates we currently provide. Instead, the subproblems would be automatically derived, providing opportunities for automatic reformulations and pre-solving. This planned feature is conceptually possible, but it has not been implemented in libALE at present. Regarding the support of subsolver features,

libALE already supports certain thermodynamic functions as intrinsic functions, enabling specialized convex relaxation in supporting solvers such as MAiNGO, c.f., Najman et al. (2019). Lastly, using our own domain-specific language also has the benefit that it can be directly customized and further developed to meet the demands of libDIPS and MAiNGO.

Requirement item (ii) holds importance because the algorithms reviewed in Section 3.3 all follow the same principle: First, single-level subproblems derived from the original (multi-level) problem are defined. Then, these single-level subproblems are iteratively and repeatedly solved using existing optimizers employed as subsolvers. The single-level subproblems are manipulated in each iteration, depending on the solution of the previous subproblem(s). Therefore, there is a significant overlap between the algorithms in terms of the specialized need for subproblem manipulations and operations, e.g., fixing variables, extending the discretization sets, and increasing the dimension of variables. This overlap demands a customizable and general data structure. In the case of libDIPS, the representation and manipulation of algebraic and logical conditions are embodied through libALE. libALE can represent real, index, or boolean-valued expressions and supports sets and tensors of these types. All expressions are stored as expression trees, where nodes are either operations or symbols. The latter includes constant values or named symbols. Named symbols are dereferenced through a symbol table, which links them with the correct values, i.e., parameter values or variables. This allows us to dynamically adjust parameter values or add elements to the discretization sets by using a modified symbol table while keeping a constant expression representation of the subproblem itself.

In addition, the separation of the problem structure from the referenced values through a symbol table also allows for many common tasks in discretization algorithms to be implemented only once because the algorithms primarily interact with the subproblems through high-level operations on the symbol table. Using the example in listing 3.1 and listing 3.2 of Section 3.4.1.1 in the context of *B&F*, the expressions comprising the subproblems are not changed between iterations. Instead, the discretization of the set $\mathcal{Y}^{\mathrm{LBP}}$ in (LBP), which corresponds to the symbol `lbp_y_disc`, is realized through modifying the set-valued entry of the dereferenced symbol `lbp_y_disc` and `lbp_k_disc` in the symbol table. The dereferenced symbol `lbp_y_disc` is populated with the solution of (LLP), and the set `lbp_k_disc` is expanded by one element. Similarly, the upper-level variables in (LLP) can be fixed while using the same variable names as in (LBP). Fixing variables is achieved by a scope-wise definition of the symbol `x` as a variable in the scope of listing 3.1 and as a parameter in the scope of listing 3.2. Instead of keeping modified copies of the symbol table for each scope, we support scoped changes, which are thus reversible. Additional shared tasks include updating parameters (such as the restriction parameter ε^{r}) or setting initial values for the variables.

For requirement (iii), i.e., providing an abstract interface for external optimization solvers (subsolvers), libDIPS utilizes the (sub)problem description as expression trees provided by libALE. As described in Section 3.4.1, libDIPS interfaces several state-of-the-art optimization solvers with vastly different internal problem representations. The conversion of the subproblems from their internal representation in libALE to the respective solver format is realized through a solver-specific parser that utilizes a visitor pattern.

Concerning (iv), more advanced visitor patterns are also implemented for auxiliary functionality, such as printing expressions or for categorizing the problem types, c.f., Section 3.5.

3.4.2.1 Implementation Changes of the Original Algorithms in libDIPS

As revisited in Section 3.2, the implemented algorithms are conceptually closely related. Hence, we were able to enhance the implemented solvers by leveraging ideas from the publications of other algorithms. For example, some of the original publications assume that the subproblems are solved exactly, i.e., the original algorithm statements do not account for the fact that NLPs may only be solved with a given tolerance, while others do.

Additionally, we extended the solvers to, e.g., be able to handle multiple semi-infinite constraints. Appendix C.6 provides a detailed overview of the improvements of the implemented algorithms in libDIPS.

3.5 Library of Test Problems

As mentioned in the introduction, one of the primary objectives of this chapter is to compare the different algorithms across a more comprehensive range of problem instances. For this task, we collected problem instances from the literature of the different problem classes. All problem instances are available in the libDIPS repository. The vast majority of the collected problem instances are academic examples. Nonetheless, the collected examples contain numerous challenging problems that the respective authors specifically chose to highlight and pinpoint specific problems in existing/previous algorithms.

With libDIPS, we provide easy-to-use software for solving hierarchical programs, allowing real-life problem instances to be implemented and, in the future, incorporated into the library of test problems. We encourage such submission to the libDIPS repository. Further, users can easily compare new approaches against the algorithms in libDIPS on the same machine for problems they have selected from the library of test problems. We provide a command-line tool that facilitates such a comparison by running the appropriate implemented algorithms for a user-defined list of problem instances, as described in Section 3.4.1.2.

We want to emphasize that, while this library of test problems is the most comprehensive used in this field, particularly for SIPs, our intention is not to establish it as a fixed benchmark for comparing hierarchical optimization solvers that employ different solution approaches. While our collection of problems, which have already appeared as test problems in the literature, can serve as a basis for creating a benchmark set, it should be expanded through community contributions and carefully curated, similar to the work in more established fields, e.g., Gleixner et al. (2021). More specifically, creating a proper benchmark test set would require both submissions of real-life instances and an understanding of how the problem type, structure, and formulation interact with different solution approaches. Ideally, this would be a collaborative effort of multiple research groups working on different solution approaches, and ideally include a standard problem file format. Unfortunately, this is outside the scope of the work presented in this chapter.

In several cases, the problems could be reformulated to be more suitable for solving with our framework. We did not exploit such individual opinionated problem formulations within the library of test problems; instead, we attempted to keep the problem formulations as close as possible to the initially published problem formulation. We believe this will make it easier to reuse the presented problem instances in other/future works that may not allow for opinionated problem formulations.

However, there are several main changes that we made during problem instance implementation, as we

- added appropriate (sufficiently large) variable bounds if none were provided.
- performed trivial transformations to fit the problem instances in our template, e.g., from minimization to maximization problems.
- performed trivial reformulations to avoid division by zero.
- made reasonable assumptions in case of ambiguous notation or typos.
- replaced open or half-open host sets by closed host sets because all implemented solvers assume closed host sets.
- normalized minmax approximation problems in the MINMAX program category to use the same squared objective. Specifically, $\min_{\boldsymbol{x}} \max_{\boldsymbol{y}} |e(\boldsymbol{x}, \boldsymbol{y})|$ was changed to the equivalent formulation $\min_{\boldsymbol{x}} \max_{\boldsymbol{y}} (e(\boldsymbol{x}, \boldsymbol{y}))^2$ to be consistent with problems with ambiguous formulation in the original literature and to avoid creating artificial duplicates. The latter formulation is chosen as it is preferable for the used subsolver.

We noted further non-trivial changes as supplementary information in the `def.txt` file of the corresponding problem instances.

We employed a two-step heuristic procedure to avoid duplicate problem instances in the presented library of test problems. First, we automatically searched for potential duplicates, i.e., problem instances with the same number of variables, and close optimal objective values or solution points. Second, we manually compared the potential duplicates. When we found a duplicate, we named the problem instance according to the older published source and added the newer source as supplementary information to the `def.txt` file. For the library of BLP test problems, we automatically checked for symbolic equivalence of the objective functions to narrow down the number of potential duplicates in the first step. We also marked problems that are nearly duplicates as similar, e.g., when the problem instances are almost identical but differ only by an additional constraint in the upper or lower level, or if the problem instances were created by a parametric formulation, e.g., in the number of variables. Nevertheless, both problem instances were added to the library of problem instances, even if the change did not change the optimal solution, as equivalence is not straightforward to derive from the problem statement, and these changes might influence the performance of the algorithms. We proceeded in the same manner when an original problem instance was cited, but the presented problem instance was different. For similar problem instances, we added a remark in the corresponding `def.txt` files. We ensured that the similarity relationships expressed by these remarks are transitive and symmetric.

For SIPs, we started with the existing test sets from Mitsos (2011), Vaz et al. (2004b), Watson (1983). Additionally, we have included problem instances from various publications. Table 3.4 shows a complete list of the used publications. Note that in Table 3.4 and all following listings, only the oldest found publication with the implemented problem instance is listed. After removing duplicates, the library of test problems contains 336 SIPs; for a complete list of implemented problem instances, see Table C.2 in Appendix C.9.

For MINMAX, we collected problem instances from numerous publications. After removing duplicates, the library of test problems contains 83 problem instances of various publications; for a complete list of implemented problem instances and summary of used publications, see Table 3.4 and Table C.3 in Appendix C.9, respectively.

For GSIPs, we started with the existing test set of Mitsos and Tsoukalas (2015) and added problem instances from GSIPLib&Gen (Seidel and Schwientek, 2017). The problem instances taken from Seidel and Schwientek (2017) were reformulated to be compatible with our framework. We also added additional problem instances from various other publications. This led to a total of 86 problem instances after removing duplicates; for a complete list of implemented problem instances and summary of used publications, see Table 3.4 and Table C.4 in Appendix C.9, respectively.

For BLPs, we started with the problem instances from Mitsos et al. (2008b) and combined them with the existing benchmark test set BASLib (Paulavičius and Adjiman, 2020) and BOLIB v2 (Zhou et al., 2020). For some of the problem instances, we observed that there are constraints implemented as part of the lower-level feasible set $\mathcal{Y}(\boldsymbol{x})$ but only depend on the upper-level variables $\boldsymbol{x}$. We assumed that the original intent was to include these constraints as upper-level ones. In these cases, we moved these constraints to the definition of the upper-level feasible set. Our motivation for this transformation is that these constraints can be trivially detected; without it, the solvers encounter numerical issues or performance losses. Additionally, this detail does not change the feasible set of individual problem instances. After removing duplicates, we obtained 167 problem instances by combining the existing benchmark test sets.

We categorized all problem instances according to the problem class of their respective ULP and LLP using the following categories

- *LP* – linear objective function subject to linear constraints
- *QP* – quadratic objective function subject to linear constraints
- *QCQP* – quadratic objective function subject to quadratic constraints
- *NLP* – all problem instances not fitting the aforementioned problem categories

Note that we do not differentiate in terms of convexity because the convexity of the subproblems can vary from iteration to iteration. For example, consider the expression xy^2 in the context of the semi-infinite constraint in (SIP) with x being an upper-level variable and y being a lower-level variable. The convexity with respect to the lower-level variables y depends on the sign of the fixed upper-level variable, which might change in each iteration. The resulting composition of the library of test problems is shown in Table 3.5, which also indicates the (small) number of problems that include discrete variables.

Table 3.4: List of publications used in the library of test problems.

Problem Class	Used Publications (Oldest Found)
SIP	Anderson and Lewis (1989), Barragán and Camacho-Vallejo (2022), Ben-Tal and Nemirovski (1999), Betro (2004), Bhattacharjee et al. (2005a,b), Blankenship and Falk (1976), Bonnans and Shapiro (2000), Cánovas et al. (1999), Cerulli et al. (2022), Coope and Price (1998), Diehl et al. (2013), Fang et al. (1994), Floudas and Stein (2008), Glashoff and Gustafson (1983), Goberna and López (1998), Grossmann and Sargent (1978), Guo and Sun (2020), Guo and Zhang (2023), Gustafson (1979), Gustafson and Kortanek (1973), Harwood (2019), Hettich and van Honstede (1979), Hettich and Zencke (1982), Kostyukova and Tchemisova (2010), Lemonidis (2008), Leon and Vercher (1992), Li et al. (2004), Lin et al. (1994), Liu and Wang (2013), Liu et al. (1999), Lo Bianco and Piazzi (2001), López and Still (2007), Mehrotra and Papp (2014), Mitsos (2009), Mitsos et al. (2008b), Pang and Wu (2022), Pang et al. (2016), Polak et al. (1999), Powell (1989), Price (1992), Price and Coope (1996), Qi et al. (2003), Rückmann and Shapiro (2009), Rudnick-Cohen et al. (2020), Schwientek et al. (2021), Seidel and Küfer (2022), Spjøtvold et al. (2005), Stein and Steuermann (2012), Still (2001b), Streit and Nuttall (1982), Su et al. (2018), Tanaka et al. (1988a), Teo et al. (2000), Tichatschke and Nebeling (1988), Tsoukalas and Rustem (2011), Vaz et al. (2004b), Watson (1975, 1983), Wu et al. (2005), Xu et al. (2013), Žaković and Rustem (2003), Zhang et al. (2010), Zhou and Tits (1996)
GSIP	Barragán and Camacho-Vallejo (2022), Chen (2013), Diehl et al. (2013), Harwood (2019), Harwood and Barton (2017), Jongen et al. (1998), Kanzi (2013), Lemonidis (2008), Li (2003), Liu and Wang (2013), Mitsos and Tsoukalas (2015), Royset et al. (2003), Rückmann and Shapiro (1999, 2001), Schwientek et al. (2021), Seidel and Schwientek (2017), Selassie (2004), Stein (2003), Stein and Steuermann (2012), Still (1999, 2001a, 2002), Tezel Özturan (2015), Tsoukalas et al. (2009), Vázquez and Rückmann (2005)
BLP	Mitsos et al. (2008a), Paulavičius and Adjiman (2020), Zhou et al. (2020)
MINMAX	Andreassen and Watson (1976), Atkinson and Federov (1975), Barrodale et al. (1978), Curtis and Powell (1966), Diehl et al. (2013), Duarte et al. (2015), Ferris and Philpott (1989), Glashoff and Gustafson (1983), Görner (1997), Guo and Sun (2020), Hettich (1976a, 1979, 1986), Hettich and Zencke (1982), Mehrotra and Papp (2014), Pang et al. (2016), Žaković and Rustem (2003), Zhou and Tits (1996)

Table 3.5: Composition of the libraries of test problems for the different problem categories. UL = problem class of the upper level, specifically the respective (LBP) with a given (non-empty) discretization, LL = problem class of the lower level with fixed upper-level variables.

(a) Library of SIP test problems. For a breakdown of the number of SIP test problems with discrete variables, see Table 3.5b.

UL \ LL	LP	QP	QCQP	NLP
LP	22	22	1	75
QP	1	47	2	21
QCQP	2	12	5	11
NLP	9	3	3	100

(b) Breakdown of the number of SIP test problems included in Table 3.5a, with discrete variables in the upper/lower level, in that order.

UL \ LL	LP	QP	QCQP	NLP
LP	10|0	3|0	0|0	1|2
QP	0|0	0|0	0|0	0|0
QCQP	0|1	0|0	0|0	0|0
NLP	0|0	0|0	0|0	3|4

(c) Library of MINMAX test problems. The library includes one problem instance with integers in the lower-level (shown in brackets).

UL \ LL	LP	QP	QCQP	NLP
LP	0	1	0	1
QP	0	0	0	0
QCQP	0	13	0	44
NLP	0	3	0	21 (1)

(d) Library of GSIP test problems. The library does not include any integer problem instances.

UL \ LL	LP	QP	QCQP	NLP
LP	11	0	4	4
QP	8	3	6	1
QCQP	6	7	8	8
NLP	1	0	4	15

(e) Library of BLP test problems. The library does not include any integer problem instances.

UL \ LL	LP	QP	QCQP	NLP
LP	31	0	0	0
QP	7	0	0	0
QCQP	18	41	7	0
NLP	2	9	10	42

3.6 Numerical Experiments

All calculations were conducted on the RWTH High-Performance Computing cluster running Rocky Linux 8. No parallelization was used, and each computation was performed on a single core with an Intel Xeon Platinum 8160 Processor "SkyLake" running at 2.1 GHz with up to 20 Gbit RAM. This precluded the calculation for five large instances from Cerulli et al. (2022), which are not included in the following results. All subproblems are solved using MAiNGO version 0.7.1, configured to utilize CPLEX version 22.1.1 to solve linear or quadratic problems. We use MAiNGO for the numerical experiments for two main reasons: Firstly, as MAiNGO is open-source, we configured the libDIPS compilation file to compile MAiNGO and automatically make it a subsolver. As a result, MAiNGO is our default solver. Secondly, MAiNGO supports trigonometric functions as well as the functions abs, min, and max, which occur in some of the problem instances. For example, BARON does not support trigonometric functions. The maximum CPU time per problem instance was set to 20 min. If a problem instance is not solved within that time, the solution procedure is aborted, and the instance is considered not solved (CPU time is set to infinity). The absolute inequality tolerance `deltaIneq` of MAiNGO was set as small as possible to 1×10^{-9}. In Section 3.6.1, to follow, we shortly discuss the implications of `deltaIneq` being > 0. Note that for some automatically generated subproblems, we use specific fixed settings to speed up the convergence of MAiNGO. For example, to compute an initial UBD in *Oracle*, the objective function is maximized absent any constraints, and the relative optimality tolerance `rel_tol` of MAiNGO is set to $\max\{\texttt{rel_tol_lbp}, 0.5\}$. For other subproblems, users can easily set the optimality tolerances. The specific tolerances used for each solver can be found in Table C.1 in Appendix C.7. All other settings of MAiNGO, except for the output settings, have been left at their default values.

Within the chapter, we primarily focus on time as the relevant quantity, as we investigate global deterministic optimization algorithms. We predominantly show, discuss, and compare the performance of the individual solvers within a problem class utilizing Dolan-Moré profiles; we plot the empirical cumulative distribution functions of the ratio of solve CPU time for a given solver against the virtual best, defined as

$$\textit{CPU time performance ratio} = \frac{\textit{CPU time of problem instance } i}{\textit{CPU time of problem instance } i \textit{ of the fastest solver}}. \tag{PR}$$

Our motivation for using CPU time performance ratio profiles is to minimize the importance of problem instance size.

While CPU time performance ratio profiles are widely used and allow for a comprehensive and visual comparison of different algorithms over larger problem sets, they also have disadvantages. For problems with very small solution times, even minor absolute time differences can result in a significant difference in terms of the CPU time performance ratio. To combat this, all CPU times under 1 second (s) have been rounded up to 1 s as we can not fairly differentiate relative runtime measurements for run times under 1 s. Furthermore, CPU time performance ratio profiles can be sensitive to the addition or removal of considered algorithms. However, since the performance profiles based on absolute run times align with our conclusions, we contend this sensitivity does not affect the validity of our findings. Due to the known disadvantages, performance profiles for other metrics, such as CPU time and

the number of subproblems solved, as well as additional plots, are included in Appendix C.8 and referenced at the appropriate places in the text.

We briefly investigated the consistency of the measured run times by measuring three repeated runs of *RRHS* on the SIP library of test problems. On average, the standard deviation over the three repeated runs, normalized by the mean, is 0.02 and the maximum of this normalized standard deviation is 0.16. Since we consider the deviations insignificant, we refrain from repeating the computations of the more extensive computational studies, which are discussed later in this section.

We ascertained that there is relatively little overhead outside of the subsolver MAiNGO, i.e., the time spent in our C++ code outside the subsolver is negligible compared to the CPU time spent solving the subproblems by the subsolver. For normal-sized problems, on average 99 % of the total computation time (wall clock time) is spent solving the subproblems in MAiNGO.

In the following subsections, we present several numerical experiments examining the performance of the implemented solvers across different problem classes with varying parameters. We use the problem instances from the library of test problems presented in Section 3.5 for the comparison, excluding the large instances from Cerulli et al. (2022) that could not be run due to memory limitations. Since *B&F* terminates with SIP-ε^{a}-feasible points (with $\varepsilon^{\mathrm{a}} > 0$), we exclude *B&F* from the performance tests.

Anticipating the results of the following subsections, in summary, the solvers reliably solve most collected problem instances within their respective problem classes. However, we found that, on the one hand, the library of test problems contained many problem instances that were quickly solvable (global solution found in less than 10 s) for all solvers with the considered parameter values. On the other hand, many problem instances could not be solved by any solver within the time limit.

We briefly investigated whether a reliable prediction regarding problem difficulty, i.e., if a given problem instance is *solved quickly* (solved in < 10 s) by all solvers or remains *unsolved* within the time limit (20 min), can be based on easily observable problem instance characteristics. The considered problem instance characteristics are the subproblem class, the number of variables at each level, and the number of distinct constraint types. We attempted to classify the problem instances based on these characteristics using a decision tree with monotonicity constraints. Unfortunately, this analysis did not produce much insight. This is not surprising, as predicting problem difficulty straightforwardly is already challenging for nonconvex NLPs. It remains to be seen whether more sophisticated analyses of problem properties for nonconvex NLPs (to be developed) will also allow a deeper understanding of the factors related to the complexity of hierarchical optimization programs. The trends that we could derive are

- SIPs with less than four upper-level variables are likely to be solved *quickly* (only 20 % of SIPs with less than four upper-level variables are not solved *quickly* and 16 % of SIPs with more than four upper-level variables are solved *quickly*)
- SIPs where the lower-level is not an NLP are likely to be *solved*; ignoring the 25 unsolved large-scale instances from Cerulli et al. (2022) (4 % of *unsolved* problem instances have an LLP that is not an NLP; less than 1 % of problem instances with an LLP that is not NLP are *unsolved*)

- GSIPs with less than four upper-level variables are likely to be *solved* (10 % of such GSIP are *unsolved*, 3 % of *unsolved* cases have less than four upper-level variables)

A breakdown by the introduced categorization by subproblem type is shown in Table 3.6.

Table 3.6: Number of problem instances that are always solved *quickly*, i.e., solved in $< 10\,\mathrm{s}$, and problem instances that are *unsolvable* within the time limit across all numerical experiments for the different problem categories, in that order. UL = problem class of the upper level, specifically the respective (LBP) with a given (non-empty) discretization, LL = problem class of the lower level with fixed upper-level variables.

(a) Library of SIP test problems.

UL \ LL	LP	QP	QCQP	NLP
LP	22|0	18|0	0	45|7
QP	1|0	3|25	1|0	5|2
QCQP	2|0	9|0	4|0	6|2
NLP	7|1	2|0	1|0	33|16

(b) Library of GSIP test problems.

UL \ LL	LP	QP	QCQP	NLP
LP	4|1	0	1|0	3|0
QP	2|1	0|2	2|0	0
QCQP	0|4	1|4	2|1	3|1
NLP	0	0	1|0	3|7

(c) Library of MINMAX test problems.

UL \ LL	LP	QP	QCQP	NLP
LP	0	1|0	0	0
QP	0	0	0	0
QCQP	0	7|0	0	12|7
NLP	0	1|0	0	3|11

(d) Library of BLP test problems.

UL \ LL	LP	QP	QCQP	NLP
LP	5|3	0	0	0
QP	2|1	0	0	0
QCQP	2|5	2|7	2|0	0
NLP	0|1	3|2	2|2	15|4

We want to emphasize that the summary of our empirical results presented above, as well as the detailed results in the following subsections, depend on both the library of test problems used and the subsolver employed. To the best of our knowledge, the library of test problems used is the largest in the known literature to date. Still, it can not be ruled out that we will obtain different empirical results with another (even more) representative library of test problems. This is especially the case for problems with discrete variables because our library of test problems only includes 24 SIPs and 1 MINMAX problem instances with discrete upper- and/or lower-level variables, c.f., Table 3.5.

3.6.1 Definition of Feasibility and Numerical Challenges

For SIPs and GSIPs, we use relatively strict criteria for an upper-level point $\bar{\boldsymbol{x}}$ to be considered feasible. In both problem classes, an upper-level point $\bar{\boldsymbol{x}}$ is only feasible if the objective value at the solution of the LLP is less or equal to zero. In the following experiments, we thus do not consider any explicit tolerance on this constraint. This is in contrast to the case of BLPs, where we allow a corresponding tolerance, c.f., Definition 5 in Appendix B.1, of 1×10^{-4} which is likely unavoidable in the nonconvex case (Mitsos, 2006).

Consistent with this strict feasibility requirement in the case of SIPs and GSIPs, we set the numerical tolerance for inequality constraints as small as possible in the ULPs ; recall in MAiNGO this corresponds to `deltaIneq` $\geq 1 \times 10^{-9}$. As discussed in Beck et al. (2023) for the case of BLPs, constraint tolerances in the solution of the LLPs can significantly impact the solution of the overall problem when nonlinear constraints are considered. We thus also consider this strict constraint tolerance in the LLPs .

Note that we adaptively refine the optimality tolerances used for the lower level as outlined in Djelassi and Mitsos (2017) for the SIP case, c.f., Appendix C.6. While being necessary, we observed that some numerical issues and long computation times are related to this refinement. For some problems, later iterations require smaller and smaller optimality tolerances in the lower level, as it becomes more likely to be inconclusive, i.e., the computed LBD and UBD on the optimal objective value of (LLP) bracket zero. In these cases, no statement can be made whether the candidate point $\bar{\boldsymbol{x}}$ of (LBP) is SIP-feasible, c.f., Definition 4 in Appendix B.1.

Introducing looser tolerances bears the possibility of deviating from the rigorous mathematical formulation, which was thus avoided in this chapter. Still, a user in a specific application should critically evaluate their decision on the chosen tolerances, as they may be overly restrictive compared to the model accuracy and significantly influence the solution time or even solvability.

Another frequent numerical challenge arises due to large coefficients, for example, when working with relatively wide variable bounds in conjunction with large powers. For example, in Wu et al. (2005), the following SIP is posed

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X}=\mathbb{R}^{20}} \quad & \boldsymbol{x}^T\boldsymbol{x} \\ \text{s.t.} \quad & g(\boldsymbol{x},y) = 3 + 4.5\cos\left(\tfrac{4.7\pi(y-1.23)}{8}\right) - \textstyle\sum_{i=1}^{20} x_i y^{i-1} \leq 0, \quad \forall y \in [0,200]. \end{aligned}$$

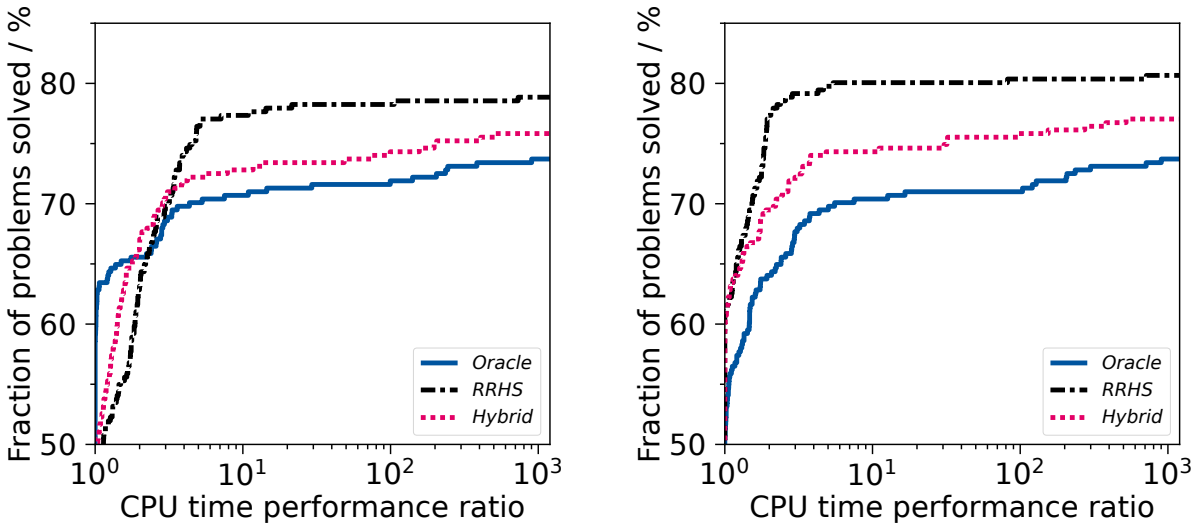
All considered SIP solvers fail for this problem, even after bounding the domain of the upper-level variable to $\mathcal{X} = [-100,100]^{20}$. The initial upper-level point is $\boldsymbol{x}^0 = \mathbf{0}$. If the smallest lower-level optimizer $y^0 \approx 1.23$ is found, some of the approaches converge. However, if another global optimizer of the LLP, e.g., $y^0 \approx 4.63$, is selected, the resulting coefficient of x_{20} is so large that the solution of the next ULP can not be numerically differentiated from $\mathbf{0}$, leading to an infinite loop.

3.6.2 Performance Results of SIP Solvers

3.6.2.1 "Out of the box performance"

Figure 3.2a shows the "out of the box performance" of the SIP solvers with their default parameters, i.e., the parameters of each solver are set to the values of their respective original publication, c.f., Table 3.7. For completeness, we provide additional performance profiles in Appendix C: for CPU time, subproblem number, and subproblem ratio performance performance profiles see Figures C.1a, C.2a and C.2c in Appendix C.8.1, for performance profiles of problem instances with discrete variables in the upper and/or lower level see Figure C.6 in Appendix C.8.1.3, for performance profiles of problem instances with purely continuous variables see Figures C.4 and C.5 in Appendix C.8.1.2. As outlined in Section 3.3, discrete variables do not pose a challenge for the SIP solvers. However, it remains to be seen if this result also holds for a more extensive library of test problems.

Quantitatively, the performance of the solvers on our more extensive library of test problems is similar to the results published in Djelassi (2020): *Oracle* is the fastest for the largest fractions of problems. Still, when it is not the fastest, it is often significantly slower and is the least robust in the sense that it solves the fewest problems for large CPU time performance ratios within the time limit. In contrast, *RRHS* can solve most problems within the time limit and solves most problems when allowing for CPU time performance ratios exceeding 10. However, in this test, it is the fastest solver for the least number of problems. The solver *Hybrid* can solve significantly more problems than *Oracle* and outperforms *RRHS* for the time-factor range up to factor 3. Overall, the *Hybrid* solver is the most well-rounded with default parameters. However, as shown in Figure 3.2b, after parameter tuning, *RRHS* is likely the better choice. In the following, we take a closer look at the impact of tuning on two solvers: *RRHS* and *Hybrid*.



(a) "Out of the box performance;" parameters as in the original publications.

(b) Solvers with tuned parameters.

Figure 3.2: CPU time performance ratio profiles of SIP solvers: *RRHS*, *Oracle*, and *Hybrid*. Without tuning, a clear trade-off between the fraction of problems solved overall and the fraction of problems solved in the shortest time exists between the solvers. With tuning *RRHS* performs best. Note: Adapted from Jungen et al. (2023a).

3.6.2.2 Parameter Tuning for *RRHS* and *Hybrid*

Recall that the performance of the upper-bounding procedure, and hence overall performance, of *RRHS* and *Hybrid* is influenced by the initial restriction parameter $\varepsilon^{\mathrm{r},0}$; the rate of reduction, i.e., $\varepsilon^{\mathrm{red}}$; and the initial discretization. Therefore, we performed a parameter study for *RRHS* and *Hybrid* varying the two parameters $\varepsilon^{\mathrm{r},0} = \{0.1, 1, 2, 5, 8, 10\}$ and $\varepsilon^{\mathrm{red}} = \{1.2, 1.5, 2, 2.5, 5, 10\}$. The results of the parameter study are shown in Figures 3.3 and 3.4.

RRHS, for a fixed $\varepsilon^{\mathrm{r},0} = 1$ (value in original publication Mitsos (2011)), performs best with a fast reduction of ε^{r}, c.f., Figure 3.3a. Furthermore, a small initial restriction $\varepsilon^{\mathrm{r},0}$ is beneficial, c.f., Figure 3.3b. Hence, choosing a big value for $\varepsilon^{\mathrm{red}}$ and a small $\varepsilon^{\mathrm{r},0}$ seems

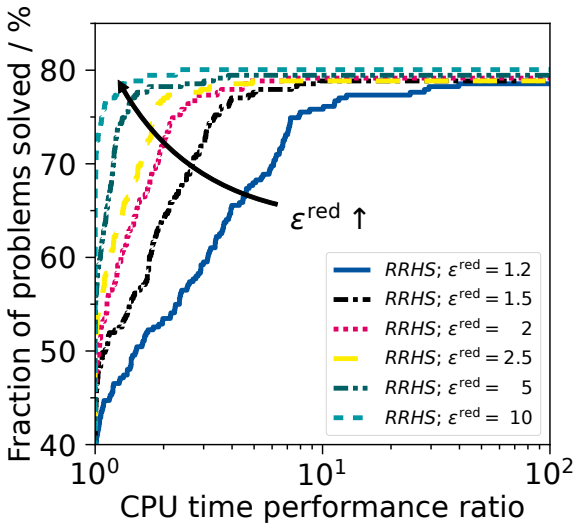
Table 3.7: Parameter values used in the original publications (Djelassi and Mitsos, 2017, Mitsos, 2011, Mitsos and Tsoukalas, 2015, Tsoukalas and Rustem, 2011) and tuned parameter values. *GSIP-RRHS*: Note that Mitsos and Tsoukalas (2015) introduces restriction and reduction parameters that may differ between $\boldsymbol{g}^{\mathrm{u}}$ and for $\boldsymbol{g}^{\mathrm{l}}$; however Mitsos and Tsoukalas (2015) uses the same values for both. $*$: α^{red} is introduced by us.

Problem Class	Solver Name	Paremeters in Original Publication	Tuned Parameters
SIP	*RRHS*	$\varepsilon^{\mathrm{r},0} = 1$, $\varepsilon^{\mathrm{red}} = 1.5$	$\varepsilon^{\mathrm{r},0} = 0.1$, $\varepsilon^{\mathrm{red}} = 10$
	Oracle	–	–
	Hybrid	$\varepsilon^{\mathrm{r},0} = 1$, $\varepsilon^{\mathrm{red}} = 1.2$	$\varepsilon^{\mathrm{r},0} = 0.1$, $\varepsilon^{\mathrm{red}} = 10$
GSIP	*GSIP-RRHS*	$\varepsilon^{\mathrm{r},0} = 1$, $\varepsilon^{\mathrm{red}} = 2$, $\alpha^{0} = 0.5$, $\alpha^{\mathrm{red}} = *$	$\varepsilon^{\mathrm{r},0} = 1$, $\varepsilon^{\mathrm{red}} = 2$, $\alpha^{0} = 0.25$, $\alpha^{\mathrm{red}} = 1.2$
	RRHS	–	$\varepsilon^{\mathrm{r},0} = 5$, $\varepsilon^{\mathrm{red}} = 5$
	Oracle	–	–
	Hybrid	–	$\varepsilon^{\mathrm{r},0} = 0.1$, $\varepsilon^{\mathrm{red}} = 5$

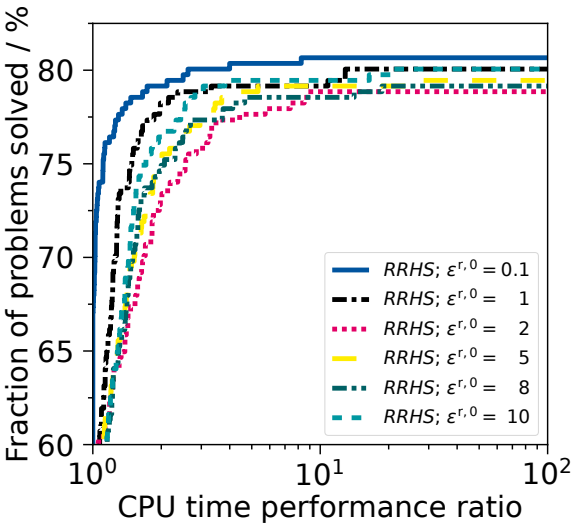
beneficial for the considered library of SIP test problems. With tuned parameters, i.e., $\varepsilon^{\mathrm{red}} = 10$ and $\varepsilon^{\mathrm{r},0} = 0.1$, *RRHS* was able to solve 81 % of the problem instances within 20 min. For a detailed listing of the reasons why the remaining problem instances were not solved, refer to Table 3.8.

Hybrid, for a fixed $\varepsilon^{\mathrm{r},0} = 1$ (value in original publication Djelassi and Mitsos (2017)), performs best with a fast reduction of ε^{r}, c.f., Figure 3.4a. Furthermore, a small initial restriction $\varepsilon^{\mathrm{r},0}$ is beneficial, c.f., Figure 3.4b. With tuned parameters, i.e., $\varepsilon^{\mathrm{red}} = 10$ and $\varepsilon^{\mathrm{r},0} = 0.1$, *Hybrid* was able to solve 77 % of the problem instances within 20 min. For a detailed listing of the reasons why the remaining problem instances were not solved, refer to Table 3.8.

Figure 3.2b shows that after parameter tuning *RRHS* outperforms the other solvers (for the CPU time, subproblem number, and subproblem ratio performance profiles see Figures C.1b, C.2b and C.2d in Appendix C.8.1). Furthermore, by comparing the sensitivity of the solvers *RRHS* and *Hybrid* (Figures 3.3 and 3.4), we can conclude that *Hybrid* is less sensitive to the choice of the parameters. This is most likely due to the additional subproblem (RES), which is used to generate *optimal* updates for the restriction parameter ε^{r}. However, this update procedure comes with additional computational costs and can not compensate for the reduced computational costs of *RRHS* if the *tuned* parameters are chosen. A reason for the additional computational costs might be that *Hybrid* solves the subproblem (RES), which, beyond being an additional subproblem, might belong to a different problem class than (LBP). For example, if (LBP) is a QP, (RES) becomes a QCQP. However, we also noticed instances where (RES) took much longer than the respective (UBP), even though the ULP was a general NLP. We were unable to confirm a structural reason for this behavior. In summary, the performance of *RRHS* is superior compared to the other solvers, especially if parameter tuning can be performed.

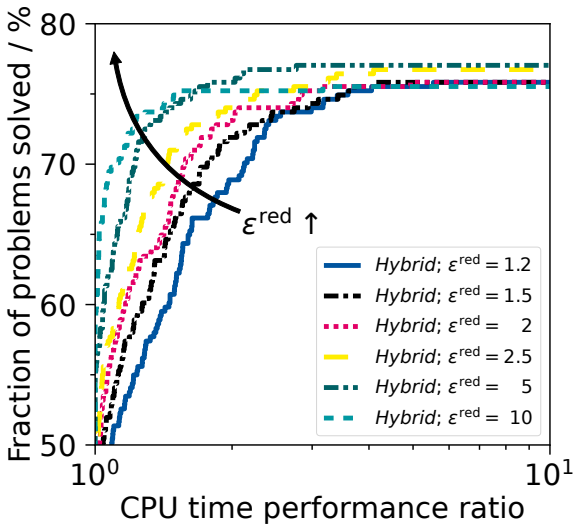


(a) *RRHS*: $\varepsilon^{\text{r},0} = 1$ and $\varepsilon^{\text{red}} = \{1.2, 1.5, 2, 2.5, 5, 10\}$.

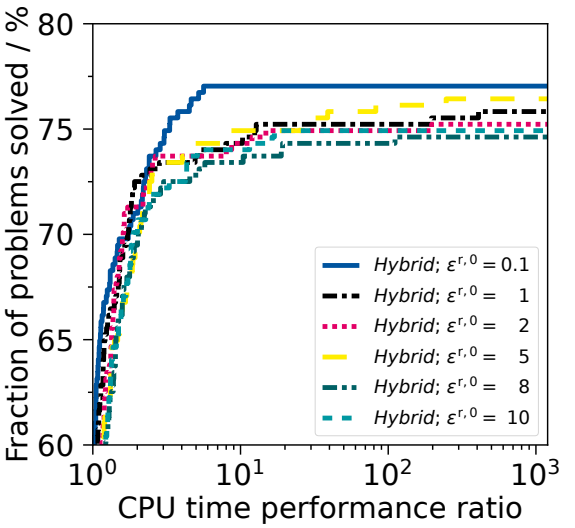


(b) *RRHS*: $\varepsilon^{\text{r},0} = \{0.1, 1, 2, 5, 8, 10\}$ and $\varepsilon^{\text{red}} = 10$.

Figure 3.3: CPU time performance ratio profiles of the SIP parameter study for *RRHS*. Performance improves with increasing ε^{red}. Note: Adapted from Jungen et al. (2023a).



(a) *Hybrid*: $\varepsilon^{\text{r},0} = 1$ and $\varepsilon^{\text{red}} = \{1.2, 1.5, 2, 2.5, 5, 10\}$.



(b) *Hybrid*: $\varepsilon^{\text{r},0} = \{0.1, 1, 2, 5, 8, 10\}$ and $\varepsilon^{\text{red}} = 10$.

Figure 3.4: CPU time performance ratio profiles of the SIP parameter study for *Hybrid*. Performance improves with increasing ε^{red}. No systematic behavior is seen for $\varepsilon^{\text{r},0}$. Note: Adapted from Jungen et al. (2023a).

Table 3.8: Number of errors encountered in the computational study using tuned parameters, c.f., Table 3.7. The subsolver errors category encompasses problem instances where MAiNGO failed to solve a subproblem globally or MAiNGO threw an exception. The algorithmic errors category includes problem instances where the maximum number of iterations `max_iter` was exceeded or where algorithmic parameters, e.g., ε^{r}, were reduced below a certain threshold, e.g., `min_eps_res`, c.f., Table C.1 in Appendix C.7.

Problem Class	Solver Name	Non-Solved / Total	Exceeded max. Time	Subsolver Errors	Algorithmic Errors
SIP	*RRHS*	64/331	33	18	13
	Oracle	87/331	36	41	10
	Hybrid	76/331	35	28	13
GSIP	*GSIP-RRHS*	34/ 86	8	12	15
	RRHS	26/ 86	8	17	1
	Oracle	24/ 86	6	18	–
	Hybrid	26/ 86	6	19	1
BLP	*BLP-noBox*	28/167	6	17	5
	BLP-Box	84/167	31	41	12
MINMAX	*Minmax*	20/ 83	–	20	–

3.6.2.3 Results on a Reduced SIP Test Set

Our library of SIP test problems comprises numerous problem instances from the literature, and curation is only carried out to a basic level, as described in Section 3.5. In addition to the previously mentioned lack of real-world instances, the results could be skewed by "clusters" of very similar problem instances, which would bias the results in favor of solvers and parameter sets that work exceptionally well for these "clusters."

It is, however, rather difficult to quantify the kind of similarity that would cause such a bias. For example, as previously noted, we observed that adding a single simple constraint to a problem can significantly impact the problem's solution or the solver's behavior. We have already mentioned that the results obtained with standard parameters are qualitatively similar to those obtained on a smaller test set in Djelassi (2020).

Nevertheless, for SIPs, we investigate the stability of the results for solvers with tuned parameters when using a diverse subsample of the problem set, i.e., a reduced test set. For BLPs, we use two established benchmark sets. For GSIPs and MINMAX, the problem sets are already relatively small, so further reducing the test set would likely impair the validity of the reduced test set.

To create the reduced test set, we first performed a prefiltering step, where we retained only a single representative from a group of problems marked as similar. For problem instances parameterized in the number of variables, only the largest problem instance is considered a candidate in this step. This resulted in 228 problem instances.

Then, we used two approaches to generate a diverse subsample of problem instances. The first is based on k-medoids clustering, utilizing the problem class of upper and lower levels, as well as the number of discrete and continuous variables as features. While the

shifted geometric means (with a shift of 10 s) of *RRHS*, *Hybrid*, and *Oracle* over the cluster medoids change when varying the number of clusters, the qualitative results stay consistent after $k = 50$: *RRHS* performs best, *Hybrid* performs slightly worse than *RRHS*, and *Oracle* performs worst. The results for $k = 50$ are shown in Table 3.9. In the second approach, we used a binning-based subsampling approach. In contrast to the k-medoid approach, which is not suitable for a high number of features, we incorporated additional features: the number of inequality and equality constraints in the respective levels and the number of semi-infinite constraints. We divided the numerical features into five quantiles, assigning each problem to a bin depending on its features. We selected our subsamples by choosing a problem from each bin. For bins containing multiple problems, a single representative was selected randomly. Still, if possible, a problem was chosen that was solved within the time limit by at least one solver in our tests, and for all solvers, more than one second was required to solve it.

From the results displayed in Table 3.9, we conclude that qualitatively, our ranking of the solvers with tuned parameters is not skewed by clusters of similar problems. We also observe that the inclusion of large-scale instances from Cerulli et al. (2022) significantly alters the geometric means, as most of these problems remain unsolved or require a comparatively long time to solve. Comparing the geometric means over the problems excluding the ones from Cerulli et al. (2022), we see that there is no considerable change with prefiltering, i.e., only considering a single instance from similar problems. One explanation is that, as mentioned, even minor modifications, such as an additional linear constraint in either level, can change the behavior of the solvers. Additionally, we found that the separate groups of similar problems were diverse, suggesting that there is no overall bias towards a specific structure. *Oracle* seems to perform worse on the reduced problem sets. This appears to support our analysis, indicating that it is especially suitable for specific problems.

Table 3.9: Shifted geometric means (with a shift of 10 s) of run times over different sets of problems for the different SIP solvers with tuned parameters. Measured times are rounded up to 1 s.

	Full Problem Set	Without Cerulli et al. (2022)	Pre-filtered Problem Set	50-Medoids	Bin-Based
RRHS	30.4 s	18.2 s	17.8 s	14.3 s	21.3 s
Hybrid	35.4 s	21.6 s	21.2 s	20.2 s	22.7 s
Oracle	44.9 s	28.2 s	27.1 s	31.4 s	38.0 s

3.6.2.4 Performance on Unsolved SIP Problem Instances

We also investigated the behavior of the SIP solvers with tuned parameters (c.f., Table 3.7) in problem instances where the solver(s) could not solve the problem instance within the given time limit of 20 min. Suppose a solver is unable to solve a problem instance up to the specified optimality tolerance. In that case, we are interested in the remaining objective value gap and LBD on the objective. Naturally, bounds on the optimal objective value are preferred over not providing any bounds at all, even if the provided bounds on the optimal

objective value are looser than the optimality tolerances used for termination. However, no UBD, i.e., feasible solution, is provided by any solver for most problem instances that are unsolved by all considered solvers (with tuned parameters) within the time limit.

For problem instances where at least one of the solvers failed to solve the instance within the time limit, we plot in Figure 3.5 the remaining gap given by the minimum of the absolute and relative difference of the UBD (UBD) and the and LBD (LBD) on the objective, i.e.,

$$Gap = \min\left\{UBD - LBD, \frac{UBD - LBD}{UBD}\right\}.$$

Similar to the results over all problem instances (solved and unsolved), c.f., Figure 3.2b,

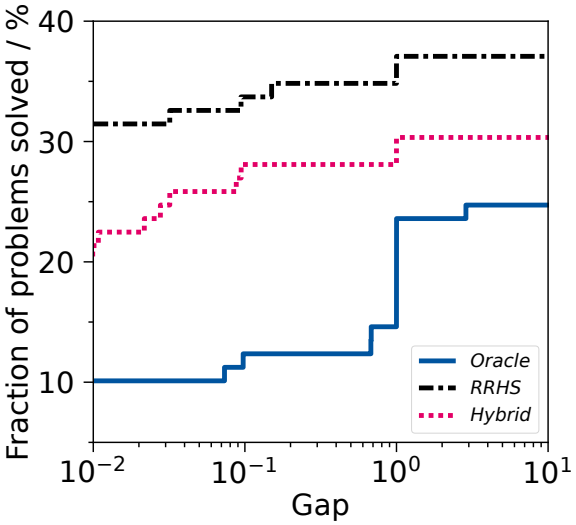


Figure 3.5: Fraction of problem instances that are not solved by one of *RRHS*, *Hybrid*, and *Oracle* but would have been considered solved with a higher allowed absolute and relative objective tolerance according to the minimum of the final absolute and relative gap. Solvers with tuned parameters, c.f., Table 3.7. Note: Adapted from Jungen et al. (2023a).

RRHS solved the most problems with the prescribed gap of 1×10^{-2}. We see that the number of solved problems and the results for large CPU time performance ratios in Figure 3.2 are not sensitive to an increase in the optimality tolerance. All solvers seem to behave similarly concerning the ability to solve an instance within the given time limit with looser termination tolerances. Assuming the problem instance is solved, we usually observe that looser termination tolerances generally facilitate faster convergence. Therefore, users should usually critically assess the necessity of (strict) tolerances. However, it appears that loosening termination tolerances does not significantly enhance solvability.

3.6.2.5 Performance Result of *Minmax*

Figure 3.6 shows the performance of *Minmax* on the library of MINMAX test problems. We refrained from a comparison to SIP solvers using the reformulation given in (MINMAX-REF) as the specialized algorithm is preferable: The specialized algorithm provides the best upper-bound for each candidate point of the upper-level variables $\bar{\boldsymbol{x}}$ and the problems (RES), (ORA), and (UBP); used in the upper-bounding of the respective solvers; will not produce different candidate points (except when there are multiple possible global

solutions, in which case better performance would only be due to randomness). *Minmax* was able to solve 79 % of the problem instances within 20 min. The remaining 21 % of problem instances were not solved due to subsolver errors, c.f., Table 3.8.

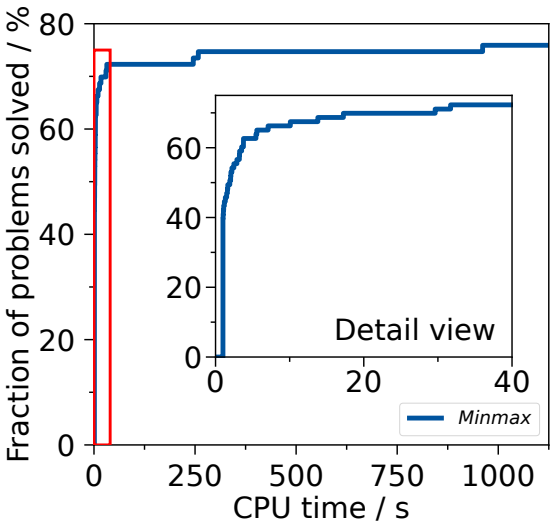


Figure 3.6: CPU time performance profiles of *Minmax*. Most problems are solved within 40 s. Note: Adapted from Jungen et al. (2023a).

3.6.3 Performance Results of GSIP Solvers

We conducted a similar parameter study for the GSIP solvers as for the SIP solvers. The problem instances of the GSIP benchmark study were solved using *GSIP-RRHS* and the corresponding derived SIPs, as described in Section 3.3.2, with the SIP solvers *Oracle*, *RRHS*, and *Hybrid*. We varied the parameters $\varepsilon^{\mathrm{r},0} = \{0.1, 1, 5\}$, $\varepsilon^{\mathrm{red}} = \{1.2, 2, 5\}$, $\alpha^0 = \{0.25, 0.5, 0.95\}$, and $\alpha^{\mathrm{red}} = \{1.2, 2, 5\}$ of the algorithms *GSIP-RRHS*, *RRHS*, and *Hybrid*, respectively. Figure 3.7 clearly shows that *GSIP-RRHS* performs best for many problems for tuned parameters (for CPU time, subproblem number, and subproblem performance ratio profiles see Figures C.7 and C.8 in Appendix C.8.2). One possible reason is that the SIP solvers solve the derived (LLP) with (GSIP-REF), while *GSIP-RRHS* solves the (GSIP-LLP) and (GSIP-AUX). The derived (LLP) is more expensive than (GSIP-LLP) and (GSIP-AUX) (c.f., Figure C.9 in Appendix C.8.2). This is probably due to the non-smooth min in (GSIP-REF), which might lead to numerical disadvantages in the employed subsolver. However, it remains to be seen whether an improved opinionated subproblem formulation of (LLP) could alter these findings. We also want to point out that although *GSIP-RRHS* mostly outperforms the other solvers, it has more problems where it does not converge. An analysis analogous to the one in Section 3.6.2.4 shows that this is not changed significantly by increasing the optimality tolerance. This is connected to the issue raised by Harwood (2019), which we try to address by reducing α iteratively whenever a GSIP-LLP-Slater point is not attained by (GSIP-AUX), c.f., Sections 3.3.2 and 3.4.2.1. However, numerical issues connected to identifying GSIP-LLP-Slater points persist. For a detailed listing of the reasons why some of the problem instances were not solved, refer to Table 3.8.

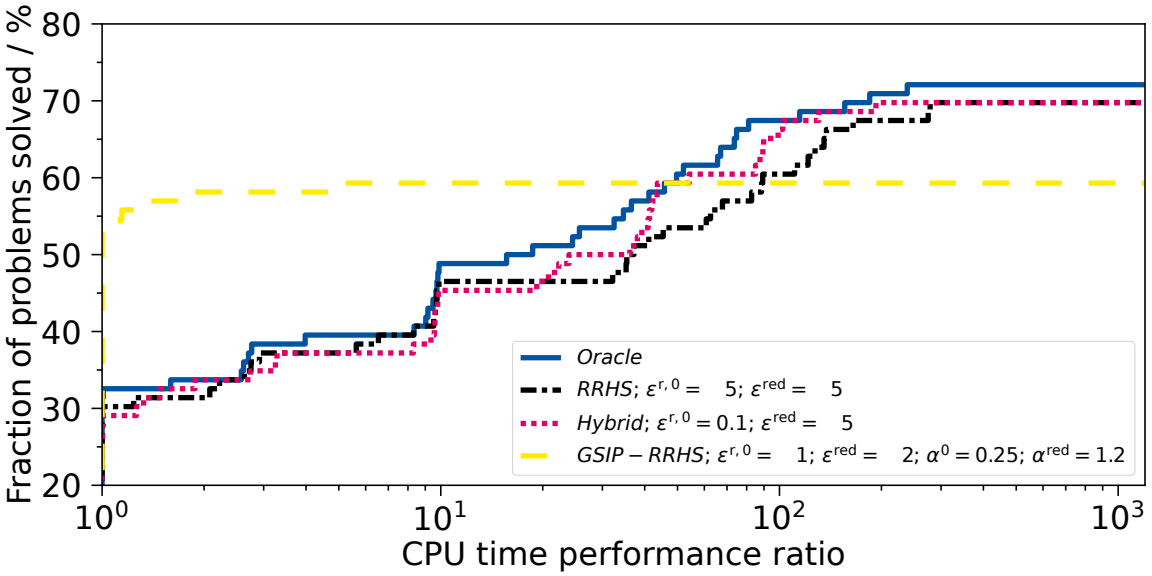


Figure 3.7: CPU time performance ratio profiles of GSIP parameter study for *GSIP-RRHS*, *Oracle*, *RRHS*, and *Hybrid*, using tuned parameters according to Table 3.7. *GSIP-RRHS* is the fastest, most often, but solves fewer problems within the time limit. Note: Adapted from Jungen et al. (2023a).

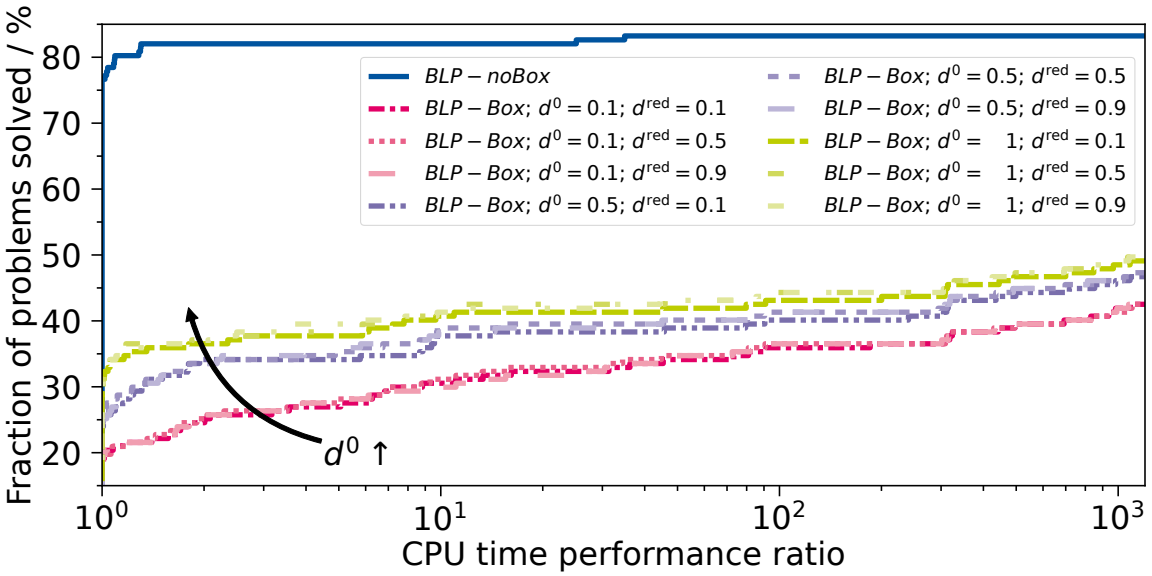


Figure 3.8: CPU time performance ratio profiles of BLP parameter study for *BLP-noBox* and *BLP-Box* with $d^0 = \{0.1, 0.5, 1\}$ and $d^{\text{red}} = \{0.1, 0.5, 0.9\}$. *BLP-noBox* outperforms *BLP-Box* for all tested parameter values. Lines with the same color correspond to the same value of the parameter d^{red}. Changes in d^{red} appear to have little effect, while increasing d^0 improves performance. Note: Adapted from Jungen et al. (2023a).

3.6.4 Performance Results of BLP Solvers

We conducted a similar parameter tuning study for the BLP solvers. For *BLP-Box*, we varied $d^0 = \{0.1, 0.5, 1\}$ and $d^{\text{red}} = \{0.1, 0.5, 0.9\}$. *BLP-noBox* performs best, c.f., Figure 3.8 (for CPU time, subproblem number, and subproblem performance ratio profiles see Figures C.10 and C.11 in Appendix C.8.3). Note that a large part of *BLP-Box*'s performance disadvantage is attributed to subsolver errors, c.f., Table 3.8. The *best* parameters for *BLP-Box* are $d^0 = 1$ and $d^{\text{red}} = 0.9$, while the influence of d^{red} is less significant. We conclude that the approach of choosing $\tilde{\mathcal{X}}$ of Djelassi et al. (2019) is superior to Mitsos et al. (2008a).

For problem instances solved by none of the solvers with their respective parameters, the solvers did not provide meaningful UBDs for a vast majority ($> 90\,\%$) of problem instances. In contrast to the analysis in Section 3.6.2.4, the fraction of unsolved problems (termination tolerance not met within the time limit) increased significantly by around $10\,\%$ for larger gaps (around 0.5) for *BLP-Box*.

3.6.5 Testing of Two Exemplary Scientific Hypotheses

The extensive library of test problems, in combination with libDIPS, offers the advantage of quickly testing scientific hypotheses for the algorithms. The following sections briefly describe two hypotheses designed to accelerate convergence. Both were implemented in libDIPS and tested on the previously introduced library of test problems.

3.6.5.1 Introduction of a Guard for the Upper-bounding Procedure in *RRHS*

RRHS, with tuned parameters ($\varepsilon^{\text{r},0} = 0.1$, $\varepsilon^{\text{red}} = 10$), spends about $24\,\%$ of the total CPU time for the upper-bounding procedure over all solved problem instances (c.f, Figure C.3 in Appendix C.8.1.1). Recall that (UBP) of *RRHS* is generally neither a relaxation nor a restriction. If ε^{r} is a small value, the discretization must be "very" fine for the upper-bounding procedure to yield an SIP-feasible point solution and thus a "good" UBD. Hence, if $\varepsilon^{\text{r},0}$ is a small value and we start with an empty discretization $\mathcal{Y}^{\text{UBP}} = \emptyset$, the first few upper-bounding iterations will most likely be unsuccessful. In an attempt to further reduce computation time, we added a guard such that the upper-bounding procedure of *RRHS* is skipped until the lower-bounding procedure produces ε^{a}-SIP-feasible points with $\varepsilon^{\text{a}} > 0$. Here, we choose $\varepsilon^{\text{a}} = 0.01$ and refer to this introduced guard parameter as $g^{\text{UBP}} = 0.01$. Whenever the upper-bounding procedure is skipped, the discretization of the $\mathcal{Y}^{\text{UBP}}$ is populated with the discretization point used in the lower-bounding procedure. Hence, the upper-bounding procedure is skipped until the discretization of $\mathcal{Y}^{\text{UBP}}$ has reached a sufficient density (in combination with ε^{r}). For a pseudocode of the described adapted algorithm, please refer to Algorithm C.2 in Appendix C.3. In our experiments, this heuristic benefits problems where the solver *RRHS* was already competitive with a small CPU time performance ratio and does not negatively affect performance on the other problems. Indeed, in Figure 3.9, *RRHS* with the heuristic is the fastest solver for most problems, but the number of problems solved within the time limit does not change significantly. For subproblem number and subproblem ratio performance profiles, see Figure C.12 in Appendix C.8.4. Note that this heuristic does not impede the convergence guarantees.

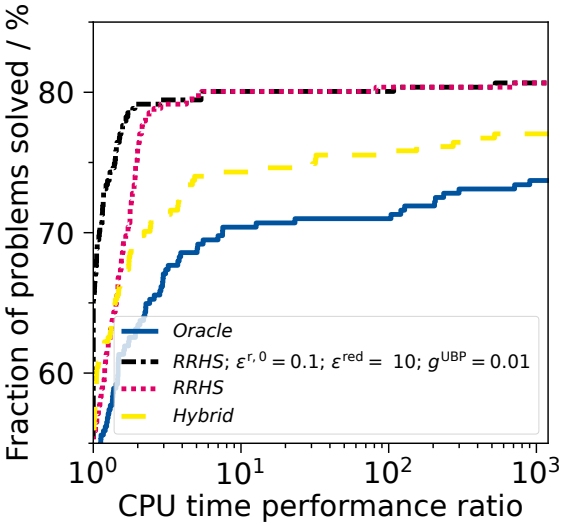


Figure 3.9: CPU time performance ratio profiles for the introduced guard parameter as $g^{\mathrm{UBP}} = 0.01$ in *RRHS*. Comparison with tuned parameters for the respective solvers. For small CPU time performance ratios, the guard significantly improves performance. Note: Adapted from Jungen et al. (2023a).

3.6.5.2 Bracketing of the Objective Function by Current Upper and Lower Bound in *RRHS*

We test whether additional bracketing of the objective function speeds up convergence, i.e., we added to (LBP) and (UBP) the constraint

$$LBD \leq f(x) \leq UBD, \tag{3.6}$$

where LBD and UBD are the current LBD and UBD. The idea is that bracketing of the objective function makes it easier for the used subsolvers to bound the solution. Additionally, if ε^{r} is "too" big but (UBP) is not directly infeasible, it is more likely that through the added bracketing constraints, (UBP) will become infeasible. Whenever (UBP) is infeasible, ε^{r} is deemed "too" large; it will be reduced, and the subsequent (unnecessary) solution of (LLP) is skipped.

Exemplary, we test our hypothesis on *RRHS* with tuned parameters, c.f, Table 3.7. We excluded the problems from Cerulli et al. (2022) that were already shown in previous experiments to not converge within the time limit. In these problem instances, the ULP is transformed from a QP to a QCQP by imposing constraints on the quadratic objective, potentially making the problem instances even more challenging.

The results shown in Figure 3.10 indicate that the additional bracketing of the objective function can have a considerable positive impact on a given problem instance. However, we do not see an overall improvement across the library of test problems. The bracketing of the objective function does not reduce the number of subproblems solved (for subproblem number and performance ratio profiles, see Figure C.13 in Appendix C.8.4). In summary, bracketing the objective function is not a promising approach for reducing computation time and the number of subproblems to be solved.

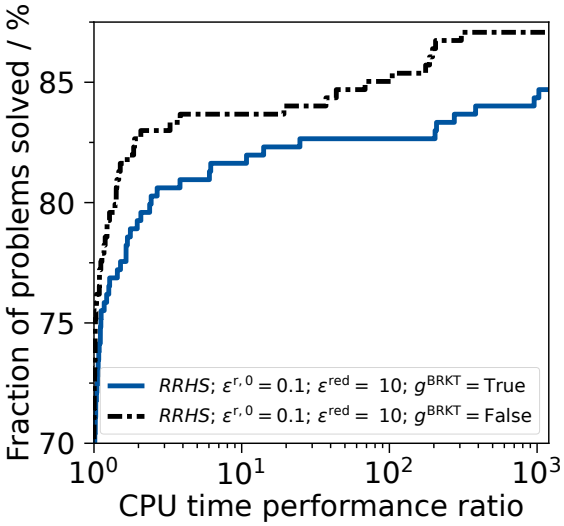


Figure 3.10: CPU time performance ratio profiles comparing the approach using bracketing (g^{BRKT}=True) against the default of not using it (g^{BRKT}=False). Comparison with tuned parameters of *RRHS*. Bracketing does not improve performance. Problems from Cerulli et al. (2022) are excluded. Note: Adapted from Jungen et al. (2023a).

3.7 Conclusion and Outlook

We presented libDIPS, an open-source software for adaptive discretization-based algorithms for (G)SIPs, BLPs, and MINMAX, which provides solvers based on the algorithms proposed and further developed by Blankenship and Falk (1976), Djelassi and Mitsos (2017, 2021), Djelassi et al. (2019), Mitsos and Tsoukalas (2015), Mitsos et al. (2008a), Tsoukalas and Rustem (2011). We optimized the algorithmic parameters of the solvers and compared these "tuned" solvers on an extensive library of test problems comprising over 600 problem instances, which unified eight existing test sets using MAiNGO as a subsolver. We found that for SIPs and BLPs, the simpler algorithms outperform the more advanced ones, while for GSIPs, the more advanced tailored algorithm outperforms the simpler ones.

The implemented solvers, which all belong to the class of adaptive discretization-based algorithms, highly rely on the used subsolver. If the used subsolver performs poorly, the algorithm's performance is directly negatively affected. Therefore, it is recommended to try different subsolvers, as some subsolvers have strengths and weaknesses depending on the problem class (i.e., LP, NLP,...) and the problem formulation. Implicitly, subproblems with linear constraints and quadratic objectives are already handled by a specialized solver in our tests, as MAiNGO utilizes CPLEX in these cases. However, other problem classes or characteristics, such as convexity, could be exploited for faster and more robust optimization. Indeed, in our tests, we encountered a significant number of numerical difficulties in solving the subproblems. This is partially caused by the fact that with each level, stronger tolerances must be enforced.

As a result, the user should continually and critically evaluate their decision on the chosen feasibility tolerances, as they may be overly restrictive compared to the model accuracy. Restrictive feasibility tolerances can significantly influence the solution time. Additionally, they should explore whether a solution approach, which guarantees a feasible point upon

termination, is necessary or if an ε^{a}-feasible point (with $\varepsilon^{\mathrm{a}} > 0$) is sufficient, as generating a feasible point is more expensive.

The idea of trying different solvers also applies to the investigated adaptive discretization-based solvers. *Oracle* has excellent performance for some problems, while it struggles with other problems. Hence, it is likely beneficial to start with *Oracle* and try a different solver if *Oracle* fails to solve the problem instance in a reasonable time.

The presented library of test problems consists of a large portion of problems that are solved quickly, several problems are unsolved, and only a small number of problems are solved within the time limit but take more than a minute to solve for the tested solvers. Following the presented investigation, it appears that many of the problems constructed as a challenge in previous publications are fast to solve after subsequent hardware and algorithmic improvements. Meanwhile, others are hardly tractable with the investigated approaches (in combination with the used subsolver MAiNGO). Therefore, it is likely necessary to use expert knowledge of the problem to facilitate its solution (if the problem is hard). Note that libDIPS offers the possibility of formulating the subproblems independently in an opinionated way. It remains to be seen whether the unsolved problems are inherently difficult to solve or whether other (non-)adaptive discretization-based solution approaches would excel at solving them. Classifying the problems into difficulty classes would also be a necessary step in refining our test collection to a proper benchmark. Ideally, a standard file format would greatly benefit such a community benchmark.

Similar to the potential gaps identified in the difficulty level, the categorization of the instances in the library of test problems revealed that problems of several problem categories, e.g., MINMAX, with an upper level being QCQP, are underrepresented. This also holds for problems with discrete variables and application-based test problem instances. Hence, future work should consider more test problem instances, especially application-based ones. To facilitate easy extension and reuse of the library of test problems, interested readers may suggest additional problem instances via the Git issue system in the libDIPS repository.

Apart from further expanding and improving the library of test problems, we believe there are three main promising directions for improvement: initialization strategies for the discretization, subproblem formulations and adaptations, and algorithm adaptations, including heuristics and extensions. One could apply an (advanced) initialization of the discretized sets instead of starting with empty ones, e.g., by initializing the sets with Karush-Kuhn-Tucker (KKT) points of the LLP or with edge points of semi-infinite sets. To reduce the number of iterations, the subproblems could be adapted by using better-suited formulations for the used subsolver, adding KKT conditions to the ULPs, or considering higher-order cuts, as in Djelassi (2020). Note that the first proposed adaptation is already possible in libDIPS. However, this will likely make the subproblems more expensive to solve. Last but not least, algorithms can be adapted and additional ones can be implemented, e.g., the interval-based method of Bhattacharjee et al. (2005a,b). For SIPs, promising ideas for algorithm adaptations include employing the solver *B&F*, and then, after a given feasibility tolerance is met, using (RES) to search a feasible point that meets the given optimality tolerance; and testing the impact of adapting *Hybrid* to a local solution of the (RES) subproblem. For GSIPs, hybridization in the generation of the discretization points of *GSIP-RRHS* is promising. *GSIP-RRHS* outperforms the other solvers. However, if *GSIP-RRHS* fails to converge, i.e., when no GSIP-LLP-Slater point can be produced through (GSIP-AUX), using the (LLP) with (GSIP-REF) might increase the number of

solved problems. Alternatively, one could solve (LLP) with (GSIP-REF) every n iterations and populate the discretization with its solution.

The impact of all these adaptations can be easily evaluated using libDIPS and the presented library of test problems. We will show in the following chapter that this is indeed the case by implementing an existing discretization-based approach for the solution of BLPs in libDIPS, presenting an adaptation thereof, and by evaluating whether our proposed adaptation is beneficial.

4 An Improved Oracle Adaptation for Bilevel Programs

As outlined in Section 3.1, BLPs with a nonconvex LLP occur in many applications, and a reliable, global, and fast solution to such optimization programs is paramount. For their solution, we can already rely on the solvers *BLP-Box* and *BLP-noBox*, which we reviewed in Section 3.3.3 and implemented in libDIPS. However, these solvers still have potential for improvement, as the best performing solver solves less than 85 % of the problem instances in our library of test problems within the time limit, c.f., Figure 3.8 in Section 3.6.4. Therefore, we explore in this chapter the potential of the approach by Tsoukalas et al. (2009) for the solution of (BLP) and propose an adaptation of their approach to further improve performance.

4.1 Background

The approach of Tsoukalas et al. (2009) can be seen as a generalization of the *Oracle* solver for SIPs, which we reviewed in Section 3.3.1. Just like *Oracle*, the generalized approach of Tsoukalas et al. (2009), is an adaptive discretization-based algorithm that performs a bisection search in the objective space by iteratively solving an oracle problem (and subsequent an LLP) to determine whether a target objective value f^{t} of the upper-level objective is attainable or not. This oracle problem is an unconstrained minmax problem, where the objective function combines the upper- and lower-level objectives and constraints. Tsoukalas et al. (2009) solve this minmax problem through a smoothing technique based on the Laplace method. The approach of Tsoukalas et al. (2009) seems promising for two reasons. First, recall that for SIPs, *Oracle* supersedes "out of the box" the performance of *RRHS* and *Hybrid* for the largest fractions of problems, c.f., Figure 3.2a in Section 3.6.2.1. This raises the question whether the generalized approach for BLPs by Tsoukalas et al. (2009) exhibits comparable performance. Second, closely related to the first point, *Oracle* as well as its generalized approach to BLPs do not have inherent algorithmic parameters. The latter might be especially beneficial if no strenuous parameter tuning, as carried out in Section 3.6.4, can be performed.

To further improve the performance of the approach of Tsoukalas et al. (2009), we propose an adaptation thereof where we change the oracle problem to minimize directly the lower-level objective, with the target objective value of the upper-level objective inscribed into its constraints. With this formulation, we aim to enhance the chance that the computed candidate point is lower-level optimal, which may lead to a faster generation of good UBDs. Although the unconstrained minmax problem, as proposed by Tsoukalas et al. (2009), might be easier to solve, we do not expect any performance losses since we require a global solution of the subproblems.

4.2 Algorithm Description and Subproblem Formulations

For the bisection search in the objective space, we need an initial LBD and UBD on the objective. We compute an initial LBD for (BLP) by

$$\begin{aligned} LBD^{\text{init}} = \min_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x})} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\text{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0}, \\ \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\text{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y}(\boldsymbol{x}) \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{g}^{\text{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\} \end{aligned} \tag{BLP-LBP$^{\text{init}}$}$$

and an initial UBD by

$$\begin{aligned} UBD^{\text{init}} = \max_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x})} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\text{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0}, \\ \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\text{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y}(\boldsymbol{x}) \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{g}^{\text{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}. \end{aligned} \tag{BLP-UBP$^{\text{init}}$}$$

Note that the point produced by (BLP-UBP$^{\text{init}}$), is in general not ε^{a}-BLP feasible, c.f., Definition 5 in Appendix B.1. Once LBD^{init} and UBD^{init} are computed, the incumbent LBD and UBD and the target objective value are initialized, i.e., $LBD \leftarrow LBD^{\text{init}}$, $UBD \leftarrow UBD^{\text{init}}$, and $f^{\text{t}} \leftarrow \frac{UBD+LBD}{2}$.

We formulate our adapted oracle problem as

$$\begin{aligned} h^{\text{BLP-ORA}} = \min_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x})} \quad & h(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{g}^{\text{u}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \min\left\{h(\boldsymbol{x},\boldsymbol{y}) - h\big(\boldsymbol{x},\boldsymbol{y}^k\big), \min_{i\in\{1,\dots,n_{\text{gl}}\}} -g_i^{\text{l}}\big(\boldsymbol{x},\boldsymbol{y}^k\big)\right\} \leq 0, \\ & \qquad\qquad \forall \boldsymbol{y}^k \in \mathcal{Y}^{\text{ORA}} \\ & f(\boldsymbol{x},\boldsymbol{y}) \leq f^{\text{t}}, \\ \\ \text{with} \quad & \mathcal{X} \coloneqq \left\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\text{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{eu}}(\boldsymbol{x}) = \boldsymbol{0}\right\} \\ & \mathcal{Y}(\boldsymbol{x}) \coloneqq \left\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{g}^{\text{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\boldsymbol{y}) = \boldsymbol{0}\right\}. \end{aligned} \tag{BLP-ORA}$$

If (BLP-ORA) is infeasible, then f^{t} is not attainable. Then, the current LBD is set to f^{t} and (BLP-ORA) is solved again. Whenever (BLP-ORA) is feasible, first, feasibility of the computed candidate point $\bar{\boldsymbol{x}}$ is checked by solving (BLP-LLP), see Section 3.3.3. If the candidate point $\bar{\boldsymbol{x}}$ is ε^{a}-BLP-feasible or the corresponding upper level objective value is better than UBD, f^{t} is updated, and (BLP-ORA) is solved again. If the candidate point is infeasible, the discretization $\mathcal{Y}^{\text{ORA}}$ is populated with a suitable discretization point, and (BLP-ORA) is solved again.

An auxiliary problem needs to be solved to compute a valid discretization point[1], i.e., (BLP-AUX), see Appendix C.2.3.

We call the solver following the above-described procedure *BLP-AdaptOracle*. A simplified pseudocode of *BLP-AdaptOracle* is given in Algorithm 4.1.

4.3 Outline of Proof of Convergence

The following assumptions are primarily standard in global optimization and are very similar to the assumptions used in Mitsos et al. (2008a), which we implemented in libDIPS as *BLP-Box*, c.f., Section 3.3.3.

Assumption 1 (*Compactness of Sets*) The sets $\bar{\mathcal{X}}$ and $\bar{\mathcal{Y}}$ are compact, c.f., (3.1) in Section 3.2.

Assumption 2 (*Continuous Functions*) All occurring functions are continuous on their respective host sets.

Assumption 3 (*Global Solution of Subproblems*) All subproblems are solved to global optimality.

Assumption 4 (*Inner Problem*) For each point $\bar{\boldsymbol{x}} \in \mathcal{X}^{\text{outer}} \bigcap \mathcal{X}^{\text{inner}}$ with $\mathcal{X}^{\text{outer}} = \{\boldsymbol{x} \in \mathcal{X} : \exists \bar{\boldsymbol{y}} \in \bar{\mathcal{Y}} : \boldsymbol{g}^{\text{u}}(\boldsymbol{x}, \bar{\boldsymbol{y}})\}$ and $\mathcal{X}^{\text{inner}} = \{\boldsymbol{x} \in \mathcal{X} : \exists \bar{\boldsymbol{y}} \in \mathcal{Y} : \boldsymbol{g}^{\text{l}}(\boldsymbol{x}, \bar{\boldsymbol{y}}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\bar{\boldsymbol{y}}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\bar{\boldsymbol{y}}) = \boldsymbol{0}\}$ it holds: For any $\varepsilon^{\text{a}} > 0$ there exists a point $\bar{\boldsymbol{z}} \in \bar{\mathcal{Y}}$ such that

$$\begin{aligned} \boldsymbol{g}^{\text{u}}(\boldsymbol{x}, \bar{\boldsymbol{z}}) &\leq \boldsymbol{0}, \\ \boldsymbol{g}^{\text{l}}(\boldsymbol{x}, \bar{\boldsymbol{z}}) &< \boldsymbol{0}, \\ \boldsymbol{v}^{\text{il}}(\bar{\boldsymbol{z}}) &\leq \boldsymbol{0}, \\ \boldsymbol{v}^{\text{el}}(\bar{\boldsymbol{z}}) &= \boldsymbol{0}, \\ h(\boldsymbol{x}, \bar{\boldsymbol{z}}) &\leq h^*(\bar{\boldsymbol{x}}) + \varepsilon^{\text{a}} \end{aligned}$$

Note that Assumption 4 is stronger than Assumption 4 in Mitsos et al. (2008a).

First, we argue that if $f^{\text{t}} < f^*$, i.e., the target objective value is not attainable, (BLP-ORA) will become infeasible after finitely many iterations. The only interesting iterates are the pairs $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ satisfying all lower- and upper-level constraints but $\hat{\boldsymbol{y}}$ not satisfying optimality in (BLP-LLP). Suppose a pair $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ is found, which exhibits this. The discretization point $\boldsymbol{y}^k$ computed by (BLP-AUX), which is added, forbids in all following iterations to revisit $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$, and a neighborhood around it. Therefore, after finitely many iterations, (BLP-ORA) becomes infeasible, and it is proven that f^{t} is not attainable. Second, we arge that if $f^{\text{t}} \geq f^*$, i.e., the target objective value is attainable, (BLP-ORA)

[1] In the original publication (Jungen and Mitsos, 2024a), an auxiliary problem similar to Mitsos and Tsoukalas (2015) for GSIPs was solved. This subproblem was flawed in the sense that (i) additional assumptions had to be made about the optima of h, (ii) some of the generated discretization points were suboptimal, and (iii) additional algorithmic parameters were necessary. Algorithm 4.1 is adapted accordingly for the usage of (BLP-AUX). Note that these changes do not alter the main findings and algorithmic idea.

Algorithm 4.1: Simplified pseudocode of *BLP-AdaptOracle* to solve (BLP). All subproblems are solved globally. User inputs are the initial discretization set $\mathcal{Y}^0 \subsetneq \mathcal{Y}$, the termination tolerance $\varepsilon^{\mathrm{f}} > 0$, and feasibility tolerance $\varepsilon^{\mathrm{a}} > 0$.

// initialization procedure
1 solve (BLP-LBP$^{\mathrm{init}}$) to obtain LBD^{init};
2 **if** (BLP-LBP$^{\mathrm{init}}$) *is infeasible* **then**
    // (BLP) is is infeasible
3   set $LBD \leftarrow +\infty$, and $UBD \leftarrow +\infty$;
4   terminate;
5 solve (BLP-UBP$^{\mathrm{init}}$) to obtain UBD^{init};
6 set $LBD \leftarrow LBD^{\mathrm{init}}$, $UBD \leftarrow UBD^{\mathrm{init}}$, $f^{\mathrm{t}} \leftarrow \frac{UBD+LBD}{2}$, and $\mathcal{Y}^{\mathrm{ORA}} \leftarrow \mathcal{Y}^0$;
// check termination
7 **if** $UBD - LBD \leq \varepsilon^{\mathrm{f}}$ **then**
8   terminate;
// oracle procedure
9 solve (BLP-ORA) to obtain $\boldsymbol{x}^{\text{BLP-ORA}}$, $\boldsymbol{y}^{\text{BLP-ORA}}$ and $h^{\text{BLP-ORA}}$;
10 **if** (BLP-ORA) *is infeasible* **then**
    // f^{t} is not attainable
11   set $LBD \leftarrow f^{\mathrm{t}}$, and $f^{\mathrm{t}} \leftarrow \frac{UBD+LBD}{2}$;
12   go to Line 9;
// check ε^{a}-BLP-feasibility of $\left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-ORA}}\right)$ in (BLP)
13 solve (BLP-LLP) for fixed $\boldsymbol{x}^{\text{BLP-ORA}}$ to obtain $h^*\left(\boldsymbol{x}^{\text{BLP-ORA}}\right)$ and $\boldsymbol{y}^{\text{BLP-LLP}}$;
14 **if** $h^{\text{BLP-ORA}} \leq h^*\left(\boldsymbol{x}^{\text{BLP-ORA}}\right) + \varepsilon^{\mathrm{a}}$ *and* $\boldsymbol{g}^{\mathrm{u}}\left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-LLP}}\right) \leq \mathbf{0}$ **then**
15   set $UBD \leftarrow f^{\mathrm{t}}$, $(\boldsymbol{x}^*, \boldsymbol{y}^*) \leftarrow \left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-ORA}}\right)$, and $f^{\mathrm{t}} \leftarrow \frac{UBD+LBD}{2}$;
16   go to Line 7;
// check if solution of (BLP-LLP) produced better UBD
17 **if** $f\left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-LLP}}\right) < UBD$ *and* $\boldsymbol{g}^{\mathrm{u}}\left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-LLP}}\right) \leq 0$ **then**
18   set $UBD \leftarrow f\left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-LLP}}\right)$, $(\boldsymbol{x}^*, \boldsymbol{y}^*) \leftarrow \left(\boldsymbol{x}^{\text{BLP-ORA}}, \boldsymbol{y}^{\text{BLP-LLP}}\right)$, and $f^{\mathrm{t}} \leftarrow \frac{UBD+LBD}{2}$;
19   go to Line 7;
// generate discretization
20 solve (BLP-AUX) for fixed $\boldsymbol{x}^{\text{BLP-ORA}}$ to obtain $\boldsymbol{y}^{\text{BLP-AUX}}$;
21 set $\mathcal{Y}^{\text{ORA}} \leftarrow \mathcal{Y}^{\text{ORA}} \cup \{\boldsymbol{y}^{\text{BLP-AUX}}\}$;
22 got to Line 7;

will furnish a ε^{a}-BLP-feasible point after finitely many iterations. Like in the first case above, we exclude all pairs $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ that do not satisfy lower-level optimality after a finite number of iterations. Because at least one pair $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ fulfills all lower- and upper-level constraints and is optimal in the lower level, we must obtain such a point after finitely many iterations. Whenever the target value f^{t} is proven to be not attainable or a feasible pair $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ is found satisfying $f(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}) \leq f^{\mathrm{t}}$, the target objective value f^{t} is updated via bisection. Through bisection, we terminate finitely with proof of ε^{f}-optimality.

4.4 Numerical Experiments

We use libDIPS to compare *BLP-AdaptOracle* to our implementation of the approach of Tsoukalas et al. (2009), i.e., *BLP-Oracle*, and the solvers *BLP-Box* and *BLP-noBox*, c.f., Section 3.3.3. To enable a fair comparison to the underlying algorithmic idea of Tsoukalas et al. (2009), we do not apply the smoothing method proposed in the original application but rather directly solve the discrete minmax problem, and all other occurring subproblems, using the subsolver MAiNGO version 0.8.2 (Bongartz et al., 2018) with its default settings, configured to utilize CPLEX version 22.1.1 (International Business Machines Corporation, 2022) to solve LPs and QPs. We use the BLP problem instances in our library of test problems for the numerical experiments, c.f., Section 3.5. All calculations were conducted on the RWTH High-Performance Computing cluster CLAIX running Rocky Linux 9 using a single core with an Xeon 8468 Sapphire processors running at 2.1 GHz with up to 2540 Mbit RAM. We use the "best-working" hyperparameters of *BLP-Box*, as reported in Section 3.6.4.

Figure 4.1 shows the CPU time performance ratio for the different solvers. It can be seen that *BLP-AdaptOracle* exceeds *BLP-Oracle* as well as slightly outperforms *BLP-Box*. However, *BLP-noBox* is still by far the best solver.

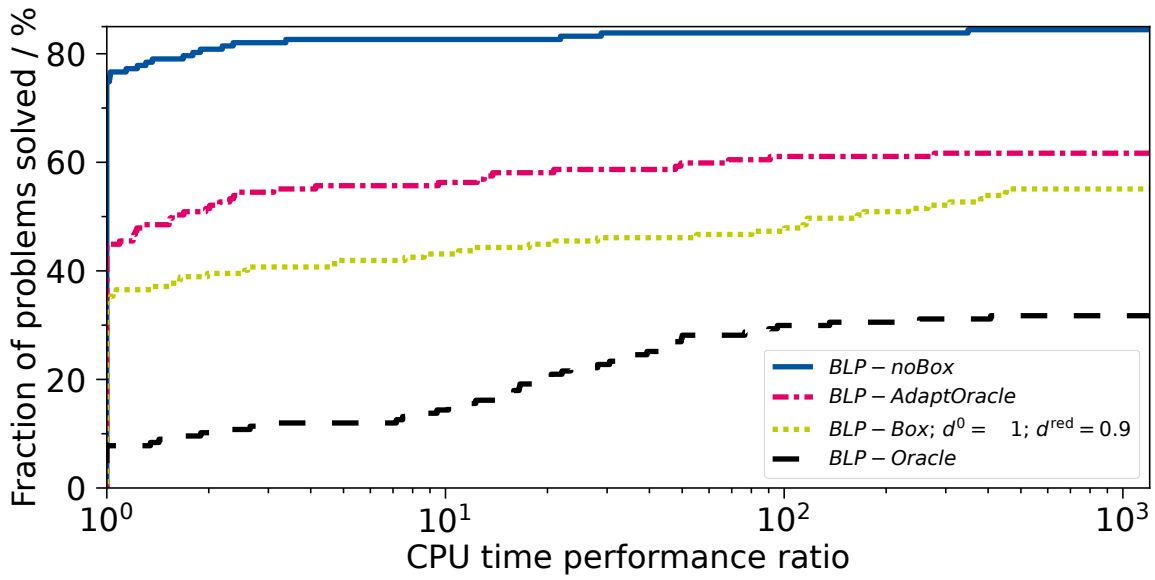


Figure 4.1: CPU time performance ratio showing the numerical performance of the solvers *BLP-noBox* and *BLP-Box*, c.f., Section 3.3.3; our algorithm adaptation *BLP-AdaptOracle*; and *BLP-Oracle*, our implementation of Tsoukalas et al. (2009). Note: Adapted from Jungen and Mitsos (2024a), © Elsevier B.V. 2024.

4.5 Conclusion

In this chapter, we presented an adaptation of the BLP algorithm proposed by Tsoukalas et al. (2009). While our adaptation outperforms our implementation of the original approach and the solver *BLP-Box*, it is superseded by *BLP-noBox*. However, as outlined in Section 3.6, our results (likely) depend on the composition of the benchmark test set, the subproblem formulation, and the used subsolver. The BLP community might benefit from our algorithm adaptation and its integration into libDIPS because the seamless transition between the BLP solvers within libDIPS enhances accessibility, and our algorithm adaptation might be advantageous in specific applications. Furthermore, our contribution shows that there is likely some undetected potential in the existing approaches, as a simple adaptation resulted in significant performance improvements.

By implementing the approach of Tsoukalas et al. (2009) and our adaptation thereof, we have shown that libDIPS serves as an ideal framework for implementing and comparing both existing and new/adapted discretization-based algorithms. This will also be further illustrated in the next chapter, where we implement and apply a specialized adaptive discretization-based algorithm to solve an OED for GPE problem.

5 Optimal Experimental Design for Guaranteed Parameter Estimation with Rigorous Calculation of Liquid-Liquid Equilibrium Embedded

Having discussed the notions and concepts of hierarchical programs in Chapters 3 and 4, we can now focus on our objective of increasing the parameters reliability in mechanistic models. In this chapter, we propose an OED for GPE formulation for systems whose input-output relation is implicitly given by an equation system with additional semi-infinite constraints, which corresponds in our application to the rigorous computation of an LLE. The resulting problem is a hierarchical program with multiple levels and additional (generalized) semi-infinite constraints. We propose an adaptive discretization-based solution procedure to our OED problem. Specifically, our procedure builds on the approaches of Blankenship and Falk (1976) (*B&F*) and its successors Djelassi et al. (2019), Mitsos and Tsoukalas (2015) (*GSIP-RRHS*), and Falk and Hoffman (1977) (*Minmax*). We implement our approach in libDIPS, which serves as a suitable development environment, because (i) our approach calls *B&F* as a subprocedure and (ii) we can utilize the implementation infrastructure of libDIPS, see Section 3.4.2, thereby obviating the need to develop every component from the ground up. As outlined in Section 3.4.2, we can take advantage of, e.g., the internal functions for populating discretized sets and the pre-existing interfaces to the subsolver.

This chapter is organized as follows. In Section 5.1, we give an introduction to the concepts of OED for GPE, provide a literature review of OED for GPE, and an introduction to the main approaches for modeling LLEs. We recapitulate in Section 5.2 the necessary thermodynamic background for our application. Section 5.3 covers OED for GPE and its application to the intended LLE application. In this section, we present our specialized adaptive discretization-based procedure for solving the introduced OED for GPE problem. In Section 5.5, we provide a proof-of-concept using the Redlich-Kister model (Redlich and Kister, 1948) as the underlying excess model to plan the OED to reduce parameter uncertainty. We conclude this chapter in Section 5.6 and provide an outlook and potential directions for future research.

5.1 Introduction and Literature Review

Process simulations are integral to chemical engineering at nearly every stage, from plant design to operation. In chemical engineering, the models used in these simulations, usually rely on semi-empirical thermodynamic models, e.g., NRTL (Renon and Prausnitz, 1968), Redlich-Kister (Redlich and Kister, 1948), UNIQUAC (Abrams and Prausnitz, 1975), modified Wilson (Tsuboka and Katayama, 1975), or other excess models (Pfennig, 2004).

Exceptions are predictive models; e.g.; (PC(P))-SAFT (Chapman et al., 1990, Gross and Sadowski, 2001, Gross and Vrabec, 2006), predictive PCP-SAFT (van Nhu et al., 2008), COSMO-RS (Klamt, 1995) and data-driven predictive models using; e.g.; transformers (Winter et al., 2022), graph neural networks (Qin et al., 2023, Rittig et al., 2023, Sanchez Medina et al., 2022, 2023), or matrix completion methods (Chen et al., 2021, Jirasek et al., 2022); which we do not consider in this chapter. The semi-empirical models inherently include mixture-dependent parameters that need to be estimated based on time-consuming and costly experiments (Kontogeorgis et al., 2021). Therefore, it is essential to design experiments with great care, which are then conducted to collect measurement data. In this chapter, we advance OED for GPE to increase parameter precision. For OED applications for model discrimination refer to, e.g., (Box and Hill, 1967).

Bounded-error OED methods perform a worst-case experimental design, which safeguards against experiments that perform well on average but poorly for certain parameter values. Bounded-error OED methods assume that the measurement error is bounded and additive, with no other hypothesis concerning their statistical distribution (Pronzato and Walter, 1990, Walz et al., 2018). Measuring equipment can often supply bounds on the measurement error, provided the equipment certifies a tolerance class (Belforte et al., 1988).

More specifically, (bounded-error) OED for GPE computes an experimental design (to be carried out) that minimizes the estimated size of the feasible parameter set or an over-approximation thereof (Pronzato and Walter, 1990). The feasible parameter set[1] consists of all parameter values consistent with the model, the measurements, and their unknown but bounded measurement errors.

OED for GPE methods builds on the concepts of GPE, also known as set-membership or bounded-noise parameter estimation. The interested reader may refer to Milanese and Vicino (1991), Walter and Piet-Lahanier (1990) and the references therein for an introduction to GPE. A small size of the feasible parameter set corresponds to high parameter reliability. As computing the exact feasible parameter set, and therefore also the OED for GPE, is often time-consuming (Milanese and Belforte, 1982, Norton, 1987, Walz et al., 2018), ellipsoid or orthotropic approximations are frequently used (Belforte et al., 1990, Milanese and Vicino, 1991). Minimizing the volume of the feasible parameter set (which corresponds to the 100 % non-asymptotic confidence region (Pronzato and Walter, 1989)), also called V-optimal design, can be seen as the natural counterpart to classical/ statistical D-optimal design, where the joint confidence ellipsoids are minimized (Theorem 1 in (Pronzato and Walter, 1990)). Other metrics exist for assessing the (approximated) size of the feasible parameter set. For example, Belforte and coworkers consider in a series of papers (Belforte and Gay, 2000, 2004, Belforte and Tay, 1993) the radius of the feasible set (which corresponds to the maximum estimation error in a suitable norm) and the largest width of the parameter uncertainty intervals for finite impulse response (FIR) systems. Schweppe (1968) considers ellipsoids to over-approximate the feasible parameter set.

Pronzato and Walter (1990) point out that, in the past, OED for GPE has received limited attention, at least compared to classical/statistical OED methods. For earlier works, refer to Belforte et al. (1984, 1987), Norton (1987), Pronzato and Walter (1989), who give an overview of linear models, and (Pronzato and Walter, 1990) for the nonlinear

[1]The feasible parameter set is also called feasible parameter region (Norton, 1987), parameter uncertainty set (Belforte et al., 1988), set of consistent parameters (SCP) (Hasenauer et al., 2010), consistent parameter set (Borchers et al., 2011), or (model) membership set (Pronzato and Walter, 1989).

case, where linearity with respect to the parameters is of relevance. In the linear case, the OED for GPE can be analytically computed, and the feasible parameter set takes the form of a convex polyhedron (Pronzato and Walter, 1989). In the nonlinear case, the feasible parameter set may be disconnected and nonconvex (Belforte et al., 1988). Pronzato and Walter (1990) suggest strategies to prevent disconnected feasible parameter sets. For further differences and commonalities between the linear and nonlinear case, the interested reader may refer to Belforte et al. (1988), who give a tabular overview.

In more recent years, Borchers et al. (2011) employ OED for GPE to discrete-time systems that are linear in the parameters and compute a one-step input design to minimize the volume of the feasible parameter set. Based on their previous work, Borchers and Findeisen (2011) extend their approach to the multi-step case. Marvel and Williams (2012) consider nonlinear continuous-time systems and leverage interval analysis to compute the next experiment design. Hasenauer et al. (2010) consider a class of implicit nonlinear discrete-time systems and apply their OED for GPE approach to a biochemical reaction network. Both, Hasenauer et al. (2010), Marvel and Williams (2012), compute discrete optimal inputs.

In the following, we provide a more detailed review of some of the contributions of Gottu Mukkula and Paulen, as these are closely related to our contribution. Gottu Mukkula and Paulen (2016a) consider dynamic nonlinear systems and compute the OED for GPE by first overapproximating the feasible parameter set and then determining the sequence of control inputs to minimize the estimated overapproximation of the feasible parameter set. Through over-approximation of the feasible parameter set by a box, the OED for GPE problem becomes a minmax BLP. This BLP is, up to this point, a rigorous and general method of OED for GPE for general dynamic nonlinear systems. Gottu Mukkula and Paulen (2016a) solve the BLP by replacing the LLP with its KKT conditions, which leads to a mathematical program with equilibrium constraints (MPEC). They note that one must ensure that the solution of the MPEC is globally optimal in the LLP. However, the employed optimization method by Gottu Mukkula and Paulen (2016a) does not guarantee this property. In related work, Gottu Mukkula and Paulen (2016b) apply the same OED for GPE method to the Lotka-Volterra (predator-prey) model, modified to control the prey population. They apply the same KKT-based relaxation, solve the resulting MPEC problem using the same method to achieve local optimality, and then perform an *a posteriori* check to verify that the LLP is globally optimal. However, Gottu Mukkula and Paulen (2016b) do not propose a follow-up procedure if the obtained solution is not globally optimal in the lower level. Gottu Mukkula and Paulen (2017a) discuss their preliminary results (Gottu Mukkula and Paulen, 2016a,b) in more detail and further expand their work. Regarding our contribution, it is particularly noteworthy that in their dynamic case study, Gottu Mukkula and Paulen (2017a) identify multiple local optima of the LLP during their solution process. In contrast to their previous work (Gottu Mukkula and Paulen, 2016a,b), Gottu Mukkula and Paulen (2017a) present and apply a heuristic on how to proceed if the *a posteriori* check fails, i.e., the solution of the LLP is not globally optimal. The heuristic entails the (re-)initialization of the MPEC problem using the results of a set-inversion algorithm used to compute the feasible parameter set. Gottu Mukkula and Paulen (2017a) note that their heuristic leads to an increased probability of identifying local optima of the OED problem.

Walz et al. (2018) build on the work of Gottu Mukkula and Paulen (2017a) by applying a solution method that guarantees a global solution to the OED problem. Walz et al. (2018) employ an algorithm based on the work of Mitsos and Tsoukalas (2015) for GSIP

(*GSIP-RRHS*, see Section 3.3.2), which is simplified to the minmax bilevel structure of the OED problem. Recall that first-order necessary conditions of the LLP are not used therein. Walz et al. (2018) were able to obtain a global solution for simple case studies, but not for the more complex case studies. The bottleneck is whether the used subsolver converges in a reasonable time.

Gottu Mukkula and Paulen (2017b) build on their previous contribution (Gottu Mukkula and Paulen, 2017a) and investigate the case where the OED for GPE can be significantly suboptimal due to the nominal parameter values being far from the true value. Based on existing multi-stage robust optimization techniques, they propose robustification techniques. In related work, Gottu Mukkula and Paulen (2022), Gottu Mukkula et al. (2021) further investigate robustification of OED in the context of nonlinear maximum-likelihood parameter and nonlinear least-squares parameter estimation using linearized confidence regions. Denis-Vidal et al. (2019) propose an approach in the OED for GPE context that does not require specifying any nominal parameter values. For other closely connected approaches, the interested reader may refer to Asprey and Macchietto (2002), Welsh and Rojas (2009). In adjacent work, Walz et al. (2020) further advance the idea of OED for GPE by tailoring the OED towards the intended model application, i.e., a process that inherently uses the model with its corresponding model parameters. The proposed min-max-min formulation mitigates the worst-case realizations of the process design costs, rather than minimizing parameter uncertainty. Walz et al. (2020) solve the hierarchical programs by a brute-force discretization of the first level and an adaptation of the GSIP algorithm by Djelassi et al. (2019) for the remaining max-min problem.

5.1.1 Approaches for Modeling and Fitting Liquid-Liquid Equilibria

Two main approaches exist for modeling LLEs. The first approach relies on solving nonlinear equations, which correspond to the first derivatives of the Gibbs free energy, i.e., equal activities, and the mass balances. The second approach directly minimizes the Gibbs free energy or solves a related optimization problem.

Methods of the first approach rely on the isopotential condition, which uses the property that a system is in equilibrium if the thermodynamic potential is equal across all phases. Many numerical methods have been developed to solve these challenging nonlinear equations; however, a fundamental problem is that thermodynamically incorrect solutions are often predicted, such as erroneous phases or trivial solutions that mistakenly suggest a homogeneous mixture. This is because the first approach relies on the isopotential condition to compute stationary points, i.e., local minima, maxima, or saddle points of the Gibbs free energy curve. The isopotential condition is only a necessary condition for phase stability, and the predicted solutions can be unstable (Sørensen et al., 1979). However, these methods are used more frequently (often, with appropriate *good* initial guesses to avoid trivial solutions) because the second approach, while more reliable in practice, is computationally more demanding.

Methods of the second approach can be broadly categorized into three categories (Marcilla et al., 2010): (i) Michelsen (1982a,b) present a stability condition, which can be geometrically interpreted as the vertical distance between the tangent hyperplane and the Gibbs free energy surface, and use this condition in their phase split calculations. The presented stability condition of Michelsen (1982a,b) is closely connected to the tangent plane (TP) criterion of Baker et al. (1982), also called Gibbs TP. Mitsos and Barton

(2007) propose a stability condition based on the Lagrangian dual of the Gibbs TP stability criterion. The solution of the dual problem describes the stable phase split. Notably, their method permits very weak assumptions: An unknown, even infinite, number of phases is allowed; it is not required that all species are present in every phase; and the Gibbs free energy must only be continuous. (ii) Eubank et al. (1992) present the area method, where they maximize the integrated area between the TP and the Gibbs surface. (iii) Eubank and Hall (1995) present an equal area rule (EAR) which uses a similar concept to the *van der Waals* loop. The EAR and modifications thereof, e.g., Sapkowski and Hofman (2023), find the common tangent line, which corresponds to finding an equal area construction for the derivative of the total Gibbs free energy.

Partially due to the inherent flexibility of the excess model, LLEs have been successfully predicted with both approaches. While this flexibility is advantageous in some cases, it can also be problematic. Parameter values can be *problematic* in the sense that incorrect phase splits or more phases than observed are predicted (Bollas et al., 2009, Mitsos et al., 2009b). These issues can arise regardless of the approach used to compute the LLE. Marcilla et al. (2017) show that an unexpectedly high number of (randomly selected) publications include *problematic* parameter values and predicted LLE data that is inconsistent, i.e., predictions do not agree with experimental data, tie-lines do not satisfy Gibbs stability criteria, or even the isopotential condition is not satisfied. Eight years later, Marcilla et al. (2025) show that these challenges remain.

In this chapter, we use ideas from of Mitsos et al. (2009b), which resolves the limitations of standard parameter estimation methods that may produce *problematic* parameter values. Mitsos et al. (2009b) use the Gibbs TP criterion to check stability *during* parameter estimation and, additionally, introduces a series of conditions to exclude *problematic* parameter values. The conditions for excluding *problematic* parameter values essentially constrain the curvature of the Gibbs surface through local convexity conditions and confining the Gibbs surface between tangent and secant lines. The Gibbs TP criterion and the additional conditions to exclude *problematic* parameter values are written as (generalized) semi-infinite constraints and ensure that only the observed phase behavior is predicted with the obtained parameter values.

In a somewhat related fashion to Mitsos et al. (2009b), Marcilla et al. (2010) present a software that uses a robust computation algorithm to fit parameter values of excess models to equilibria data. Their method is robust in the sense that they exclude *problematic* parameter values. The method employs some topology analysis of the Gibbs energy of mixing. They enforce, as in Mitsos et al. (2009b), local convexity conditions and also some empirically derived restrictions. In contrast to Mitsos et al. (2009b), they do not write their conditions as (generalized) semi-infinite constraints, but enforce them on a grid of points. They successfully apply their method to multiple multi-component systems (e.g., Marcilla Gomis (2011), Olaya et al. (2008), Reyes-Labarta et al. (2009)).

5.1.2 Optimal Experimental Design for Liquid-Liquid Equilibria Characterization

If the semi-empirical model used in process simulations is not predictive, its parameters must be fitted to experimental data. However, the thermodynamic functions, e.g., the activity, can not be measured directly. LLE data can be used to estimate parameters included in excess models, as the chemical potentials of the components are equal at equilibrium. A standard method for characterizing an LLE involves adding the components to a closed vessel in known quantities, stirring, and tempering the vessel until equilibrium is reached. After equilibration, samples are taken from both phases, and their compositions are subsequently determined. Finally, the parameters of the excess model are fitted to the LLE data (Pfennig, 2004).

OED possesses great potential for LLE characterization because LLE measurements require a significant amount of work and can therefore be costly. Usually, Fisher information-based criteria are used for OED and the LLE is modeled using the isopotential condition (Dechambre et al., 2014, Duarte et al., 2019). However, due to the high cost of experiments, measurement data may be scarce, which raises the question of whether the necessary assumptions for the classical/statistical methods, i.e., assumptions concerning the stochastic characteristics of the measurement uncertainties, are satisfied. Additionally, the planned experimental designs may be suboptimal in the sense that the planned experiment does not provide informative measurement data, leading to additional experimental effort. For example, the experimental design could suggest compositions to the experimenter where the available data does not indicate any phase splitting. This may be caused by wrong LLE predictions resulting from, e.g., modeling the LLE solely with the isopotential condition or using *problematic* parameter values for the local composition models.

As the utmost goal should be to minimize experimental effort, we propose to use an OED for GPE approach for planning the next experiment design, (i) base our computations of the LLE predictions on the sufficient condition for phase stability, i.e., the Baker's criterion (Baker et al., 1982), and (ii) impose the additional conditions of Mitsos et al. (2009b) onto the parameter values such that erroneous predictions due to *problematic* parameter values are avoided and the feasible parameter set is more tightly approximated by excluding *problematic* parameter values. Our proposed formulation leads to a MINMAX with embedded (generalized) semi-infinite constraints. We solve this problem using a specialization of the generalized semi-infinite algorithm, based on Djelassi et al. (2019), Falk and Hoffman (1977), Mitsos and Tsoukalas (2015), c.f., Section 3.3.

5.2 Thermodynamic Background

In the following, we provide a short review of the thermodynamic background necessary for modeling LLEs and concepts presented by Mitsos et al. (2009b) to exclude *problematic* parameter values that could result in inaccurate predictions. We consider a binary two-phase system, as local composition models such as NRTL (Renon and Prausnitz, 1968) are based exclusively on binary interaction parameters. Three- and higher multi-component systems are described with permutations of the binary interaction parameters between all species. Hence, LLE data of binary systems is already sufficient, avoiding the need for very

costly multi-species experiments. Note that an extension to multicomponent systems is, in principle, straightforward, but would drastically increase the computational effort.

We assume non-vanishing phases and that all species are present in all phases. As is typically the case in LLE computations, we neglect the pressure influence. Because we do not consider reactions, the total number of moles (of each species) is constant, and we can omit the molar Gibbs free energy of formation. Therefore, we can base our computations using the mole fractions and molar quantities of the Gibbs free energy of mixing $\bar{G}^{\mathrm{M}}$. Using the closing condition and mass balances, we can eliminate all describing state variables except for the mole fraction of the first species in the first phase, x_1, and the mole fraction of the first species in the second phase, x_2. For notational convenience, we arbitrarily enforce without loss of generality that $x_1 \leq x_2$. We restrict the mole fractions to $x \in [x_{\min}, 1 - x_{\min}] \coloneqq \mathcal{X}$, with $x_{\min} > 0$ because the derivative of the molar Gibbs free energy of mixing $\frac{\partial \bar{G}^{\mathrm{M}}}{\partial x}\Big|_{\bar{T},\bar{\boldsymbol{p}},x}$ is unbounded at $x = 0$ and $x = 1$. However, for small $x_{\min}$ this is not restrictive (Mitsos et al., 2009b). For ease of notation, we define $x_{\mathrm{L}} = x_{\min}$ and $x_{\mathrm{U}} = 1 - x_{\min}$ and aggregate the mole fractions of the phases in the vector $\boldsymbol{x} = (x_1, x_2)^T$.

5.2.1 Modeling Liquid-Liquid Equilbria

In the following, we reiterate the isopotential condition and TP criterion by Baker et al. (1982), which we use to model the LLE. We will highlight how the TP criterion results in a semi-infinite constraint. The isopotential condition reads for a (binary) two-phase system described by $\bar{\boldsymbol{x}}$ at constant temperature $\bar{T}$

$$\begin{aligned} \ln\big(\gamma_{1,1}\big(\bar{T},\bar{\boldsymbol{p}},\bar{\boldsymbol{x}}\big)\big)\bar{x}_1 &= \ln\big(\gamma_{2,1}\big(\bar{T},\bar{\boldsymbol{p}},\bar{\boldsymbol{x}}\big)\big)\bar{x}_{2,1} \\ \ln\big(\gamma_{1,2}\big(\bar{T},\bar{\boldsymbol{p}},\bar{\boldsymbol{x}}\big)\big)(1-\bar{x}_1) &= \ln\big(\gamma_{2,2}\big(\bar{T},\bar{\boldsymbol{p}},\bar{\boldsymbol{x}}\big)\big)(1-\bar{x}_{2,1}), \end{aligned} \tag{C-ip}$$

with $\gamma_{j,i}$ being the activity coefficient of phase j and component i, and $\bar{\boldsymbol{p}}$ being the parameters of the underlying excess model (Pfennig, 2004). Recall that (C-ip) together with the mass balances, c.f, (MB) in Appendix D.1.1, is only a necessary condition for phase stability, which, if solely used, may lead to wrong predictions. Incorrect predictions may lead to additional experimental effort. To minimize experimental effort in the laboratory, we also enforce the TP criterion.

The TP criterion reads (Baker et al., 1982)

$$\bar{G}^{\mathrm{M}}\big(\bar{T},\bar{\boldsymbol{p}},\bar{x}_1\big) + \frac{\partial \bar{G}^{\mathrm{M}}}{\partial x}\bigg|_{\bar{T},\bar{\boldsymbol{p}},\bar{x}_1} (y - \bar{x}_1) \leq \bar{G}^{\mathrm{M}}\big(\bar{T},\bar{\boldsymbol{p}},y\big),\ \forall y \in \mathcal{X}. \tag{C-TP-SIP}$$

The dummy symbol y is introduced instead of x to emphasize the key differences between them. (C-TP-SIP) is a semi-infinite constraint and can not be implemented in or handled by standard NLP solvers. Recall that in adaptive-discretization based methods for SIPs (Section 3.3), the constraint is discretized and relaxed to

$$\bar{G}^{\mathrm{M}}\big(\bar{T},\bar{\boldsymbol{p}},\bar{x}_1\big) + \frac{\partial \bar{G}^{\mathrm{M}}}{\partial x}\bigg|_{\bar{T},\bar{\boldsymbol{p}},\bar{x}_1} \big(y^l - \bar{x}_1\big) \leq \bar{G}^{\mathrm{M}}\big(\bar{T},\bar{\boldsymbol{p}},y^l\big) \tag{C-TP}$$

with $y^l \in \mathcal{X}^{\mathrm{DISC}} \subsetneq \mathcal{X}$. In the implementation, we approximate the derivative $\frac{\partial \bar{G}^{\mathrm{M}}}{\partial x}\Big|_{\bar{T},\bar{\boldsymbol{p}},\bar{x}_1}$ with $\frac{\bar{G}^{\mathrm{M}}(\bar{T},\bar{\boldsymbol{p}},\bar{x}_2) - \bar{G}^{\mathrm{M}}(\bar{T},\bar{\boldsymbol{p}},\bar{x}_1)}{\bar{x}_2 - \bar{x}_1}$ because preliminary investigations have shown that this formulation slightly reduces the computation burden for binary systems.

The solution procedure employed in this chapter uses the relaxed TP criterion (C-TP) in conjunction with (C-ip), and (MB). The predicted phase split $\bar{\boldsymbol{x}}$ (for given $\bar{T}$ and $\bar{\boldsymbol{p}}$) is stable, if the optimal objective value of

$$f^{\text{P-TP}} = \max_{\boldsymbol{y}\in\mathcal{X}^2} \; \bar{G}^{\text{M}}\big(\bar{T},\bar{\boldsymbol{p}},\bar{x}_1\big) + \left.\frac{\partial \bar{G}^{\text{M}}}{\partial x}\right|_{\bar{T},\bar{\boldsymbol{p}},\bar{x}_1} (y-\bar{x}_1) - \bar{G}^{\text{M}}\big(\bar{T},\bar{\boldsymbol{p}},y\big) \tag{P-TP}$$

is equal to 0. Standard NLP solvers can only solve an NLP to a certain tolerance. Therefore, we consider $\bar{\boldsymbol{x}}$ feasible, i.e., it describes the LLE, if the optimal objective value of (P-TP) is less equal to a small $\varepsilon^{\text{a,P-TP}} > 0$. If $f^{\text{P-TP}} > \varepsilon^{\text{a,P-TP}}$, the predicted phase split is not stable and the solution of (P-TP) is used to refine the discretization $\mathcal{X}^{\text{DISC}}$ and the LLE computations are repeated.

In our experience, additionally adding the constraint that $\bar{G}^{\text{M}}$ is locally convex at x_1 and x_2, c.f., (C-locc) in Appendix D.1.2, accelerates convergence (Mitsos et al., 2009b). Hence, in the implementation, (C-locc) is added as a constraint.

The excess Gibbs free energy included in $\bar{G}^{\text{M}}$, c.f., Appendix D.1.2, is typically computed using NRTL (Renon and Prausnitz, 1968), Redlich-Kister (Redlich and Kister, 1948), UNIQUAC model (Abrams and Prausnitz, 1975), modified Wilson model (Tsuboka and Katayama, 1975), or other excess models (Pfennig, 2004). Pragmatically, the Redlich-Kister model, c.f., (D.3) in Appendix D.1.2, is chosen in this contribution as a proof-of-concept. This relatively simple excess model is chosen to reduce the computational burden. Note that the proposed OED for GPE method is directly applicable to other excess models; however, it would result in higher computational demand.

5.2.2 Conditions for the Exclusion of Problematic Parameter Values

While local composition models have been successfully used to predict LLEs, their flexibility has been proven to lead to wrong predictions (predicting wrong phase splits or more phases than measured, modeling homogeneous azeotropes as heterogeneous, etc. (Bollas et al., 2009, Marcilla et al., 2017, 2025, Mitsos et al., 2009b)). For example, consider a binary mixture that has one observed liquid-liquid split at x_1 and x_2 for a total composition x_{total}. Figure 5.1a shows the Gibbs free energy of mixing $\bar{G}^{\text{M}}$ used to model this LLE. If the total composition is between $x_1 < x_{\text{total}} < x_2$ and a rigorous method is used for the predictions, the observed phase split is predicted. However, if the total composition is, e.g., x'_{total}, the spurious phase split at $\approx x'_1$ and $\approx x_1$ is predicted; even if a rigorous method is being used. This is due to $\bar{G}^{\text{M}}$ being non-convex on the left-hand side of x_1. Therefore, the corresponding parameter values are considered *problematic*. This example also highlights the drawbacks if only the necessary isopotential condition (C-ip) is used for the predictions without (C-TP-SIP). Then, for x_{total} the phase split with the common tangent line at x'_1 and $\approx x_2$ may be wrongly predicted. Note that the wrong phase split prediction between x'_1 and $\approx x_2$ fulfill condition (C-ip) but not (C-TP-SIP). To exclude such *problematic* parameter values, the model predictions are usually checked *after* parameter estimation. Otherwise, as in our example, incorrect predictions may be encountered in the modeled processes if these *problematic* parameters are used. The method proposed by Mitsos et al. (2009b) addresses the limitations of standard parameter estimation methods. Mitsos et al. (2009b) introduce a series of conditions to check stability *during* parameter estimation to exclude these *problematic* parameter values. The inclusion of these conditions into the parameter estimation leads to a BLP.

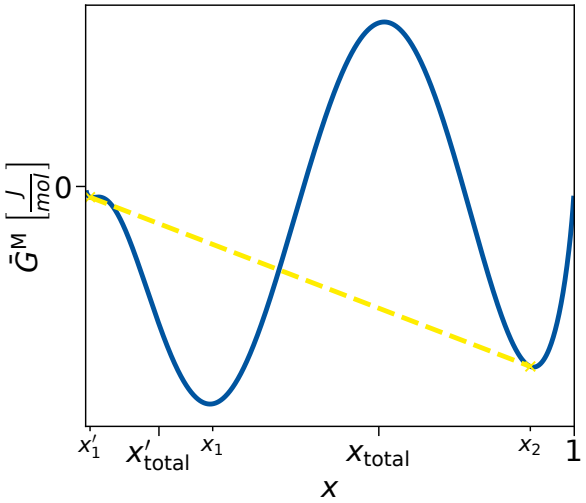
Several extensions (Bollas et al., 2009, Glass and Mitsos, 2017, 2019, Glass et al., 2017, 2018, Mitsos et al., 2009a) have utilized and extended the rigorous parameter estimation method, excluding *problematic* parameter values and considering a variety of thermodynamic systems. Note that the methods have been implemented in the software package Bilevel Optimization Algorithm for Rigorous & Robust Parameter Estimation in Thermodynamics (BOARPET) (Mitsos et al., 2018) and that BOARPET is also available as a plugin to the DECHEMA package DPP (DECHEMA e.V., 2024).

In this contribution, we utilize the idea of Mitsos et al. (2009b) of checking phase stability during parameter estimation by including it directly into the OED for GPE. In contrast to Mitsos et al. (2009b), who require that their additional conditions for parameter values $\bar{\boldsymbol{p}}$ *only* hold for the measured temperatures, we require the conditions to hold for *all* possible temperature values, i.e., over a continuous temperature range. By pursuing the idea in Mitsos et al. (2009b) with greater rigor, the corresponding (generalized) semi-infinite constraints become multidimensional and, as such, more difficult. If the conditions outlined by Mitsos et al. (2009b) are satisfied over the entire temperature range for a given set of parameter values, we postulate that the resulting LLE predictions are unique, e.g., no additional (wrong) phases are predicted. By including phase stability checks into the OED framework, we aim to ensure that (i) the planned experiments are meaningful —this may not be the case if a wrong phase split is predicted — and (ii) the prediction of $\boldsymbol{x}$ is unique. The second part is already included in the first, but it should be emphasized as it is required by the solution algorithm, which is heavily based on Djelassi et al. (2019) (and specialized with ideas from Falk and Hoffman (1977), Mitsos and Tsoukalas (2015)).

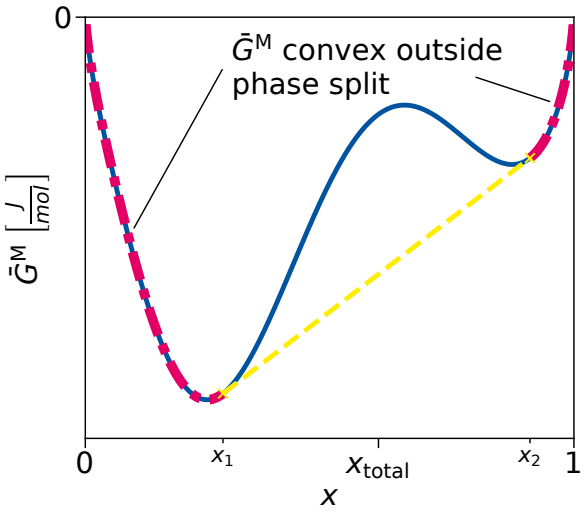
Mitsos et al. (2009b) consider a binary mixture with one liquid-liquid split over the temperature (and pressure) range under consideration. Mitsos et al. (2009b) motivate that parameter values $\bar{\boldsymbol{p}}$ should only be considered feasible if $\bar{G}^{\mathrm{M}}$ is convex in x outside the phase split, the Gibbs TPs developed at at x_{L} and x_{U} lies below the Gibbs free energy surface, and the secant, which is constructed approximately at the phase split, is strictly below the Gibbs surface within the phase split, c.f., Figures 5.1b to 5.1d.

We call the extended condition that $\bar{G}^{\mathrm{M}}$ must be convex on the whole temperature range with respect to the mole fraction outside the phase split (C-c-GSIP). The mathematical definition of (C-c-GSIP), along with its corresponding subproblems, is provided in Appendix D.2.1. Figure 5.1b visualizes (C-c-GSIP) which is a generalized semi-infinite constraint. Reformulation, relaxation, and discretization (akin (C-TP)) of (C-c-GSIP) yield two semi-infinite constraints (for the left- and right-hand side outside the phase split), which we call (C-cl) and (C-cr), respectively. For fixed parameter values $\bar{\boldsymbol{p}}$, we can check if these parameter values fulfill (C-c-GSIP) by solving two subproblems which are SIPs and which we call (SIP-cl) and (SIP-cr). If (the final lower bound of) the optimal objective value of these two subproblems is less equal to $\varepsilon^{\mathrm{a,SIP\text{-}cr}}$, $\bar{\boldsymbol{p}}$ fulfills the corresponding condition. If not, the optimal solution T^* and y^* of the subproblems are used to discretize and refine the discretized (C-cl) and (C-cr).

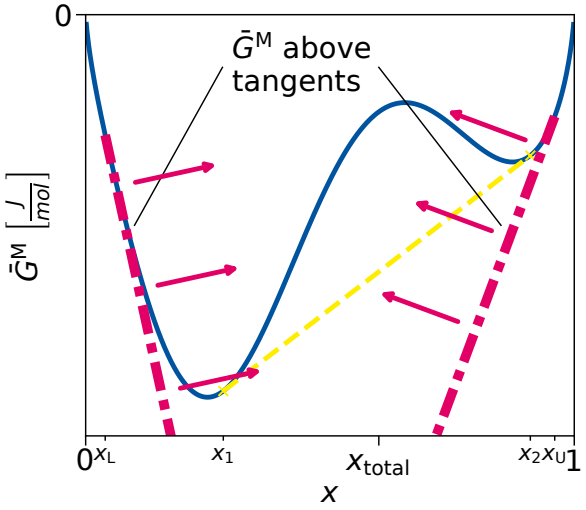
Additionally, the Gibbs TPs developed at the x_{L} and x_{U} must lie below $\bar{G}^{\mathrm{M}}$. We call this condition (C-tpl-SIP) and (C-tpr-SIP) for the right- and left- hand side, respectively (for mathematical definition, derived conditions and respective subproblems (to follow) refer to Appendix D.2.2). Figure 5.1c visualizes (C-tpl-SIP) and (C-tpr-SIP) which are semi-infinite constraints. These two conditions exclude parameter values that will predict additional spurious phase splits outside the true one (case E in Mitsos et al. (2009b)), e.g., parameter values producing a $\bar{G}^{\mathrm{M}}$ curve as depicted in Figure 5.1a would be excluded.



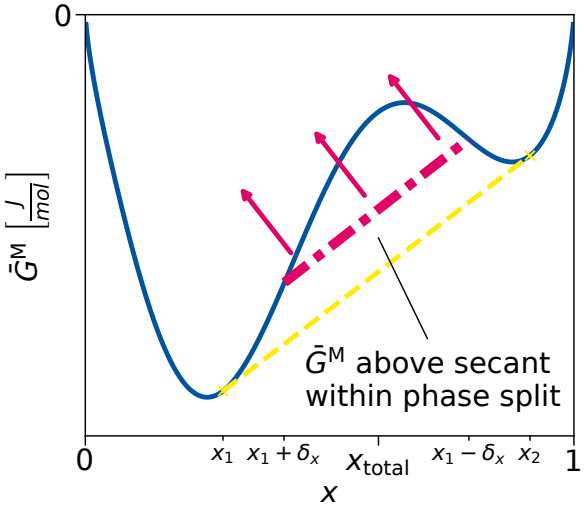
(a) Visualization of $\bar{G}^{\mathrm{M}}$ of parameter values $\boldsymbol{p}$, which may lead to wrong predictions (due to $\bar{G}^{\mathrm{M}}$ being non-convex left of x_1), making them *problematic*, and, thus, considered infeasible. Note that the shown common tangent line fulfills (C-ip) but not (C-TP-SIP). This tangent is only found if a non-rigorous solution method is used.



(b) Visualization of (C-c-GSIP). To avoid erroneous predictions, $\bar{G}^{\mathrm{M}}$ should be convex with respect to the mole fraction outside the phase split. Parameter values, as used in Figure 5.1a, are excluded by the shown condition.



(c) Visualization of (C-tpl-SIP) and (C-tpr-SIP). The centers of the tangent expansion are x_{L} and x_{U}, respectively. To avoid erroneous predictions, $\bar{G}^{\mathrm{M}}$ should lie above the tangents. Parameter values, as used in Figure 5.1a, are excluded by the shown conditions.



(d) Visualization of (C-sec-GSIP). To avoid erroneous predictions, $\bar{G}^{\mathrm{M}}$ should lie above the shown secant in the interior of the phase split. Note that parameter values, as used in Figure 5.1a, are not excluded by the shown condition because the respective secant in the interior of the phase split x_1 and x_2 fulfills the shown condition.

Figure 5.1: Figure 5.1a visualizes $\bar{G}^{\mathrm{M}}$ for parameter values which are not considered feasible. Figures 5.1b to 5.1d visualize the conditions for parameter values $\bar{\boldsymbol{p}}$ to be considered feasible by Mitsos et al. (2009b). Note that the visualization shows the respective conditions for a fixed temperature and not, as required in this contribution, over the whole temperature range $\mathcal{T}$. The phase split is denoted by x_1 and x_2. Yellow dashed lines represent the common tangent lines. Note: Reprinted from Jungen et al. (2026), CC BY 4.0, © The Author(s) 2026. Published by Elsevier Ltd.

Again, through discretization, we obtain two constraints which we call (C-tpl) and (C-tpr). Similarly, we can check if parameter values $\bar{\boldsymbol{p}}$ fulfill conditions (C-tpl-SIP) and (C-tpr-SIP) by solving subproblems which we call (P-tpl) and (P-tpr). Again, if (the final lower bound of) the optimal objective value of these two subproblems is less equal to $\varepsilon^{\text{a,P-tp}}$, $\bar{\boldsymbol{p}}$ fulfills the corresponding conditions.

To exclude additional phases from the predicted phase split, Mitsos et al. (2009b) additionally require that the secant, which is constructed approximately at the phase split, is strictly below the Gibbs surface in the interior of the predicted phase split (case C in Mitsos et al. (2009b)), c.f. Figure 5.1d. We approximate the interior by $[x_1 + \delta, x_2 - \delta]$, with some small back-up $\delta > 0$. We call this condition (C-sec-GSIP), which depends on the temperature T and parameter values $\bar{\boldsymbol{p}}$ (mathematical definition and respective subproblems see Appendix D.2.3). The condition is a generalized semi-infinite constraint. Reformulation, relaxation, and subsequent discretization yield a discretized constraint which we call (C-sec). Again, feasibility of parameter values $\bar{\boldsymbol{p}}$ in (C-sec-GSIP) can be checked by solving a subproblem, called (SIP-sec), which itself is an SIP. If (the final lower bound of) the optimal objective value of this subproblem is less equal to a small $\varepsilon^{\text{a,SIP-sec}}$, $\bar{\boldsymbol{p}}$ fulfills the corresponding condition. Otherwise the optimal solution T^* and y^* of (SIP-sec) are used to discretize (C-sec).

5.3 Optimal Experimental Design for Guaranteed Parameter Estimation

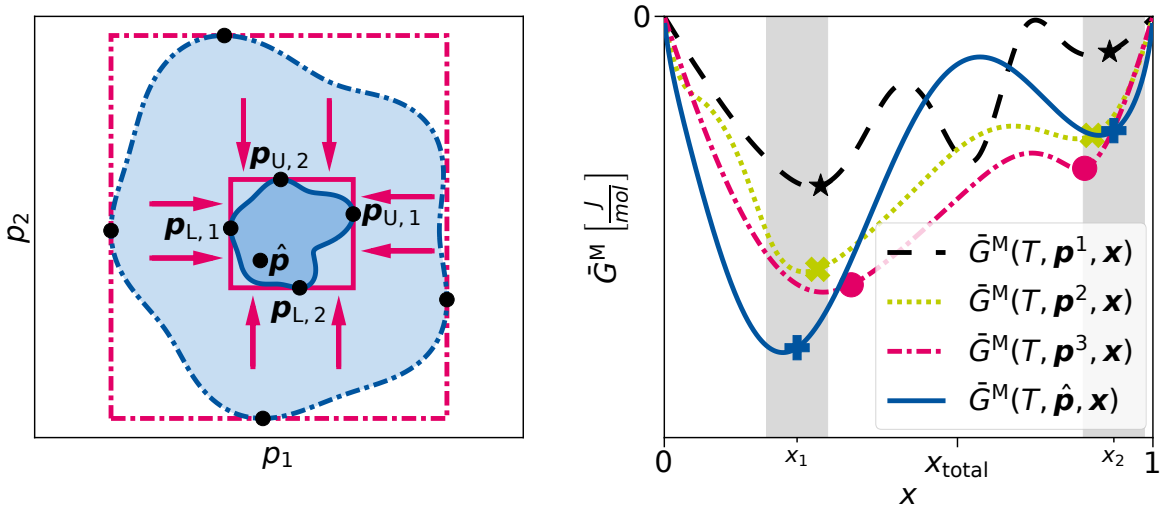
The OED for GPE problem reads (Gottu Mukkula and Paulen, 2016a,b, 2017a, Walz et al., 2018)

$$\begin{aligned} \min_{\boldsymbol{v}\in\mathcal{V}} \max_{\boldsymbol{p}_{j,i}\in\mathcal{P}(\boldsymbol{v})} \quad & \textstyle\sum_{i=1}^{n_\text{p}} (p_{\text{U},i,i} - p_{\text{L},i,i}) \\ \text{with} \quad \mathcal{P}(\boldsymbol{v}) = & \left\{ \boldsymbol{p} \in \tilde{\mathcal{P}} \middle| \begin{array}{l} \boldsymbol{2}\eta_\text{L} \leq \hat{\boldsymbol{x}} - \boldsymbol{x} \\ \boldsymbol{2}\eta_\text{U} \geq \hat{\boldsymbol{x}} - \boldsymbol{x} \\ \hat{\boldsymbol{x}} \text{ is the model system output for } \boldsymbol{v} \text{ and } \hat{\boldsymbol{p}} \\ \boldsymbol{x} \text{ is the model system output for } \boldsymbol{v} \text{ and } \boldsymbol{p} \end{array} \right\}, \end{aligned} \tag{5.1}$$

where $\boldsymbol{p}_{j,i} = \left(p_{j,i,1}, \ldots, p_{j,i,n_\text{p}}\right)^T$ with $j \in \mathcal{J} = \{\text{L}, \text{U}\}$ and $i \in \mathcal{I} = \{1, \ldots, n_\text{p}\}$, $\tilde{\mathcal{P}}$ and $\mathcal{V}$ are subsets of the real numbers of dimension n_p and n_v, $\hat{\boldsymbol{p}}$ are the nominal parameter values, η_L and η_U denote the maximum upper- and lower measurement errors, and $\boldsymbol{x}$ is the measured model system output. In (5.1), we aim to find the experimental conditions $\boldsymbol{v}$ that minimize the size of the estimated feasible parameter set. Considering the trade-off between computational burden and precision, we pragmatically choose to over-approximate the size of the feasible parameter set using an orthotope, c.f., Figure 5.2a, and minimize its circumference. This orthotope, which encloses the feasible parameter set, is defined by $2n_\text{p}$ points in the parametric space, i.e., $\boldsymbol{p}_{\text{L},i}$ and $\boldsymbol{p}_{\text{U},i}$, each being a vector of size n_p. Extending the approach to other (tighter) convex over-approximations is straightforward, e.g., minimizing the volume of the orthotope, similar to A-optimal designs of statistical approaches that minimize the dimensions of the enclosing box around the joint confidence interval (Asprey and Macchietto, 2002). The nominal parameter values $\hat{\boldsymbol{p}}$ are known *a priori* and are consistent with the measurements, i.e., they are contained in $\mathcal{P}(v)$ (for all

possible v). Therefore, in OED for GPE, the feasible set of the lower level $\mathcal{P}(v)$ is never empty; in contrast to GPE, where this might be the case (Walz et al., 2018). Therefore, we call (5.1) a minmax BLP, analogous to BLPs where the lower level is also never infeasible. To obtain meaningful results, the number of experiments should be equal to the number of uncertain parameter values, i.e., $n_\mathrm{v} = n_\mathrm{p}$. Similar to Gottu Mukkula and Paulen (2017a), we assume that $\mathcal{P}$ is not limited by $\tilde{\mathcal{P}}$, i.e., $\partial\mathcal{P}(v) \cap \tilde{\mathcal{P}} = \emptyset$, $\forall v$. One could naively set $\tilde{\mathcal{P}} = \mathbb{R}^{n_\mathrm{P}}$; however, this leads to profound implications for the employed solution algorithm, as will be discussed in Chapter 6 for the case of SIPs. Therefore, we pragmatically chose $\tilde{\mathcal{P}}$ as a compact set that is sufficiently large.

For now, assume an analytical input-output relationship for $\boldsymbol{x}$ exists, i.e., we have an analytical function that gives us $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{v}, \boldsymbol{p})$. Figure 5.2b shows, assuming such an analytical input-output relationship for the mole fractions $\boldsymbol{x}$ exists, the prediction of the LLE for different parameter values. The prediction with parameter values $\boldsymbol{p}^3$ is not within the measurement error, therefore not considered feasible.



(a) Visualization of the feasible parameter set for $n_\mathrm{p} = 2$ and aspired minimization of (the overapproximation) thereof. Based on Gottu Mukkula and Paulen (2016a), Walz et al. (2018).

(b) Visualization of the measurement uncertainty (shaded gray of width $\eta_\mathrm{U} - \eta_\mathrm{L}$) and $\bar{G}^\mathrm{M}$ for different parameter values. The predicted phase split with the nominal parameter values $\hat{\boldsymbol{p}}$ is denoted with x_1 and x_2.

Figure 5.2: Visualization of the feasible parameter set, model outputs, and measurement uncertainty. Note: Reprinted from Jungen et al. (2026), CC BY 4.0, © The Author(s) 2026. Published by Elsevier Ltd.

5.3.1 Optimal Experimental Design for Liquid-Liquid Equilibrium Systems

In the closely related work of Gottu Mukkula and Paulen (2016a,b, 2017a), Walz et al. (2018), an explicit input-output relation is required for the underlying model of (5.1), i.e., $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{v}, \boldsymbol{p})$, where the model function $\boldsymbol{g} : \mathbb{R}^{n_\mathrm{v}} \times \mathbb{R}^{n_\mathrm{p}} \to \mathbb{R}^{n_x}$, corresponds to an analytical input-output relationship of the system outputs $\boldsymbol{x}$. This restriction might be one of the reasons why OED for GPE is still considered infeasible in flow sheeting software (Asprion et al., 2020). In this section, we extend the OED for GPE approach reviewed in Section 5.3 to the case where the model output itself is implicitly defined by an optimization problem, as in the case for computing LLEs with (P-TP). Note that the applied solution strategy (to follow) requires that the solution of this optimization problem is unique. In our considered OED application, for a fixed total composition, we aim to find the optimal temperature T to reduce the parameter uncertainty in the underlying excess model. We consider the mole fractions of the phases as the model outputs. Since, at a given temperature, the mole fractions of the phases are uniquely determined, the uniqueness assumption is fulfilled in our application. This is presumably the case in *real-life* as well as for our *predictions*. Note that extending to multiple experiments is conceptually straightforward but is omitted for notational convenience.

The resulting OED problem with rigorous LLE computations embedded reads

$$\min_{T \in \mathcal{T}} \max_{\boldsymbol{p}_{j,i} \in \mathcal{P}(T)} \quad \sum_{i=1}^{n_\mathrm{p}} (p_{\mathrm{U},i,i} - p_{\mathrm{L},i,i})$$

$$\text{with} \quad \mathcal{P}(T) = \left\{ \boldsymbol{p} \in \tilde{\mathcal{P}} \;\middle|\; \begin{array}{l} \boldsymbol{2}\eta_\mathrm{L} \leq \hat{\boldsymbol{x}} - \boldsymbol{x} \\ \boldsymbol{2}\eta_\mathrm{U} \geq \hat{\boldsymbol{x}} - \boldsymbol{x} \\ \text{(C-ip) \& (MB) hold for } T, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}} \\ \text{(C-ip) \& (MB) hold for } T, \boldsymbol{p}, \boldsymbol{x} \\ \text{(C-TP-SIP) holds for } T, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}} \\ \text{(C-TP-SIP) holds for } T, \boldsymbol{p}, \boldsymbol{x} \\ \text{(C-c-GSIP) holds for } \boldsymbol{p} \\ \text{(C-tpl-SIP) holds for } \boldsymbol{p} \\ \text{(C-tpr-SIP) holds for } \boldsymbol{p} \\ \text{(C-sec-GSIP) holds for } \boldsymbol{p} \end{array} \right\}. \tag{OED}$$

We assume that the nominal parameter values $\hat{\boldsymbol{p}}$ fulfill the conditions (C-c-GSIP), (C-tpl-SIP), (C-tpr-SIP), (C-sec-GSIP). This is not restrictive as, e.g., libDIPS comes with the capabilities of estimating parameter values, enforcing conditions like (C-c-GSIP), (C-tpl-SIP), (C-tpr-SIP), (C-sec-GSIP) on a temperature range.

Note that (OED) is, by assumption, feasible. In contrast to previous work, the measured model system output is determined implicitly by (C-ip), (MB), and a semi-infinite constraint, i.e., (C-TP-SIP). Additional semi-infinite and generalized semi-infinite constraints are imposed, i.e., the conditions of Mitsos et al. (2009b), to exclude *problematic* parameter values. Revisiting Figure 5.2b, in addition to parameter values $\boldsymbol{p}^3$ (prediction not within measurement errors), now $\boldsymbol{p}^1$ and $\boldsymbol{p}^2$ are also not considered feasible. The prediction with $\boldsymbol{p}^1$ does not fulfill (C-sec-GSIP), and $\boldsymbol{p}^2$ does not fulfill (C-c-GSIP).

Using the epigraph reformulation, we reformulate (OED), eliminating the T–dependency of $\mathcal{P}(T)$, to

$$\begin{aligned}
\min_{\substack{T\in\mathcal{T},\mu\in\mathbb{R},\\ \hat{\boldsymbol{x}},\boldsymbol{x}_{j,i}\in\mathcal{X}^2}} \quad & \mu \\
\text{s.t.} \quad & \left\{ \begin{array}{l} \displaystyle\sum_{i=1}^{n_\mathrm{p}} (p_{\mathrm{U},i,i} - p_{\mathrm{L},i,i}) - \mu \leq 0 \\ \vee \exists j, i : \begin{Bmatrix} \boldsymbol{2}\eta_\mathrm{L} \leq \hat{\boldsymbol{x}} - \boldsymbol{x}_{j,i} \vee \\ \boldsymbol{2}\eta_\mathrm{U} \geq \hat{\boldsymbol{x}} - \boldsymbol{x}_{j,i} \end{Bmatrix} \end{array} \right\}, \ \forall p_{j,i} \in \mathcal{P} \\
& \text{(C-ip) \& (MB) hold for } T, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}} \\
& \text{(C-ip) \& (MB) hold for } T, \boldsymbol{p}, \boldsymbol{x}_{j,i} \\
& \text{(C-TP-SIP) holds for } T, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}} \\
& \text{(C-TP-SIP) holds for } T, \boldsymbol{p}, \boldsymbol{x}_{j,i}
\end{aligned} \tag{5.2}$$

$$\text{with} \quad \mathcal{P} = \left\{ \boldsymbol{p} \in \tilde{\mathcal{P}} \middle| \begin{array}{l} \text{(C-c-GSIP) holds for } \boldsymbol{p} \\ \text{(C-tpl-SIP) holds for } \boldsymbol{p} \\ \text{(C-tpr-SIP) holds for } \boldsymbol{p} \\ \text{(C-sec-GSIP) holds for } \boldsymbol{p} \end{array} \right\}.$$

Note that we have introduced the additional variables μ, $\hat{\boldsymbol{x}}$, and $\boldsymbol{x}_{j,i}$, with $j \in \mathcal{J} = \{\mathrm{L}, \mathrm{U}\}$ and $i \in \mathcal{I} = \{1, \ldots, n_\mathrm{p}\}$.

Discretizing (5.2) and relaxing the strict inequalities (akin Walz et al. (2018)) yields the upper-level program

$$\begin{aligned}
\min_{\substack{T\in\mathcal{T},\mu\in\mathbb{R},\\ \hat{\boldsymbol{x}},\boldsymbol{x}^d_{j,i}\in\mathcal{X}^2}} \quad & \mu \\
\text{s.t.} \quad & \left\{ \begin{array}{l} \displaystyle\sum_{i=1}^{n_\mathrm{p}} \left(p^d_{\mathrm{U},i,i} - p^d_{\mathrm{L},i,i}\right) - \mu \leq 0 \\ \vee \exists j, i : \begin{Bmatrix} \boldsymbol{2}\eta_\mathrm{L} \geq \hat{\boldsymbol{x}} - \boldsymbol{x}^d_{j,i} \vee \\ \boldsymbol{2}\eta_\mathrm{U} \leq \hat{\boldsymbol{x}} - \boldsymbol{x}^d_{j,i} \end{Bmatrix} \end{array} \right\}, \quad \forall d \in \mathcal{D}^{\text{OED-ULP}} \\
& \text{(C-ip) \& (MB) hold for } T, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}} \\
& \text{(C-ip) \& (MB) hold for } T, \boldsymbol{p}, \boldsymbol{x}^d_{j,i} \quad \forall d \in \mathcal{D}^{\text{OED-ULP}} \\
& \text{(C-TP) holds for } T, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}}, \qquad\qquad\qquad\qquad \forall l \in \mathcal{D}^{\text{C-TP}} \\
& \text{(C-TP) holds for } T, \boldsymbol{p}^d_{j,i}, \boldsymbol{x}^d_{j,i}, \quad \forall d \in \mathcal{D}^{\text{OED-ULP}} \quad \forall l \in \mathcal{D}^{\text{C-TP}}
\end{aligned} \tag{OED-ULP}$$

The conjunction in (OED-ULP) can be reformulated using the min-operator, making it solvable with standard NLP solvers. Note that the number of optimization variables depends on the discretization $\mathcal{D}^{\text{OED-ULP}}$. (OED-ULP) has $4 + 2n_\mathrm{p}\left|\mathcal{D}^{\text{OED-ULP}}\right|$ variables. (OED-ULP) has (P-TP) as a LLP for $\hat{\boldsymbol{x}}$ and for each $\boldsymbol{x}^d_{j,i}$. The solutions thereof are used to populate the set $\mathcal{D}^{\text{C-TP}}$.

Although one discretization set $\mathcal{D}^{\text{C-TP}}$ suffices, in the numerical case study in Section 5.5, a separate discretization set is introduced for each semi-infinite constraint. As a result, the number of constraints and, therefore, the computational burden is reduced.

To populate the discretization set $\mathcal{D}^{\text{OED-ULP}}$, a medial-level problem, and in some cases an additional auxiliary problem (to follow) must be solved. The medial-level problem is

developed from

$$
\begin{aligned}
\max_{\substack{\boldsymbol{p}_{j,i}\in\bar{\mathcal{P}},\\ \boldsymbol{x}_{j,i},\boldsymbol{x}^d_{c,j,i}\in\mathcal{X}}} \quad & \textstyle\sum_{i=1}^{n_\mathrm{p}}(p_{\mathrm{U},i,i}-p_{\mathrm{L},i,i})-\bar{\mu} \\
\text{s.t.} \quad & \mathbf{2}\eta_\mathrm{L} \leq \bar{\hat{\boldsymbol{x}}}-\boldsymbol{x}_{j,i} \\
& \mathbf{2}\eta_\mathrm{U} \geq \bar{\hat{\boldsymbol{x}}}-\boldsymbol{x}_{j,i} \\
& \\
& \text{(C-ip) \& (MB) hold for } \bar{T},\boldsymbol{p},,\boldsymbol{x}_{j,i} \\
& \text{(C-TP-SIP) holds for } \bar{T},\boldsymbol{p}_{j,i},\boldsymbol{x}_{j,i} \\
& \\
& \text{(C-c-GSIP) holds for } \boldsymbol{p}_{j,i} \\
& \text{(C-tpl-SIP) holds for } \boldsymbol{p}_{j,i} \\
& \text{(C-tpr-SIP) holds for } \boldsymbol{p}_{j,i} \\
& \text{(C-sec-GSIP) holds for } \boldsymbol{p}_{j,i},
\end{aligned}
\tag{5.3}
$$

with $\bar{T}$, $\bar{\hat{\boldsymbol{x}}}$, and $\bar{\mu}$ being fixed to the optimal values of (OED-ULP). Note that $\bar{\mu}$ is retained in the objective function solely for notational convenience. (5.3) computes the overapproximated feasible parameter set for a given temperature $\bar{T}$. Using the same strategy as before, discretizing yields

$$
\begin{aligned}
f^{\text{OED-MLP}} = \quad & \max_{\substack{\boldsymbol{p}_{j,i}\in\bar{\mathcal{P}},\\ \boldsymbol{x}_{j,i},\boldsymbol{x}^d_{c,j,i}\in\mathcal{X}}} \quad \textstyle\sum_{i=1}^{n_\mathrm{p}}(p_{\mathrm{U},i,i}-p_{\mathrm{L},i,i})-\bar{\mu} \\
\text{s.t.} \quad & \mathbf{2}\eta_\mathrm{L} \leq \bar{\hat{\boldsymbol{x}}}-\boldsymbol{x}_{j,i} \\
& \mathbf{2}\eta_\mathrm{U} \geq \bar{\hat{\boldsymbol{x}}}-\boldsymbol{x}_{j,i} \\
& \\
& \text{(C-ip) \& (MB) hold for } \bar{T},\boldsymbol{p}_{j,i},\boldsymbol{x}_{j,i} \\
& \text{(C-TP) holds for } \bar{T},\boldsymbol{p}_{j,i},\boldsymbol{x}_{j,i} && \forall l\in\mathcal{D}^{\text{C-TP}} \\
& \\
& \text{(C-TP) holds for } T^d_{\text{C-cl}},\boldsymbol{p}_{j,i},\boldsymbol{x}^d_{\text{C-cl},j,i}, && \forall d\in\mathcal{D}^{\text{C-cl}}, \quad \forall l\in\mathcal{D}^{\text{C-TP}} \\
& \text{(C-cl) holds for } T^d_{\text{C-cl}},\boldsymbol{p}_{j,i},\boldsymbol{x}^d_{\text{C-cl},j,i}, && \forall d\in\mathcal{D}^{\text{C-cl}} \\
& \\
& \text{(C-TP) holds for } T^d_{\text{C-cr}},\boldsymbol{p}_{j,i},\boldsymbol{x}^d_{\text{C-cr},j,i}, && \forall d\in\mathcal{D}^{\text{C-cr}}, \quad \forall l\in\mathcal{D}^{\text{C-TP}} \\
& \text{(C-cr) holds for } T^d_{\text{C-cr}},\boldsymbol{p}_{j,i},\boldsymbol{x}^d_{\text{C-cr},j,i}, && \forall d\in\mathcal{D}^{\text{C-cr}} \\
& \\
& \text{(C-tpl) holds for } \boldsymbol{p}_{j,i} && \forall d\in\mathcal{D}^{\text{C-tpl}} \\
& \text{(C-tpr) holds for } \boldsymbol{p}_{j,i} && \forall d\in\mathcal{D}^{\text{C-tpr}} \\
& \\
& \text{(C-TP) holds for } T^d_{\text{C-sec}},\boldsymbol{p}_{j,i},\boldsymbol{x}^d_{\text{C-sec},j,i}, && \forall d\in\mathcal{D}^{\text{C-sec}}, \quad \forall l\in\mathcal{D}^{\text{C-TP}} \\
& \text{(C-sec) holds for } T^d_{\text{C-sec}},\boldsymbol{p}_{j,i},\boldsymbol{x}^d_{\text{C-sec},j,i}, && \forall d\in\mathcal{D}^{\text{C-sec}},
\end{aligned}
\tag{OED-MLP}
$$

with $c\in\mathcal{C}=\{\text{C-cl},\text{C-cr},\text{C-sec}\}$. (OED-MLP) has $6n_\mathrm{p}(1+\left|\mathcal{D}^{\text{C-cl}}\right|)$ variables. (OED-MLP) has the following LLPs: (P-TP) for $\hat{\boldsymbol{x}}$, each $\boldsymbol{x}_{j,i}$, and each $\boldsymbol{x}^d_{c,j,i}$; (SIP-cl); (SIP-cr); (P-tpl); (P-tpr); and (SIP-sec), which are used to populate the respective discretized sets. Similar to the solution approach by Mitsos and Tsoukalas (2015), which we implemented as *GSIP-RRHS* in libDIPS, see Section 3.3.2, an auxiliary problem has to be solved to generate

the discretization for $\mathcal{D}^{\text{OED-ULP}}$ if no GSIP-like Slater point is obtained in (OED-MLP). For the definition of a GSIP-LLP-Slater point refer to Definition 7 in Appendix B.2. If a GSIP-like Slater point is already obtained in (OED-MLP), its solution is used to populate $\mathcal{D}^{\text{OED-ULP}}$. Note that it is also possible to solve a subproblem, which directly maximizes the semi-infinite constraint of (OED-ULP), instead of first solving (OED-MLP) and then, if necessary, an additional auxiliary problem. This approach is pursued by Djelassi et al. (2019) but comes with the drawback that this subproblem is of increased size. Recall that this approach corresponds to solving the SIP which has been derived from the original GSIP, see Section 3.3.2. Our computational results in Section 5.5 (to follow) indicate that the complexity of the subproblem is a limiting factor, suggesting that this approach may not be optimal for our application.

For fixed $\bar{T}$, $\bar{\hat{\boldsymbol{x}}}$, and $\bar{\mu}$ from (OED-ULP) and $\alpha > 0$, the auxiliary problem is defined as

$$
\begin{aligned}
f^{\text{OED-AUX}} = \quad & \max_{\substack{\zeta\in\mathbb{R},\boldsymbol{p}_{j,i}\in\tilde{\mathcal{P}},\\ \boldsymbol{x}_{j,i},\boldsymbol{x}^d_{c,j,i}\in\mathcal{X}}} \zeta \\
\text{s.t.} \quad & \textstyle\sum_{i=1}^{n_\text{p}}(p_{\text{U},i,i} - p_{\text{L},i,i}) - \bar{\mu} \geq \alpha f^{\text{OED-MLP}} \\
& \boldsymbol{2}\eta_\text{L} - \left(\bar{\hat{\boldsymbol{x}}} - \boldsymbol{x}_{j,i}\right) \leq \boldsymbol{1}\zeta \\
& -\boldsymbol{2}\eta_\text{U} + \left(\bar{\hat{\boldsymbol{x}}} - \boldsymbol{x}_{j,i}\right) \leq \boldsymbol{1}\zeta \\
& \text{(C-ip) \& (MB) hold for } \bar{T}, \boldsymbol{p}_{j,i}, \boldsymbol{x}_{j,i} \\
& \text{(C-TP) holds for } \bar{T}, \boldsymbol{p}_{j,i}, \boldsymbol{x}_{j,i}, && \forall l \in \mathcal{D}^{\text{C-TP}} \\
& \text{(C-TP) holds for } T^d_{\text{C-cl}}, \boldsymbol{p}_{j,i}, \boldsymbol{x}^d_{\text{C-cl},j,i}, && \forall d \in \mathcal{D}^{\text{C-cl}}, \quad \forall l \in \mathcal{D}^{\text{C-TP}} \\
& \text{(C-cl) holds for } T^d_{\text{C-cl}}, \boldsymbol{p}_{j,i}, \boldsymbol{x}^d_{\text{C-cl},j,i}, && \forall d \in \mathcal{D}^{\text{C-cl}} \\
& \text{(C-TP) holds for } T^d_{\text{C-cr}}, \boldsymbol{p}_{j,i}, \boldsymbol{x}^d_{\text{C-cr},j,i}, && \forall d \in \mathcal{D}^{\text{C-cr}}, \quad \forall l \in \mathcal{D}^{\text{C-TP}} \\
& \text{(C-cr) holds for } T^d_{\text{C-cr}}, \boldsymbol{p}_{j,i}, \boldsymbol{x}^d_{\text{C-cr},j,i}, && \forall d \in \mathcal{D}^{\text{C-cr}} \\
& \text{(C-tpl) holds for } \boldsymbol{p}_{j,i} && \forall d \in \mathcal{D}^{\text{C-tpl}} \\
& \text{(C-tpr) holds for } \boldsymbol{p}_{j,i} && \forall d \in \mathcal{D}^{\text{C-tpr}} \\
& \text{(C-TP) holds for } T^d_{\text{C-sec}}, \boldsymbol{p}_{j,i}, \boldsymbol{x}^d_{\text{C-sec},j,i}, && \forall d \in \mathcal{D}^{\text{C-sec}}, \quad \forall l \in \mathcal{D}^{\text{C-TP}} \\
& \text{(C-sec) holds for } T^d_{\text{C-sec}}, \boldsymbol{p}_{j,i}, \boldsymbol{x}^d_{\text{C-sec},j,i}, && \forall d \in \mathcal{D}^{\text{C-sec}}.
\end{aligned}
\tag{OED-AUX}
$$

(OED-AUX) has $6n_\text{p}(1 + \left|\mathcal{D}^{\text{C-cl}}\right|) + 1$ variables. The subproblems and discretization procedures of (OED-AUX) are identical to the ones of (OED-MLP).

Given the above-defined subproblems, we can solve (OED) by an adaptation of the algorithms proposed by Djelassi et al. (2019), Falk and Hoffman (1977), Mitsos and Tsoukalas (2015). From Djelassi et al. (2019), we adapt the idea to handle equality constraints, from Mitsos and Tsoukalas (2015) (see solver *GSIP-RRHS* in Section 3.3.2) we adapt the idea to solve an auxiliary problem to generate the discretization, and from Falk and Hoffman (1977) (see solver *Minmax* in Section 3.3.1), we adapt the idea to use the solution of (OED-MLP) as the upper-bound. The latter is also done by Walz et al. (2018). Note that these successors (Djelassi et al., 2019, Mitsos and Tsoukalas, 2015) utilize an upper-bounding procedure to find a global solution that is (SIP-)feasible, c.f., Definition 4 in Appendix B.1.

However, in our application, the necessary assumption for the upper-bounding procedures is not satisfied due to the reformulations we applied. As a result, we rely solely on the relaxation-based lower-bounding procedure. The simplified pseudocode is given by Algorithm 5.1, which uses Algorithm 5.2 and the solver *B&F* (simplified pseudocode see Algorithm C.1 in Appendix C.1), as subprocedures. Note that in the loop starting in line 7 of Algorithm 5.2, we check conditions (C-cl), (C-cr), (C-sec). Therefore, we loop over $c \in \mathcal{C} = \{\text{C-cl}, \text{C-cr}, \text{C-sec}\}$.

As the proposed algorithms are adaptations of Djelassi et al. (2019), Falk and Hoffman (1977), Mitsos and Tsoukalas (2015), which have been proven to converge within finite time, we argue in the following that their convergence guarantees are transferable to our adaptation. The assumptions necessary by Falk and Hoffman (1977) and Mitsos and Tsoukalas (2015) are included in Djelassi et al. (2019) which assume; Assumption 1: compact host sets; Assumption 2: continuous functions; Assumption 3: approximate solution of subproblems; Assumption 4: the solution of the coupling equality constraints is unique and exists; Assumption 5: the infimum of GSIP is equal to the optimum of the relaxed problem; Assumption 6: existence of ε^{f}-optimal GSIP-Slater point. Assumptions 1, 2, and 6 hold for our case. Assumptions 3 and 5 are standard technical assumptions. Assumption 4 of Djelassi et al. (2019) warrants further discussion. (C-ip) is an equality constraint which couples the upper and media level, while (C-TP-SIP) is a selection constraint, ensuring that the solution of (C-ip), i.e., the predicted phase split, is unique. There exists a discretization of $\mathcal{D}^{\text{C-TP}}$, such that when using criterion (C-TP-SIP) and (used and discretized) condition (C-TP) predict the same unique phase split (within some tolerance). Through the algorithm, we ensure that such a discretization is found. Now, assume parameter values are chosen by the solver (when solving (OED-MLP)) that do not fulfill a condition reviewed in Section 5.2. These parameter values are subsequently excluded by solving the corresponding subproblem and refining the respective discretization(s), c.f., Algorithm 5.2. Therefore, *problematic* parameter values, which might produce non-unique predictions, are excluded.

Note that some issues concerning the generation of GSIP-like Slater points likely persist, c.f. Section 3.3.2. However, this could be remedied by directly maximizing the semi-infinite constraint of (OED-ULP), instead of first solving (OED-MLP) and then, if necessary, the additional auxiliary problem. This approach is pursued by Djelassi et al. (2019); however, due to the increased computational burden, we refrain from using this approach.

5.4 Computational Setup

We implemented Algorithms 5.1 and 5.2 in libDIPS, see Section 3.4, and use the already implemented solver *B&F* as a subprocedure. The implementation of Algorithms 5.1 and 5.2 is strongly based on existing solvers implemented in libDIPS and can be seen as an adaptation. In the implementation, Algorithms 5.1 and 5.2 have been extended to support planning multiple experiments at different temperatures. This extension is straightforward. Note that planning multiple experiments leads to more complex and an increased number of subproblems.

The computations are conducted on the RWTH High-Performance Computing cluster CLAIX running Rocky Linux 9 using on each core two Intel Xeon 8468 Sapphire processors running at 2.1 GHz with up to 2540 Mbit RAM and 96 cores per cluster node. All

Algorithm 5.1: Simplified pseudocode for solving (OED). All subproblems are solved globally. User inputs are the initial discretization sets $\mathcal{D}^{\text{OED-ULP},0}$, $\mathcal{D}^{\text{C-TP},0}$, parameters α_{init}, α_{red}, and tolerances ε^{a}, $\varepsilon^{\text{a,G}}$.

1 set $\mathcal{D}^{\text{OED-ULP}} \leftarrow \mathcal{D}^{\text{OED-ULP},0}$, $\mathcal{D}^{\text{C-TP}} \leftarrow \mathcal{D}^{\text{C-TP},0}$, $LBD \leftarrow -\infty$, and $UDB \leftarrow \infty$;
2 solve (OED-ULP) to obtain $\mu^{\text{OED-ULP}}$, $T^{\text{OED-ULP}}$, $\hat{\boldsymbol{x}}^{\text{OED-ULP}}$, and $\boldsymbol{x}_{j,i}^{d,\text{OED-ULP}}$;
`// check stability of` $\hat{\boldsymbol{x}}^{\text{OED-ULP}}$
3 solve (P-TP) for $T^{\text{OED-ULP}}$, $\hat{\boldsymbol{p}}$, $\hat{\boldsymbol{x}}^{\text{OED-ULP}}$ to obtain $f^{\text{P-TP}}$ and $\boldsymbol{y}^*$;
4 **if** $f^{\text{P-TP}} > \varepsilon^{\text{a,G}}$ **then**
5 $\quad \mathcal{D}^{\text{C-TP}} \leftarrow \mathcal{D}^{\text{C-TP}} \bigcup \{\boldsymbol{y}^*\}$;
6 $\quad$ go to line 2;
`// check stability of` $\boldsymbol{x}_{j,i}^{d,\text{OED-ULP}}$
7 **for** *all* $j, i, d \in \mathcal{J}, \mathcal{I}, \mathcal{D}^{\text{OED-ULP}}$ **do**
8 $\quad$ solve (P-TP) for $T^{\text{OED-ULP}}$, $\boldsymbol{p}_{j,i}^{d}$, $\boldsymbol{x}_{j,i}^{d,\text{OED-ULP}}$ to obtain $f^{\text{P-TP}}$ and $\boldsymbol{y}^*$;
9 $\quad$ **if** $f^{\text{P-TP}} > \varepsilon^{\text{a,G}}$ **then**
10 $\quad\quad \mathcal{D}^{\text{C-TP}} \leftarrow \mathcal{D}^{\text{C-TP}} \bigcup \{\boldsymbol{y}^*\}$;
11 $\quad\quad$ go to line 2;
12 set $LBD \leftarrow \mu^{\text{OED-ULP}}$;
13 solve (OED-MLP) using Algorithm 5.2 for fixed $T^{\text{OED-ULP}}$ and $\mu^{\text{OED-ULP}}$ to obtain $f^{\text{OED-MLP}}$ and $\boldsymbol{p}_{j,i}^{\text{OED-MLP}}$;
14 **if** $f^{\text{OED-MLP}} > 0$ **then**
15 $\quad$ **if** $\mathbf{2}\eta_{\text{L}} < \hat{\boldsymbol{x}}^{\text{OED-ULP}} - \boldsymbol{x}_{j,i}^{\text{OED-MLP}} \wedge \mathbf{2}\eta_{\text{U}} > \hat{\boldsymbol{x}}^{\text{OED-ULP}} - \boldsymbol{x}_{j,i}^{\text{OED-MLP}}$ **then**
$\quad\quad$ `// Slater condition fulfilled`
16 $\quad\quad \mathcal{D}^{\text{OED-ULP}} \leftarrow \mathcal{D}^{\text{OED-ULP}} \bigcup \boldsymbol{p}_{j,i}^{\text{OED-MLP}}$
17 $\quad$ **else**
$\quad\quad$ `// generate valid discretization with` (OED-AUX)
18 $\quad\quad \alpha \leftarrow \alpha_{\text{init}}$;
19 $\quad\quad$ **do**
20 $\quad\quad\quad$ solve (OED-AUX) using Algorithm 5.2 for fixed $T^{\text{OED-ULP}}$, $\mu^{\text{OED-ULP}}$, and $f^{\text{OED-MLP}}$ to obtain $f^{\text{OED-AUX}}$ and $\boldsymbol{p}_{j,i}^{\text{OED-AUX}}$;
21 $\quad\quad\quad \alpha \leftarrow \alpha / \alpha_{\text{red}}$;
22 $\quad\quad$ **while** $f^{\text{OED-AUX}} > 0$;
23 $\quad\quad \mathcal{D}^{\text{OED-ULP}} \leftarrow \mathcal{D}^{\text{OED-ULP}} \bigcup \boldsymbol{p}_{j,i}^{\text{OED-AUX}}$
24 **else**
25 $\quad$ terminate;
26 **if** $F^*_{\textit{OED-MLP}} + \mu^* \leq UBD$ **then**
27 $\quad$ set $UBD \leftarrow F^*_{\text{OED-MLP}} + \mu^*$;
28 **if** $UBD - LBD < \varepsilon^{\text{a}}$ **then**
29 $\quad$ terminate;
30 Go to line 2;

Algorithm 5.2: Simplified pseudocode for solving subproblem (SP), i.e., (OED-MLP) or (OED-AUX). All subproblems are solved globally. User inputs are the solution point of (OED-ULP) $T^{\text{OED-ULP}}$, $\mu^{\text{OED-ULP}}$, the initial discretization sets $\mathcal{D}^{\text{C-TP},0}$, $\mathcal{D}^{\text{C-cl},0}$, $\mathcal{D}^{\text{C-cr},0}$, $\mathcal{D}^{\text{C-tpl},0}$, $\mathcal{D}^{\text{C-sec},0}$, parameters α_{init}, α_{red}, and tolerances $\varepsilon^{\text{a,G}}$, $\varepsilon^{\text{a,C-cl}}$, $\varepsilon^{\text{a,C-cr}}$, $\varepsilon^{\text{a,C-sec}}$, $\varepsilon^{\text{a,C-tp}}$. Note that $c \in \mathcal{C} = \{\text{C-cl}, \text{C-cr}, \text{C-sec}\}$.)

1 set $\mathcal{D}^{\text{C-TP}} \leftarrow \mathcal{D}^{\text{C-TP},0}$, $\mathcal{D}^{\text{C-cl}} \leftarrow \mathcal{D}^{\text{C-cl},0}$, $\mathcal{D}^{\text{C-cr}} \leftarrow \mathcal{D}^{\text{C-cr},0}$, $\mathcal{D}^{\text{C-tpl}} \leftarrow \mathcal{D}^{\text{C-tpl},0}$, $\mathcal{D}^{\text{C-sec}} \leftarrow \mathcal{D}^{\text{C-sec},0}$. solve (SP) for fixed $T^{\text{OED-ULP}}$ and $\mu^{\text{OED-ULP}}$ to obtain f^{SP}, $\boldsymbol{p}_{j,i}^{\text{SP}}$, $\boldsymbol{x}_{j,i}^{\text{SP}}$, and $\boldsymbol{x}_{c,j,i}^{d,\text{SIP}}$;
2 **for** *all* $j, i \in \mathcal{J}, \mathcal{I}$ **do**
    `// check stability of` $\boldsymbol{x}_{j,i}^{\text{SP}}$
3 solve (P-TP) for $T^{\text{OED-ULP}}$, $\boldsymbol{p}_{j,i}^{\text{SP}}$, $\boldsymbol{x}_{j,i}^{\text{SP}}$ to obtain $f^{\text{P-TP}}$ and $\boldsymbol{y}^*$;
4 **if** $f^{\text{P-TP}} > \varepsilon^{\text{a,G}}$ **then**
5 $\mathcal{D}^{\text{C-TP}} \leftarrow \mathcal{D}^{\text{C-TP}} \bigcup \{\boldsymbol{y}^*\}$;
6 go to line 1;
7 **for** *all* $c, j, i, d(c) \in \mathcal{J}, \mathcal{I}, \mathcal{C}, \mathcal{D}^c$ **do**
    `// check stability of` $x_{c,j,i}^{d,\text{SP}}$
8 solve (P-TP) for fixed T_c^d, $\boldsymbol{p}_{j,i}^{\text{SP}}$, $\boldsymbol{x}_{c,j,i}^{d,\text{SIP}}$ to obtain $f^{\text{P-TP}}$, and $\boldsymbol{y}^*$;
9 **if** $f^{\text{P-TP}} > \varepsilon^{\text{a,G}}$ **then**
10 $\mathcal{D}^{\text{C-TP}} \leftarrow \mathcal{D}^{\text{C-TP}} \bigcup \{\boldsymbol{y}^*\}$;
11 go to line 1;
    `// check condition` c
12 solve (P-c) using *B&F* (Algorithm C.1) for fixed $\boldsymbol{p}_{j,i}^{\text{SP}}$ to obtain $f^{\text{P-c}}$ and y^*;
13 **if** $f^{P\text{-}c} > \varepsilon^{\text{a,c}}$ **then**
14 $\mathcal{D}^c \leftarrow \mathcal{D}^c \bigcup \{y^*\}$;
15 go to line 1;
16 solve (P-tpl) for fixed $T^{\text{OED-ULP}}$, $\boldsymbol{p}_{j,i}^{\text{SP}}$ to obtain $f^{\text{P-tpl}}$, and y^*;
17 **if** $f^{\text{P-tpl}} > \varepsilon^{\text{a,C-tp}}$ **then**
18 $\mathcal{D}^{\text{C-tpl}} \leftarrow \mathcal{D}^{\text{C-tpl}} \bigcup \{y^*\}$;
19 go to line 1;
20 solve (P-tpr) for fixed $T^{\text{OED-ULP}}$, $\boldsymbol{p}_{j,i}^{\text{SP}}$ to obtain $f^{\text{P-tpr}}$, and y^*;
21 **if** $f^{\text{P-tpr}} > \varepsilon^{\text{a,C-tp}}$ **then**
22 $\mathcal{D}^{\text{C-tpr}} \leftarrow \mathcal{D}^{\text{C-tpr}} \bigcup \{y^*\}$;
23 go to line 1;
24 return f^{SP} and $\boldsymbol{p}_{j,i}^{\text{SP}}$;

subproblems are solved using MAiNGO version 0.8.1 (Bongartz et al., 2018), configured to utilize CPLEX version 22.1.1 (International Business Machines Corporation, 2022) to solve LPs or QPs. The specific tolerances and settings used can be found in Appendix D.3.

5.5 Case Study

As an illustrative case study, we use a 5^{th}-order Redlich-Kister model as the underlying $\bar{G}^{\text{E}}$ model. The presented case study serves as a proof-of-concept, demonstrating the feasibility of our method while also highlighting the computational cost of the solution. Table 5.1 provides an overview of the used values. Pragmatically, we only consider parameter p_4 to be uncertain. Therefore, only p_4 is the nominal parameter with $\hat{p}_4 = p_4$. The respective initial bounds $p_{4,\text{lb}}$ and $p_{4,\text{ub}}$ are given in Tables 5.1a and 5.1b. Recall that nominal parameters are known *a priori* and are assumed to be consistent with the measurements and model. All other Redlich-Kister parameters are considered to have no uncertainty, are fixed, and are equal to the values in Table 5.1. We selected parameter p_4 along with its uncertainty range for demonstrational purposes to show that our approach effectively works to exclude *problematic* parameter values not fulfilling the conditions reviewed in Section 5.2.2. Figure 5.3a exemplary shows $\bar{G}^{\text{M}}$ for different parameter values of p_4 and temperatures. For $p_4 = -12\,\text{kJ/mol}$, $\bar{G}^{\text{M}}$ is nonconvex on the left side of the phase split, violating (C-c-GSIP) (and additionally (C-tpl-SIP)). Therefore, these parameter values are *problematic* and not considered feasible in our setting.

Table 5.1: Values used in the OED for GPE case study.

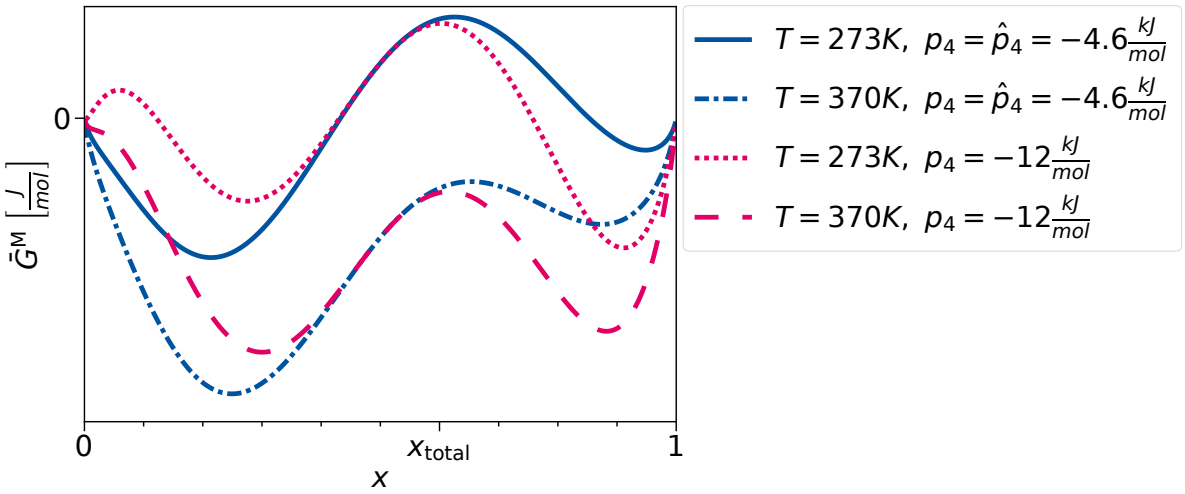
(a) Redlich-Kister parameters used in the case study. Note that we only consider p_4 to be uncertain, i.e., only p_4 is a nominal parameter with $\hat{p}_4 = p_4$. All other parameters are considered to have no uncertainty.

Parameter	Value
p_1	$7\,\text{kJ/mol}$
p_2	$4.5\,\text{kJ/mol}$
p_3	$-4.5\,\text{kJ/mol}$
p_4	$-4.6\,\text{kJ/mol}$
p_5	$5\,\text{kJ/mol}$

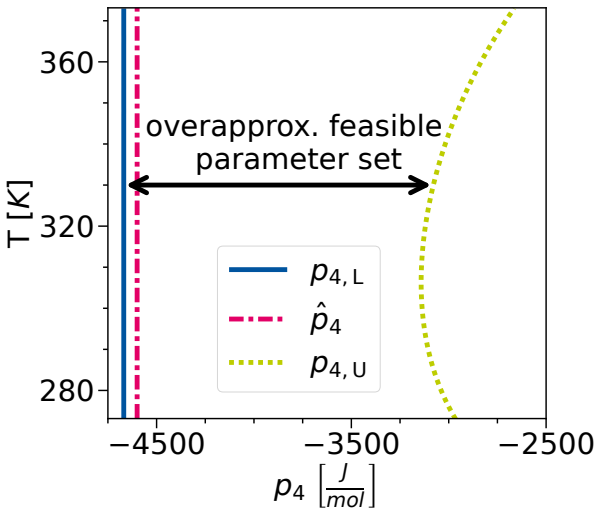
(b) Total composition, measurement bounds, temperature range, and initial parameter bounds used to define $\tilde{\mathcal{P}}$ in (OED).

Parameter	Value
x_{total}	0.6
η_{L}	-0.01
η_{U}	0.01
$\mathcal{T}$	$[273.15\,\text{K},\ 373.15\,\text{K}]$
$p_{4,\text{lb}}$	$-12\,\text{kJ/mol}$
$p_{4,\text{ub}}$	$-2\,\text{kJ/mol}$
$\tilde{\mathcal{P}}$	$[p_{4,\text{lb}}, p_{4,\text{ub}}]$

The estimated parameter uncertainty of p_4 for an additional measurement at the respective temperature is depicted in Figure 5.3b. For $p_4 < -4.67\,\text{kJ/mol}$, at least one of the conditions reviewed in Section 5.2 is violated for at least one temperature within $\mathcal{T}$ and, therefore, these values are not considered feasible. Because the conditions reviewed in Section 5.2 (and not the measurement uncertainty, i.e., $\mathbf{2}\eta_{\text{L}} \leq \hat{\boldsymbol{x}} - \boldsymbol{x} \leq \mathbf{2}\eta_{\text{U}}$), is restrictive for $p_{4,\text{lb}}$, the predicted $p_{4,\text{lb}}$ does not depend on the temperature. On the other hand, the predicted $p_{4,\text{ub}}$ depends on the temperature because for larger values of p_4 the measurement uncertainty is restrictive. Using the proposed method of Section 5.3.1, the optimal temperature for the next experiment is reported to be at 306 K with an estimated overapproximated feasible parameter set of $p_4 \in [-4.67\,\text{kJ/mol}, -3.14\,\text{kJ/mol}]$. This is in agreement with the brute-force prediction depicted in Figure 5.3b. Considering the initial bounds on p_4, c.f., $\tilde{\mathcal{P}}$ in Table 5.1b, the parameter uncertainty is reduced by 85 %. If the feasible parameter set is assumed to be already narrowed by the conditions reviewed in Section 5.2, the improvement is still 10 %.



(a) $\bar{G}^{\mathrm{M}}$ for different parameter values of p_4 and temperatures.



(b) Estimated parameter uncertainty of p_4 for an additional measurement at the respective temperature. Data generated by solving (OED-MLP) for various temperature values within $\mathcal{T}$, in increments of one unit.

Figure 5.3: Influence of p_4 on the curvature of $\bar{G}^{\mathrm{M}}$ and estimated feasible parameter set. All other parameter values are set to the values given in, c.f., Table 5.1a. Note: Reprinted from Jungen et al. (2026), CC BY 4.0, © The Author(s) 2026. Published by Elsevier Ltd.

Note that for the considered case study, condition (C-cr) was never observed to be violated. However, it is still enforced for consistency reasons. We utilize one full cluster-node (96 cores) and terminate after 8 iterations (in Algorithm 5.1) and 11 min (wall-clock time). The overall computation is thus comparable to an LLE experiment, which often takes only 20 min to reach equilibrium (Pfennig, 2004). Recall, however, that the case study is only a proof-of-concept due to the simple thermodynamic model and the single uncertain parameter.

We also test whether we can extend the case study by considering more uncertain parameters. Additional to p_4 also p_5 is imposed with uncertainty, i.e, $p_{5,\mathrm{lb}} = 2.5\,\mathrm{kJ/mol}$, $p_{5,\mathrm{ub}} = 7.5\,\mathrm{kJ/mol}$, $\tilde{\mathcal{P}} = [p_{4,\mathrm{lb}}, p_{4,\mathrm{ub}}] \times [p_{5,\mathrm{lb}}, p_{5,\mathrm{ub}}]$. To obtain meaningful results, two experiments are planned. Recall that the implemented algorithms support this, c.f., Section 5.4. The OED problem could not be solved by utilizing two full cluster-nodes (192 cores) and a maximum wall-clock time of 12 h. In the first iteration of Algorithm 5.1, the subproblems (OED-ULP) and (P-TP) are all solved in less than 30 s. Then, Algorithm 5.2 is called, and in the first main iteration, all subproblems and the solver *B&F* used for checking and computing the discretization for (C-cl), (C-cr), and (C-sec), terminate in less than 5 min. After discretization, (OED-MLP) is untrackable for MAiNGO. After more than 11 h, the absolute gap is still 3.2×10^3. The upper and lower bounds reported by MAiNGO remain unchanged within this time, indicating that the optimal solution is most likely found during preprocessing. This motivates ad-hoc approaches, e.g., Walz et al. (2018), who assume that the global optimum is found, even though the lower-bounding procedure did not confirm it.

5.6 Conclusion and Outlook

We applied OED for GPE to an LLE. To be able to incorporate LLE computations into the OED for GPE formulation, we extended the existing formulation and solution procedure (Gottu Mukkula and Paulen, 2016a,b, 2017a, Walz et al., 2018) to systems whose input-output relation is implicitly defined through an equation system and additional semi-infinite constraints. We computed the LLE using a rigorous method, employing Baker's criterion (Baker et al., 1982), and enforced additional requirements to avoid *problematic* parameter values that may lead to erroneous predictions. In our numerical case study, we presented a proof-of-concept using the Redlich-Kister model to plan an OED that significantly reduced the predicted parameter uncertainty. Our proposed solution procedure is computationally demanding compared to the duration of LLE experiments. Therefore, the computations should be highly parallelized to apply our method to real-life scenarios, where more uncertain parameters are considered.

The overarching challenge of the OED for GPE approaches is that it comes with high computational costs. This is especially the case for our application because the embedded LLE computations and additional requirements to avoid *problematic* parameter values further increase the computational burden.

In future work, first, the computational burden should be reduced. Reducing the computational burden may entail simple reformulations, e.g., using a big-M formulation for reformulating the encountered conjunctions in the subproblems, as done by Walz et al. (2018). Moreover, at the level of subsolvers, custom relaxations of functions have a significant impact on the computation time (Floudas and Gounaris, 2009, Tawarmalani and

Sahinidis, 2004). In particular for NRTL, Najman et al. (2019) showed that custom relaxations can significantly reduce the computation time of global optimization in chemical process design. However, therein, Najman et al. (2019) assume constant parameter values, implying that generalization would be necessary for our OED application.

At the level of solution of the GSIPs and SIPs, there surely is hidden potential for speeding up the computations. For example, we demonstrated in Chapter 4 that a slight adaptation of an existing adaptive discretization-based algorithm resulted in significant speed-ups. Djelassi (2020), Seidel (2020), Zingler and Mitsos (2025) investigate how the sensitivity information of the lower-level objective function might lead to fewer iterations necessary to converge. Recall that the optimal solution of the lower level is used for populating the discretization. These discretization points, in essence, cut off infeasible points, thereby iteratively improving the approximation. With each discretization point, a larger set of points can be excluded by using additional sensitivity information of the lower-level objective function. Djelassi (2020), Seidel (2020), Zingler and Mitsos (2025) show that this can reduce the number of iterations necessary to converge. However, incorporating this sensitivity information comes with the additional cost that the subproblems might become harder to solve, hinting that his approach may not be ideal for our application, especially since the subproblems are already computationally very expensive.

If rigor is not required, simple ad-hoc solution approaches, such as not guaranteeing global optimality within the subproblems, are also possible. This is done by, e.g., Walz et al. (2018), and justified by the UBD reported by the subsolver during solution. A non-changing UBD is an indicator that the optimal solution has already been found. However, the global solution of the subproblems is required for the employed solution algorithm. Hence, this ad-hoc solution approach should be treated with caution. Tailoring the approach to the system at hand also holds potential to reduce the computational burden. Olaya et al. (2008) demonstrated that for a ternary liquid mixture with closed miscibility gaps, additional empirical constraints on the parameter values can be derived. In our application, this would result in a smaller search space, potentially speeding up computations.

These first steps to reduce computational burden hold considerable potential, as we found that extending to more uncertain parameters already becomes computationally infeasible, i.e., the employed subsolver does not solve the respective subproblems in a reasonable time. However, the recent developments and improvements of (MI)NLP solvers give hope that even with (possibly) not-so-optimal computationally efficient algorithms for hierarchical programs, applications to real-life problems will become more and more feasible. This positive outlook is at least supported by the increasing number of publications and real-life applications in which hierarchical programs have been solved over the last decade.

Once numerical challenges are overcome, extensions to more phases should be investigated. Recall that we pragmatically considered a binary two-phase LLE with non-vanishing phases. In real-life applications, binary systems at a given temperature and pressure exhibit predominantly at most one LLE split. Therefore, our restriction to only two phases seems warranted. However, the question remains of how to handle LLEs where phases disappear, and/or the total number of phases is not known *a priori*. Mitsos and Barton (2007) present an approach where the number of phases must not be known *a priori* to compute the LLE. Adapting the approach of Mitsos and Barton (2007) appears promising for overcoming this challenge. Furthermore, the overall composition should be considered a experimental design variable.

Other future work might entail tighter approximations of the feasible parameter set, as the orthotropic approximation is very pessimistic (Pronzato and Walter, 1990). Walz et al. (2018) note that for one of their case studies, the computed design leads to the smallest over-approximated feasible parameter set; however, this does not correspond to the actual smallest possible feasible parameter set.

Furthermore, tailoring the OED for GPE to the intended application remains the ultimate goal, as done by Walz et al. (2020). A comparison to statistical approaches, e.g., Fleitmann et al. (2021, 2022), remains to be seen. However, one must note that such a comparison is difficult, as the *best* OED approach is likely strongly case-dependent.

Recall that the employed approach for the solution of our OED problem relies on the assumption of compact sets. As do the procedures that we base our approach on. In Section 5.3, we noted that we pragmatically choose the set $\tilde{\mathcal{P}}$, which corresponds to the set of all "considered parameter" values, as a compact set that is sufficiently large. However, in many applications, it is unclear what constitutes a sufficiently large set. This raises the question whether, the assumption of compact sets can be relaxed to, e.g., allow unbounded variables. In the following chapter, we address this question and consider SIPs with unbounded variables.

6 Adaptive Discretization-Based Algorithms for Semi-Infinite Programs with Unbounded Variables

The proof of convergence of the respective algorithms for the global solution of the problems encountered in the previous chapters usually relies, among other assumptions, on compact host sets. This is true for the encountered discretization-based approaches in Chapters 3 to 5, and in general also for all other global and local methods. Typically, the optimization variables have a technical or physical meaning and thus, in most cases, inherit finite bounds from their physical or technical origin. Because these finite bounds are usually attainable, we obtain closed and therefore compact host sets.

However, in certain cases, the host sets at hand may not be comapact. This might be the case if one does not know the finite bounds of the variables or the hierarchical programs stem from specific applications or reformulations that exhibit or introduce unbounded variables, and hence unbounded host sets. We have already encountered these cases: the bounds on the parameter values for GPE are usually not known *a priori*, c.f., Section 5.3, and unbounded variables are introduced through reformulations in, e.g., (MINMAX-REF) in Section 3.2 or (RES) in Section 3.3.1. In these cases, we have pragmatically chosen arbitrary, but sufficiently large, bounds. Note that in some cases, finite bounds of the variables might be computed or generated by additional assumptions. This might also the case for the following examples of SIPs where unbounded host sets can occur. As mentioned, in most of the following examples, arbitrary (hopefully sufficiently) large bounds are usually used in practice. An unbounded lower-level set might occur, e.g., in approximation theory, where one wants to approximate a function with a minimal estimation error, not only over a compact set, but over all $\mathbb{R}^n$ (Chebyshev approximation). An unbounded upper-level variable host set occurs, e.g., in design centering, in epigraph reformulations of min-max programs (recall that (MINMAX-REF) in Section 3.2 belongs to this category) or in approximation theory, e.g., classical Chebyshev problem, where the parameter values are unbounded, or reverse Chebyshev approximation, where the approximation error is fixed and the region where the approximation is not worse than the fixed approximation error is computed (Guerra-Vázquez et al., 2008, Still, 1999).

In order to address these and other applications with unbounded host sets, we will investigate whether adaptive discretization-based algorithms are directly applicable to these cases, i.e., whether the assumption of compact host sets can be relaxed. By relaxing the assumption of compact host sets, we will also consider problems with bounded but non-compact host sets, e.g., host sets consisting of half-open intervals or open intervals.

Note that another popular assumption within literature is the one of a compact lower-level set (Reemtsen and Rückmann, 1998). This assumption is implied by the usually used assumptions: the SIP is feasible, the host sets are compact, and the objective and constraint

functions are continuous. The advantage of the assumption of compact lower-level sets is that it already allows special cases of SIPs with unbounded upper-level host sets. However, unbounded lower-level sets are not covered.

In this chapter, we pragmatically restrict ourselves to the lower-bounding procedure and assumptions of Mitsos (2011) for SIPs. Recall that the full procedure of Mitsos (2011) is implemented in libDIPS as *RRHS* and that the lower-bounding procedure thereof is equivalent to the approach Blankenship and Falk (1976), which corresponds to the solver *B&F*. Notably, the assumptions Mitsos (2011) uses to prove convergence are already slightly relaxed than those used by Blankenship and Falk (1976). Further recall that many adaptive discretization-based algorithms for (G)SIPs, MINMAX, and BLPs have an identical predecessor, i.e., Blankenship and Falk (1976), which itself is based on Remez (1962), c.f., Section 3.3. Therefore, and also based on our additional findings, c.f., Appendix E.2, we expect that the results within this chapter, i.e., convergence guarantees with sharper, slightly relaxed, assumptions and recovered convergence guarantees in case of unbounded variables, directly carry over from one algorithm to the other.

This chapter is organized as follows. In Section 6.1, we first introduce the basic notation and simplifications used throughout this chapter and review the assumptions in Mitsos (2011). Second, we prove the convergence of the lower-bounding procedure. In the proof, we use sharper and slightly relaxed assumptions compared to Mitsos (2011), which in turn are already relaxed compared to Blankenship and Falk (1976). We show that our relaxed assumptions are implied by the original assumptions of Mitsos (2011). Section 6.2 shows that the lower-bounding procedure may exhibit convergence problems if the host sets are not compact. In Section 6.3, we give additional assumptions to apply the lower-bounding procedure to SIPs with non-compact and unbounded host sets. In Section 6.4, we present two case studies as a proof-of-concept for our findings. Finally, we give a conclusion and outlook in Section 6.5.

6.1 Preliminaries

In this section, we briefly review the notation, (adapted) formulations, definitions, and assumptions of the lower-bounding procedure in Mitsos (2011). For a detailed description of the lower-bounding procedure, refer to Section 3.3.1, which is unchanged for the case of non-compact host sets. Compared to (SIP) in Section 3.2, we restrict ourselves to a scalar constraint function g, do not consider integer variables and equality constraints, *but* allow for the occurring sets to be non-compact. These restrictions are made for notational convenience, and an extension to the form of (SIP) in Section 3.2 should be straightforward. Note that the following presented subproblem formulations are slightly adapted to the case of non-compact host sets: The min (max) operator is *replaced* with inf (sup), because in non-compact sets, the minimum (maximum) may not be attainable. The respective LBP and LLP of the SIP with non-compact host sets are, apart from the above-mentioned notational changes, equivalent to (LBP) and (LLP) in Section 3.2. Furthermore, note that for notational convenience, we write the semi-infinite constraint using the universal quantifier notation, which we will use throughout this chapter.

6.1.1 Notation, Formulation, and Subproblems

In this chapter, we consider SIPs of the (simplified) form

$$\begin{aligned} f^* = \inf_{\boldsymbol{x}\in\mathcal{X}} & f(\boldsymbol{x}) \\ \text{s.t. } & \forall \boldsymbol{y} \in \mathcal{Y}[g(\boldsymbol{x},\boldsymbol{y}) \leq 0], \end{aligned} \tag{SIP$^{\text{unb}}$}$$

with the sets $\mathcal{X} \subseteq \mathbb{R}^{n_\text{x}}$ and $\mathcal{Y} \subseteq \mathbb{R}^{n_\text{y}}$ with $|\mathcal{Y}| = \infty$, the constraint function $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, and the objective function $f : \mathcal{X} \to \mathbb{R}$ with f^* being the optimal objective value. As we consider possibly non-compact host sets, we redefine the LBP of (SIP$^{\text{unb}}$) as

$$\begin{aligned} f^k = \inf_{\boldsymbol{x}\in\mathcal{X}} & f(\boldsymbol{x}) \\ \text{s.t. } & \forall \boldsymbol{y} \in \mathcal{Y}^k[g(\boldsymbol{x},\boldsymbol{y}) \leq 0], \end{aligned} \tag{LBP$^{\text{unb}}$}$$

with $\mathcal{Y}^k \subsetneq \mathcal{Y}$ and k denoting the iteration index. Just like in the case of compact host sets, due to $\mathcal{Y}^k \subsetneq \mathcal{Y}$, (LBP$^{\text{unb}}$) is a relaxation of (SIP$^{\text{unb}}$) and it holds $f^k \leq f^*$. For the optimal solutions of (LBP$^{\text{unb}}$), we use the notation

Notation 1 (*Optimal Solution of* (LBP$^{\text{unb}}$)) Assuming the optimal solution of (LBP$^{\text{unb}}$) exists in all iterations, we denote the optimal solution point by $\boldsymbol{x}^k$ and the sequence of optimal solutions by $\{\boldsymbol{x}^k\}_{k=0}^m = \{\boldsymbol{x}^0, \boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^m\}$. To simplify the notation, we omit indexing numbers wherever possible, i.e., $\{\boldsymbol{x}^k\}$.

If (LBP$^{\text{unb}}$) is determined unbounded, then by Assumption 8 (to follow), also (SIP$^{\text{unb}}$) is unbounded. If (LBP$^{\text{unb}}$) is determined infeasible, also (SIP$^{\text{unb}}$) is infeasible. Similarly, we redefine the LLP of (SIP$^{\text{unb}}$) for fixed upper-level variables $\boldsymbol{x}^k$ as

$$g^*(\boldsymbol{x}^k) = \sup_{\boldsymbol{y}\in\mathcal{Y}} g(\boldsymbol{x}^k, \boldsymbol{y}). \tag{LLP$^{\text{unb}}$}$$

Likewise to the case of compact host sets (Definition 4), the iterate $\boldsymbol{x}^k$ is SIP-feasible if $\sup_{\boldsymbol{y}\in\mathcal{Y}} g(\boldsymbol{x}^k, \boldsymbol{y}) \leq 0$, c.f., Definition 4 in Appendix B.1. Note that according to the later used Assumption 7, (LLP$^{\text{unb}}$) may only be solved approximately. For the optimal and approximate solution of (LLP$^{\text{unb}}$), we use the notation

Notation 2 (*Optimal Solution of* (LLP$^{\text{unb}}$)) Assuming the optimal solution of (LLP$^{\text{unb}}$) exists in all iterations, we denote the optimal solution point by $\boldsymbol{y}^{*,k}$ and the sequence of optimal solutions by $\{\boldsymbol{y}^{*,k}\}_{k=0}^m = \{\boldsymbol{y}^{*,0}, \boldsymbol{y}^{*,1}, \boldsymbol{y}^{*,2}, ..., \boldsymbol{y}^{*,m}\}$. To simplify the notation, we omit indexing numbers wherever possible, i.e., $\{\boldsymbol{y}^{*,k}\}$.

Notation 3 (*Approx. Solution of* (LLP$^{\text{unb}}$)) Assuming the approximate solution of (LLP$^{\text{unb}}$) exists in all iterations, we denote the approximate solution point by $\boldsymbol{y}^k$ and the sequence of approximate solutions by $\{\boldsymbol{y}^k\}_{k=0}^m = \{\boldsymbol{y}^0, \boldsymbol{y}^1, \boldsymbol{y}^2, ..., \boldsymbol{y}^m\}$. To simplify the notation, we omit indexing numbers wherever possible, i.e., $\{\boldsymbol{y}^k\}$.

To ease notation, we also use Notation 1 to 3 for infinite sequences. Note that this is abuse of notation as in these cases no maximum iteration index m exists.

The following is a summary of additional notation and the definition of compact sets used in this chapter.

Notation 4 (*Feasible Set*) We use for the set of all feasible points in the host set of (SIP^{unb}) the notation

$$\mathcal{X}^{\text{feas}} := \{\boldsymbol{x} \in \mathcal{X} : \forall \boldsymbol{y} \in \mathcal{Y}[g(\boldsymbol{x}, \boldsymbol{y}) \leq 0]\}. \tag{6.1}$$

Notation 5 (*Infeasible Set*) We use for the set of all infeasible points in the host set of (SIP^{unb}) the notation

$$\mathcal{X}^{\text{infeas}} := \{\boldsymbol{x} \in \mathcal{X} : \exists \boldsymbol{y} \in \mathcal{Y}[g(\boldsymbol{x}, \boldsymbol{y}) > 0]\}. \tag{6.2}$$

Definition 1 (*Compact Set*) $\mathcal{K} \subsetneq \mathbb{R}^n$ is compact if any open cover of $\mathcal{K}$ has a finite subcover. More explicitly, $\mathcal{K}$ is compact if whenever $\mathcal{K} \subseteq \bigcup_{\alpha \in \mathcal{I}} \mathcal{A}_\alpha$, where each $\mathcal{A}_\alpha$ is open, there exists a finite number of indices $\alpha_1, \alpha_2, ..., \alpha_m \in \mathcal{I}$ such that $\mathcal{K} \subseteq \bigcup_{j=1}^{j=m} \mathcal{A}_{\alpha_j}$ (c.f., Jänich (2008)).

6.1.2 Assumptions

Recall that the considered lower-bounding procedure of Mitsos (2011) uses a global solver to compute the subproblems (LBP^{unb}) and (LLP^{unb}) which are (MI)(N)LPs. By considering SIPs with unbounded host sets, we obviously inherit the need for global solvers that can handle optimization problems with unbounded host sets. In the case of LPs, this does not pose a problem as, e.g., the simplex method can theoretically handle unbounded host sets (Nocedal and Wright, 1999). In the more general case of NLPs, e.g., BARON is able to treat some problems with unbounded variables systematically by trying to compute appropriate bounds from problem constraints (Khajavirad and Sahinidis, 2018) but substantial theoretical and practical challenges remain. In what follows, we focus on the SIP algorithm's convergence properties for unbounded host sets and do not discuss these challenges.

The considered lower-bounding procedure of Mitsos (2011) relies on the following assumptions (c.f., Lemma 2.2 in Mitsos (2011), revised in Lemma 2 in Harwood et al. (2021)):

Assumption 5 (*Compactness of Sets*) The sets $\mathcal{X} \subsetneq \mathbb{R}^{n_\text{x}}$ and $\mathcal{Y} \subsetneq \mathbb{R}^{n_\text{y}}$ are compact.

Assumption 6 (*Continuous Functions*) The functions f and g are continuous on $\mathcal{X}$ and $\mathcal{X} \times \mathcal{Y}$, respectively.

Assumption 7 (*Appr. Solution of* (LLP^{unb})) At each iteration k, (LLP^{unb}) is solved approximately for the solution of the lower-bounding problem $\boldsymbol{x}^k$ either establishing $\sup_{\boldsymbol{y} \in \mathcal{Y}} g\big(\boldsymbol{x}^k, \boldsymbol{y}\big) \leq 0$, or furnishing a point $\boldsymbol{y}^k$ such that $g\big(\boldsymbol{x}^k, \boldsymbol{y}^k\big) \geq \alpha g\big(\boldsymbol{x}^k, \boldsymbol{y}^{*,k}\big) > 0$. With α being constant over all iterations and $\alpha \in (0, 1]$.

Assumption 7 is relaxed compared to the assumption in Blankenship and Falk (1976), where the exact solution of (LLP^{unb}) is assumed. It is also slightly relaxed to Harwood et al. (2021) wherein α is restricted to $(0, 1)$. However, as will be shown below, the problems associated with unbounded host sets persist even if (LLP^{unb}) is solved exactly.

6.1.3 Proof of Convergence of the Lower-Bounding Procedure

Recall that finite termination of the lower-bounding procedure to a feasible point is not guaranteed and, therefore, in practice a feasibility tolerance $\varepsilon^{\mathrm{a}} > 0$ is usually introduced as a termination criterion. In this case, the algorithm may terminate with an SIP-ε^{a}-feasible point, see Definition 2 in Appendix B.1. A feasibility tolerance $\varepsilon^{\mathrm{a}} > 0$ is also used in the numerical case studies in Section 6.4. However, note that in the theoretical considerations (to follow), no feasibility tolerance is introduced, or equivalently the feasibility tolerance is set to zero, i.e., $\varepsilon^{\mathrm{a}} = 0$. Therefore, in the following theoretical analysis of the lower-bounding procedure, especially in the proof of convergence, one does not terminate *early*, but considers the full (infinite) sequence of iterates.

The presented proof relies on slightly relaxed assumptions compared to those made by Mitsos (2011) and Reemtsen and Rückmann (1998). Basically, we split the assumptions made by Mitsos (2011) and some properties, which result from them, into multiple sharpened assumptions. Additionally, we use the idea of level sets by Reemtsen and Rückmann (1998). These sharpened assumptions are often challenging to prove *a priori*. In these cases, we later present alternative, stricter, assumptions, which are easier to prove and that imply the sharpened assumptions. The sharpened assumptions motivate the additional, stricter, assumptions, which might be easier to prove, for SIPs with unbounded host sets in Section 6.3.

Assumption 5 is relaxed to

Assumption 8 It holds (a) $\mathrm{cl}(\Lambda(\mathcal{Y}^0)) \subseteq \mathcal{X}$ and (b) $\Lambda(\mathcal{Y}^0)$ is bounded with the set $\Lambda(\mathcal{Y}^0)$ defined as

$$\begin{aligned}\Lambda\big(\mathcal{Y}^0\big) = \{\boldsymbol{x} \in \mathcal{X} :\ & f(\boldsymbol{x}) \leq f^*\} \\ & \cap \big\{\boldsymbol{x} \in \mathcal{X} :\ \boldsymbol{y} \in \mathcal{Y}^0[g(\boldsymbol{x}, \boldsymbol{y}) \leq 0]\big\} \\ & \cap \mathcal{X}^{\mathrm{infeas}}\end{aligned} \tag{6.3}$$

whereas f^* is the optimal objective value of ($\mathrm{SIP}^{\mathrm{unb}}$).

Remark 2 Assumption 8 builds on the consideration that the algorithm must only exclude points which

- are super-optimal, i.e., points which belong to the lower level-set $\{\boldsymbol{x} \in \mathcal{X} :\ f(\boldsymbol{x}) \leq f^*\}$,
- are feasible within the initially chosen discretization $\mathcal{Y}^0$, and
- belong to $\mathcal{X}^{\mathrm{infeas}}$.

If the iterate is not an element of $\Lambda(\mathcal{Y}^0)$, the iterate is feasible and optimal by assumption. Note that if ($\mathrm{SIP}^{\mathrm{unb}}$) is infeasible we have by convention $f^* = +\infty$ and therefore $\Lambda(\mathcal{Y}^0) = \{\boldsymbol{x} \in \mathcal{X} :\ \boldsymbol{y} \in \mathcal{Y}^0[g(\boldsymbol{x}, \boldsymbol{y}) \leq 0]\} \cap \mathcal{X}^{\mathrm{infeas}}$.

Assumption 8 allows for unbounded host sets and also for bounded but not closed sets. In Section 6.3 we give examples of stronger assumptions or checks which can be performed since the assumption is difficult to verify *a priori*. The property of uniform continuity of f and g on $\mathcal{X}$ and $\mathcal{X} \times \mathcal{Y}$, respectively, follow from Assumptions 5 and 6. In the following we relax these two assumptions.

Assumption 9 The function f is lower semi-continuous at all $\boldsymbol{x} \in \partial\Lambda(\mathcal{Y}^0)$.

Assumption 10 It holds

$$\begin{aligned}\forall \varepsilon > 0\ \exists \delta_1 > 0 :\ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \Lambda(\mathcal{Y}^0), \boldsymbol{x} \in \Lambda(\mathcal{Y}^0)\, [\|\boldsymbol{x}^k - \boldsymbol{x}\| < \delta_1] \\ \Rightarrow |g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k)| < \varepsilon.\end{aligned} \tag{6.4}$$

We first prove that the proposed assumptions are indeed relaxed compared to the original assumptions, i.e., the latter imply the former.

Lemma 1 *Assumptions 8 to 10 hold if Assumptions 5 and 6 are satisfied.*

Proof. First, we show that Assumption 5 implies Assumption 8. $\mathcal{X}$ is compact according to Assumption 5. Since $\Lambda(\mathcal{Y}^0) \subseteq \mathcal{X}$, we directly get $\mathrm{cl}(\Lambda(\mathcal{Y}^0)) \subseteq \mathcal{X}$. $\mathcal{X}$ is bounded by compactness and, therefore, $\Lambda(\mathcal{Y}^0)$ is also bounded.
Second, according to Assumption 6, f is continuous on $\mathcal{X}$ and hence lower semi-continuous on $\partial\Lambda(\mathcal{Y}^0) \subseteq \mathcal{X}$, or Assumption 9 holds.
Third, from Assumptions 5 and 6 follows uniform continuity of g on $\mathcal{X} \times \mathcal{Y}$ and we have

$$\forall \varepsilon > 0\ \exists \delta_1 > 0 :\ \forall \boldsymbol{x}^k, \boldsymbol{x} \in \mathcal{X}\, [\|\boldsymbol{x}^k - \boldsymbol{x}\| < \delta_1] \Rightarrow |g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k)| < \varepsilon. \tag{6.5}$$

Then, it also holds

$$\begin{aligned}\forall \varepsilon > 0\ \exists \delta_1 > 0 :\ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \Lambda(\mathcal{Y}^0), \boldsymbol{x} \in \Lambda(\mathcal{Y}^0)\, [\|\boldsymbol{x}^k - \boldsymbol{x}\| < \delta_1] \\ \Rightarrow |g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k)| < \varepsilon\end{aligned} \tag{6.6}$$

because $\{\boldsymbol{x}^k\} \cap \Lambda(\mathcal{Y}^0) \subseteq \mathcal{X}$ and $\Lambda(\mathcal{Y}^0) \subseteq \mathcal{X}$. □

Next, we prove convergence of the lower-bounding procedure using Assumption 7 and the relaxed Assumptions 8 to 10. Recall, that no feasibility tolerance ε^{a} is introduced and, hence, the full (infinite) sequence of iterates is considered.

Theorem 1 *If Assumptions 7 to 10 are satisfied, the adaptive discretization-based lower-bounding procedure in Mitsos (2011) terminates finitely with the optimal objective value or converges to the optimal objective value, i.e., $f^k \to f^*$ for $k \to \infty$. If* (SIP$^{\mathrm{unb}}$) *is infeasible or unbounded, the lower-bounding procedure terminates finitely with proof of infeasibility or unboundedness, respectively.*

Note that Reemtsen and Rückmann (1998) and Mitsos (2011) exclude infeasible SIPs by assumption, c.f., Assumption 16 in Appendix E.2.

Proof. We first show that we move away from any *compact set of infeasible points* within finitely many iterations. Second, we consider the case of an *infeasible* SIP. We show that the algorithm terminates finitely with proof of infeasibility. Third, we consider the case of a *feasible* SIP and show that the algorithm terminates finitely with a globally optimal solution, *or* the algorithm produces the optimal solution in the limit.

1. Consider a *compact set of infeasible points* $\mathcal{K}^{\text{infeas}} \subseteq \Lambda(\mathcal{Y}^0)$. In the following, we restrict the iterations to be in this set, i.e., $\boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \mathcal{K}^{\text{infeas}}$. Recall that $g(\boldsymbol{x}^k, \boldsymbol{y}^{*,k}) \leq 0$ would imply that $\boldsymbol{x}^k$ is feasible, $\boldsymbol{x}^k$ is not a member of $\mathcal{K}^{\text{infeas}}$, and we have left $\mathcal{K}^{\text{infeas}}$. Due to Assumption 7, we have

$$\exists \mu > 0 : \ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \mathcal{K}^{\text{infeas}} \big[g(\boldsymbol{x}^k, \boldsymbol{y}^k) \geq \alpha \cdot g(\boldsymbol{x}^k, \boldsymbol{y}^{*,k}) \geq \mu > 0 \big]. \tag{6.7}$$

Since Assumption 10 holds for all $\varepsilon > 0$, it also holds

$$\begin{aligned} \exists \delta_1 > 0 : \ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \mathcal{K}^{\text{infeas}}, \boldsymbol{x} \in \mathcal{K}^{\text{infeas}} \big[\|\boldsymbol{x}^k - \boldsymbol{x}\| < \delta_1 \big] \\ \Rightarrow \big| g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k) \big| < \mu. \end{aligned} \tag{6.8}$$

We obtain for the deduction in (6.8) the two cases for $\boldsymbol{x} \in \mathcal{K}^{\text{infeas}}$

Case 1: $g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k) \geq 0$:

$$\begin{aligned} &\Rightarrow && g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k) < \mu \\ &\overset{(6.7)}{\Longrightarrow} && \mu - g(\boldsymbol{x}, \boldsymbol{y}^k) \leq g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k) < \mu \\ &\Rightarrow && 0 < g(\boldsymbol{x}, \boldsymbol{y}^k) \end{aligned} \tag{6.9}$$

Case 2: $g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k) < 0$:

$$\begin{aligned} &\Rightarrow && g(\boldsymbol{x}^k, \boldsymbol{y}^k) < g(\boldsymbol{x}, \boldsymbol{y}^k) \\ &\overset{(6.7)}{\Longrightarrow} && 0 < \mu \leq g(\boldsymbol{x}^k, \boldsymbol{y}^k) < g(\boldsymbol{x}, \boldsymbol{y}^k) \end{aligned} \tag{6.10}$$

Therefore, it holds

$$\begin{aligned} \exists \delta_1 > 0 : \ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \mathcal{K}^{\text{infeas}}, \boldsymbol{x} \in \mathcal{K}^{\text{infeas}} \big[\|\boldsymbol{x}^k - \boldsymbol{x}\| < \delta_1 \big] \\ \Rightarrow 0 < g(\boldsymbol{x}, \boldsymbol{y}^k). \end{aligned} \tag{6.11}$$

Thus, with each iteration, the open neighborhood $\mathcal{N}_{\delta_1}(\boldsymbol{x}^k) \cap \mathcal{K}^{\text{infeas}}$ is infeasible for the following iterations (and therefore we do not (re)visit points in this neighborhood). Since the excluded neighborhoods cannot be revisited and $\mathcal{K}^{\text{infeas}}$ is compact, after at most finitely many iterations, these neighborhoods form a finite cover of $\mathcal{K}^{\text{infeas}}$. Therefore, we only need a finite number of iterations until we have covered $\mathcal{K}^{\text{infeas}}$ (c.f., Definition 1), i.e., we prove infeasibility of all points in $\mathcal{K}^{\text{infeas}}$ after a finite number of iterations.

2. Consider now an *infeasible* SIP. $\Lambda(\mathcal{Y}^0)$ is compact by Assumption 8. The proof of finite termination with proof of infeasibility directly follows from the proof for case 1 above with $\mathcal{K}^{\text{infeas}} = \Lambda(\mathcal{Y}^0)$. It follows that (LBP$^{\text{unb}}$) is infeasible after a finite number of iterations.

3. Finally, consider a *feasible* SIP.
   a) If (SIP$^{\text{unb}}$) is unbounded, the problem will be determined to be unbounded in the first iteration. (LBP$^{\text{unb}}$) is a valid relaxation of (SIP$^{\text{unb}}$), i.e., $f^k \leq f^*$. As (SIP$^{\text{unb}}$) is by assumption unbounded, also (LBP$^{\text{unb}}$) is in the first iteration unbounded, i.e., $\|\boldsymbol{x}^*\| = \infty$. By Assumption 8, $\|\boldsymbol{x}^*\| \notin \mathcal{X}^{\text{infeas}}$ and the unbounded solution is feasible.

b) If the optimal solution is finite, i.e., $\|\boldsymbol{x}^*\| < \infty$, we will show by contradiction that a feasible and optimal point is generated in the limit.
By Assumption 8, there exists a compact set $\widetilde{\mathcal{X}} \supsetneq \Lambda(\mathcal{Y}^0)$ containing the optimal point. By compactness of $\widetilde{\mathcal{X}}$, we can choose an infinite subsequence $\{\boldsymbol{x}^k\}$ that converges to $\hat{\boldsymbol{x}} \in \widetilde{\mathcal{X}}$.
Now, assume the *limit point is infeasible* $\hat{\boldsymbol{x}} \in \mathcal{X}^{\text{infeas}}$. There exists a compact set $\mathcal{K}^{\text{infeas}} \ni \hat{\boldsymbol{x}}$. By proof from case 1 above, we move away from any infeasible set $\mathcal{K}^{\text{infeas}}$ within finite time. Therefore, the infeasible point $\hat{\boldsymbol{x}}$ is not a limit point that gives us the desired contradiction.
It remains to show that the *feasible limit point* is optimal. (LBP$^{\text{unb}}$) is a valid relaxation of (SIP$^{\text{unb}}$). Hence, we have $f^k \leq f^*$. The limit point $\hat{\boldsymbol{x}}$ is feasible, i.e., $f(\hat{\boldsymbol{x}}) \geq f^*$ and in the boundary of $\Lambda(\mathcal{Y}^0)$. With lower semi-continuous of f at all $\forall \boldsymbol{x} \in \partial\Lambda(\mathcal{Y}^0)$ and with $f^k \leq f^*$ follows $f(\hat{\boldsymbol{x}}) = f^*$. □

Following the proof of Lemma 1 and Theorem 1, we can also prove that the lower-bounding procedure in Mitsos (2011) converges in the infeasible case under the original Assumptions 5 to 7. The proof is also applicable with slight changes for a lower-semicontinuous constraint function g on $\mathcal{X}$ for all $\boldsymbol{y} \in \mathcal{Y}$. This property might be of interest in cases where the constraint function resembles the solution of an embedded optimization problem. The reader may refer to Appendix E.1.1 for the corresponding proposition and proof.

Recall that the upper-bounding procedure in Mitsos (2011) is conceptually similar to the lower-bounding procedure, see Section 3.3.1. The proof of convergence and the slightly changed assumptions for the case of unbounded host sets of the upper-bounding procedure are shown in Appendix E.2.3. This particular supports our claim that the results of this chapter can be directly transferred to other conceptually similar procedures.

6.2 Illustrative Examples of SIPs With Unbounded Host Sets

We first show examples where Assumptions 7 to 10 do not hold and discuss how the lower-bounding procedure might fail. All examples follow the same pattern. They contain an upper- or lower-level variable that (i) has an unbounded host set and (ii) can be chosen arbitrarily in the sense that the variable does not affect the (LBP$^{\text{unb}}$)- or (LLP$^{\text{unb}}$)-objective, respectively. Using this property, we can choose a sequence of points that generates arbitrarily weak discretization cuts, thus violating Assumption 10. This leads to a convergence to an infeasible point in the limit with no proof of infeasibility

6.2.1 Illustrative Example for $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded

Consider the SIP

$$\begin{aligned} f^* = \inf_{x \in \mathcal{X}} \; & -x \\ \text{s.t. } & \forall \boldsymbol{y} \in \mathcal{Y}\big[-y_1^2(y_2 - x)^2 + 1 \leq 0\big], \\ \text{with } & \mathcal{X} \coloneqq [0, 2] \\ & \mathcal{Y} \coloneqq \mathbb{R}^2. \end{aligned} \tag{E1}$$

Note that the corresponding LLP has infinitely many solutions. The optimal solutions of (LLP$^{\text{unb}}$) are $\boldsymbol{y}^*(\bar{x}) = (\tilde{y}, \bar{x})^T$ or $\boldsymbol{y}^*(\bar{x}) = (0, \tilde{y})^T$ with arbitrary $\tilde{y} \in \mathbb{R}$. We have $\forall x \in \mathcal{X}[g(x, \boldsymbol{y}^*) = 1 > 0]$. Therefore, (E1) is infeasible.

Now, we show that the lower-bounding procedure may converge to an infeasible point. We start with an empty discretization $\mathcal{Y}^0 = \emptyset$. The optimal solution of the (LBP$^{\text{unb}}$) is $x^0 = 2$ with an optimal objective value of $f^0 = -2$. Then, $\boldsymbol{y}^{*,0} = (2, 2)^T$ is a globally optimal solution of (LLP$^{\text{unb}}$). Using $\boldsymbol{y}^{*,k} = \left(2^{k+1}, x^k\right)^T$ for all subsequent iterations and considering the (LBP$^{\text{unb}}$) objective, the last introduced discretization point determines the optimal solution in the next iteration. See also Figure 6.1 for graphical illustration. x^{k+1} can be computed by

$$\begin{aligned} & g\left(x^{k+1}, \boldsymbol{y}^{*,k}\right) && \leq 0 \\ & - \left(y_1^{*,k}\right)^2 \left(y_2^{*,k} - x^{k+1}\right)^2 + 1 && \leq 0 \\ & - \left(2^{k+1}\right)^2 \left(x^k - x^{k+1}\right)^2 + 1 && \leq 0 \\ & \Rightarrow x^{k+1} = x^k - \frac{1}{2^{k+1}}. \end{aligned}$$

Therefore, we have for any iteration $k \geq 1$

$$x^k = x^0 - \sum_{i=1}^{k} \frac{1}{2^i}.$$

With $\sum_{i=1}^{k} \frac{1}{2^i}$ being the geometrical series, we obtain $\lim_{k\to\infty} x^0 - \sum_{i=1}^{k} \frac{1}{2^i} = 1$ with an

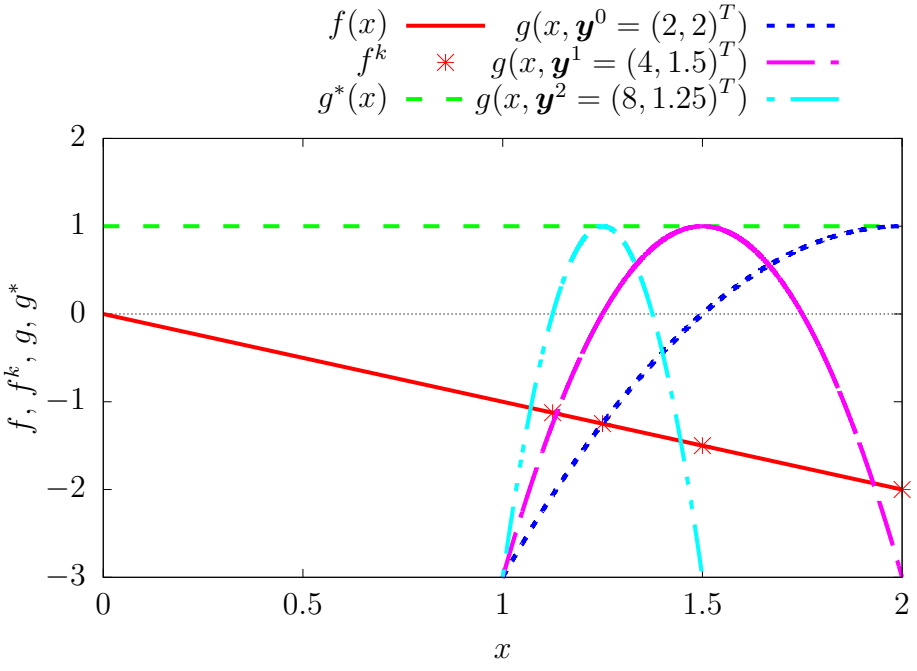


Figure 6.1: Illustrative example (E1) for $\mathcal{X}$ compact and $\mathcal{Y}$ unbounded with parametric solutions of (LLP$^{\text{unb}}$). (E1) is infeasible. Note: Adapted from Jungen et al. (2022) with minor formatting and typesetting changes, CC BY 4.0, © The Author(s) 2022.

objective value of $f^{k\to\infty} = -1$. Therefore, the lower-bounding procedure converges to an infeasible point in the limit, and no proof of infeasibility is given within finite time.

The same convergence issues arise when we consider an SIP with $\mathcal{X}$ compact and $\mathcal{Y}$ bounded but not closed, c.f., Appendix E.1.2.1. For the sake of simplicity, we considered the infeasible SIP (E1). Recall that Mitsos (2011), Reemtsen and Rückmann (1998) exclude infeasible SIPs. Therefore, (E1) would not be considered (assuming the extension to unbounded host sets). The reader may refer to the Appendix E.1.2.2 for a feasible but conceptually equivalent SIP that exhibits the same convergence issues.

We show in (E1) that in certain cases the lower-bounding procedure can converge to an infeasible point. Similarly, also, the upper-bounding procedure in Mitsos (2011) may never produce a SIP-feasible point (see Definition 4 in Appendix B.1), supporting our claim that our results carry over to other related adaptive discretization-based procedures. For a detailed example, the reader may refer to Appendix E.2.2.

6.2.2 Illustrative Example for $\mathcal{X}$ Unbounded and $\mathcal{Y}$ Compact

Consider the SIP

$$\begin{aligned} f^* = \inf_{\boldsymbol{x}\in\mathcal{X}} & \begin{cases} -1, & x_2 \geq 1 \\ -x_2, & x_2 < 1 \end{cases} \\ & \text{s.t.}\, \forall y \in \mathcal{Y}\big[-x_1(x_2-y)^2+1 \leq 0\big], \end{aligned} \tag{E3}$$

$$\begin{aligned} \text{with}\, \mathcal{X} &\coloneqq [0,+\infty) \times [0,2] \\ \mathcal{Y} &\coloneqq [0,2]. \end{aligned}$$

The optimal value function of the corresponding (LLP^{unb}) is $g(\bar{\boldsymbol{x}}, \boldsymbol{y}^*) = 1 > 0,\ \forall \bar{\boldsymbol{x}} \in \mathcal{X}$ with the optimal solution $y^*(\bar{\boldsymbol{x}}) = \bar{x}_2$. Therefore, (E3) is infeasible. Note that the objective function only depends on x_2 and is constant in the interval $x_2 \in [1,2]$. Again, we show below that the lower-bounding procedure can converge to an infeasible point in the limit. First, start with an empty discretization set $\mathcal{Y}^0 = \emptyset$. For iteration $k = 0$, choose $\boldsymbol{x}^0 = (4,2)^T$ as the optimal solution, with an objective value of $f^0 = -1$. For the corresponding (LLP^{unb}), we have $y^{*,0} = 2$. Using for iteration $k \geq 1$ and all following iterations $x_1^k = 2^{k+2}$ and considering the (LBP^{unb}) objective, we can choose $x_2^k = 2 - \sum_{i=1}^{k} \frac{1}{2^i}$, see Figure 6.2 for graphical illustration. With $\sum_{i=1}^{k} \frac{1}{2^i}$ being the geometrical series, we obtain $\lim_{k\to\infty} x_2 = \lim_{k\to\infty} x_2^0 - \sum_{i=1}^{k} \frac{1}{2^i} = 1$ with an objective value of $f^* = -1$. Therefore, the lower-bounding procedure does not prove infeasibility of (E3) within finite time.

Again, for the sake of simplicity, an infeasible SIP with a non-differentiable function f has been considered. Similar adaptations as in Appendix E.1.2.2 can be made to obtain a feasible SIP with n-times differentiable functions, which exhibit the same convergence issues. Furthermore, similar adaptions as in Appendix E.1.2.1 can be made to obtain an SIP with compact $\mathcal{Y}$ and bounded but non-compact $\mathcal{X}$, which exhibits the same convergence issues.

6.3 Examples of Assumptions That Are Easier to Prove Than Assumptions 8 to 10

As already shown in Section 6.1, the often encountered assumptions in literature for SIPs imply our slightly relaxed assumptions. However, for SIPs with unbounded host sets the

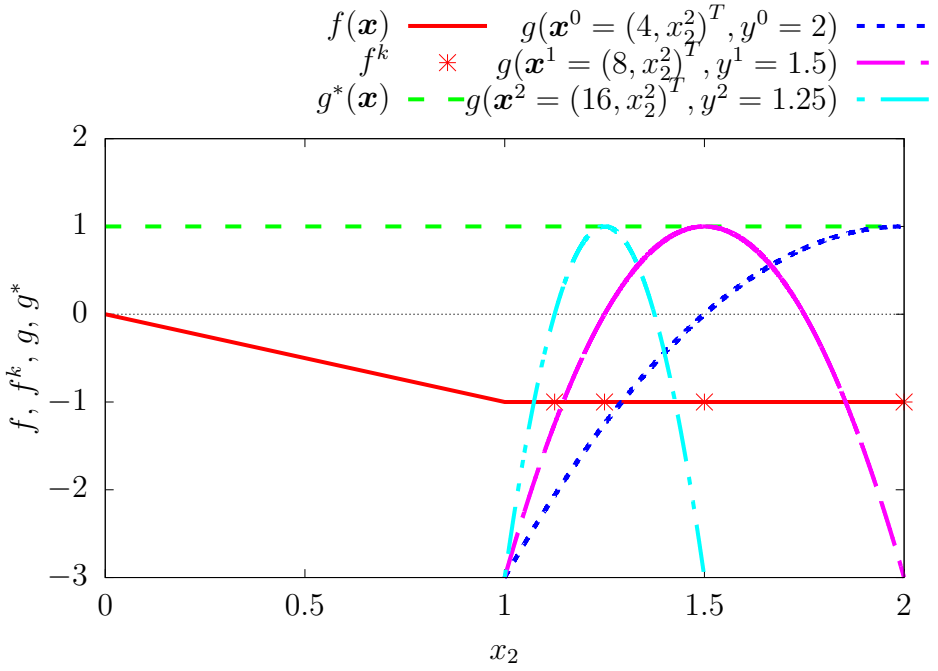


Figure 6.2: Illustrative example (E3) for $\mathcal{X}$ unbounded and $\mathcal{Y}$ compact with parametric solutions of LLP. (E3) is infeasible. Note: Adapted from Jungen et al. (2022) with minor formatting and typesetting changes, CC BY 4.0, © The Author(s) 2022.

former assumptions do not hold. In these cases, it is often not obvious whether the relaxed Assumptions 8 to 10 hold. This section, discusses additional (stricter) assumptions and criteria for different cases of SIPs with unbounded host sets which can be often checked *a priori* or during run-time to enable application.

6.3.1 Case 1: $\mathcal{X}$ Unbounded $\mathcal{Y}$ Compact

First, note that if the lower-bounding procedure produces a point outside $\Lambda(\mathcal{Y}^0)$, the algorithm terminates, see Remark 2. Hence, we restrict our discussion to the following cases:

1.1 $\Lambda(\mathcal{Y}^0)$ is unbounded

1.2 $\Lambda(\mathcal{Y}^0)$ is bounded

Feasible SIPs can belong to cases 1.1 or 1.2. Infeasible SIPs usually belong to case 1.1 if the initial discretization does not directly lead to an infeasible (LBP$^{\text{unb}}$). If this is not the case, it follows from $\mathcal{X}$ unbounded and $\mathcal{X}^{\text{feas}} = \emptyset$ that $\Lambda(\mathcal{Y}^0)$ is unbounded.

In the general case 1.1, the existence of a finite cover of the set $\Lambda(\mathcal{Y}^0)$ cannot be guaranteed. Even if we exclude an open neighborhood of points whose size tends to infinity in each iteration, we cannot guarantee convergence. A different perspective offer Reemtsen and Rückmann (1998) who note that due to unboundness the proof of convergence fails as no limit point of $\{\boldsymbol{x}^k\}$ may exist.

As a counterexample, revisit (E3) and replace g with the piecewise-defined function

$$g(x, \boldsymbol{y}) = \begin{cases} -x_2(x_1 - y)^2 + 1, & x_1 \leq y \\ 1, & x_1 > y \end{cases} \tag{6.12}$$

In this case, the open ball we exclude in each iteration is infinite. However, the same convergence issues arise when using the same sequence of (LLP$^{\text{unb}}$) solutions given in (E3).

The function g in (6.12) is continuous but not differentiable on $\mathcal{X} \times \mathcal{Y}$. Similar to Remark 3, we can obtain a function belonging to differentiability class C^n that produces the same convergence issues.

Therefore, we concentrate in the following on case 1.2, i.e., $\Lambda(\mathcal{Y}^0)$ is bounded. We will discuss stricter assumptions or checks which can often be applied to verify that case 1.2 holds irrespectively of $\mathcal{X}$ unbounded. In detail, we will discuss

1.2.1 $\mathcal{X}^{\text{feas}} \neq \emptyset$ and f is continuous and coercive on $\mathcal{X}$, i.e., $\lim_{\|\boldsymbol{x}\| \to +\infty} f(\boldsymbol{x}) = +\infty$.

1.2.2 the (SIP$^{\text{unb}}$) stems from the MINMAX (abusing notation)

$$\begin{aligned} f^* = \inf_{\boldsymbol{x} \in \mathcal{X}} \sup_{\boldsymbol{y} \in \mathcal{Y}} q(\boldsymbol{x}, \boldsymbol{y}), \\ \text{with } \mathcal{Y} \subsetneq \mathbb{R}^{n_y} \\ \mathcal{X} \subseteq \mathbb{R}^{n_x}, \end{aligned} \tag{6.13}$$

$q : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and q being coercive in $\boldsymbol{x}$ for all $\boldsymbol{y}$. In this case, we consider in the following the epigraph reformulation of this MINMAX, which reads

$$\begin{aligned} f^* = \inf_{\boldsymbol{x} \in \mathcal{X}, \mu \in \mathbb{R}} \mu \\ \text{s.t. } \forall \boldsymbol{y} \in \mathcal{Y}[q(\boldsymbol{x}, \boldsymbol{y}) \leq \mu], \\ \text{with } \mathcal{Y} \subsetneq \mathbb{R}^{n_y} \\ \mathcal{X} \subseteq \mathbb{R}^{n_x}. \end{aligned} \tag{6.14}$$

1.2.3 additional computational checks

6.3.1.1 Case 1.2.1: f is Continuous and Coercive on $\mathcal{X}$ and $\mathcal{X}^{\text{feas}} \neq \emptyset$

If f is continuous and coercive on $\mathcal{X}$ and $\mathcal{X}^{\text{feas}} \neq \emptyset$, we can prove convergence of the lower-bounding procedure.

Assumption 11 The set $\mathcal{Y} \subsetneq \mathbb{R}^{n_y}$ is compact and $\text{cl}\big(\mathcal{X}^{\text{infeas}}\big) \subseteq \mathcal{X} \subseteq \mathbb{R}^{n_x}$. f is continuous and coercive on $\mathcal{X}$ and $\mathcal{X}^{\text{feas}} \neq \emptyset$

Proposition 1 *Under Assumptions 6, 7 and 11, the lower-bounding procedure converges.*

Proof. Because f is coercive on $\mathcal{X}$ and $\mathcal{X}^{\text{feas}} \neq \emptyset$, we have $f^* < +\infty$ and the set $\{\boldsymbol{x} \in \mathcal{X} : \ f(\boldsymbol{x}) \leq f^*\}$ is bounded. Hence, also $\Lambda(\mathcal{Y}^0)$ is bounded irrespective of $\mathcal{Y}^0$. From boundness of $\Lambda(\mathcal{Y}^0)$ and Assumption 11 follows $\text{cl}(\Lambda(\mathcal{Y}^0)) \subseteq \mathcal{X}$. First, Assumption 8 is therefore satisfied. Second, from Assumption 11 follows Assumption 9. Third, from continuity of g (Assumption 6) and $\text{cl}(\Lambda(\mathcal{Y}^0)) \subseteq \mathcal{X}$ and compactness of $\mathcal{Y}$ follow uniform continuity of g and Assumption 10 is satisfied. By Theorem 1, the lower-bounding procedure converges. □

6.3.1.2 Case 1.2.2: SIP Has the Form of an Epigraph Reformulation

In the special case that (SIP$^{\text{unb}}$) is the epigraph reformulation of a MINMAX with a coercive objective function, we can show that Assumption 8 holds irrespectively of unboundness of $\mathcal{X}$.

Assumption 12 (SIP$^{\text{unb}}$) has the form of (6.14), q is continuous and coercive in $\boldsymbol{x}$ for all $\boldsymbol{y}$, and $\mathcal{Y}$ is compact.

Proposition 2 *Under Assumptions 7 and 12, the lower-bounding procedure converges.*

Proof. From continuity and coerciveness of q in $\boldsymbol{x}$ $\forall \boldsymbol{y}$ follows the optimal point exists, is finite, and attainable. It also follows, $\exists M$ such that $\forall \boldsymbol{y}, \boldsymbol{x} : \|\boldsymbol{x}\| > M \Big[q(\boldsymbol{x}, \boldsymbol{y}) \geq \sup_{\boldsymbol{y} \in \mathcal{Y}} q(\boldsymbol{x}^*, \boldsymbol{y}) = \mu^* \Big]$. Because of the epigraph form of (6.14), the construction of (LBP$^{\text{unb}}$), and the global solution of the subproblems, we also have $\forall k, l : \ k > l \big[\mu^* \geq q\big(\boldsymbol{x}^k, \boldsymbol{y}^l\big) \big]$. Combining these results above, there exists a compact $\mathcal{K}$ such that $\big\{\boldsymbol{x}^k\big\} \subseteq \mathcal{K} \subsetneq \mathcal{X}$ and $\boldsymbol{x}^* \in \mathcal{K}$. Using $\mathcal{K}$ instead of $\mathcal{X}$ in (6.14), we obtain a new SIP, called $\widetilde{\text{SIP}}$. From compactness of $\mathcal{K}$ follows Assumption 8 is satisfied. From Assumption 12 follows Assumption 9. From continuity of q and compactness of $\mathcal{K}$ and $\mathcal{Y}$ follows Assumption 10. By Theorem 1, the lower-bounding procedure converges. It remains to show that the solution of $\widetilde{\text{SIP}}$ and of the original SIP are equivalent. $\widetilde{\text{SIP}}$ is a restriction of (6.14) because $\mathcal{K} \subsetneq \mathcal{X}$. Since the optimal point of (6.14) is also feasible in $\widetilde{\text{SIP}}$, the optimal solutions are equivalent. □

6.3.1.3 Case 1.2.3: Additional Computational Checks

In some instances, one can show (analytically) that Assumption 8 is satisfied by proving that $\lim_{\|\boldsymbol{x}\| \to +\infty} \sup_{\boldsymbol{y}} \ g(\boldsymbol{x}, \boldsymbol{y}) \ < 0$ holds for any sequence $\big\{\boldsymbol{x}^k\big\}$ with $\big\|\boldsymbol{x}^k\big\| \to \infty$ for $k \to \infty$. This can also be done in some cases computationally by solving the two problems (6.15) and (6.16) *a priori*. The computational burden is in most cases tractable because the problems are adaptions of (LLP$^{\text{unb}}$). However, one must assume that g is continuous on $\text{cl}(\mathcal{X})$.

$$
\begin{aligned}
x^{\text{infeas,UBD}} = \sup_{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{y} \in \mathcal{Y}} \ & \|\boldsymbol{x}\| \\
\text{s.t. } & g(\boldsymbol{x}, \boldsymbol{y}) \geq 0 \\
& \forall \bar{\boldsymbol{y}} \in \mathcal{Y}^0 [g(\boldsymbol{x}, \bar{\boldsymbol{y}}) \leq 0].
\end{aligned}
\tag{6.15}
$$

If $x^{\text{infeas,UBD}}$ is bounded, $\Lambda(\mathcal{Y}^0)$ is also bounded (Assumption 8(b)). We now verify if Assumption 8(a) holds. If the optimal objective value of

$$\begin{aligned} \sup_{\boldsymbol{x}\in\text{cl}(\text{cl}(\mathcal{X})\setminus\mathcal{X}),\boldsymbol{y}\in\mathcal{Y}} \quad & g(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t. } \forall \bar{\boldsymbol{y}} \in \mathcal{Y}^0 & [g(\boldsymbol{x},\bar{\boldsymbol{y}}) \leq 0]. \end{aligned} \tag{6.16}$$

is greater than zero, there exist no points that belong to $\text{cl}\big(\mathcal{X}^{\text{infeas}}\big)$ but not to $\mathcal{X}$. Because $\text{cl}\big(\mathcal{X}^{\text{infeas}}\big) \supseteq \Lambda(\mathcal{Y}^0)$, Assumption 8(a) holds.

6.3.2 Case 2: $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded

If one of the following assumptions holds instead of Assumption 5, no convergence issues occur.

Assumption 13 The set $\mathcal{X} \subsetneq \mathbb{R}^{n_x}$ is compact and $\forall\varepsilon > 0\ \exists\delta_2 > 0 :\ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\}, \boldsymbol{x} \in \mathcal{X}\big[\big\|\boldsymbol{x}^k - \boldsymbol{x}\big\| < \delta_2\big]$ holds $\big|g\big(\boldsymbol{x}^k,\boldsymbol{y}^k\big) - g\big(\boldsymbol{x},\boldsymbol{y}^k\big)\big| < \varepsilon$.

Assumption 14 The optimal solution $\boldsymbol{y}^k$ of (LLP$^{\text{unb}}$) exists for all iterates $\boldsymbol{x}^k$, and the sequence $\{\boldsymbol{y}^k\}$ does not diverge, i.e., $\lim_{k\to\infty}\big\|\boldsymbol{y}^k\big\| < \infty$.

As shown before, Assumption 9 is satisfied if f is continuous on $\mathcal{X}$. Furthermore, Assumption 13 holds if g is uniformly continuous on $\mathcal{X}\times\mathcal{Y}$ and $\mathcal{X}$ is compact.

Proposition 3 *Under Assumptions 7, 9 and 13, the lower-bounding procedure converges.*

Proof. First, from Assumption 13 follows Assumption 8. Second, Assumption 9 is fulfilled by Proposition 3. Third, Assumption 10 is fulfilled by Assumption 13. By Theorem 1, the lower-bounding procedure converges. □

Proposition 4 *Under Assumptions 6, 7 and 14, the lower-bounding procedure converges.*

Proof. Consider

$$\begin{aligned} \inf_{\tilde{\boldsymbol{x}}\in\widetilde{\mathcal{X}}} \quad & f(\tilde{\boldsymbol{x}}) \\ \text{s.t. } \forall \tilde{\boldsymbol{y}} \in \widetilde{\mathcal{Y}} & [g(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}) \leq 0] \end{aligned} \tag{$\widetilde{\text{SIP}}$}$$

with $\widetilde{\mathcal{X}} = \mathcal{X}$ and the compact interval $\widetilde{\mathcal{Y}} \subseteq \mathcal{Y}$ such that $\{\boldsymbol{y}^k\} \in \widetilde{\mathcal{Y}}$. With Assumption 6 follows uniform continuity of g on $\widetilde{\mathcal{Y}}\times\widetilde{\mathcal{X}}$. ($\widetilde{\text{SIP}}$) fulfills Assumptions 5 to 7. Therefore, Theorem 1 and its proof are applicable.
We will show by contradiction that the optimal objective value of the limit point of ($\widetilde{\text{SIP}}$) and the original SIP are equivalent, i.e., $f(\tilde{\boldsymbol{x}}^*) = f(\boldsymbol{x}^*)$. First, assume $f(\tilde{\boldsymbol{x}}^*) < f(\boldsymbol{x}^*)$. Since the functions g and f are the same for ($\widetilde{\text{SIP}}$) and the original SIP, we have $\exists\boldsymbol{y} \in \mathcal{Y} : g(\tilde{\boldsymbol{x}}^*,\boldsymbol{y}) > 0$, i.e., $\tilde{\boldsymbol{x}}^*$ is infeasible in the original SIP. Assumption 14 and $\lim_{k\to\infty} g\big(\tilde{\boldsymbol{x}}^*,\tilde{\boldsymbol{y}}^k\big) > 0$ prove $\tilde{\boldsymbol{x}}^*$ infeasible which gives us the desired contradiction. Second, assume $f(\tilde{\boldsymbol{x}}^*) > f(\boldsymbol{x}^*)$ which gives us (due to the same functions g and f) $\widetilde{\mathcal{X}}^{\text{feas}} \subsetneq \mathcal{X}^{\text{feas}}$. But from $\widetilde{\mathcal{Y}} \subseteq \mathcal{Y}$, follows $\widetilde{\mathcal{X}}^{\text{feas}} \supseteq \mathcal{X}^{\text{feas}}$ which gives us the desired contradiction. From $f(\tilde{\boldsymbol{x}}^*) \not< f(\boldsymbol{x}^*)$ and $f(\tilde{\boldsymbol{x}}^*) \not> f(\boldsymbol{x}^*)$ follows $f(\tilde{\boldsymbol{x}}^*) = f(\boldsymbol{x}^*)$. □

Assumption 14 generally cannot be proven *a priori*. Therefore, it is not directly applicable. However, stronger conditions than required for Assumption 14 can be checked *a priori*. For example, one may choose some large number M *a priori* and check during runtime whether $\|\boldsymbol{y}^k\| < M$ holds. One way to achieve this may be to require the solver to minimize the magnitude of the variable values among all global solutions of (LLP$^{\text{unb}}$). The auxiliary problem

$$\begin{aligned} &\inf_{\boldsymbol{y}\in\mathcal{Y}} \|\boldsymbol{y}\|_{\max} \\ &\text{s.t. } g(\bar{\boldsymbol{x}},\bar{\boldsymbol{y}}) = g(\bar{\boldsymbol{x}},\boldsymbol{y}). \end{aligned} \tag{SIP-AUX}$$

is solved with $\bar{y}$ being the optimal solution of the (LLP$^{\text{unb}}$). Note that only an approximate solution of the (LLP$^{\text{unb}}$) is required, c.f., Assumption 7. Therefore, a relaxation of the equality constraint is possible. The optimal solution of (SIP-AUX) is then used to populate the set $\mathcal{Y}^k$.
We must point out that if one chooses a too large M in combination with a too restrictive maximum number iteration, the test may be inconclusive. Furthermore, populating the set $\mathcal{Y}^k$ with the solution of (SIP-AUX) may be counterproductive in certain cases. One could also prove that Assumption 14 holds, by computing an UBD and an LBD for each lower-level variable y_i *a priori*. The following optimization-based bound tightening based approach computes upper and lower bounds for all y_i, i.e., y_i^{UBD} and y_i^{LBD}, respectively. However, there is no guarantee of success.

$$\begin{aligned} y_i^{\text{LBD}} = \inf_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}} & y_i \\ \text{s.t. } & g(\boldsymbol{x},\boldsymbol{y}) \geq 0 \\ & \forall \bar{\boldsymbol{y}} \in \mathcal{Y}^0[g(\boldsymbol{x},\bar{\boldsymbol{y}}) \leq 0] \end{aligned} \tag{6.17}$$

$$\begin{aligned} y_i^{\text{UBD}} = \sup_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}} & y_i \\ \text{s.t. } & g(\boldsymbol{x},\boldsymbol{y}) \geq 0 \\ & \forall \bar{\boldsymbol{y}} \in \mathcal{Y}^0[g(\boldsymbol{x},\bar{\boldsymbol{y}}) \leq 0] \end{aligned} \tag{6.18}$$

Note the direction of the inequality in (6.17) and (6.18) as we want to compute the maximum and minimum value of y_i on $\left(\text{cl}\left(\mathcal{X}^{\text{infeas}}\right) \cap \{\boldsymbol{x}\in\mathcal{X} : \ \boldsymbol{y}\in\mathcal{Y}^0[g(\boldsymbol{x},\boldsymbol{y})\leq 0]\}\right) \times \mathcal{Y}$.

6.3.3 Case 3: $\mathcal{X}$ Unbounded and $\mathcal{Y}$ Unbounded

For SIPs with unconstrained upper- and lower-level hosts, convergence can be guaranteed if a suitable combination of the assumptions presented in Section 6.3.2 and Section 6.3.1 are adopted. Note that this is possible, as none of the corresponding pairs are mutually exclusive.

6.4 Numerical Case Studies

We present two illustrative case studies from Chebyshev approximation. These are a proof-of-concept for our findings rather than a complete numerical study which is beyond the scope of this dissertation. For the presented case studies, we utelize the in libDIPS implemented solver *B&F*, c.f., Section 3.3.1 (which is equivalent to the lower-bounding

procedure of Mitsos (2011)). All subproblems are solved using BARON version 19.12.7., accessed through GAMS version 30.2.0 (GAMS Development Corporation, 2019, Khajavirad and Sahinidis, 2018). The termination tolerance of the lower-bounding procedure is set to $\varepsilon^{\mathrm{a}} = 10^{-3}$. The default values for BARON are used with the exception of the optimality tolerances `EpsA` and `EpsR` which are set to 10^{-9}. The numerical case studies are carried out on a Windows Server 2016 Standard with an Intel(R) Xeon(R) CPU E5-2640 v3 @2.60GHz processor and 128GB of RAM. Note that BARON cannot handle trigonometric functions, which limits the applicability, and does not give certificate of global optimality for complex subproblems.

6.4.1 Case Study for $\mathcal{X}$ Unbounded and $\mathcal{Y}$ Compact

Consider the minmax problem inspired by Chebyshev-approximation

$$\begin{aligned} f^* = \inf_{\boldsymbol{x}\in\mathcal{X}} \sup_{\boldsymbol{y}\in\mathcal{Y}} \Big\| h(\boldsymbol{x}) - p(\boldsymbol{x},\boldsymbol{y}) \Big\|_2, \\ \text{with } \mathcal{X} = \mathbb{R}^5 \\ \mathcal{Y} = [1,5] \end{aligned} \tag{6.19}$$

and the auxiliary functions p and h defined as

$$\begin{aligned} h(\boldsymbol{x}) &= -\exp\big(-(y-3)^2\big) + 0.2\exp(y-3) + 1 \\ p(\boldsymbol{x},\boldsymbol{y}) &= x_1 + x_2 y + x_3 y^2 + x_4 y^3 + x_5 y^4. \end{aligned}$$

We reformulate (6.19) to

$$\begin{aligned} f^* = \inf_{\boldsymbol{x}\in\mathcal{X},\mu\in\mathcal{E}} \mu \\ \text{s.t. } \forall \boldsymbol{y}\in\mathcal{Y}\big[(h(\boldsymbol{x}) - p(\boldsymbol{x},\boldsymbol{y}))^2 \le \mu\big], \\ \text{with } \mathcal{X} = \mathbb{R}^5 \\ \mathcal{Y} = [1,5] \\ \mathcal{E} = [0,+\infty). \end{aligned} \tag{6.20}$$

Note that we have in this example the case of Section 6.3.1.2, i.e., the SIP has the form of an epigraph reformulation. The lower-bounding procedure terminates after 11 iterations (total CPU time 19.5 s) with an optimal solution of $\boldsymbol{x}^* = (-2.111,\ 6.471,\ -4.377,\ 1.091,\ -0.088)^T$ and with a maximal error of the fit of 0.028, see Figure 6.3.

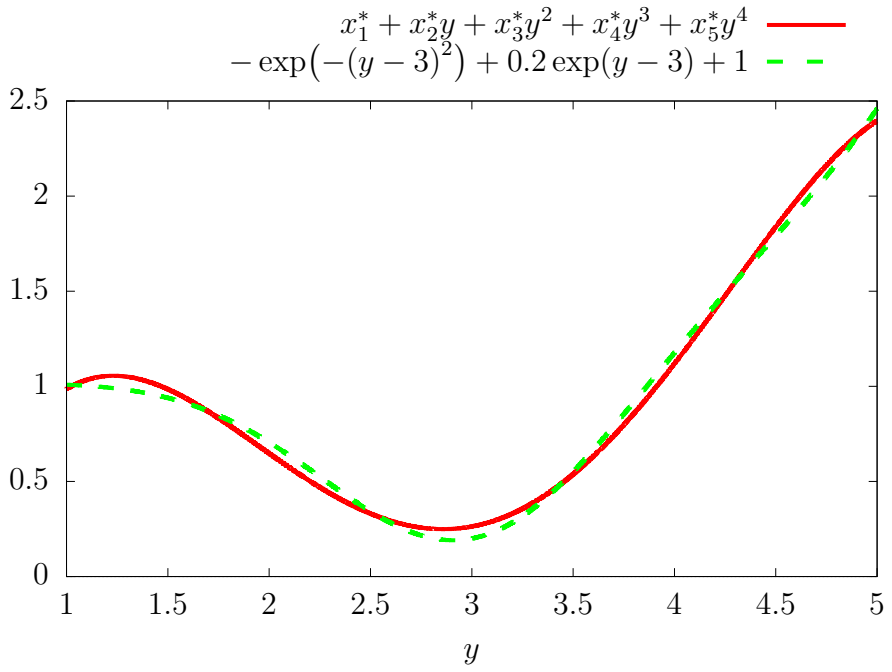


Figure 6.3: Case study for $\mathcal{X}$ unbounded and $\mathcal{Y}$ compact.The original function is plotted as a green dashed line and the approximation function as a red solid line. Note: Adapted from Jungen et al. (2022) with minor formatting and typesetting changes, CC BY 4.0, © The Author(s) 2022.

6.4.2 Case Study for $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded

Consider

$$\begin{aligned} f^* = \inf_{(\mu,\boldsymbol{x})^T \in \mathcal{X}} & \mu \\ & \text{s.t.}\, \forall \boldsymbol{y} \in \mathcal{Y} \left[\left(\left(\frac{3}{y^2} - \frac{2}{y^3} \right) - \left(\exp\left(x_1 (x_2 y + x_3)^2 \right) \right) \right)^2 \leq \mu \right], \\ & \text{with}\, \mathcal{X} = [0, 100] \times [-10, 10]^3 \\ & \quad\; \mathcal{Y} = [1, +\infty). \end{aligned} \tag{6.21}$$

(6.21) corresponds to a reformulated minmax Chebyshev-approximation problem, where $\left(\frac{3}{y^2} - \frac{2}{y^3}\right)$ is approximated by $\exp\left(x_1(x_2 y + x_3)^2\right)$ over $[1, +\infty)$ with $\boldsymbol{x}$ being the parameters to be estimated. For (6.21), Assumption 14 holds. The lower-bounding procedure terminates after 8 iterations (total CPU time 23.3 s) with the optimal parameters $\boldsymbol{x}^* = (-0.010, -4.285, 1.248)^T$, see Figure 6.4.

6.5 Conclusion and Outlook

Unconstrained upper and/or lower variable host sets in hierarchical programs arise in many applications. Specifically in SIPs, they may arise, e.g., in (inverse) Chebyshev approximation, epigraphic reformulations, or design centering. In this chapter, we pragmatically

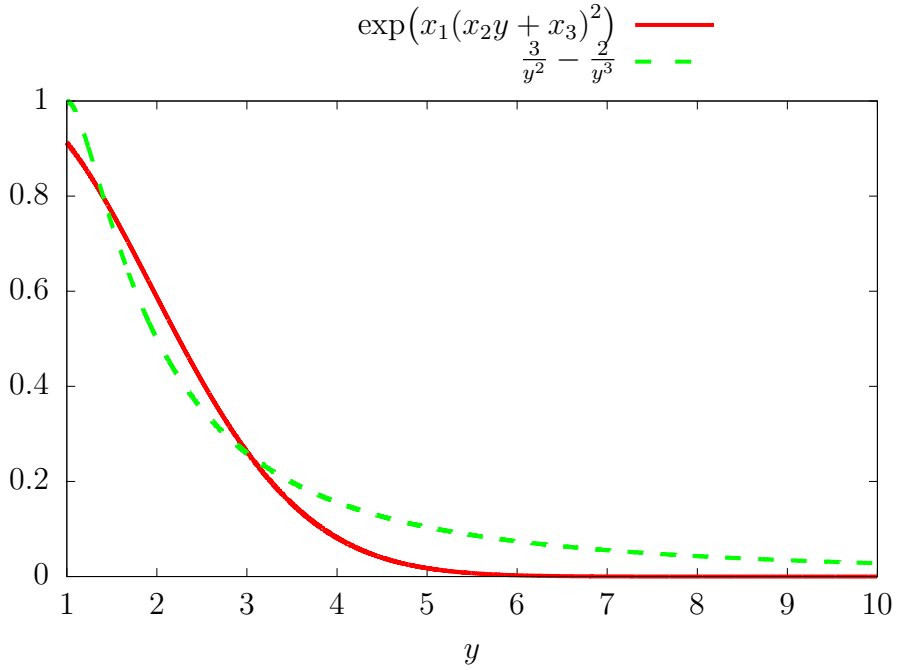


Figure 6.4: Case study for $\mathcal{X}$ compact and $\mathcal{Y}$ unbounded. The original function is plotted as a green dashed line and the approximation function as a red solid line. Note: Adapted from Jungen et al. (2022) with minor formatting and typesetting changes, CC BY 4.0, © The Author(s) 2022.

restricted ourselves to SIPs and showed that adaptive discretization-based algorithms are not suitable for all SIPs with unbounded host sets since convergence problems may arise.

Therefore, we investigated under which circumstances adaptive discretization-based algorithms are applicable to SIPs with unbounded host sets. The study was carried out using the lower-bounding procedure of Mitsos (2011), which can be considered representative of the class, c.f., Section 3.3. First, we briefly reviewed the assumptions of the lower-bounding procedure. Instead of the original assumptions, we used sharper, (slightly) relaxed, assumptions for our proof of the convergence of the lower-bounding procedure. In essence, the sharpened, slightly relaxed, assumptions establish a weaker form of uniform continuity of the constraint function on the set of all infeasible points in the host set of the SIP. In addition, the objective function must be at least semi-continuous on the boundary of a subset of the infeasible points.

For SIPs with unbounded host sets, it is often not obvious whether the relaxed assumptions hold. We discuss additional (stricter) assumptions and criteria for different cases of SIPs with unbounded host sets which can be often checked *a priori* or during run-time to enable application. The check of the proposed criteria are expected to be, at most, of the same computational burden as the subproblems of the SIPs. The additional assumptions and criteria are expected to apply to many applications. Finally, we give two numerical case studies of SIPs with unbounded host sets, as a proof-of-concept.

Moreover, we assume that our results are transferable to conceptually related adaptive discretization-based procedures as the ones reviewed in Section 3.3, as well as the used specialized procedure in Section 5.3.1. We show that this assumption is justified by also considering the conceptually related upper-bounding procedure of *RRHS* of Mitsos (2011),

see Appendix E.2. However, in future work, one should investigate whether our results carry over to other discretization-based algorithms.

7 Conclusion and Outlook

Bayesian optimization (BO) and optimal experimental design (OED) for guaranteed parameter estimation (GPE) can reduce experimental efforts while increasing knowledge gain. In this dissertation, we leveraged global and hierarchical optimization for these two complementary approaches and extended the applicability of hierarchical optimization methods.

Notably, one of the main challenges encountered in Chapters 2 to 6 is the solution of the subproblems to global optimality. Hence, the global solution to the underlying optimization subproblems is crucial for real-world applications. It remains to be seen whether significant improvements at the level of (MI)NLP solvers will be made in the future, which will likely further boost the applicability of the investigated approaches. In case the subproblem is not tractable by the employed subsolver, ad-hoc solution approaches are promising in some cases.

To maintain an overall acceptable computational time in Chapter 2, we terminated each optimization run after a predefined time limit, *assuming* that the current solution is indeed a globally near-optimal solution of the acquisition function. Recall that in Chapter 2, we investigated the advantages and disadvantages of using a deterministic global subsolver, i.e., MAiNGO, compared to two conventional subsolvers for optimizing the acquisition function. We showed that using a deterministic global subsolver in the BO context can reduce the number of iterations needed for convergence, and discussed practical implications. Notably, using MAiNGO was found to be advantageous for problems of increased dimensionality. Aside from the obvious drawback — lacking the *guarantee* of a global solution when using the ad-hoc approach of terminating prematurely after a predefined time limit — this does not constitute a significant obstacle in BO since the method inherently lacks complete precision. This imprecision can be caused by non-representative initial data or suboptimal acquisition function parameters. BO heavily relies on the initial data, at least as much as on the chosen solver, which can render the global solution of the acquisition function in some cases of secondary importance or even disadvantageous. If the original data set is not representative, the effort involved in optimizing the acquisition function might be futile or even disadvantageous. Therefore, obtaining a representative initial data set is of high importance, and approaches for obtaining a better initial data set should be further investigated, e.g., Bull (2011), Ren and Sweet (2024), Wang et al. (2024). The same applies for suboptimal parameters of the acquisition function. Selecting appropriate parameters is challenging without any prior knowledge, which warrants dynamic approaches, e.g., Kandasamy et al. (2015), Srinivas et al. (2012). Furthermore, in real-life applications, often a "reasonably good" solution suffices, that is, a practitioner may be satisfied with finding a "reasonably good" local minimum. If the global optimum is sought, other approaches using, e.g., mechanistic or hybrid models, are probably more promising.

Apart from these considerations, in the author's opinion and experience, BO should not be used without human intervention; instead, the (expert) knowledge gained within the BO loop should be continuously incorporated. For example, Xu et al. (2024) propose

a BO framework where a human provides binary accept or reject feedback on candidate sampling points. In particular, incorporating mechanistic knowledge about the underlying process into BO constitutes a promising research direction (Cramer et al., 2025) and could be further extended to integrate newly gained knowledge into BO on the fly. This could also lead to greater benefits when using deterministic global optimization while being less reliant on initial data.

OED for GPE is a hierarchical programming problem, and finding an effective solution procedure for these programs remains a significant challenge in both the existing literature and in this dissertation (Gottu Mukkula and Paulen, 2017a, Walz et al., 2018). This is not surprising as the global solution of a single-level problem, e.g., as in BO, is already challenging, and the challenges increase when more levels are added. In contrast to BO, in hierarchical optimization, such as in OED for GPE or in worst-case optimization, the global solution of the subproblems is of higher importance. If no global solution of the lower-level problem (LLP) is obtained, the reported solution may be, depending on the application, infeasible.

To enhance the applicability of adaptive discretization-based algorithms for these types of programs, we introduced in Chapter 3 libDIPS, an open-source software that implements multiple adaptive discretization-based subsolvers. The software includes a comprehensive library of test problems, comprising established benchmark libraries as well as numerous problems from the literature. We utilize this library to compare existing adaptive discretization-based solvers and to fine-tune the algorithmic parameters incorporated within these solvers. Although the presented library of test problems is already quite extensive, future work should curate and extend the library, especially with real-world problems. Additionally, we propose in Chapter 4 an adaptation of an existing subsolver showcasing that libDIPS serves as a development platform for adaptive discretization-based subsolvers.

The algorithm adaptation proposed in Chapter 4, as well as our numerical experiments in Section 3.6.5, showed that further performance improvements are possible through (seemingly minor) adaptations. Therefore, additional adaptations should be investigated in the future. libDIPS already allows for some basic adaptations to the subproblems, enabling customization of the problem formulation for the chosen subsolver, as the subproblems can be formulated in an opinionated way, c.f., Section 3.4.1.

During our investigations, we noticed that adaptive discretization-based algorithms can be highly sensitive to the solution reported by the subsolver of the LLP (which is used for discretizing). Although we did not observe a similar behavior in BO, the solution trajectories in BO might also differ significantly if multiple global minima exist or the optimality tolerances are too loose. Recall that some randomness is inherently introduced by the fact that standard (MI)NLP solvers can only solve the subproblems to a certain tolerance (ignoring the existence of multiple global minima in this discussion). Depending on the reported solution of the subsolver (satisfying optimality), the solution trajectories of the discretization-based algorithm can heavily deviate from one another. Some approaches, such as Djelassi (2020), Zingler and Mitsos (2025), can be assumed to be less sensitive to this randomness, as they already incorporate some sensitivity information. However, as we observed significant changes in solution times, this should be further investigated.

Future research entails customizing the subsolvers to, e.g., include cutting planes, i.e., the discretized semi-infinite constraints, into the Branch & Bound, which might be promising for hierarchical programs. This direction is pursued by, e.g., Bhattacharjee et al. (2005a), Floudas and Stein (2008), Marendet et al. (2020), Marousi and Charitopoulos (2025).

Furthermore, innovative initialization procedures of the discretization set should be investigated and compared: Adding local optima of the LLP (computed for, e.g., some randomly generated upper-level points), or advanced approaches as of Turan et al. (2025). However, initialization of the discretization set with a non-empty set comes with an additional computational burden, as a denser discretization in general leads to harder-to-solve problems. Recall that the global solution of the subproblems is a significant challenge. Furthermore, in e.g., robust control, the number of constraints may scale drastically with the size of the discretization, owing to the inclusion of discretized differential equations. Hence, "dropping" discretization points in these applications may be advantageous, as solving semi-infinite programs (SIPs) in this context remains a significant challenge (Muñoz and Marquardt, 2018). Some work in the direction of "dropping" discretization points already exists, see Blankenship and Falk (1976), Zingler et al. (2025a,b).

Another promising approach might be hybrid approaches, i.e., first employing some non-deterministic approaches (Sinha et al., 2014, 2018) and then checking whether the reported solution is indeed feasible by computing the LLP (to global optimality). If the solution is not feasible, the solution and potentially other intermediate solutions can be used for initializing the discretization and for the subsolver as initial points. Similarly, in BO, instead of only using a global deterministic solver, dynamically switching between global and local solvers is likely most promising in practice because stochastic elements are incorporated. Parallels can also be seen in machine learning, where stochastic gradient descent solvers are widely used to avoid or escape local minima of the loss function during training (Kleinberg et al., 2018).

In Chapter 5, we extended OED for GPE to systems whose input-output relation is implicitly defined through embedded optimization problems, which corresponds in our application to the rigorous computation of a liquid-liquid equilibrium (LLE). LLEs are usually modeled using excess models, which, while being very flexible and, therefore, have been shown to achieve excellent results in modelling, also have been shown to be problematic: spurious phase splits may be predicted. In OED, wrong predictions may lead to additional experimental effort. Therefore, we used ideas of Mitsos et al. (2009b) to avoid such wrong predictions. The resulting OED problem is a hierarchical program with multiple (generalized) semi-infinite constraints and coupling equality constraints. Using libDIPS as a development platform, we implemented our proposed specialized adaptive discretization-based algorithm for the resulting hierarchical program. As our assumption on the coupling equality constraints is rather restrictive, additional methods should be investigated to weaken the necessary assumptions. Zingler et al. (2025b) propose a method that is based on weakened assumptions where the equality constraints are, simplistically speaking, well-behaved.

Last but not least, for SIPs, we investigated in Chapter 6 whether we can relax the necessary assumptions, specifically the compactness of the involved sets, for the adaptive discretization-based algorithm proposed by Blankenship and Falk (1976), Mitsos (2011). We were able to relax the assumption by giving sharpened, slightly relaxed assumptions, widening the applicability of the procedure. Recall that in many real-life applications, the assumption of compact host sets may not be given. However, it remains to be seen whether our results carry over to other related algorithms and whether the employed subsolvers will be able to handle unbounded host sets.

Appendix A

Supplemental Material Regarding Merit of Deterministic Global Optimization of the Acquisition Function in Bayesian Optimization

A.1 Additional Test Cases

A.1.1 Six-Hump Camelback

2D benchmark (Jamil and Yang, 2013). Six minima, with two global minima located at $\boldsymbol{x}^* = (\pm 0.090, \mp 0.713)^T$, with objective value $f(\boldsymbol{x}^*) \approx -1.032$.

$$\begin{aligned} f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} \quad & \left(4 - 2.1x_1^2 + \tfrac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \\ \text{s.t.} \quad & x_1 \in [-3, 3],\ x_2 \in [-2, 2] \end{aligned} \tag{A.1}$$

A.1.2 Ackley

3D benchmark (Ackley, 1987). Many local minima; one global minimum at $\boldsymbol{x}^* = (0, 0, 0)^T$, with objective value $f(\boldsymbol{x}^*) = 0$.

$$\begin{aligned} f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} \quad & -20\exp\left(-0.2\sqrt{\tfrac{1}{3}\textstyle\sum_{i=1}^{3} x_i^2}\right) - \exp\left(\tfrac{1}{3}\textstyle\sum_{i=1}^{3}\cos(2\pi x_i)\right) + 20 + e \\ \text{s.t.} \quad & \boldsymbol{x} \in [-5, 5]^3 \end{aligned} \tag{A.2}$$

A.1.3 Hartmann

4D benchmark (Dixon and Szegö, 1978). One global minimum located at $\boldsymbol{x}^* = (0.1146, 0.5556, 0.8525, 0.8525)^T$, with objective value $f(\boldsymbol{x}^*) \approx -3.3224$.

$$\begin{aligned} f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} \quad & -\textstyle\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{4} A_{ij}(x_j - P_{ij})^2\right) \\ \text{s.t.} \quad & \boldsymbol{x} \in [0, 1]^4, \end{aligned} \tag{A.3}$$

with

$$\boldsymbol{\alpha} = (1.0, 1.2, 3.0, 3.2)^T,$$

$$\boldsymbol{A} = \begin{bmatrix} 10 & 3 & 17 & 3.5 \\ 0.05 & 10 & 17 & 0.1 \\ 3 & 3.5 & 1.7 & 10 \\ 17 & 8 & 0.05 & 10 \end{bmatrix},$$

$$\boldsymbol{P} = \begin{bmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 \end{bmatrix}.$$

A.2 Set Up Details

Table A.1: Settings for all case studies, detailing the number of points in the initial dataset (N), the κ value used in (LCB), the number of experiments and runs, and the termination criterion thresholds. For the κ schedules $\kappa_{t,\mathrm{S}}$ and $\kappa_{t,\mathrm{K}}$ refer to Section 2.2.4.1.

Case Study	N	κ	Experiments	Runs per Exp.	$\varepsilon_{x,1}$	$\varepsilon_{x,2}$	$\varepsilon_{f,r}$	$\varepsilon_{f,a}$
2D Müller-Brown	3	2	56	31	0.001	0.05	0.01	0.5
	3	3	25	31				
	10	2	31	31				
	20	2	41	31				
	20	0.5	16	31				
	3	$\kappa_{t,\mathrm{S}}$	21	21				
	3	$\kappa_{t,\mathrm{K}}$	51	21				
2D Camelback	3	2	41	31	0.001	0.05	0.02	0.05
3D Ackley	4	2	31	15	0.001	0.05	0.02	0.05
4D Hartmann	5	2	30	16	0.001	0.02	0.02	0.01

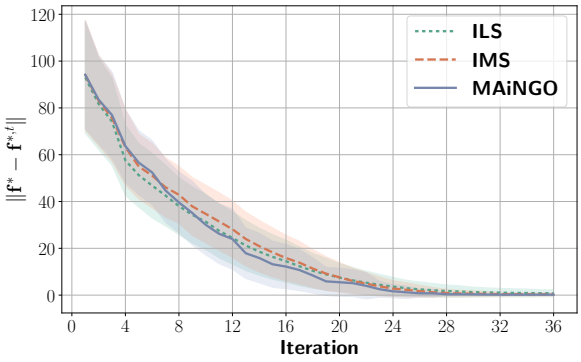
A.3 BO Frameworks

Table A.2: A non-exhaustive summary of recent Python-based BO framework software. BO packages exist in further programming languages, including C++ (e.g., MOE), R (e.g., DiceOptim, & laCP), and MATLAB (e.g., DACE).

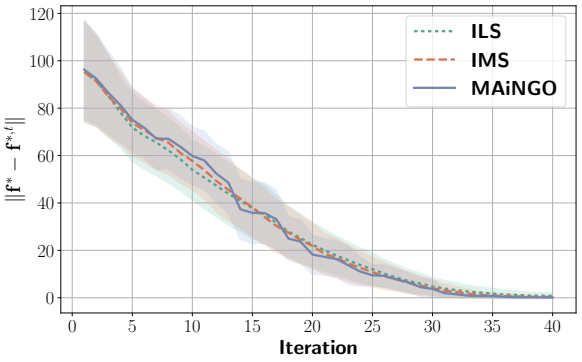
Software	Updated	GP Implementation	Acquisition Functions	Optimizer
BOTorch	2024	Wrapper around GPYTorch	PI, EI, UCB, Posterior Mean, MC-Based, +	Modular (scipy.optimize, torch.optim), with L-BFGS-B informed multi-start as default
Trieste	2024	GPFlow	PI, EI, LCB, Feasibility Based, MC-Based, +	Modular (tensorflow.optimizers, tf.keras.optimizers, scipy.optimize) with L-BFGS-B as default
bayes-optimization	2024	sklearn.GPR	PI, EI, UCB	scipy.optimize, L-BFGS-B informed multi-start
pyGPGO	2022	Native	PI, EI, UCB, Entropy	scipy.optimize, L-BFGS-B (random) multi-start
skopt	2021	sklearn.GPR	EI, LCB, PI	L-BFGS-B informed multi-start, parallel
GPyOpt	2020 (archived)	GPy	EI, LCB, PI, Entropy	scipy.optimize, L-BFGS-B multi-start, DIRECT, covariance matrix adaptation

A.4 Simple Regret

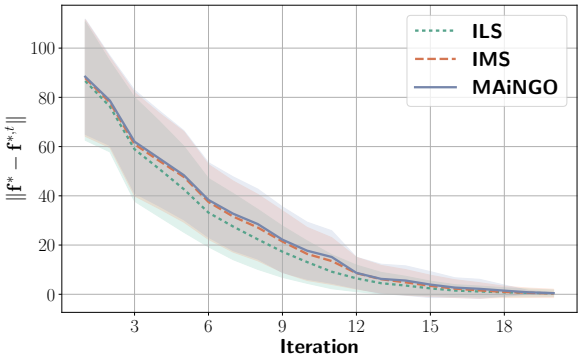
A.4.1 Considering Only Runs That Reached a Globally Near-Optimal Solution



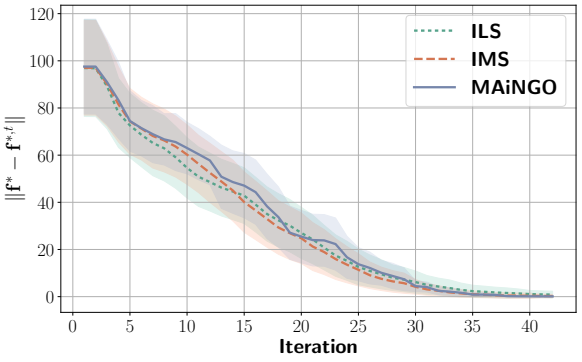
(a) Müller-Brown, $\kappa = 2$ and $N = 3$.



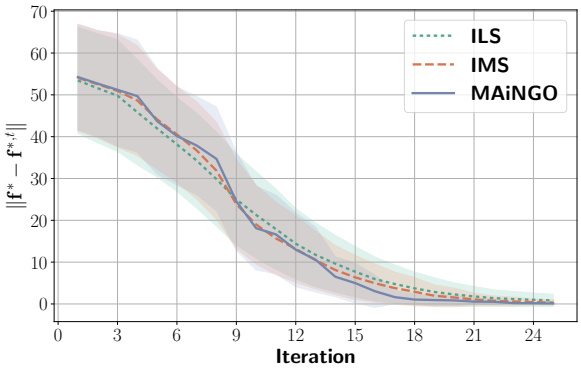
(b) Müller-Brown, $\kappa = 3$ and $N = 3$.



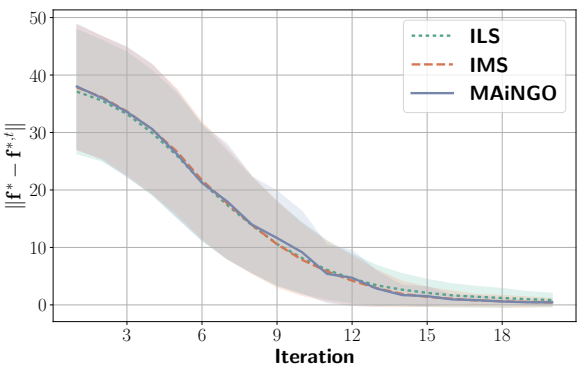
(c) Müller-Brown, $\kappa = \kappa_{t,\mathrm{K}}$ and $N = 3$.



(d) Müller-Brown, $\kappa = \kappa_{t,\mathrm{S}}$ and $N = 3$.

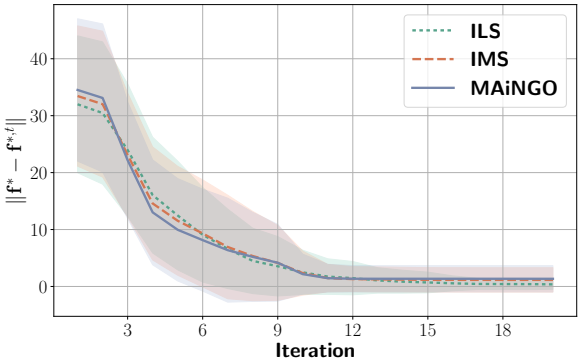


(e) Müller-Brown, $\kappa = 2$ and $N = 10$.

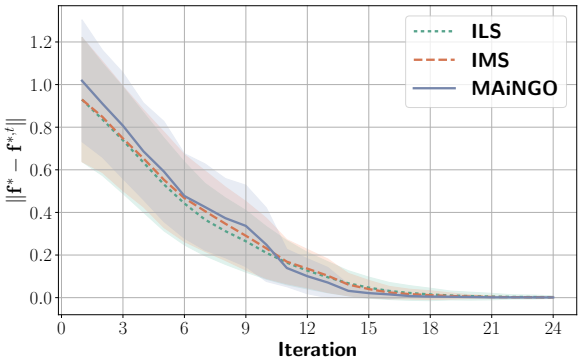


(f) Müller-Brown, $\kappa = 2$ and $N = 20$.

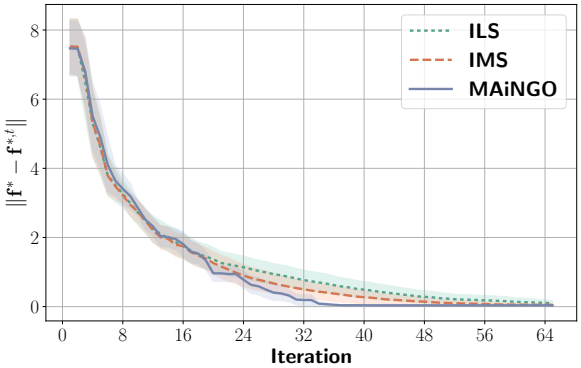
Simple regret plots (Part 1).



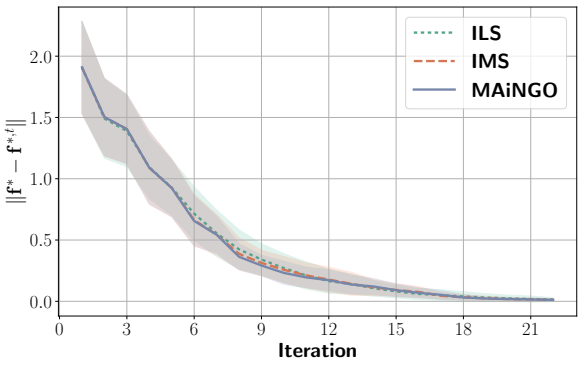
(g) Müller-Brown, $\kappa = 0.5$ and $N = 20$.



(h) 2D Camelback, $\kappa = 2$ and $N = 3$.



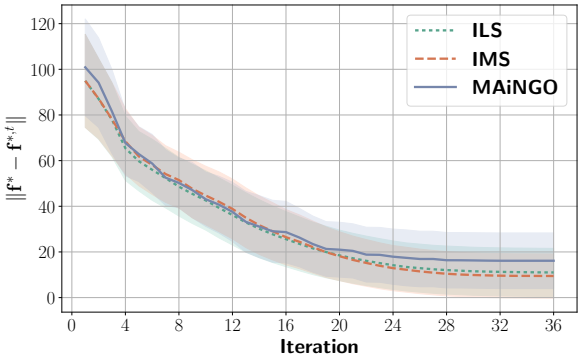
(i) 3D Ackley, $\kappa = 2$ and $N = 4$.



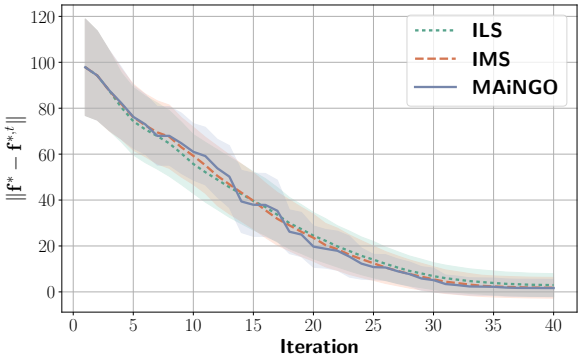
(j) 4D Hartmann, $\kappa = 2$ and $N = 5$.

Figure A.1: Simple regret plots for runs successfully converging to a globally near-optimal solution, considering only experiments where at least one run reached a globally near-optimal solution for all three solvers using (LCB). This subset of data is identical to the ones analyzed in Sections 2.3.2 and 2.3.3. Case study formulations are detailed in Appendix A.1, with experimental setup, number of runs, and termination criteria in Appendix A.2. Results align with the trends observed in Figure 2.3 and Figure 2.4, showing that MAiNGO generally terminates in fewer iterations. Lines represents the mean, while the shaded bands indicate ± 0.5 standard deviations. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

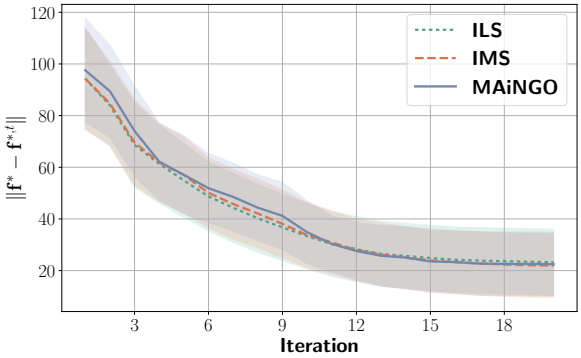
A.4.2 Considering All Runs, Despite Where They Converged



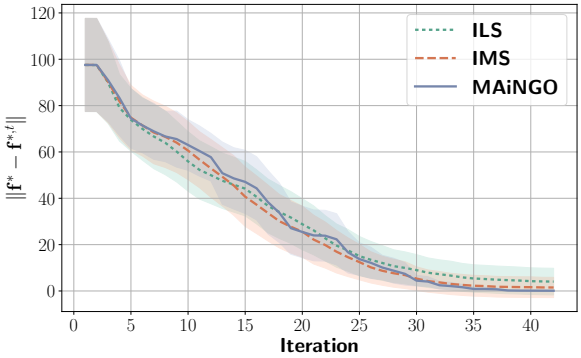
(a) Müller-Brown, $\kappa = 2$ and $N = 3$.



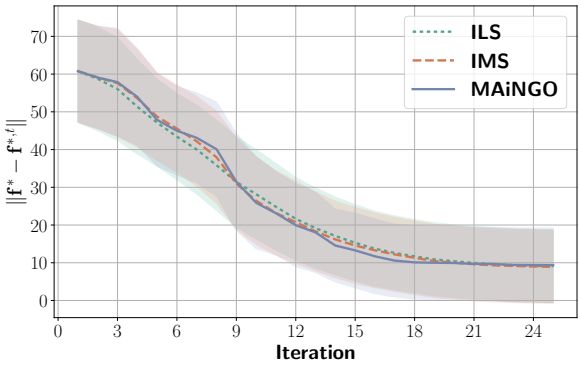
(b) Müller-Brown, $\kappa = 3$ and $N = 3$.



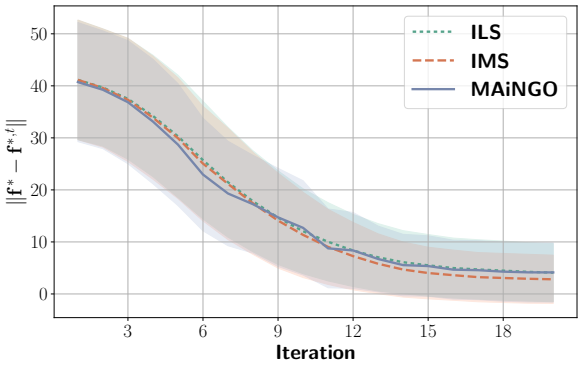
(c) Müller-Brown, $\kappa = \kappa_{t,\mathrm{K}}$ and $N = 3$.



(d) Müller-Brown, $\kappa = \kappa_{t,\mathrm{S}}$ and $N = 3$.

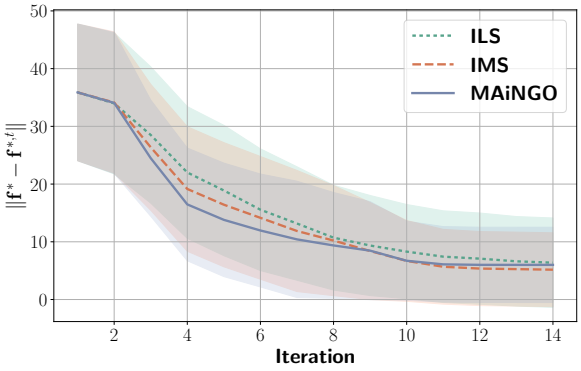


(e) Müller-Brown, $\kappa = 2$ and $N = 10$.

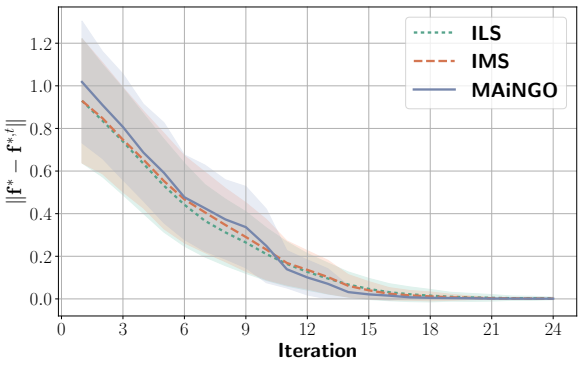


(f) Müller-Brown, $\kappa = 2$ and $N = 20$.

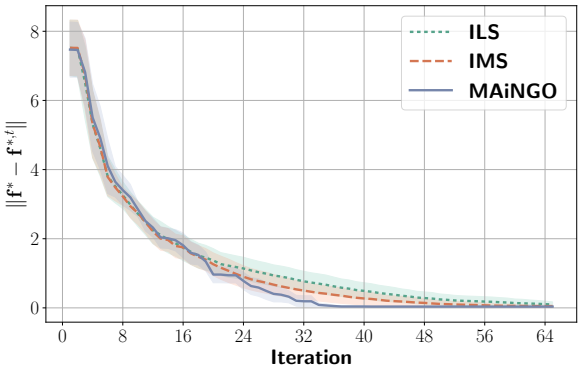
Simple regret plots (Part 1).



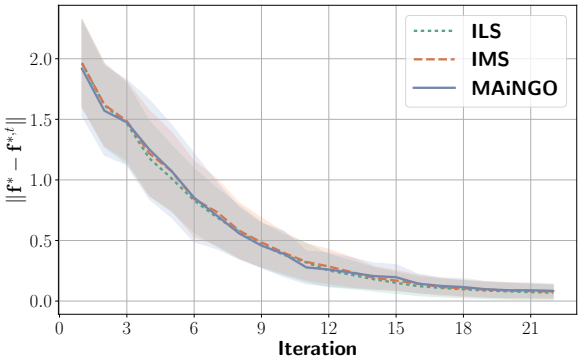
(g) Müller-Brown, $\kappa = 0.5$ and $N = 20$.



(h) 2D Camelback, $\kappa = 2$ and $N = 3$.



(i) 3D Ackley, $\kappa = 2$ and $N = 4$.



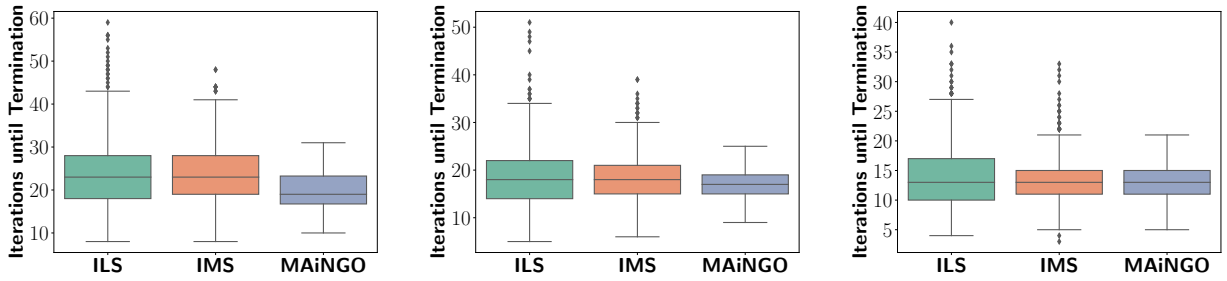
(j) 4D Hartmann, $\kappa = 2$ and $N = 5$.

Figure A.2: Simple regret plots for all conducted experiments and runs using (LCB). Case study formulations are detailed in Appendix A.1, with the experimental setup, number of runs, and termination criteria in Appendix A.2. Trajectories that do not approach 0 have a substantial probability of converging to a local minimum of the black box function. Lines represent the mean, while the shaded bands indicate ± 0.5 standard deviations. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

A.5 Iterations to Convergence for All Runs

Table A.3: Summary of convergence statistics for each case study, considering all runs of all experiments, despite which minimum each converged to. Mean, median, and standard deviation of iterations to convergence are reported for this subset of data. One-sided t-tests *paired by experiment* (initial dataset) evaluate whether MAiNGO converges in fewer iterations than IMS or ILS. MAiNGO converges in fewer iterations than ILS for all case studies and significantly fewer iterations than IMS for all but the case studies using more exploratory acquisition functions ($\kappa = 3$, $\kappa = \kappa_{t,\mathrm{S}}$). Details on the total number of experiments, runs, and termination criteria for each case study are provided in Appendix A.2.

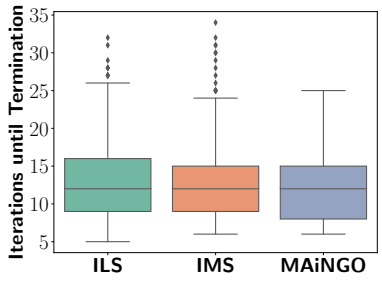
κ	N	Solver	Count	Mean	Median	Std	t-test	p-value
2	3	ILS	1736	23.6	23.0	7.55	6.53	1.100×10^{-8}
2	3	IMS	1736	23.0	23.0	5.40	3.84	1.900×10^{-7}
2	3	MAiNGO	1736	21.1	20.0	4.53	-	-
3	3	ILS	775	34.1	33.0	6.83	1.60	6.200×10^{-2}
3	3	IMS	775	33.4	33.0	5.28	0.14	4.400×10^{-1}
3	3	MAiNGO	775	33.0	34.0	4.24	-	-
2	10	ILS	961	18.5	18.0	6.10	3.21	1.600×10^{-3}
2	10	IMS	961	18.0	18.0	4.83	2.29	1.500×10^{-2}
2	10	MAiNGO	961	17.0	17.0	3.76	-	-
2	20	ILS	1271	14.0	13.0	5.01	3.70	3.200×10^{-4}
2	20	IMS	1271	13.2	13.0	3.85	1.32	9.700×10^{-2}
2	20	MAiNGO	1271	12.9	13.0	3.37	-	-
$\kappa_{t,\mathrm{S}}$	3	ILS	441	34.8	34.0	7.67	-0.73	7.600×10^{-1}
$\kappa_{t,\mathrm{S}}$	3	IMS	441	33.6	34.0	5.83	-2.21	9.800×10^{-1}
$\kappa_{t,\mathrm{S}}$	3	MAiNGO	441	35.4	36.0	3.46	-	-
$\kappa_{t,\mathrm{K}}$	3	ILS	1071	12.8	12.0	4.80	2.47	8.500×10^{-3}
$\kappa_{t,\mathrm{K}}$	3	IMS	1071	12.6	12.0	4.41	3.56	4.100×10^{-4}
$\kappa_{t,\mathrm{K}}$	3	MAiNGO	1071	11.8	12.0	4.04	-	-



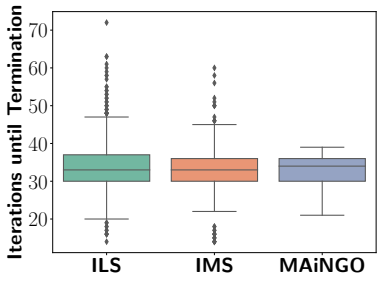
(a) $\kappa = 2$ and $N = 3$.

(b) $\kappa = 2$ and $N = 10$.

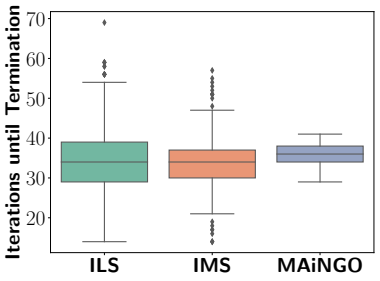
(c) $\kappa = 2$ and $N = 20$.



(d) $\kappa = \kappa_{t,\mathrm{K}}$ and $N = 3$.



(e) $\kappa = 3$ and $N = 3$.



(f) $\kappa = \kappa_{t,\mathrm{S}}$ and $N = 3$.

Figure A.3: Number of iterations until termination for all runs, despite which minimum they converged to, for a series of Müller-Brown case studies. Details on the number of experiments, runs, and termination criteria are provided in Appendix A.2. For the κ schedules $\kappa_{t,\mathrm{S}}$ and $\kappa_{t,\mathrm{K}}$ refer to Section 2.2.4.1. Note: Reprinted from Georgiou et al. (2025b), CC BY 4.0.

Appendix B

Supplemental Definitions

B.1 (ε^{a})-Feasibility

Definition 2 (*ε^{a}-SIP-Feasible Point*) A point $\bar{\boldsymbol{x}} \in \mathcal{X}$ is ε^{a}-SIP-feasible in (SIP) if

$$\boldsymbol{g}^{\mathrm{u}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \leq \varepsilon^{\mathrm{a}} \cdot \mathbf{1}, \ \forall\, \boldsymbol{y} \in \mathcal{Y},$$

with $\varepsilon^{\mathrm{a}} \geq 0$.

Definition 3 (*ε^{a}-GSIP-Feasible Point*) A point $\bar{\boldsymbol{x}} \in \mathcal{X}$ is ε^{a}-GSIP-feasible in (GSIP) if

$$\min \left\{ g_i^{\mathrm{u}}(\bar{\boldsymbol{x}}, \boldsymbol{y}), \min_{j \in \{1 \ldots n_{\mathrm{gl}}\}} - g_j^{\mathrm{l}}(\bar{\boldsymbol{x}}, \boldsymbol{y}) \right\} \leq \varepsilon^{\mathrm{a}}, \ \forall\, i \in \{1 \ldots n_{\mathrm{gu}}\}, \ \forall\, \boldsymbol{y} \in \hat{\mathcal{Y}},$$

with $\varepsilon^{\mathrm{a}} \geq 0$ and $\hat{\mathcal{Y}}$ as defined in (3.3) in Section 3.3.2.

Definition 4 (*(G)SIP-Feasible Point*) A point $\bar{\boldsymbol{x}} \in \mathcal{X}$ is called (G)SIP-feasible in (SIP) ((GSIP)) if it fulfills Definition 2 (Definition 3) with $\varepsilon^{\mathrm{a}} = 0$.

Definition 5 (*ε^{a}-BLP-Feasible Point*) A pair $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ with $\bar{\boldsymbol{x}} \in \mathcal{X}$ and $\bar{\boldsymbol{y}} \in \mathcal{Y}(\bar{\boldsymbol{x}})$ is called ε^{a}-BLP-feasible in (BLP) if $\boldsymbol{g}^{\mathrm{u}}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}) \leq \mathbf{0}$ and $\bar{\boldsymbol{y}}$ is ε^{a}-optimal in (BLP-LLP), see Mitsos et al. (2008a).

B.2 SIP-Slater and GSIP-LLP-Slater Point

Definition 6 (*SIP-Slater Point*) A point $\boldsymbol{x}^{\mathrm{S}} \in \mathcal{X}$ is called an SIP-Slater point in (SIP) if

$$\boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\big) < \mathbf{0}, \ \forall\, \boldsymbol{y} \in \mathcal{Y}.$$

Under compactness of $\mathcal{Y}$ and continuity of $\boldsymbol{g}$ if $\boldsymbol{x}^{\mathrm{S}}$ is an SIP-Slater point, there exists $\varepsilon^{\mathrm{S}} > 0$, such that

$$\boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\big) \leq -\varepsilon^{\mathrm{S}} \cdot \mathbf{1}, \ \forall\, \boldsymbol{y} \in \mathcal{Y},$$

c.f., Definition A.1 in Mitsos (2011).

Definition 7 (*GSIP-LLP-Slater Point*) A point $\boldsymbol{y}^{\mathrm{S}} \in \hat{\mathcal{Y}}$ (see (3.3) in Section 3.3.2) is called a GSIP-LLP-Slater point at $\bar{\boldsymbol{x}}$ in (GSIP) if

$$\boldsymbol{g}^{\mathrm{l}}\big(\bar{\boldsymbol{x}}, \boldsymbol{y}^{\mathrm{S}}\big) \leq -\varepsilon^{\mathrm{S}} \cdot \mathbf{1},$$

with some $\varepsilon^{\mathrm{S}} > 0$, c.f., Appendix B.2 in Mitsos and Tsoukalas (2015).

B.3 ε^{f}-optimal (G)SIP-Slater Point

Definition 8 (ε^{f}-*optimal SIP-Slater Point*) A point $\boldsymbol{x}^{\mathrm{S}} \in \mathcal{X}$ is called an ε^{f}-optimal SIP-Slater point in (SIP) if

$$f\big(\boldsymbol{x}^{\mathrm{S}}\big) \leq f^* + \varepsilon^{\mathrm{f}} \wedge \boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\big) \leq -\varepsilon^{\mathrm{S}} \cdot \mathbf{1},\ \forall\ \boldsymbol{y} \in \mathcal{Y},$$

with $\varepsilon^{\mathrm{f}}, \varepsilon^{\mathrm{S}} > 0$, c.f., Lemma 2.4 in Mitsos (2011).

Definition 9 (ε^{f}-*optimal GSIP-Slater Point*) A point $\boldsymbol{x}^{\mathrm{S}} \in \mathcal{X}$ is called an ε^{f}-optimal GSIP-Slater point in (GSIP) if

$$f\big(\boldsymbol{x}^{\mathrm{S}}\big) \leq f^* + \varepsilon^{\mathrm{f}} \wedge \big[\boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\big) \leq -\varepsilon^{\mathrm{S}} \cdot \mathbf{1} \vee \exists j : g_j^{\mathrm{l}}\big(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\big) > \varepsilon^{\mathrm{S}}\big],\ \forall\ \boldsymbol{y} \in \hat{\mathcal{Y}},$$

with $\varepsilon^{\mathrm{f}}, \varepsilon^{\mathrm{S}} > 0$, and $\hat{\mathcal{Y}}$ as defined in (3.3) in Section 3.3.2, c.f., Assumption 3 in Mitsos and Tsoukalas (2015).

Appendix C

Supplemental Material Regarding Discretization-Based Semi-Infinite and Bilevel Programming Solvers

C.1 Blankenship and Falk Solution Algorithm for SIPs

Algorithm C.1: Simplified pseudocode of the Blankenship and Falk algorithm (implemented in libDIPS as *B&F*) to solve (SIP). All subproblems are solved globally. User inputs are the initial discretization $\mathcal{Y}^0 \subsetneq \mathcal{Y}$ and feasibility tolerance $\varepsilon^{\mathrm{a}} \geq 0$.

1 set $\mathcal{Y}^{\mathrm{LBD}} \leftarrow \mathcal{Y}^0$;
2 solve (LBP) to obtain f^{LBP} and $\boldsymbol{x}^{\mathrm{LBP}}$;
3 solve (LLP) to obtain g^{LLP} and $\boldsymbol{y}^{\mathrm{LLP}}$;
4 **if** $g^{\mathrm{LLP}} > \varepsilon^{\mathrm{a}}$ **then**
5 $\quad \mathcal{Y}^{\mathrm{LBP}} \leftarrow \mathcal{Y}^{\mathrm{LBP}} \bigcup \{\boldsymbol{y}^{\mathrm{LLP}}\}$;
6 $\quad$ go to line 2
7 return f^{LBP} and $\boldsymbol{x}^{\mathrm{LBP}}$;

C.2 Further Subproblem Formulations

C.2.1 SIP Solvers

(ORA) of *Oracle* can be reformulated as

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X},\nu} \quad & \nu \\ \text{s.t.} \quad & f(\boldsymbol{x}) - f^{\mathrm{t}} \leq \nu \\ & \boldsymbol{g}^{\mathrm{u}}\big(\boldsymbol{x}, \boldsymbol{y}^k\big) \leq \nu \cdot \mathbf{1},\ \forall\, \boldsymbol{y}^k \in \mathcal{Y}^{\mathrm{ORA}} \end{aligned} \tag{ORA-REF}$$

$$\text{with} \quad \mathcal{X} \coloneqq \big\{\boldsymbol{x} \in \bar{\mathcal{X}} : \boldsymbol{v}^{\mathrm{iu}}(\boldsymbol{x}) \leq \mathbf{0},\ \boldsymbol{v}^{\mathrm{eu}}(\boldsymbol{x}) = \mathbf{0}\big\}.$$

C.2.2 MINMAX Solver

The ULP of *Minmax* reads

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{X},t} \quad & t \\ \text{s.t.} \quad & f\big(\boldsymbol{x},\boldsymbol{y}^k\big) - t \leq 0 \,\forall\, \boldsymbol{y}^k \in \mathcal{Y}^{\text{LBP}} \\ \text{with} \quad & \mathcal{X} \coloneqq \big\{\boldsymbol{x}\in\bar{\mathcal{X}} : \boldsymbol{v}^{\text{iu}}(\boldsymbol{x}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{eu}}(\boldsymbol{x}) = \boldsymbol{0}\big\}. \end{aligned} \tag{MINMAX-LBP}$$

Note that we have introduced the auxiliary variable t.

C.2.3 BLP Solvers

The auxiliary problem of *BLP-noBox* and *BLP-Box* is formulated as

$$\begin{aligned} \min_{\boldsymbol{y}\in\mathcal{Y}(\bar{\boldsymbol{x}}),u\in\mathbb{R}} \quad & u \\ \text{s.t.} \quad & h(\bar{\boldsymbol{x}},\boldsymbol{y}) \leq h^*(\bar{\boldsymbol{x}}) + \varepsilon^{\text{AUX}} \\ & \boldsymbol{g}^l(\bar{\boldsymbol{x}},\boldsymbol{y}) \leq u\cdot\boldsymbol{1} \\ \text{with} \quad & \mathcal{Y}(\boldsymbol{x}) \coloneqq \big\{\boldsymbol{y}\in\bar{\mathcal{Y}} : \boldsymbol{g}^{\text{l}}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{il}}(\boldsymbol{y}) \leq \boldsymbol{0},\ \boldsymbol{v}^{\text{el}}(\boldsymbol{y}) = \boldsymbol{0}\big\}. \end{aligned} \tag{BLP-AUX}$$

The auxiliary problem for *BLP-Box* for deciding whether or not a given box $\mathcal{X}_{\text{box}}$ for the upper-level variables is small enough such that the given discretization point $\bar{\boldsymbol{y}}$ stays feasible in the lower level is given by

$$\min_{\boldsymbol{x}\in\mathcal{X}_{\text{box}}} \; - \max_{j\in\{1\ldots n_{gl}\}} \boldsymbol{g}^l(\boldsymbol{x},\bar{\boldsymbol{y}}). \tag{BLP-AUX-V}$$

Note that Mitsos et al. (2008a) uses interval analysis to determine if $\mathcal{X}_{\text{box}}$ is valid.

C.3 Details on UBP-Guard

The pseudocode in Algorithm C.2 illustrates the experimental change described in Section 3.6.5. It also includes one of the changes mentioned in Section 3.4.2.1, namely the recovery step (see Appendix C.6). Other details, such as consideration of optimality tolerances, are omitted for brevity.

Algorithm C.2: Simplified pseudocode of *RRHS* with guard g^{UBP} for the upper-bounding procedure and recovery step.

$LBD \leftarrow -\infty$, $UBD \leftarrow \infty$, $\mathcal{Y}^{\mathrm{LBP}} \leftarrow \emptyset$, $\mathcal{Y}^{\mathrm{UBP}} \leftarrow \emptyset$, $\varepsilon^{\mathrm{r}} \leftarrow \varepsilon^{\mathrm{r},0}$, $NEXT \leftarrow LBP$, $i \leftarrow 0$;
while ($i <$ *maxiter*) ***and*** *LBD and UBD* ***not*** *sufficiently close* **do**
  switch $NEXT$ **do**
    case *LBP* **do**
      Solve (LBP) to obtain f^{LBP} and $\bar{\boldsymbol{x}}$;
      if *Infeasible* **then return** Infeasible ;
      $LBD \leftarrow f^{\mathrm{LBP}}$;
      Solve (LLP) given $\bar{\boldsymbol{x}}$ to obtain g^{LLP} and $\bar{\boldsymbol{y}}$;
      if *Infeasible* ***or*** $g^{\mathrm{LLP}} \leq 0$ **then**
        $UBD \leftarrow f^{\mathrm{LBP}}$;
        return GLOBAL;
      $\mathcal{Y}^{\mathrm{LBP}} \leftarrow \mathcal{Y}^{\mathrm{LBP}} \cup \{\bar{\boldsymbol{y}}\}$;
      if $g^{\mathrm{LLP}} < g^{\mathrm{UBP}}$ ***or*** $g^{\mathrm{UBP}} \leq 0$ **then**
        $NEXT \leftarrow UBP$;
      else
        $\mathcal{Y}^{\mathrm{UBP}} \leftarrow \mathcal{Y}^{\mathrm{UBP}} \cup \{\bar{\boldsymbol{y}}\}$, $NEXT \leftarrow LBP$;
    case *UBP* **do**
      Solve (UBP) to obtain f^{UBP} and $\bar{\boldsymbol{x}}$;
      if *Infeasible* **then**
        $\varepsilon^{\mathrm{r}} \leftarrow \varepsilon^{\mathrm{r}}/\varepsilon^{\mathrm{red}}$;
        if $\varepsilon^{\mathrm{r}} < \varepsilon^{\mathrm{r,min}}$ **then** $NEXT \leftarrow RECOVER$;
      else
        Solve (LLP) given $\bar{\boldsymbol{x}}$ to obtain g^{LLP} and $\bar{\boldsymbol{y}}$;
        if *Infeasible* ***or*** $g^{\mathrm{LLP}} \leq 0$ **then**
          $UBD \leftarrow f^{\mathrm{UBP}}$, $\varepsilon^{\mathrm{r}} \leftarrow \varepsilon^{\mathrm{r}}/\varepsilon^{\mathrm{red}}$;
          if $\varepsilon^{\mathrm{r}} < \varepsilon^{\mathrm{r,min}}$ **then** $NEXT \leftarrow RECOVER$;
        else
          $\mathcal{Y}^{\mathrm{UBP}} \leftarrow \mathcal{Y}^{\mathrm{UBP}} \cup \{\bar{\boldsymbol{y}}\}$;
    case *RECOVER* **do**
      Solve (UBP) with $\varepsilon^{\mathrm{r}} = 0$ and obtain f^{REC} and $\bar{\boldsymbol{x}}$;
      if *Infeasible* **then return** Infeasible ;
      Solve (LLP) given $\bar{\boldsymbol{x}}$ to obtain g^{LLP} and $\bar{\boldsymbol{y}}$;
      if *Infeasible* ***or*** $g^{\mathrm{LLP}} \leq 0$ **then**
        $LBD \leftarrow f^{\mathrm{REC}}$, $UBD \leftarrow f^{\mathrm{REC}}$;
        return GLOBAL;
      return ASSUMPTION_VIOLATION_LIKELY;

C.4 Note on Integer Capabilities of the Solvers

In the following, we argue that although some of the publications on which we base our implementation do not explicitly cover the integer case, the corresponding implementation handles discrete variables. The exception is *BLP-Box*, as we base our implementation on Mitsos et al. (2008a). Although the extension in Mitsos (2010) explicitly covers the integer case, the required modifications have not yet been implemented.

Djelassi et al. (2019) considers BLPs and GSIPs and Mitsos (2010) considers BLPs with mixed-integers. Both approaches provide a proof of finite convergence for their respective methods. Recall that the solver *BLP-noBox* is based on Djelassi et al. (2019) and Mitsos (2010) is the extension to Mitsos et al. (2008a), which is implemented as *BLP-Box*.

The approach presented in Djelassi et al. (2019) extends the methods of Blankenship and Falk (1976) (*B&F*), Mitsos (2011) (*RRHS*), and Mitsos and Tsoukalas (2015) (*GSIP-RRHS*). The convergence proof provided in Djelassi et al. (2019) can be adapted to these earlier approaches to establish finite convergence. Similarly, although Tsoukalas and Rustem (2011) (*Oracle*) and Djelassi and Mitsos (2017) (*Hybrid*) do not explicitly address integer variables, the algorithms are based on Blankenship and Falk (1976), and the same convergence argument applies.

To illustrate the underlying concept, we note that all approaches approximate the set $\mathcal{Y}$ with a finite discrete set. Similarly, there are only finitely many possible values for the discrete lower-variables within the host set. Thus, in many cases, it is straightforward to incorporate host sets $\bar{\mathcal{Y}}$ that restrict a subset of the variables to be discrete without affecting convergence guarantees. In fact, discretization of the discrete dimensions is inherently more effective, as the set of feasible values is already finite. Therefore, assuming finite convergence is established for fixed values of the discrete variables, the approximation will be exact with respect to the discrete variables after a finite number of additional iterations (a more detailed discussion, applicable to the other algorithms as well, can be found in Djelassi et al. (2019), Mitsos (2010)). In the worst case, this may require complete enumeration of the discrete lower-level variables. However, in our experience with applications involving integer variables, only a few iterations are typically required (see, e.g., Djelassi et al. (2018), Zingler et al. (2025a) and our computational results for SIPs with discrete variables in the upper and/or lower level, c.f., Figure C.6 in Appendix C.8.1.3).

C.5 Example Subproblem Formulation for *RRHS*

For using *RRHS* for the solution of (Ex) in Section 3.4.1.1, the same problem files `lbp.txt` and `llp.txt` are needed; c.f., listing 3.1 and listing 3.2 in Section 3.4.1.1, respectively. Additionally, the (UBP) in Section 3.3.1 is transcribed to the `ubp.txt` file, c.f., listing C.1. Besides the additional subproblem file, the user has to extend the `def.txt` file, as shown in listing C.2. For *RRHS*, the user is required to provide, additionally to the information required in the case of *B&F*, the lower-level variable name, in this case, `y`, the discretized semi-infinite set of the UBP, `ubp_y_disc`, the index set `ubp_k_disc` of the UBP, as well as the name of the parameter representing the restriction of the right-hand side, `eps_res`.

```
definitions:
real[2] x in [-3, 3];

set{index} ubp_k_disc := {};
real[0,2] ubp_y_disc := 0;

real eps_res := 0;

objective:
2*x[1]^2+x[2]^2;

constraints:
x[1]+x[2] = 1.0;
forall k in ubp_k_disc :
    min(x[1]+ubp_y_disc[k,1],x[2]+ubp_y_disc[k,2]) <= - eps_res;
```

Listing C.1: Listing of the file `ubp.txt` for the solution of (Ex) in Section 3.4.1.1 using *RRHS*. $\mathcal{Y}^{\mathrm{UBP}}$ is represented by `ubp_y_disc` and `ubp_k_disc`, where the former contains the elements of the set and the latter contains the indices. Both are internally updated by the used solver. Note that in this example, $\mathcal{Y}^{\mathrm{UBP}}$ is initialized as an empty set; thus, the size of the first dimension of `ubp_y_disc` is zero.

```
programdefinitions("lbp"):
set_disc(1) = lbp_k_disc;
set_disc_parameter_src(1, lbp_y_disc) = y;

programdefinitions("ubp"):
set_disc(1) = ubp_k_disc;
set_disc_parameter_src(1, ubp_y_disc) = y;
set_internal("eps_res") = eps_res;
```

Listing C.2: Listing of the file `def.txt` for the solution of (Ex) in Section 3.4.1.1 using *RRHS*.

C.6 Changes in the Implemented Solvers

The following changes and improvements were made compared to the original publications.

Changes Common in All Implementations

- the termination criterion is checked more often / at different place(s) in the algorithm
- iterations are defined differently
- allow termination based on relative optimality gap

Changes in the Implementation of *B&F* Based on Blankenship and Falk (1976)

- added tolerance update addressing the approximate solution of subproblems
- support multiple LLPs, i.e., $\boldsymbol{g}^{\mathrm{u}}$, as well as multiple semi-infinite constraints with different infinite index sets
- discretization of all or only single LLP possible

Changes in the Implementation of *Minmax* Based on Falk and Hoffman (1977)

- added tolerance update addressing the approximate solution of subproblems
- support multiple LLPs, i.e., $\boldsymbol{g}^{\mathrm{u}}$, as well as multiple semi-infinite constraints with different infinite index sets
- discretization of all or only single LLP possible

Changes in the Implementation of *RRHS* Based on Mitsos (2011)

- support multiple LLPs, i.e., $\boldsymbol{g}^{\mathrm{u}}$, as well as multiple semi-infinite constraints with different infinite index sets
- discretization of all or only single LLP possible
- added (optional) feasibility tolerance to be able to terminate upon an SIP-ε^{a}-feasible point (Definition 2 in Appendix B.1)
- added $\varepsilon^{\mathrm{r,min}}$ to terminate if $\varepsilon^{\mathrm{r}} < \varepsilon^{\mathrm{r,min}}$, which likely corresponds to an assumption violation, i.e., no SIP-Slater point (Definition 6 in Appendix B.2) exists
- added tolerance update to account for the issue raised by Harwood et al. (2021)

Changes in the Implementation of *Oracle* Based on Tsoukalas and Rustem (2011)

- support multiple LLPs, i.e., $\boldsymbol{g}^{\mathrm{u}}$, as well as multiple semi-infinite constraints with different infinite index sets
- discretization of all or only single LLP possible
- added (optional) feasibility tolerance to be able to terminate upon an SIP-ε^{a}-feasible point (Definition 2 in Appendix B.1)
- solve (LBP) (see Section 3.3.1) and corresponding LLP(s) once if initial discretization set $\mathcal{Y}^{\mathrm{ORA}}$ is empty or no initial LBD is provided by the user, generating both a LBD and an initial discretization point
- solving auxiliary problem UTOPIAN-UBP if the user provides no UBD to compute a UBD (which is not necessarily feasible). Note that if the UBD is not attainable, the SIP is infeasible. Thus, we also handle the infeasible case.
- avoid looping of the algorithm if (ORA) is infeasible ((LBP) is computed if (ORA) is infeasible to provide hints on whether the (ORA)-problem is wrongly implemented; for the problem definition of (LBP) and (ORA) refer to Section 3.3.1)
- added tolerance update addressing the approximate solution of subproblems

Changes in the Implementation of *Hybrid* Based on Djelassi and Mitsos (2017)

- support multiple LLPs, i.e., $\boldsymbol{g}^{\mathrm{u}}$, as well as multiple semi-infinite constraints with different infinite index sets
- discretization of all or only single LLP possible
- added (optional) feasibility tolerance to be able to terminate upon an SIP-ε^{a}-feasible point (Definition 2 in Appendix B.1)
- added $\varepsilon^{\mathrm{r,min}}$ to terminate if $\varepsilon^{\mathrm{r}} < \varepsilon^{\mathrm{r,min}}$, which likely corresponds to an assumption violation, i.e., no SIP-Slater point (Definition 6 in Appendix B.2) exists
- allow termination based on relative optimality gap

Changes in the Implementation of *GSIP-RRHS* Based on Mitsos and Tsoukalas (2015)

- discretization of all or only single LLP possible
- added (optional) feasibility tolerance to be able to terminate upon a GSIP-ε^{a}-feasible point (Definition 3 in Appendix B.1)
- added tolerance update addressing the approximate solution of subproblems
- iterative reduction of α in the auxiliary problem if necessary
- added $\varepsilon^{\mathrm{r,min}}$ to terminate if $\varepsilon^{\mathrm{r}} < \varepsilon^{\mathrm{r,min}}$, which likely corresponds to an assumption violation, i.e., no SIP-Slater point (Definition 6 in Appendix B.2) exists

- added $\alpha^{\min}$ to terminate if $\alpha < \alpha^{\min}$, which likely corresponds to an assumption violation, i.e., no lower-level GSIP-LLP-Slater point (Definition 7 in Appendix B.2) exists at the current iterate.
- Simplifications:
  - one restriction for all constraints g_j^{l}

Changes in the Implementation of *BLP-Box* Based on Mitsos et al. (2008a)

- using (BLP-AUX-V) in Appendix C.2.3 to decide if a guessed box size is small enough instead of interval arithmetic
- added termination if the required box size becomes too small
- the UBD for the lower-level objective function is calculated by solving an optimization problem with very loose tolerances instead of using interval arithmetic

C.7 Settings and Tolerances Used in the Numerical Experiments

Table C.1: Settings used in the numerical experiments of Section 3.6. Note that some settings are varied in the numerical experiments or changed automatically by the respective solver during the solution. *: `feas_tol` is set to 1.00×10^{-4} for the BLP solvers.

Setting	Value
`feas_tol`*	0.00
`abs_tol`	1.00×10^{-2}
`rel_tol`	1.00×10^{-2}
`abs_tol_lbp`	1.00×10^{-3}
`rel_tol_lbp`	1.00×10^{-3}
`abs_tol_lbp_llp`	1.00×10^{-4}
`rel_tol_lbp_llp`	1.00×10^{-4}
`abs_tol_lbp_aux`	1.00×10^{-5}
`rel_tol_lbp_aux`	1.00×10^{-5}
`abs_tol_ubp`	1.00×10^{-3}
`rel_tol_ubp`	1.00×10^{-3}
`abs_tol_ubp_llp`	1.00×10^{-4}
`rel_tol_ubp_llp`	1.00×10^{-4}
`abs_tol_ora`	1.00×10^{-3}
`rel_tol_ora`	1.00×10^{-3}

Table C.1: continued.

Setting	Value
`abs_tol_ora_llp`	1.00×10^{-4}
`rel_tol_ora_llp`	1.00×10^{-4}
`abs_tol_llp_ora_aux`	1.00×10^{-5}
`rel_tol_llp_ora_aux`	1.00×10^{-5}
`abs_tol_aux`	1.00×10^{-4}
`rel_tol_aux`	1.00×10^{-4}
`abs_tol_aux_v`	1.00×10^{-9}
`rel_tol_aux_v`	1.00×10^{-4}
`abs_tol_ora_eps_h`	1.00×10^{-5}
`abs_tol_ubp_eps_h`	1.00×10^{-5}
`abs_tol_aux_eps_h2`	1.00×10^{-9}
`abs_tol_res`	1.00×10^{-4}
`rel_tol_res`	1.00×10^{-4}
`abs_tol_res_llp`	1.00×10^{-5}
`rel_tol_res_llp`	1.00×10^{-5}
`init_res`	1.00
`red_res`	2.00
`min_eps_res`	1.00×10^{-9}
`init_res_u`	1.00
`red_res_u`	2.00
`min_eps_res_u`	1.00×10^{-9}
`init_res_l`	1.00
`red_res_l`	2.00
`min_eps_res_l`	1.00×10^{-9}
`init_d_box`	1.00
`red_d`	5.00×10^{-1}
`min_d_box`	1.00×10^{-9}
`init_alpha`	5.00×10^{-1}
`red_alpha`	1.50
`min_alpha`	1.00×10^{-9}
`tol_llp_slater`	1.00×10^{-5}
`red_tol_lbp_llp`	1.25
`red_tol_ubp_llp`	1.25

Table C.1: continued.

Setting	Value
`red_tol_ora`	1.25
`red_tol_ora_llp`	1.25
`red_tol_res_llp`	1.25
`max_iter`	1.00×10^3
`max_time`	1.20×10^3
`discr_all`	`false`

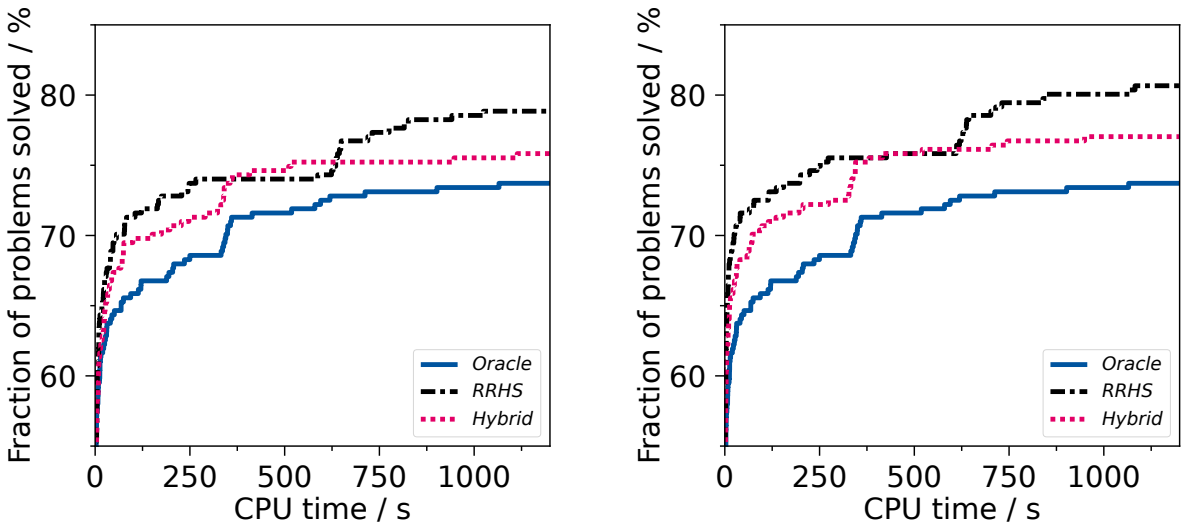
C.8 Additional Performance Profiles

Analogously to the CPU Time performance ratio, c.f., (PR) in Section 3.6, the subproblem performance ratio is defined as

$$\textit{Subproblem performance ratio} = \frac{\textit{number of subproblems solved for problem instance } i}{\textit{smallest number of subproblems solved for problem instance } i \textit{ of any solver}}. \quad \text{(S-PR)}$$

C.8.1 Additional SIP Performance Profiles

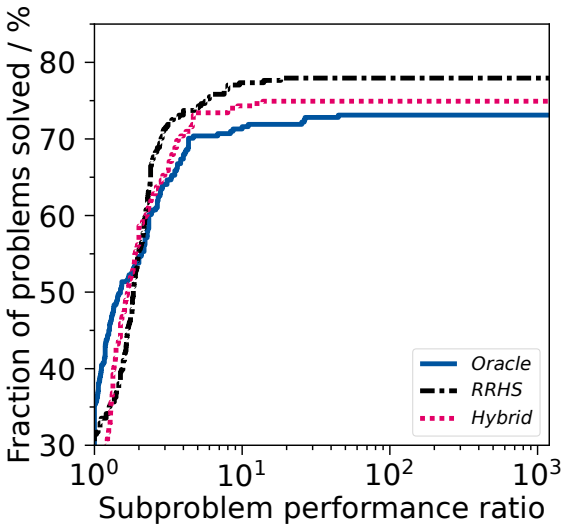
C.8.1.1 Considering All Problem Instances



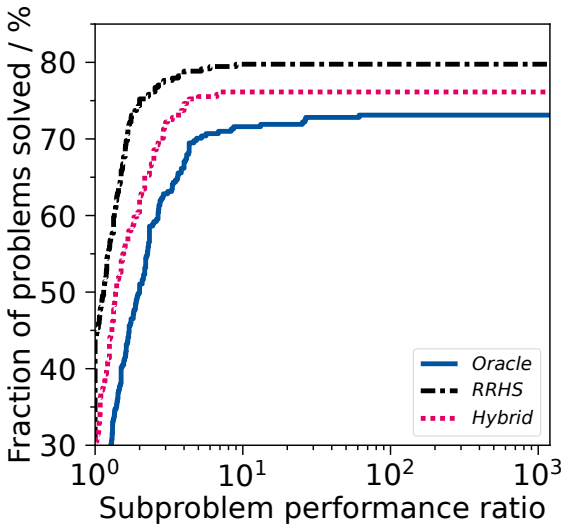
(a) "Out of the box performance"; parameters as in the original publications.

(b) Solvers with tuned parameters.

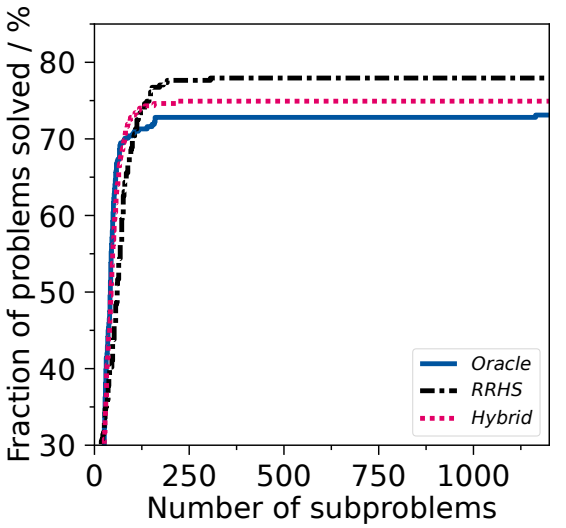
Figure C.1: CPU time performance profiles of SIP solvers: *RRHS*, *Oracle*, and *Hybrid*. "Out of the box performance" of the SIP solvers with their default parameters, i.e., the parameters of each solver are set to the values of their respective original publication, vs. SIP solvers with tuned parameters. Parameter values are listed in Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).



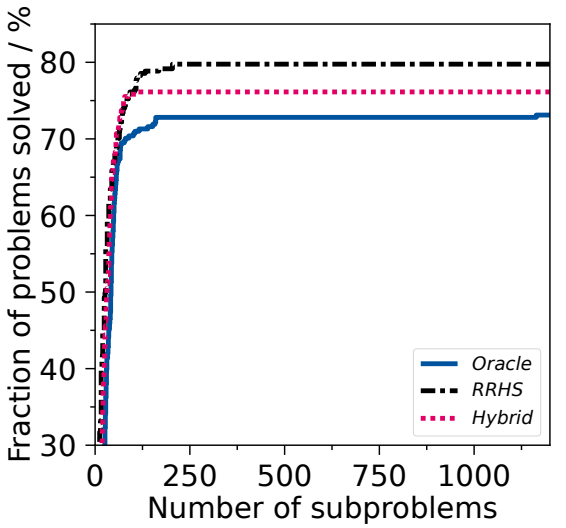
(a) "Out of the box performance"; parameters as in the original publications.



(b) Solvers with tuned parameters.



(c) "Out of the box performance"; parameters as in the original publications.



(d) Solvers with tuned parameters.

Figure C.2: Subproblem number and subproblem ratio profiles of SIP solvers: *RRHS*, *Oracle*, and *Hybrid*. "Out of the box performance" of the SIP solvers with their default parameters, i.e., the parameters of each solver are set to the values of their respective original publication, vs. SIP solvers with tuned parameters. Parameter values are listed in Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).

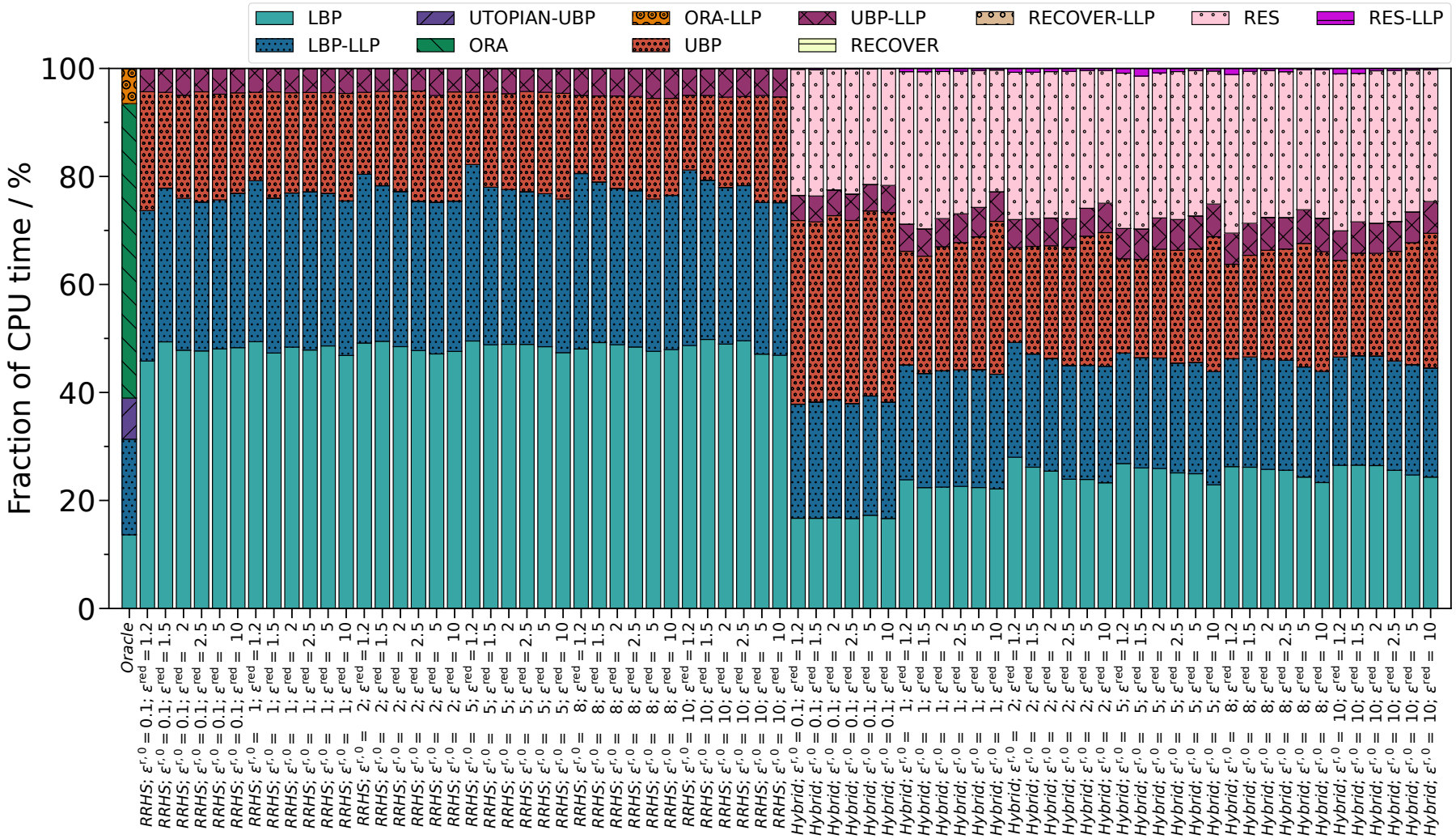
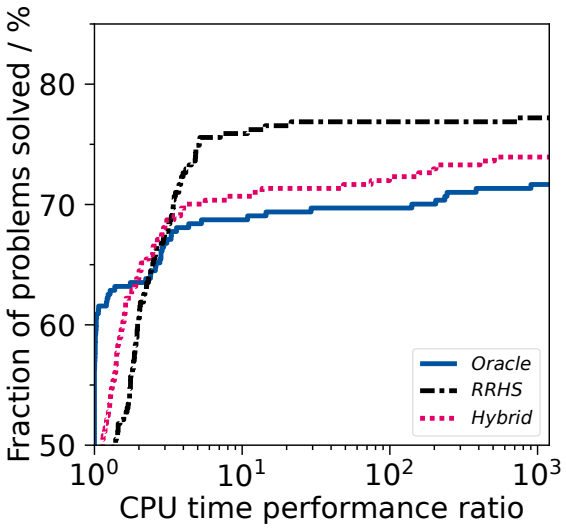
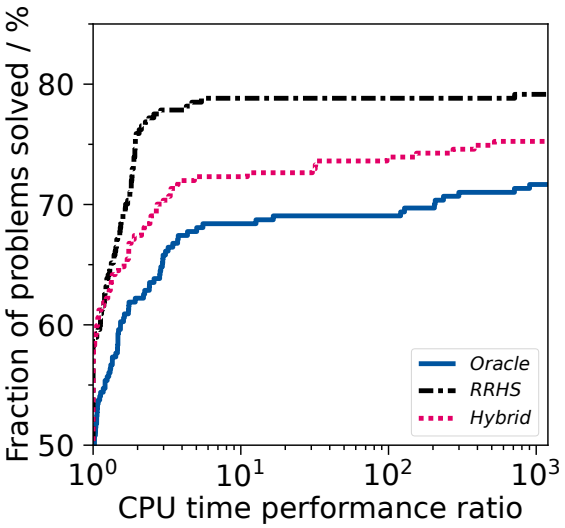


Figure C.3: Total percentage of CPU time spent in each subproblem. SIP solvers *Oracle*, *RRHS*, *Hybrid*. All solved problem instances are considered. LLPs are prefixed with their corresponding ULP, e.g., ULP-LLP. UTOPIAN-UBP is an auxiliary problem generated in *Oracle* to generate an initial UBD. RECOVER and RECOVER-LLP are auxiliary problems generated in (rare) situations in *RRHS* when ε^{r} is reduced below $\varepsilon^{\mathrm{r,min}}$. $\varepsilon^{\mathrm{r,min}}$ is set to the subsolver tolerance for inequality constraints (called `deltaIneq` in MAiNGO), see Table C.1. The respective fractions are too small to be discernible. Note: Adapted from Jungen et al. (2023a).

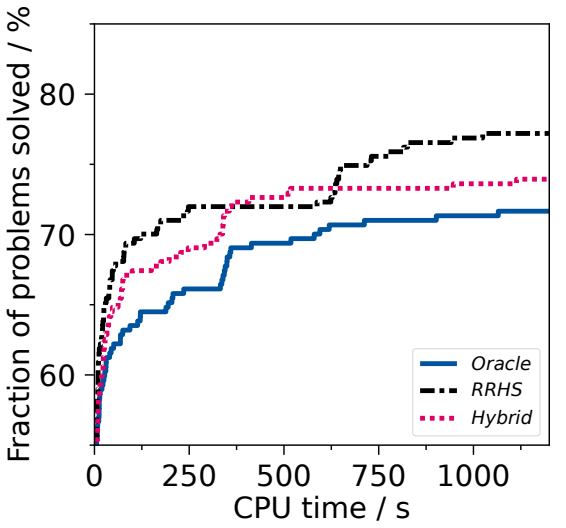
C.8.1.2 Considering Only Problem Instances With Purely Continuous Variables



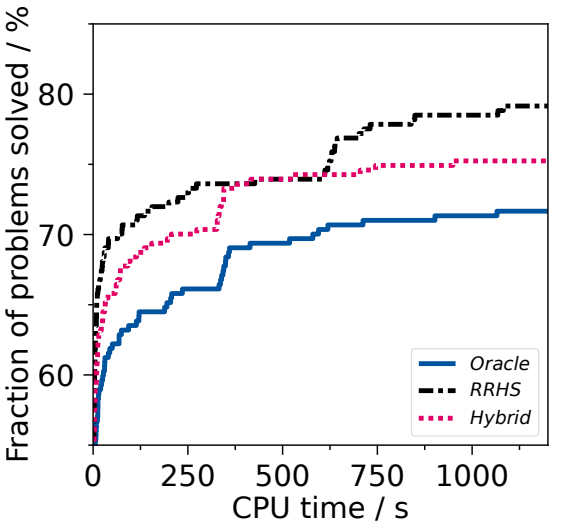
(a) "Out of the box performance;" parameters as in the original publications.



(b) Solvers with tuned parameters.

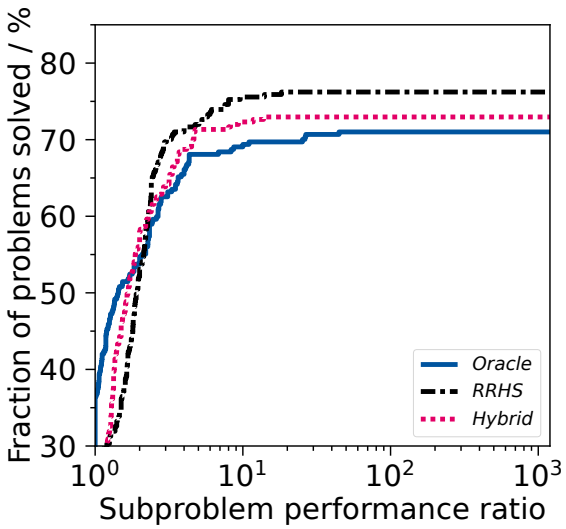


(c) "Out of the box performance"; parameters as in the original publications.

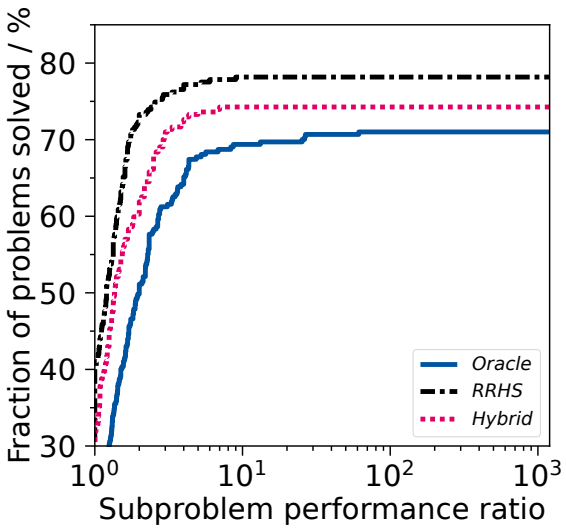


(d) Solvers with tuned parameters.

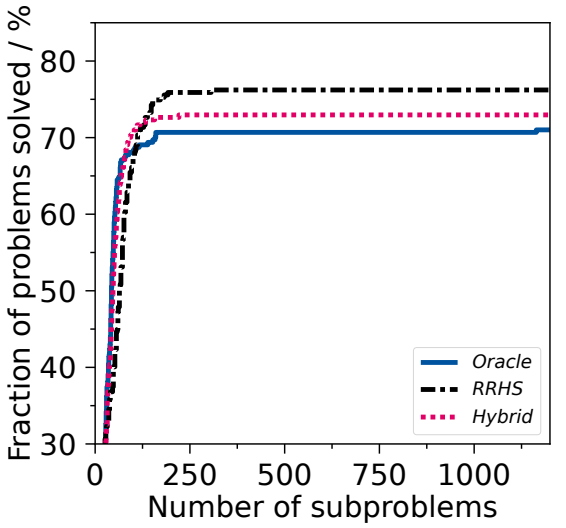
Figure C.4: CPU time performance ratio profiles and CPU time performance profiles of SIP solvers for problem instances with purely continuous variables in the upper and lower level: *RRHS*, *Oracle*, and *Hybrid*. "Out of the box performance" of the SIP solvers with their default parameters, i.e., the parameters of each solver are set to the values of their respective original publication, vs. SIP solvers with tuned parameters. Parameter values are listed in Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).



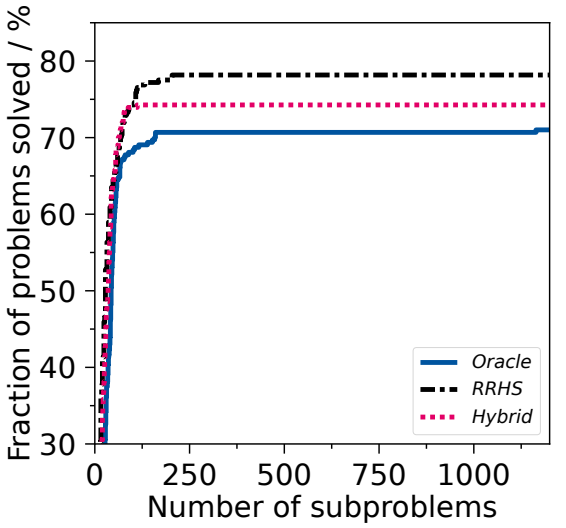
(a) "Out of the box performance"; parameters as in the original publications.



(b) Solvers with tuned parameters.



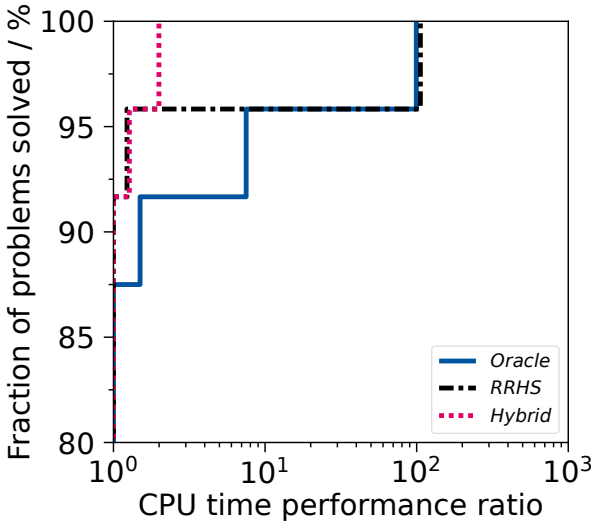
(c) "Out of the box performance"; parameters as in the original publications.



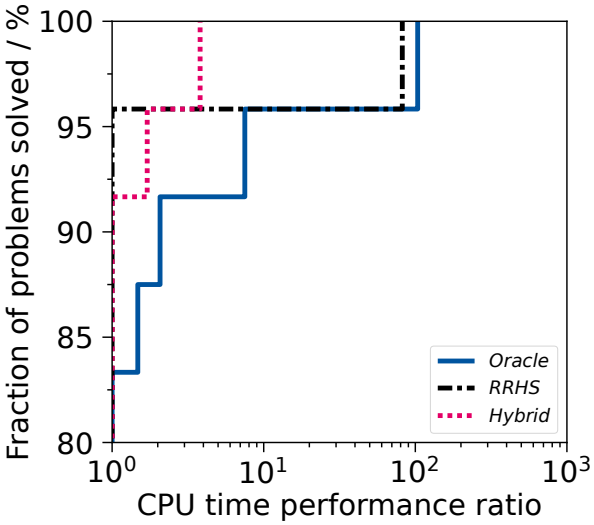
(d) Solvers with tuned parameters.

Figure C.5: Subproblem number and subproblem ratio profiles of SIP solvers for problem instances with purely continuous variables in the upper and lower level: *RRHS*, *Oracle*, and *Hybrid*. "Out of the box performance" of the SIP solvers with their default parameters, i.e., the parameters of each solver are set to the values of their respective original publication, vs. SIP solvers with tuned parameters. Parameter values are listed in Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).

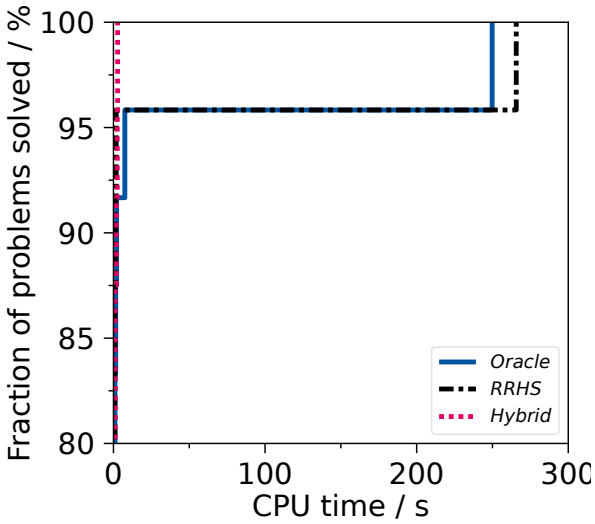
C.8.1.3 Considering Only Problem Instances With Discrete Variables in the Upper and/or Lower Level



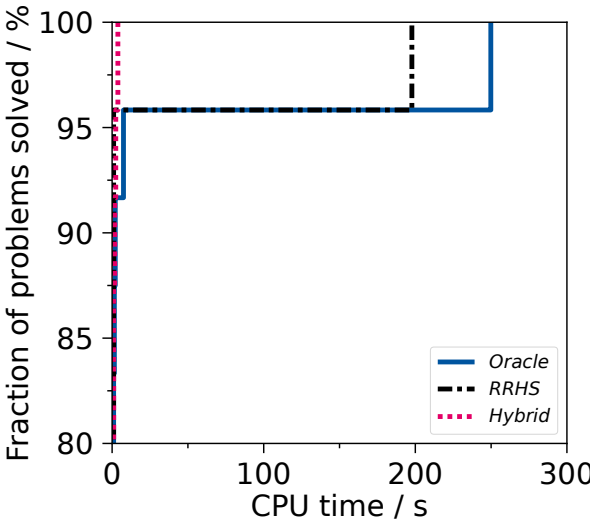
(a) "Out of the box performance;" parameters as in the original publications.



(b) Solvers with tuned parameters.



(c) "Out of the box performance"; parameters as in the original publications.



(d) Solvers with tuned parameters.

Figure C.6: CPU time performance ratio profiles and CPU time performance profiles of SIP solvers for problem instances with discrete variables in the upper and/or lower level: *RRHS*, *Oracle*, and *Hybrid*. "Out of the box performance" of the SIP solvers with their default parameters, i.e., the parameters of each solver are set to the values of their respective original publication, vs. SIP solvers with tuned parameters. Parameter values are listed in Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).

C.8.2 Additional GSIP Performance Profiles

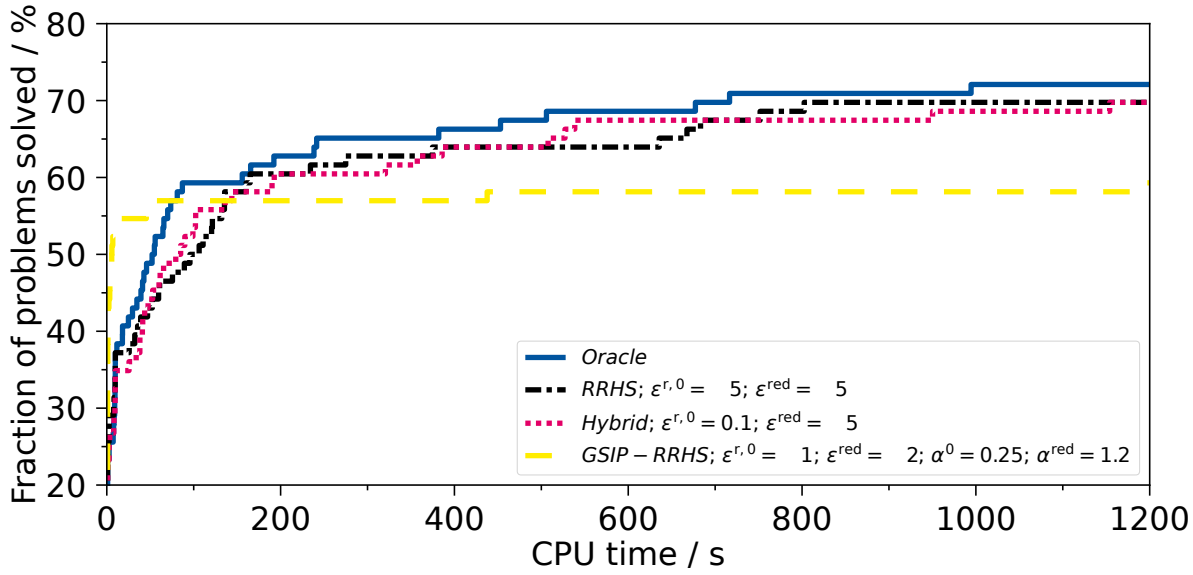
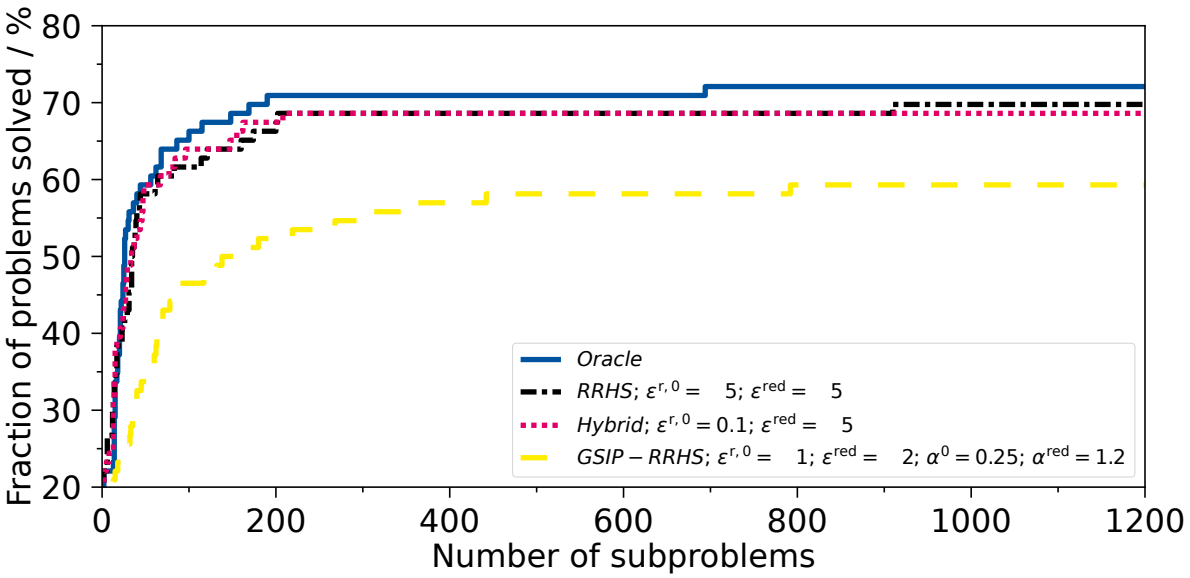
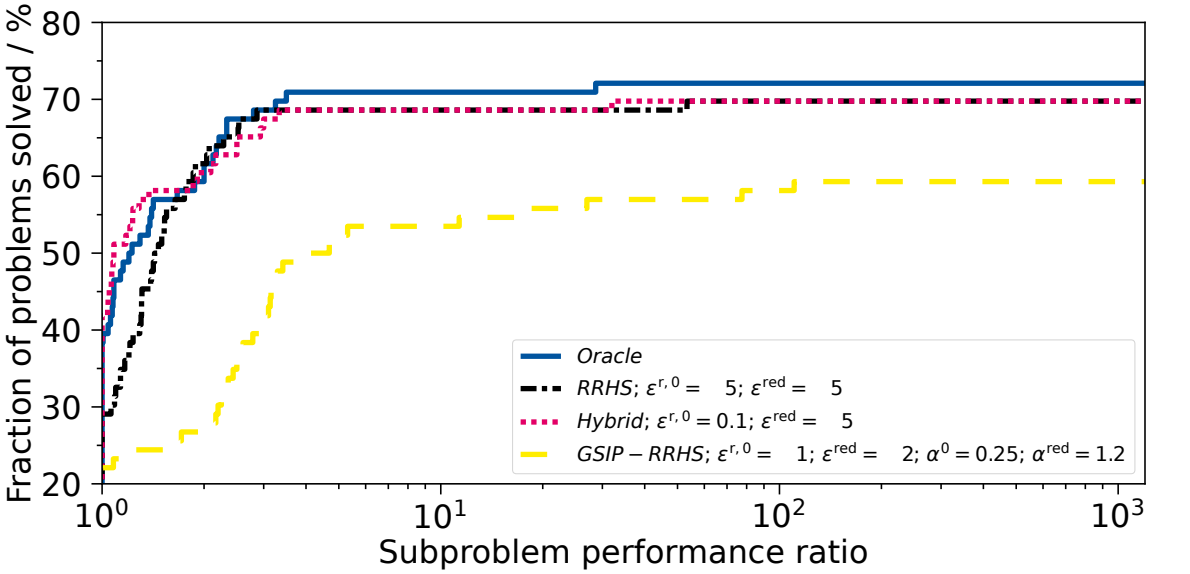


Figure C.7: CPU time performance profile of GSIP solvers *GSIP-RRHS*, *Oracle*, *RRHS*, *Hybrid*. Tuned parameters according to Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).



(a) Subproblem number performance profile.



(b) Subproblem performance ratio profiles.

Figure C.8: Subproblem number and subproblem ratio performance profiles of GSIP solvers *GSIP-RRHS*, *Oracle*, *RRHS*, *Hybrid*. Tuned parameters according to Table 3.7 in Section 3.6.2. Note: Adapted from Jungen et al. (2023a).

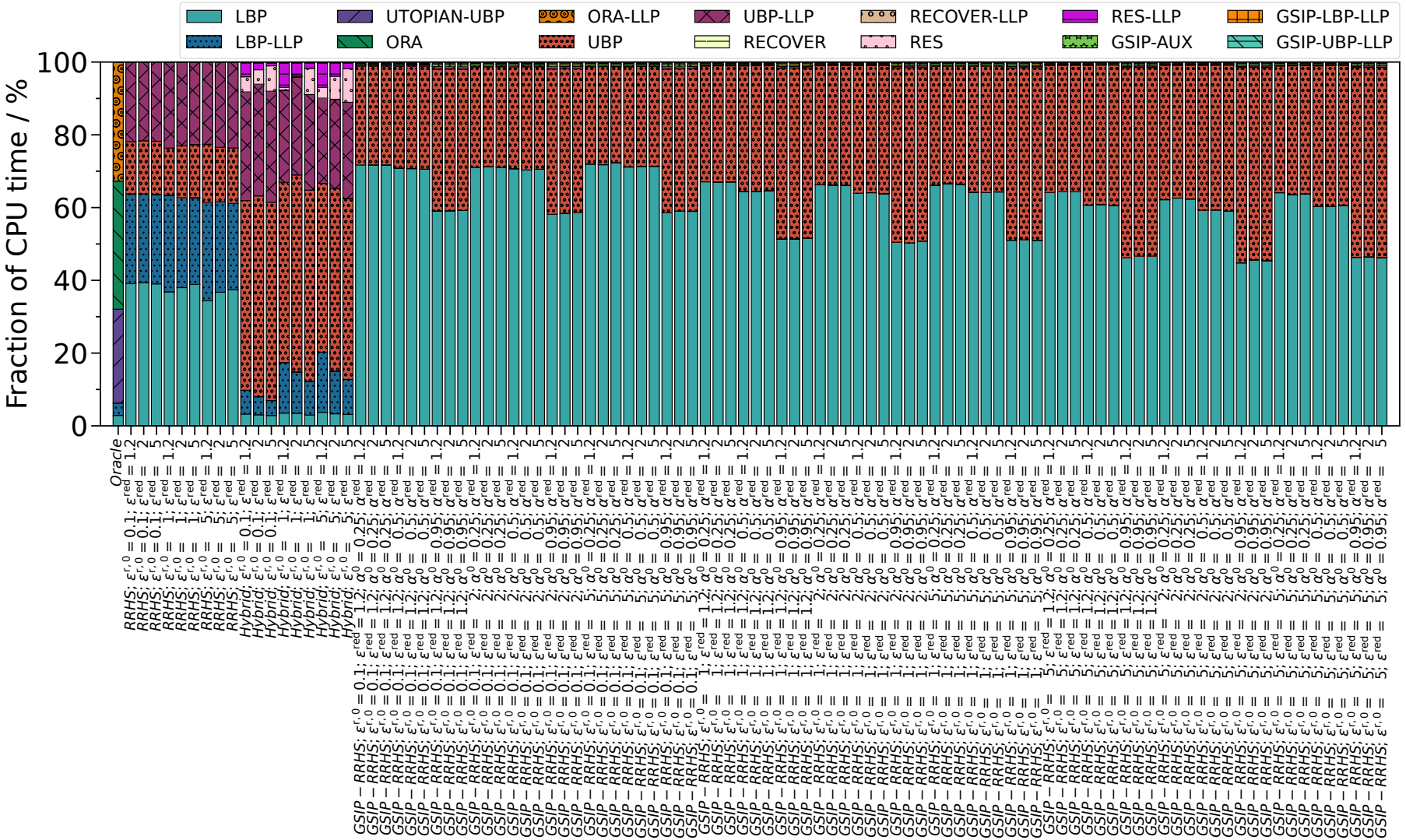


Figure C.9: Total percentage of CPU time spent in each subproblem. GSIP solvers *GSIP-RRHS*, *Oracle*, *RRHS*, *Hybrid*. All solved problem instances are considered. LLPs are prefixed with their corresponding ULP, e.g., ULP-LLP. UTOPIAN-UBP is an auxiliary problem generated in *Oracle* to generate an initial UBD. RECOVER is an auxiliary problem generated in *Oracle*. Note: Adapted from Jungen et al. (2023a).

C.8.3 Additional BLP Performance Profiles

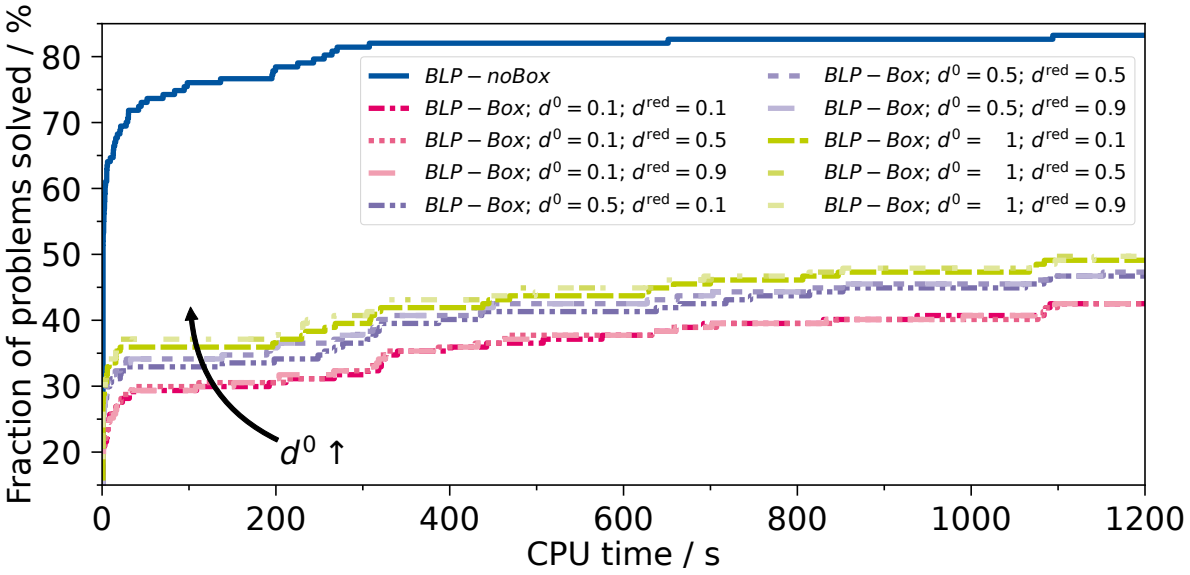
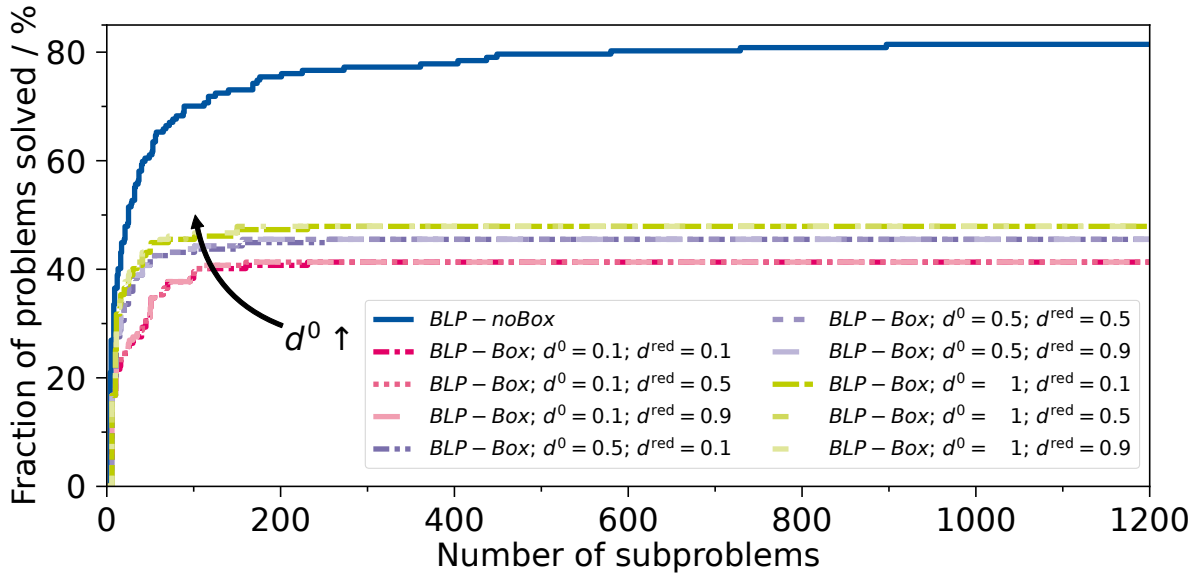
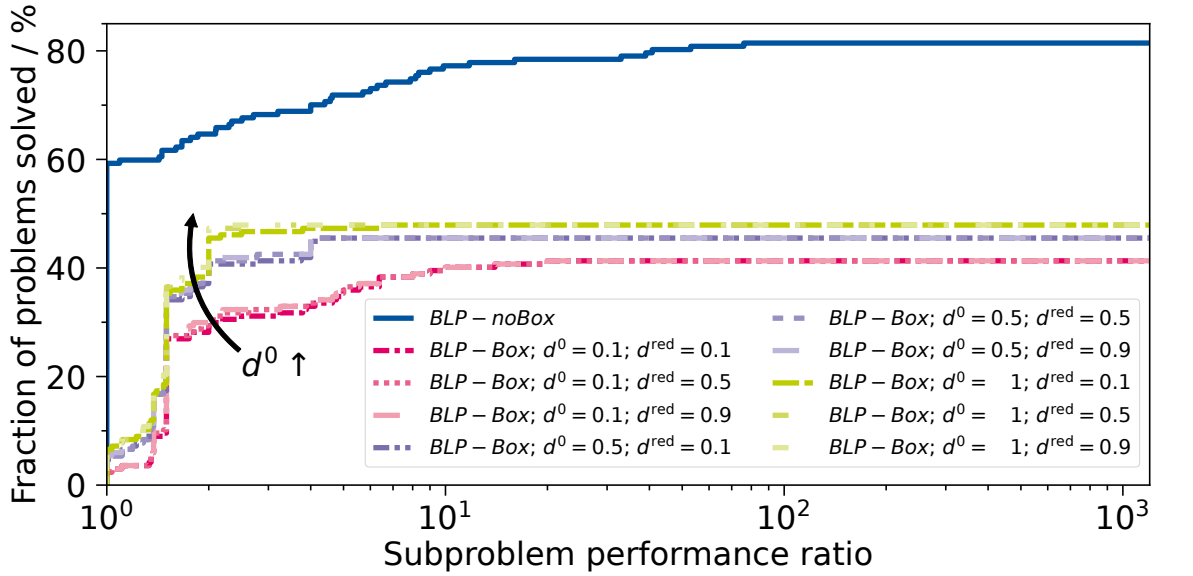


Figure C.10: CPU time performance profile of BLP solvers *BLP-Box* and *BLP-noBox*. Note: Adapted from Jungen et al. (2023a).



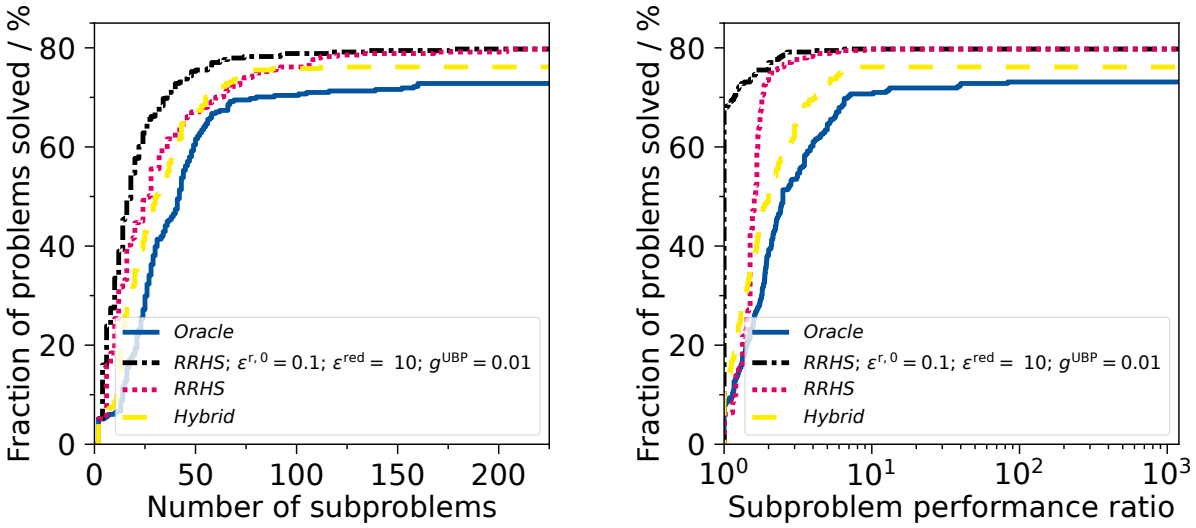
(a) Subproblem number performance profile.



(b) Subproblem performance ratio profiles.

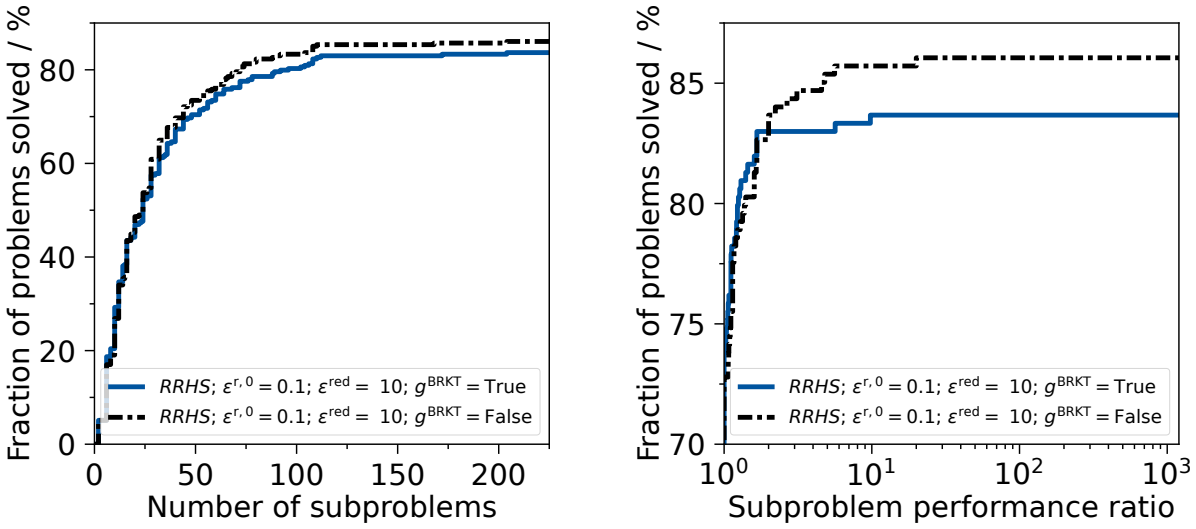
Figure C.11: Subproblem number and subproblem ratio performance profiles of BLP solvers *BLP-Box* and *BLP-noBox*. Note: Adapted from Jungen et al. (2023a).

C.8.4 Additional Performance Profiles of the Testing of Scientific Hypothesis



(a) Subproblem number performance profile. **(b)** Subproblem number performance profile.

Figure C.12: Subproblem number and subproblem ratio performance profiles for the introduced guard parameter as $g^{\mathrm{UBP}} = 0.01$. Comparison with "best" parameters of *RRHS*. Note: Adapted from Jungen et al. (2023a).



(a) Subproblem number performance profile. **(b)** Subproblem performance ratio profile.

Figure C.13: Subproblem number and subproblem ratio performance profiles comparing the use of bracketing (g^{BRKT}=True) against the default of not using it (g^{BRKT}=False). Comparison with "best" parameters of *RRHS*. Problems from Cerulli et al. (2022) are excluded. Note: Adapted from Jungen et al. (2023a).

C.9 Implemented Problem Instances

Table C.2: List of SIP problem instances with references that have the same or a similar problem.

Problem Name	Source	Also Appears in	Appears Similarly in
Anderson_1989-1	Anderson and Lewis (1989)	Betro (2004), Ferris and Philpott (1989), Goberna and López (1998), Leon and Vercher (1992), Vaz et al. (2004b)	
Anderson_1989-2	Anderson and Lewis (1989)	Liu and Wang (2013), Vaz et al. (2004b)	
Barragan_2022-1	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-2	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-3	Barragán and Camacho-Vallejo (2022)		
BenTal_1999-1-n50	Ben-Tal and Nemirovski (1999)	Stein (2003)	
Betro_2004-10	Betro (2004)		Betro (2004)
Betro_2004-11	Betro (2004)		Betro (2004)
Betro_2004-12	Betro (2004)		Betro (2004), Glashoff and Gustafson (1983)
Betro_2004-13	Betro (2004)		Betro (2004), Glashoff and Gustafson (1983)
Betro_2004-2	Betro (2004)		Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Betro_2004-6	Betro (2004)		Betro (2004)
Betro_2004-7	Betro (2004)		Betro (2004)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Betro_2004-8	Betro (2004)		Betro (2004)
Betro_2004-9	Betro (2004)		
Bhattacharjee_2005_1-H	Bhattacharjee et al. (2005b)	Djelassi (2020), Djelassi and Mitsos (2017), Lemonidis (2008), Mitsos (2009, 2011)	
Bhattacharjee_2005_2-6	Bhattacharjee et al. (2005a)		Pang et al. (2016), Watson (1983)
Bhattacharjee_2005_2-S-n3	Bhattacharjee et al. (2005a)		Bhattacharjee et al. (2005a), Price and Coope (1996)
Bhattacharjee_2005_2-S-n4	Bhattacharjee et al. (2005a)		Bhattacharjee et al. (2005a), Price and Coope (1996)
Bhattacharjee_2005_2-S-n5	Bhattacharjee et al. (2005a)		Bhattacharjee et al. (2005a), Price and Coope (1996)
Bhattacharjee_2005_2-S-n6	Bhattacharjee et al. (2005a)		Bhattacharjee et al. (2005a), Price and Coope (1996)
Blankenship_1976-3.1	Blankenship and Falk (1976)	Vaz et al. (2004b)	
Blankenship_1976-3.2	Blankenship and Falk (1976)	Vaz et al. (2004b)	
Blankenship_1976-3.3	Blankenship and Falk (1976)	Vaz et al. (2004b)	Blankenship and Falk (1976)
Blankenship_1976-3.3-corrected	Blankenship and Falk (1976)		Blankenship and Falk (1976)
Bonnans_2000-5-101	Bonnans and Shapiro (2000)		
Bonnans_2000-5-102	Bonnans and Shapiro (2000)		
Bonnans_2000-5-103	Bonnans and Shapiro (2000)		
Bonnans_2000-5-122u0	Bonnans and Shapiro (2000)		
Canovas_1999-6-1	Cánovas et al. (1999)		
Canovas_1999-6-2	Cánovas et al. (1999)		
Canovas_1999-6-3	Cánovas et al. (1999)		
Canovas_1999-6-4	Cánovas et al. (1999)		

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Canovas_1999-6-5	Cánovas et al. (1999)		Cánovas et al. (1999)
Canovas_1999-6-6	Cánovas et al. (1999)		Cánovas et al. (1999)
Canovas_1999-6-9	Cánovas et al. (1999)		
Cerulli_2022-PSD-inst1	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst10	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst2	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst3	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst4	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst5	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst6	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst7	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst8	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-PSD-inst9	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-jean-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-jean-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel4-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel4-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel5-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel5-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel6-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel6-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel7-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-myciel7-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst1	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst10	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst2	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst3	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst4	Cerulli et al. (2022)		Cerulli et al. (2022)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Cerulli_2022-notPSD-inst5	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst6	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst7	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst8	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-notPSD-inst9	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen5-5-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen5-5-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen6-6-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen6-6-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen7-7-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen7-7-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen8-12-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen8-12-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen8-8-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen8-8-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen9-9-PSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Cerulli_2022-queen9-9-notPSD	Cerulli et al. (2022)		Cerulli et al. (2022)
Coope_1985-14	Coope and Watson (1985)	Price (1992), Tanaka et al. (1988a)	Pang et al. (2016), Qi et al. (2003)
Coope_1998-1-1	Coope and Price (1998)		
Coope_1998-1-2-n20	Coope and Price (1998)	Xu et al. (2013)	Betro (2004), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Diehl_2013-5-1	Diehl et al. (2013)		
Fang_1994-1-n20	Fang et al. (1994)		

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Fang_1994-1-n50	Fang et al. (1994)		
Fang_1994-2-n20	Fang et al. (1994)		
Fang_1994-2-n50	Fang et al. (1994)		
Fang_1994-3-n20	Fang et al. (1994)		
Fang_1994-3-n50	Fang et al. (1994)		
Floudas_2008-CA	Floudas and Stein (2008)		
Floudas_2008-DC	Floudas and Stein (2008)		
Glashoff_1983-1-3-12	Glashoff and Gustafson (1983)	Goberna and López (1998), Leon and Vercher (1992)	
Glashoff_1983-2-4-7	Glashoff and Gustafson (1983)		
Glashoff_1983-2-5-2	Glashoff and Gustafson (1983)		
Glashoff_1983-2-5-5	Glashoff and Gustafson (1983)		
Glashoff_1983-2-5-8	Glashoff and Gustafson (1983)		
Glashoff_1983-4-8-12	Glashoff and Gustafson (1983)		
Glashoff_1983-4-8-13	Glashoff and Gustafson (1983)		
Glashoff_1983-5-12-40	Glashoff and Gustafson (1983)		
Glashoff_1983-5-13-13a	Glashoff and Gustafson (1983)		
Glashoff_1983-5-13-13b	Glashoff and Gustafson (1983)	Goberna and López (1998), Leon and Vercher (1992)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Glashoff_1983-7-16-12	Glashoff and Gustafson (1983)	Betro (2004), Guo and Sun (2020)	Betro (2004)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Glashoff_1983-7-16-14	Glashoff and Gustafson (1983)	Guo and Sun (2020)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Glashoff_1983-8-19-22	Glashoff and Gustafson (1983)		
Goberna_1998-1-1	Goberna and López (1998)	Goberna and López (1998)	
Goberna_1998-1-2	Goberna and López (1998)	Goberna and López (1998)	
Goberna_1998-10-13	Goberna and López (1998)		
Goberna_1998-10-2	Goberna and López (1998)		
Goberna_1998-11-1	Goberna and López (1998)		
Goberna_1998-11-2	Goberna and López (1998)		
Goberna_1998-11-2-n1	Goberna and López (1998)		Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Goberna_1998-11-2-n2	Goberna and López (1998)		Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Goberna_1998-11-2-n7	Goberna and López (1998)		Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Goberna_1998-11-3	Goberna and López (1998)		
Goberna_1998-11-8	Goberna and López (1998)		
Goberna_1998-11-9	Goberna and López (1998)		
Goberna_1998-12-10	Goberna and López (1998)		
Goberna_1998-12-11	Goberna and López (1998)		
Goberna_1998-2-1	Goberna and López (1998)		
Goberna_1998-2-10	Goberna and López (1998)		
Goberna_1998-2-7	Goberna and López (1998)		
Goberna_1998-2-8	Goberna and López (1998)		
Goberna_1998-4-14	Goberna and López (1998)		
Goberna_1998-7-1	Goberna and López (1998)		
Goberna_1998-7-9	Goberna and López (1998)	Goberna and López (1998)	
Goberna_1998-8-10	Goberna and López (1998)		
Goberna_1998-8-13	Goberna and López (1998)		
Goberna_1998-8-14	Goberna and López (1998)		
Goberna_1998-8-3	Goberna and López (1998)		
Goberna_1998-8-5	Goberna and López (1998)		
Goberna_1998-8-7	Goberna and López (1998)		
Goberna_1998-8-8	Goberna and López (1998)		
Goberna_1998-9-1	Goberna and López (1998)		
Goberna_1998-P	Goberna and López (1998)		
Grossmann_1978-pipe	Grossmann and Sargent (1978)	Žaković and Rustem (2003)	
Guo_2020-3-19	Guo and Sun (2020)		

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Guo_2020-3-20	Guo and Sun (2020)		
Guo_2020-3-7	Guo and Sun (2020)		
Guo_2020-3-8-i	Guo and Sun (2020)		
Guo_2020-3-8-ii	Guo and Sun (2020)		
Guo_2020-4-2-i	Guo and Sun (2020)		
Guo_2020-4-2-i-ref	Guo and Sun (2020)		
Guo_2020-4-2-ii	Guo and Sun (2020)		
Guo_2020-4-2-ii-ref	Guo and Sun (2020)		
Guo_2020-4-7	Guo and Sun (2020)		
Guo_2023-5-1-1	Guo and Zhang (2023)		
Guo_2023-5-1-2	Guo and Zhang (2023)		
Guo_2023-5-1-3	Guo and Zhang (2023)		
Guo_2023-5-1-4	Guo and Zhang (2023)		
Guo_2023-5-2-16	Guo and Zhang (2023)		
Guo_2023-5-3-10-6	Guo and Zhang (2023)		
Guo_2023-5-3-10-8	Guo and Zhang (2023)		
Guo_2023-5-3-12-6	Guo and Zhang (2023)		
Guo_2023-5-3-12-8	Guo and Zhang (2023)		
Guo_2023-5-3-16-4	Guo and Zhang (2023)		
Guo_2023-5-3-20-4	Guo and Zhang (2023)		
Gustafson_1973-2-1	Gustafson and Kortanek (1973)		
Gustafson_1973-2-2	Gustafson and Kortanek (1973)		
Gustafson_1973-2-3	Gustafson and Kortanek (1973)		
Gustafson_1973-2-4	Gustafson and Kortanek (1973)		

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Gustafson_1979-2	Gustafson (1979)		
Gustafson_1979-3-e0	Gustafson (1979)		
Harwood_2019-CEx1	Harwood (2019)		
Harwood_2019-CEx2	Harwood (2019)		
Hettich_1979_2-1	Hettich and van Honstede (1979)		
Hettich_1982-3-1-17	Hettich and Zencke (1982)		
Hettich_1982-3-1-3	Hettich and Zencke (1982)		
Hettich_1982-3-1-5-n20	Hettich and Zencke (1982)		
Kostyukova_2010-3-6	Kostyukova and Tchemisova (2010)		
Kostyukova_2010-3-7	Kostyukova and Tchemisova (2010)		
Lemonidis_2008-1-11	Lemonidis (2008)		
Lemonidis_2008-1-12	Lemonidis (2008)		
Lemonidis_2008-1-13	Lemonidis (2008)		
Leon_1992-1	Leon and Vercher (1992)		
Leon_1992-2	Leon and Vercher (1992)		
Leon_1992-3	Leon and Vercher (1992)		
Leon_1992-5	Leon and Vercher (1992)		
Leon_1992-6-n5	Leon and Vercher (1992)	Goberna and López (1998)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Leon_1992-6-n7	Leon and Vercher (1992)	Goberna and López (1998)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Leon_1992-6-n9	Leon and Vercher (1992)	Goberna and López (1998)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Li_2004-7	Li et al. (2004)		
Lin_1994-1	Lin et al. (1994)		
Lin_1994-2	Lin et al. (1994)		Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Lin_1994-3	Lin et al. (1994)		
Lin_1994-4	Lin et al. (1994)		
Liu_1999-6-1	Liu et al. (1999)	Vaz et al. (2004b)	
Liu_1999-6-2	Liu et al. (1999)	Liu and Wang (2013), Vaz et al. (2004b)	
Liu_1999-6-3	Liu et al. (1999)	Vaz et al. (2004b)	Wu et al. (2005)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Liu_2013-1	Liu and Wang (2013)		
Liu_2013-2	Liu and Wang (2013)		
Liu_2013-3	Liu and Wang (2013)		
Liu_2013-4	Liu and Wang (2013)		
LoBianco_2001-2	Lo Bianco and Piazzi (2001)		
LoBianco_2001-3	Lo Bianco and Piazzi (2001)		
Lopez_2007-1	López and Still (2007)		
Lopez_2007-2	López and Still (2007)		
Lopez_2007-3-A	López and Still (2007)		
Mehrotra_2014-1	Mehrotra and Papp (2014)	Pang et al. (2016)	Tichatschke and Nebeling (1988)
Mehrotra_2014-2-17	Mehrotra and Papp (2014)		Mehrotra and Papp (2014)
Mehrotra_2014-2-18	Mehrotra and Papp (2014)		Mehrotra and Papp (2014)
Mitsos_2008-5-1	Mitsos et al. (2008b)		
Mitsos_2009-4-3	Mitsos (2009)	Pang and Wu (2022)	Mitsos (2009)
Mitsos_2009-4-3-impl	Mitsos (2009)	Mitsos (2011)	Mitsos (2009)
Mitsos_2009-4-6	Mitsos (2009)		Mitsos (2009)
Mitsos_2009-4-6-impl	Mitsos (2009)	Mitsos (2011)	Mitsos (2009)
Mitsos_2009-D1-0-1	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-D1-0-2	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-D1-1-1	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-D1-1-2	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-D2-0-1	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Mitsos_2009-D2-0-2	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-D2-1-1	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-D2-1-2	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Mitsos_2009-DP	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017)	
Mitsos_2009-IP1	Mitsos (2009)		Mitsos (2009)
Mitsos_2009-IP1-corrected	Mitsos (2009)	Mitsos (2011)	Mitsos (2009)
Mitsos_2009-MINLP1	Mitsos (2009)	Mitsos (2011)	
Mitsos_2009-MINLP2	Mitsos (2009)	Mitsos (2011)	
Mitsos_2009-S	Mitsos (2009)	Djelassi (2020), Djelassi and Mitsos (2017), Mitsos (2011)	
Pang_2016-1	Pang et al. (2016)		Coope and Watson (1985), Qi et al. (2003)
Pang_2016-2	Pang et al. (2016)		Watson (1983)
Pang_2016-3	Pang et al. (2016)		Bhattacharjee et al. (2005a), Watson (1983)
Pang_2016-4	Pang et al. (2016)		
Pang_2016-5	Pang et al. (2016)	Pang and Wu (2022)	Pang et al. (2016), Watson (1983)
Pang_2016-6	Pang et al. (2016)	Pang and Wu (2022)	Pang et al. (2016), Watson (1983)
Pang_2016-7	Pang et al. (2016)	Pang and Wu (2022)	
Pang_2016-8	Pang et al. (2016)		
Pang_2022-16	Pang and Wu (2022)		
Pang_2022-17	Pang and Wu (2022)		

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Pang_2022-18	Pang and Wu (2022)		
Pang_2022-19	Pang and Wu (2022)		
Pang_2022-20	Pang and Wu (2022)		
Pang_2022-21	Pang and Wu (2022)		
Pang_2022-22	Pang and Wu (2022)		
Pang_2022-23	Pang and Wu (2022)		
Polak_1999-1	Polak et al. (1999)	Vaz et al. (2004b)	
Polak_1999-2	Polak et al. (1999)	Vaz et al. (2004b)	
Powell_1989-6-3	Powell (1989)	Liu and Wang (2013), Todd (1994), Vaz et al. (2004b)	
Price_1992-4-n4	Price (1992)	Görner (1997)	Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Price_1992-4-n5	Price (1992)		Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Price_1992-K	Price (1992)	Bhattacharjee et al. (2005a), Liu and Wang (2013), Price and Coope (1996), Vaz et al. (2004b)	
Price_1996-L	Price and Coope (1996)	Bhattacharjee et al. (2005a), Teo et al. (2000), Vaz et al. (2004b), Yu et al. (2010)	
Price_1996-M	Price and Coope (1996)	Bhattacharjee et al. (2005a), Vaz et al. (2004b)	

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Price_1996-N	Price and Coope (1996)	Bhattacharjee et al. (2005a), Djelassi (2020), Djelassi and Mitsos (2017), Lemonidis (2008), Mitsos (2009, 2011), Mitsos et al. (2008b), Vaz et al. (2004b)	
Price_1996-S-p3	Price and Coope (1996)	Liu and Wang (2013), Vaz et al. (2004b)	Bhattacharjee et al. (2005a), Price and Coope (1996)
Price_1996-S-p4	Price and Coope (1996)	Liu and Wang (2013), Vaz et al. (2004b)	Bhattacharjee et al. (2005a), Price and Coope (1996)
Price_1996-S-p5	Price and Coope (1996)	Liu and Wang (2013), Vaz et al. (2004b)	Bhattacharjee et al. (2005a), Price and Coope (1996)
Price_1996-S-p6	Price and Coope (1996)	Coope and Price (1998), Liu and Wang (2013), Vaz et al. (2004b)	Bhattacharjee et al. (2005a), Price and Coope (1996)
Price_1996-T-p3	Price and Coope (1996)	Liu and Wang (2013), Vaz et al. (2004b)	Price and Coope (1996)
Price_1996-T-p4	Price and Coope (1996)		Price and Coope (1996)
Price_1996-T-p5	Price and Coope (1996)		Price and Coope (1996)
Price_1996-T-p6	Price and Coope (1996)	Coope and Price (1998)	Price and Coope (1996)
Price_1996-U	Price and Coope (1996)	Bhattacharjee et al. (2005a), Coope and Price (1998), Liu and Wang (2013), Vaz et al. (2004b)	
Qi_2003-1	Qi et al. (2003)	Su et al. (2018)	Coope and Watson (1985), Pang et al. (2016)
RudnickCohen_2020-1	Rudnick-Cohen et al. (2020)		
RudnickCohen_2020-2	Rudnick-Cohen et al. (2020)		

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
RudnickCohen_2020-4	Rudnick-Cohen et al. (2020)		
Rueckmann_2009-1	Rückmann and Shapiro (2009)		
Rueckmann_2009-2	Rückmann and Shapiro (2009)		
Rueckmann_2009-3	Rückmann and Shapiro (2009)		
Schwientek_2021-1	Schwientek et al. (2021)		Caspari (2016), Jongen et al. (1998), Lemonidis (2008), Mitsos and Tsoukalas (2015)
Schwientek_2021-10	Schwientek et al. (2021)		Rückmann and Stein (2001)
Schwientek_2021-11	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-12	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-13	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-14	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-15	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-2	Schwientek et al. (2021)		Jongen et al. (1998)
Schwientek_2021-3	Schwientek et al. (2021)		Rückmann and Shapiro (2001)
Schwientek_2021-6	Schwientek et al. (2021)		Rückmann and Shapiro (2001)
Schwientek_2021-7	Schwientek et al. (2021)		
Schwientek_2021-8	Schwientek et al. (2021)		Lemonidis (2008)
Seidel_2019-2.1	Seidel and Küfer (2022)		
Seidel_2019-2.3	Seidel and Küfer (2022)		
Seidel_2019-5	Seidel and Küfer (2022)		
Spjotvold_2005-4	Spjøtvold et al. (2005)		
Stein_2012-DC	Stein and Steuermann (2012)		
Still_2001_2-1	Still (2001b)	Liu and Wang (2013), Vaz et al. (2004b)	
Streit_1982-A8	Streit and Nuttall (1982)	Kortanek and No (1993)	

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Su_2018-4-n15	Su et al. (2018)		Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Su_2018-4-n20	Su et al. (2018)		Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Tanaka_1988-5-n10	Tanaka et al. (1988a)	Görner (1997), Price (1992)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Tanaka_1988-5-n12	Tanaka et al. (1988a)	Görner (1997), Price (1992)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Tanaka_1988-5-n15	Tanaka et al. (1988a)	Görner (1997), Price (1992)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Tanaka_1988-5-n8	Tanaka et al. (1988a)	Price (1992)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983), Wu et al. (2005)
Tanaka_1988-8-n10	Tanaka et al. (1988a)	Görner (1997), Price (1992)	Tanaka et al. (1988a), Watson (1983)
Tanaka_1988-8-n15	Tanaka et al. (1988a)	Görner (1997), Price (1992)	Tanaka et al. (1988a), Watson (1983)
Tanaka_1988-Ex1	Tanaka et al. (1988a)		
Teo_2000-6-2	Teo et al. (2000)	Yu et al. (2010)	
Tichatschke_1988-1	Tichatschke and Nebeling (1988)	Kortanek and No (1993), Vaz et al. (2004b)	Mehrotra and Papp (2014), Tichatschke and Nebeling (1988)
Tichatschke_1988-2	Tichatschke and Nebeling (1988)		Mehrotra and Papp (2014), Tichatschke and Nebeling (1988)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Tichatschke_1988-3	Tichatschke and Nebeling (1988)	Kortanek and No (1993), Pang et al. (2016), Vaz et al. (2004b), Zhang et al. (2010)	
Tsoukalas_2011-2-1	Tsoukalas and Rustem (2011)		
Vaz_2004-1	Vaz et al. (2004b)		
Watson_1975-1-n16	Watson (1975)		Watson (1975)
Watson_1975-1-n9	Watson (1975)		Watson (1975)
Watson_1975-2-n9	Watson (1975)		
Watson_1975-3-n9	Watson (1975)		
Watson_1975-4-n9	Watson (1975)		
Watson_1975-5-n16	Watson (1975)		Watson (1975)
Watson_1975-5-n25	Watson (1975)		Watson (1975)
Watson_1975-5-n9	Watson (1975)		Watson (1975)
Watson_1975-6-n16	Watson (1975)		Watson (1975)
Watson_1975-6-n25	Watson (1975)		Watson (1975)
Watson_1975-6-n9	Watson (1975)		Watson (1975)
Watson_1975-7-n16	Watson (1975)		Watson (1975)
Watson_1975-7-n25	Watson (1975)		Watson (1975)
Watson_1975-7-n9	Watson (1975)		Watson (1975)
Watson_1983-1	Watson (1983)	Bhattacharjee et al. (2005a,b), Coope and Watson (1985), Lemonidis (2008), Liu and Wang (2013), Tanaka et al. (1988a), Vaz et al. (2004b)	

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Watson_1983-2	Watson (1983)	Bhattacharjee et al. (2005a,b), Coope and Watson (1985), Djelassi (2020), Djelassi and Mitsos (2017), Görner (1997), Lemonidis (2008), Li et al. (2004), Liu and Wang (2013), Mitsos (2009, 2011), Mitsos et al. (2008b), Pang and Wu (2022), Price (1992), Qi et al. (2003), Su et al. (2018), Tanaka et al. (1988a), Vaz et al. (2004b)	Pang et al. (2016)
Watson_1983-3	Watson (1983)	Bhattacharjee et al. (2005a,b), Coope and Watson (1985), Görner (1997), Kortanek and No (1993), Lemonidis (2008), Li et al. (2004), Liu and Wang (2013), Pang and Wu (2022), Pang et al. (2016), Price (1992), Qi et al. (2003), Schittkowski (1992), Tanaka et al. (1988a,b), Vaz et al. (2004b)	Pang et al. (2016)
Watson_1983-4-n3	Watson (1983)	Bhattacharjee et al. (2005a), Coope and Watson (1985), Görner (1997), Leon and Vercher (1992), Liu and Wang (2013), Tanaka et al. (1988a,b), Vaz et al. (2004b)	Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Watson_1983-4-n6	Watson (1983)	Bhattacharjee et al. (2005a), Coope and Watson (1985), Goberna and López (1998), Görner (1997), Leon and Vercher (1992), Liu and Wang (2013), Price (1992), Tanaka et al. (1988a,b), Vaz et al. (2004b)	Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Watson_1983-4-n8	Watson (1983)	Betro (2004), Bhattacharjee et al. (2005a), Coope and Watson (1985), Goberna and López (1998), Görner (1997), Leon and Vercher (1992), Liu and Wang (2013), Price (1992), Tanaka et al. (1988a,b), Vaz et al. (2004b)	Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Watson_1983-5	Watson (1983)	Bhattacharjee et al. (2005a), Coope and Watson (1985), Djelassi (2020), Djelassi and Mitsos (2017), Görner (1997), Lemonidis (2008), Liu and Wang (2013), Lo Bianco and Piazzi (2001), Mitsos (2009, 2011), Mitsos et al. (2008b), Pang and Wu (2022), Schittkowski (1992), Tanaka et al. (1988a,b), Tsoukalas and Rustem (2011), Vaz et al. (2004b)	Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Wu et al. (2005)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Watson_1983-6	Watson (1983)	Coope and Watson (1985), Djelassi (2020), Djelassi and Mitsos (2017), Görner (1997), Lemonidis (2008), Li et al. (2004), Liu and Wang (2013), Mitsos (2009, 2011), Mitsos et al. (2008b), Price (1992), Qi et al. (2003), Su et al. (2018), Tanaka et al. (1988a), Vaz et al. (2004b)	Bhattacharjee et al. (2005a), Pang et al. (2016)
Watson_1983-7	Watson (1983)	Bhattacharjee et al. (2005a), Coope and Watson (1985), Djelassi (2020), Djelassi and Mitsos (2017), Görner (1997), Guo and Sun (2020), Lemonidis (2008), Liu and Wang (2013), Mitsos (2009, 2011), Mitsos et al. (2008b), Price (1992), Schittkowski (1992), Tanaka et al. (1988a,b), Vaz et al. (2004b)	
Watson_1983-8	Watson (1983)		Tanaka et al. (1988a), Watson (1983)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Watson_1983-8-corrected	Watson (1983)	Bhattacharjee et al. (2005a), Coope and Watson (1985), Djelassi (2020), Djelassi and Mitsos (2017), Görner (1997), Lemonidis (2008), Liu and Wang (2013), Mitsos (2009, 2011), Mitsos et al. (2008b), Tanaka et al. (1988a), Vaz et al. (2004b)	Tanaka et al. (1988a), Watson (1983)
Watson_1983-9	Watson (1983)	Bhattacharjee et al. (2005a), Coope and Watson (1985), Djelassi (2020), Djelassi and Mitsos (2017), Lemonidis (2008), Liu and Wang (2013), Mitsos (2009, 2011), Mitsos et al. (2008b), Tanaka et al. (1988a,b), Vaz et al. (2004b)	
Wu_2005-6-1	Wu et al. (2005)		Liu et al. (1999), Wu et al. (2005)
Wu_2005-6-2	Wu et al. (2005)		Liu et al. (1999), Wu et al. (2005)
Wu_2005-6-3	Wu et al. (2005)		Betro (2004), Glashoff and Gustafson (1983), Leon and Vercher (1992), Lin et al. (1994), Price (1992), Su et al. (2018), Tanaka et al. (1988a), Watson (1983)
Wu_2005-6-4	Wu et al. (2005)		Watson (1983)

Table C.2: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Xu_2013-11-n30	Xu et al. (2013)		Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Xu_2013-11-n40	Xu et al. (2013)		Betro (2004), Coope and Price (1998), Coope and Watson (1985), Goberna and López (1998), Price (1992), Watson (1983), Xu et al. (2013)
Xu_2013-2	Xu et al. (2013)		
Xu_2013-ex-2-n20	Xu et al. (2013)		Xu et al. (2013)
Xu_2013-ex-2-n30	Xu et al. (2013)		Xu et al. (2013)
Xu_2013-ex-2-n40	Xu et al. (2013)		Xu et al. (2013)
Zakovic_2003-17	Žaković and Rustem (2003)		
Zakovic_2003-18	Žaković and Rustem (2003)		
Zakovic_2003-19	Žaković and Rustem (2003)		
Zakovic_2003-20	Žaković and Rustem (2003)		
Zakovic_2003-21	Žaković and Rustem (2003)		
Zhang_2010-2	Zhang et al. (2010)	Pang et al. (2016)	
Zhou_1996-PT-SIP	Zhou and Tits (1996)	Liu and Wang (2013), Vaz et al. (2004b)	

Table C.3: List of MINMAX problem instances with references that have the same or a similar problem.

Problem Name	Source	Also Appears in	Appears Similarly in
Andreassen_1976-2	Andreassen and Watson (1976)		
Andreassen_1976-3	Andreassen and Watson (1976)		
Andreassen_1976-4	Andreassen and Watson (1976)		
Andreassen_1976-5	Andreassen and Watson (1976)		
Andreassen_1976-6	Andreassen and Watson (1976)		
Atkinson_1975-1	Atkinson and Federov (1975)	Duarte et al. (2015)	
Atkinson_1975-2	Atkinson and Federov (1975)	Duarte et al. (2015)	
Barrodale_1978-3-n3	Barrodale et al. (1978)	Zhang et al. (2010)	Barrodale et al. (1978)
Barrodale_1978-3-n5	Barrodale et al. (1978)	Kortanek and No (1993), Zhang et al. (2010)	Barrodale et al. (1978)
Barrodale_1978-3-n7	Barrodale et al. (1978)	Kortanek and No (1993), Vaz et al. (2004b), Zhang et al. (2010)	Barrodale et al. (1978)
Curtis_1966-1	Curtis and Powell (1966)	Andreassen and Watson (1976), Glashoff and Gustafson (1983), Görner (1997), Hettich (1979), Hettich and Zencke (1982), Osborne and Watson (1969), Zhou and Tits (1996)	
Diehl_2013-CA	Diehl et al. (2013)		
Duarte_2015-2	Duarte et al. (2015)		
Duarte_2015-3	Duarte et al. (2015)		Dette et al. (2012), Duarte et al. (2015)
Duarte_2015-4	Duarte et al. (2015)		Dette et al. (2012), Duarte et al. (2015)
Duarte_2015-5	Duarte et al. (2015)		
Duarte_2015-6	Duarte et al. (2015)		
Duarte_2015-7	Duarte et al. (2015)		

Table C.3: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Duarte_2015-8	Duarte et al. (2015)		
Ferris_1989-2.3	Ferris and Philpott (1989)		
Glashoff_1983-3-6-23	Glashoff and Gustafson (1983)		
Glashoff_1983-3-6-24	Glashoff and Gustafson (1983)		
Glashoff_1983-7-17-12	Glashoff and Gustafson (1983)		
Goerner_1997-19	Görner (1997)		
Goerner_1997-20	Görner (1997)		
Goerner_1997-21	Görner (1997)		Hettich and Zencke (1982)
Goerner_1997-22	Görner (1997)		
Guo_2020-3-19	Guo and Sun (2020)		
Guo_2020-3-8-i	Guo and Sun (2020)		
Guo_2020-3-8-ii	Guo and Sun (2020)		
Hettich_1976_1-1	Hettich (1976a)	Hettich (1976b, 1979)	Hettich and Zencke (1982)
Hettich_1979_1-5-mM	Hettich (1979)		
Hettich_1979_1-6-mM	Hettich (1979)		
Hettich_1982-3-3-1	Hettich and Zencke (1982)	Görner (1997), Zhou and Tits (1996)	Hettich and Zencke (1982)
Hettich_1982-3-3-1-mod	Hettich and Zencke (1982)		Hettich and Zencke (1982)
Hettich_1982-5-2-2	Hettich and Zencke (1982)		Hettich and Zencke (1982)
Hettich_1982-5-3-8	Hettich and Zencke (1982)		Hettich (1976a)
Hettich_1982-6-1-a-n4	Hettich and Zencke (1982)	Görner (1997)	Hettich and Zencke (1982)
Hettich_1982-6-3-13	Hettich and Zencke (1982)		
Hettich_1986-1-d2-mM	Hettich (1986)		
Hettich_1986-1-d3-mM	Hettich (1986)		
Hettich_1986-1-d4-mM	Hettich (1986)		
Hettich_1986-1-d5-mM	Hettich (1986)		
Hettich_1986-1-d6-mM	Hettich (1986)		
Hettich_1986-1-d7-mM	Hettich (1986)		

Table C.3: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Hettich_1986-2-d2-mM	Hettich (1986)		
Hettich_1986-2-d3-mM	Hettich (1986)		
Hettich_1986-2-d4-mM	Hettich (1986)		
Hettich_1986-2-d5-mM	Hettich (1986)		
Hettich_1986-2-d6-mM	Hettich (1986)		
Hettich_1986-2-d7-mM	Hettich (1986)		
Mehrotra_2014-3-n10	Mehrotra and Papp (2014)		Mehrotra and Papp (2014), Pang et al. (2016)
Mehrotra_2014-3-n20	Mehrotra and Papp (2014)	Pang et al. (2016)	Mehrotra and Papp (2014), Pang et al. (2016)
Mehrotra_2014-3-n40	Mehrotra and Papp (2014)		Mehrotra and Papp (2014), Pang et al. (2016)
Mehrotra_2014-3-n5	Mehrotra and Papp (2014)	Pang et al. (2016)	Mehrotra and Papp (2014), Pang et al. (2016)
Pang_2016-14-n15	Pang et al. (2016)		Mehrotra and Papp (2014), Pang et al. (2016)
Pang_2016-14-n2	Pang et al. (2016)		Mehrotra and Papp (2014), Pang et al. (2016)
Pang_2016-14-n25	Pang et al. (2016)		Mehrotra and Papp (2014), Pang et al. (2016)
Pang_2016-14-n30	Pang et al. (2016)		Mehrotra and Papp (2014), Pang et al. (2016)
Zakovic_2003-1	Žaković and Rustem (2003)		
Zakovic_2003-10	Žaković and Rustem (2003)		
Zakovic_2003-11	Žaković and Rustem (2003)		
Zakovic_2003-12	Žaković and Rustem (2003)		
Zakovic_2003-13	Žaković and Rustem (2003)		
Zakovic_2003-14	Žaković and Rustem (2003)		

Table C.3: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Zakovic_2003-15	Žaković and Rustem (2003)		
Zakovic_2003-16	Žaković and Rustem (2003)		
Zakovic_2003-2	Žaković and Rustem (2003)		
Zakovic_2003-3	Žaković and Rustem (2003)		
Zakovic_2003-4	Žaković and Rustem (2003)		
Zakovic_2003-5	Žaković and Rustem (2003)		
Zakovic_2003-5-2	Žaković and Rustem (2003)		
Zakovic_2003-6	Žaković and Rustem (2003)		
Zakovic_2003-7	Žaković and Rustem (2003)		
Zakovic_2003-8	Žaković and Rustem (2003)		
Zakovic_2003-9	Žaković and Rustem (2003)		
Zhou_1996-OET-2	Zhou and Tits (1996)	Görner (1997)	
Zhou_1996-OET-3	Zhou and Tits (1996)	Görner (1997)	
Zhou_1996-OET-4	Zhou and Tits (1996)	Görner (1997), Vaz et al. (2004b)	
Zhou_1996-OET-5	Zhou and Tits (1996)	Görner (1997), Vaz et al. (2004b)	
Zhou_1996-OET-6	Zhou and Tits (1996)	Görner (1997)	
Zhou_1996-OET-7	Zhou and Tits (1996)	Görner (1997)	
Zhou_1996-PT	Zhou and Tits (1996)		

Table C.4: List of GSIP problem instances with references that have the same or a similar problem.

Problem Name	Source	Also Appears in	Appears Similarly in
Barragan_2022-10	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-4	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-5	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-6	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-7	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-8	Barragán and Camacho-Vallejo (2022)		
Barragan_2022-9	Barragán and Camacho-Vallejo (2022)		
Chen_2013-351	Chen (2013)		
Diehl_2013-DC-corrected	Diehl et al. (2013)		
Diehl_2013-DG	Diehl et al. (2013)		
Harwood_2017-1	Harwood and Barton (2017)		
Harwood_2017-2	Harwood and Barton (2017)		
Harwood_2019-CEx1	Harwood (2019)		
Harwood_2019-CEx2	Harwood (2019)		
Jongen_1998-4-1	Jongen et al. (1998)	Lemonidis (2008), Rückmann and Alfredo Gómez (2006), Stein (2003)	Caspari (2016), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)
Jongen_1998-4-2	Jongen et al. (1998)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015), Stein (2003)	Schwientek et al. (2021)

Table C.4: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Kanzi_2013-1	Kanzi (2013)		
Lemonidis_2008-11	Lemonidis (2008)	Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)	Schwientek et al. (2021)
Lemonidis_2008-12	Lemonidis (2008)	Caspari (2016), Djelassi et al. (2019), Mitsos and Tsoukalas (2015), Schwientek et al. (2021), Seidel and Schwientek (2017)	
Lemonidis_2008-13	Lemonidis (2008)	Caspari (2016), Djelassi et al. (2019), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)	Schwientek et al. (2021)
Lemonidis_2008-14	Lemonidis (2008)	Caspari (2016), Djelassi et al. (2019), Mitsos and Tsoukalas (2015), Schwientek et al. (2021)	Seidel and Schwientek (2017)
Lemonidis_2008-15	Lemonidis (2008)	Mitsos and Tsoukalas (2015), Schwientek et al. (2021), Seidel and Schwientek (2017)	
Lemonidis_2008-16	Lemonidis (2008)	Mitsos and Tsoukalas (2015)	Seidel and Schwientek (2017)
Lemonidis_2008-8	Lemonidis (2008)	Caspari (2016), Djelassi et al. (2019), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)	Schwientek et al. (2021), Vázquez and Rückmann (2005)
Lemonidis_2008-9	Lemonidis (2008)	Caspari (2016), Djelassi et al. (2019), Mitsos and Tsoukalas (2015)	
Li_2003-4-6-1-n-10	Li (2003)		Li (2003), Seidel and Schwientek (2017)

Table C.4: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Li_2003-4-6-1-n-15	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Li_2003-4-6-1-n-20	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Li_2003-4-6-1-n-25	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Li_2003-4-6-2-n-10	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Li_2003-4-6-2-n-15	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Li_2003-4-6-2-n-20	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Li_2003-4-6-2-n-25	Li (2003)		Li (2003), Seidel and Schwientek (2017)
Liu_2013-5	Liu and Wang (2013)		
Liu_2013-6	Liu and Wang (2013)		
Mitsos_2015-2-impl	Mitsos and Tsoukalas (2015)	Djelassi et al. (2019)	Caspari (2016), Jongen et al. (1998), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)
Royset_2003-1	Royset et al. (2003)	Seidel and Schwientek (2017), Tsoukalas et al. (2009)	
Rueckmann_1999-3-1	Rückmann and Shapiro (1999)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015), Rückmann and Alfredo Gómez (2006), Seidel and Schwientek (2017)	

Table C.4: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Rueckmann_2001-4-5	Rückmann and Shapiro (2001)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015), Rückmann and Stein (2001), Schwientek et al. (2021), Seidel and Schwientek (2017)	
Rueckmann_2001-5-1	Rückmann and Shapiro (2001)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015), Rückmann and Alfredo Gómez (2006)	Schwientek et al. (2021), Seidel and Schwientek (2017)
Rueckmann_2001-5-2	Rückmann and Shapiro (2001)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015)	Seidel and Schwientek (2017)
Schwientek_2021-1	Schwientek et al. (2021)		Jongen et al. (1998)
Schwientek_2021-11	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-13	Schwientek et al. (2021)		Lemonidis (2008)
Schwientek_2021-6	Schwientek et al. (2021)		Rückmann and Shapiro (2001), Seidel and Schwientek (2017)
Schwientek_2021-7	Schwientek et al. (2021)		Lemonidis (2008), Schwientek et al. (2021), Vázquez and Rückmann (2005)
Schwientek_2021-8	Schwientek et al. (2021)		Lemonidis (2008), Schwientek et al. (2021), Vázquez and Rückmann (2005)
Schwientek_2021-DC	Schwientek et al. (2021)		
Seidel_2017-DCProblem10	Seidel and Schwientek (2017)		

Table C.4: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Seidel_2017-DCProblem11	Seidel and Schwientek (2017)		
Seidel_2017-DCProblem12	Seidel and Schwientek (2017)		
Seidel_2017-DCProblem2	Seidel and Schwientek (2017)		Diehl et al. (2013)
Seidel_2017-DCProblem7	Seidel and Schwientek (2017)		
Seidel_2017-DCProblem8	Seidel and Schwientek (2017)	Schwientek (2013)	
Seidel_2017-DCProblem9	Seidel and Schwientek (2017)		
Seidel_2017-Problem10	Seidel and Schwientek (2017)		
Seidel_2017-Problem11	Seidel and Schwientek (2017)	Schwientek et al. (2021)	Caspari (2016), Jongen et al. (1998), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)
Seidel_2017-Problem12	Seidel and Schwientek (2017)	Schwientek et al. (2021)	Rückmann and Shapiro (2001)
Seidel_2017-Problem15-impl	Seidel and Schwientek (2017)		Rückmann and Shapiro (2001), Schwientek et al. (2021), Seidel and Schwientek (2017)
Seidel_2017-Problem2	Seidel and Schwientek (2017)		Selassie (2004)
Seidel_2017-Problem22	Seidel and Schwientek (2017)		Lemonidis (2008)
Seidel_2017-Problem24	Seidel and Schwientek (2017)		Lemonidis (2008), Seidel and Schwientek (2017)
Seidel_2017-Problem5	Seidel and Schwientek (2017)		
Seidel_2017-Problem7	Seidel and Schwientek (2017)		Caspari (2016), Jongen et al. (1998), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)
Seidel_2017-Problem8-n-5	Seidel and Schwientek (2017)		Li (2003)
Seidel_2017-Problem9-n-5	Seidel and Schwientek (2017)		Li (2003)
Selassie_2004-221	Selassie (2004)		Seidel and Schwientek (2017)

Table C.4: continued.

Problem Name	Source	Also Appears in	Appears Similarly in
Selassie_2004-501	Selassie (2004)	Seidel and Schwientek (2017)	
Selassie_2004-502	Selassie (2004)		
Selassie_2004-503	Selassie (2004)	Seidel and Schwientek (2017)	
Stein_2003-6-1-1	Stein (2003)	Seidel and Schwientek (2017)	
Stein_2003-6-1-2	Stein (2003)	Seidel and Schwientek (2017)	
Stein_2003-6-1-3	Stein (2003)	Seidel and Schwientek (2017), Stein and Winterfeld (2010)	
Stein_2003-6-1-4	Stein (2003)	Seidel and Schwientek (2017)	
Stein_2012-DC	Stein and Steuermann (2012)		
Still_1999-1	Still (1999)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)	
Still_2001-2-1	Still (2001a)		
Still_2001-2-2	Still (2001a)		
Still_2001-2-4	Still (2001a)	Seidel and Schwientek (2017)	
Still_2002-1	Still (2002)		
Still_2002-2	Still (2002)		
TezelOzturan_2015-2-6	Tezel Özturan (2015)		
Tsoukalas_2009-4	Tsoukalas et al. (2009)		
Vazquez_2005-2-2	Vázquez and Rückmann (2005)	Seidel and Schwientek (2017)	Lemonidis (2008), Schwientek et al. (2021)
Vazquez_2005-3-3	Vázquez and Rückmann (2005)	Caspari (2016), Djelassi et al. (2019), Lemonidis (2008), Mitsos and Tsoukalas (2015), Seidel and Schwientek (2017)	
Vazquez_2005-3-4	Vázquez and Rückmann (2005)	Rückmann and Alfredo Gómez (2006)	

Appendix D

Supplemental Material Regarding Optimal Experimental Design for Guaranteed Parameter Estimation with Rigorous Calculation of Liquid-Liquid Equilibrium Embedded

D.1 Liquid-liquid Equilibirum Model

D.1.1 Reduction of the Model Complexity

With the following assumptions, we can eliminate all describing state variables except for the mole fraction of the first species in the first phase and the mole fraction of the first species in the second phase. We consider two species and two phases, and assume that all species are present in all phases and non-vanishing phases. Recall that an extension to multi-component and multi-phase systems is conceptually straightforward. For notational convenience, we arbitrarily enforce that $x_{1,1} \leq x_{2,1}$, with $x_{l,i}$ being mole fraction of the i^{th} species in phase l. This directly leads to $x_{2,1} - x_{1,1} \geq \varepsilon_{\text{split}} > 0$. Using the closing condition and the definition of the phase ratios φ_j, we obtain

$$\begin{aligned} x_{l,2} &= 1 - x_{l,2}, \ \forall l \\ \varphi_1 &= \frac{x_{2,1} - x_{1,\text{total}}}{x_{2,1} - x_{1,1}} \\ \varphi_2 &= 1 - \varphi_1. \end{aligned} \tag{D.1}$$

Eliminating the second component and dropping the subscript for the components, we obtain the simplified mass balances

$$\begin{aligned} x_1 - x_{\text{total}} &\leq 0 \\ x_{\text{total}} - x_2 &\leq 0 \\ x_1 + \varepsilon_{\text{split}} - x_2 &\leq 0. \end{aligned} \tag{MB}$$

D.1.2 Thermodynamic Relations

The molar Gibbs free energy of mixing reads

$$\bar{G}^{\mathrm{M}}(T, \boldsymbol{p}, \boldsymbol{x}_l) = \bar{\mathrm{R}} T \sum_{c=1}^{n_{\mathrm{c}}} x_{l,c} \ln x_{l,c} + \bar{G}^{\mathrm{E}} \tag{D.2}$$

and is comprised of an ideal and an excess contribution. The molar excess Gibbs energy $\bar{G}^{\mathrm{E}}$ stems from an excess model, in our case, the Redlich and Kister model. The $n_{\mathrm{p}}{}^{\mathrm{th}}$-order Redlich-Kister model reads for two components (Redlich and Kister, 1948)

$$\bar{G}^{\mathrm{E}}(T, \boldsymbol{p}, \boldsymbol{x}_l) = \bar{\mathrm{R}} T x_{l,1} x_{l,2} \sum_{i=0}^{n_p} p_i (x_{l,1} - x_{l,2})^i, \tag{D.3}$$

with n_p denoting the number of parameters.

A necessary condition for stability is that $\bar{G}^{\mathrm{M}}$ is locally convex (Mitsos et al., 2009b)

$$\left.\frac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\right|_{T,\boldsymbol{p},x_i} \geq 0, \ \forall i, \tag{C-locc}$$

where x_i denotes the composition of the two phases. Adding (C-locc) to (C-ip) and (C-TP) may accelerate convergence (Mitsos et al., 2009b).

D.2 Conditions for the Exclusion of Problematic Parameter Values

The following subsections reiterate the conditions by Mitsos et al. (2009b), which state that $\bar{G}^{\mathrm{M}}$ should be convex in x outside the phase split, the Gibbs TPs constructed at $x \approx 0$ and $\approx= 1$ should lie below $\bar{G}^{\mathrm{M}}$, and the secant, which is constructed approximately at the phase split, should be strictly below the Gibbs surface within the phase split. These conditions are extended to hold over a whole temperature range $\mathcal{T}$. For a graphical visualization of these conditions, refer to Figures 5.1b to 5.1d in Section 5.2.2.

D.2.1 Convexity of $\bar{G}^{\mathrm{M}}$ Outside the Phase Split

The extended condition that $\bar{G}^{\mathrm{M}}$ must be convex on the whole temperature range with respect to the mole fraction outside the phase split, c.f. Figure 5.1b in Section 5.2.2, reads

$$\left.\frac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\right|_{T,\bar{\boldsymbol{p}},y} \geq 0, \ \forall T \in \mathcal{T}, \ \forall y \in [x_{\mathrm{L}}, x_1] \cup [x_2, x_{\mathrm{U}}]. \tag{C-c-GSIP}$$

Note that x_1 and x_2 describe the phase split, depend on the (in the case of Mitsos et al. (2009b), measured) temperature T and parameter values $\bar{\boldsymbol{p}}$, and must satisfy (C-TP-SIP). (C-c-GSIP) is a generalized semi-infinite constraint. Reformulation and relaxation yield the two semi-infinite constraints

$$\min\left\{ -\left.\frac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\right|_{T,\bar{\boldsymbol{p}},y}, -(y - x_1) \right\} \leq 0, \ \forall T \in \mathcal{T}, \ \forall y \in [x_{\mathrm{L}}, x_{\mathrm{U}}] \tag{D.4}$$

and

$$\min\left\{-\frac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\bigg|_{T,\bar{\boldsymbol{p}},y}, -(x_2 - y)\right\} \leq 0,\ \forall T \in \mathcal{T},\ \forall y \in [x_{\mathrm{L}}, x_{\mathrm{U}}]. \tag{D.5}$$

Discretizing (D.4) and (D.5) using the index d yields

$$\min\left\{-\frac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\bigg|_{T^d,\bar{\boldsymbol{p}},y^d}, -\left(x_2^d - y^d\right)\right\} \leq 0 \tag{C-cl}$$

and

$$\min\left\{-\frac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\bigg|_{T^d,\bar{\boldsymbol{p}},y^d}, -\left(y^d - x_1^d\right)\right\} \leq 0. \tag{C-cr}$$

Note that $\boldsymbol{x}^d$ is implicitly defined through T^d, $\bar{\boldsymbol{p}}$, (C-TP-SIP) and mass balances (MB). For fixed parameter values $\bar{\boldsymbol{p}}$, we can check if these parameter values fulfill (D.4) and (D.5) by solving the subproblems

$$\begin{aligned} f^{\text{SIP-cl}} = \max_{\substack{T\in\mathcal{T},y\in\mathcal{X},\\ \boldsymbol{x}\in\mathcal{X}^2}} \quad & -\tfrac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\Big|_{T,\bar{\boldsymbol{p}},y} \\ \text{s.t.} \quad & y \leq x_1 \\ & \text{(C-TP-SIP) holds for } T, \bar{\boldsymbol{p}}, \boldsymbol{x} \\ & \text{(MB) holds for } \boldsymbol{x} \end{aligned} \tag{SIP-cl}$$

and

$$\begin{aligned} f^{\text{SIP-cr}} = \max_{\substack{T\in\mathcal{T},y\in\mathcal{X},\\ \boldsymbol{x}\in\mathcal{X}^2}} \quad & -\tfrac{\partial^2 \bar{G}^{\mathrm{M}}}{\partial x^2}\Big|_{T,\bar{\boldsymbol{p}},y} \\ \text{s.t.} \quad & x_2 \leq y \\ & \text{(C-TP-SIP) holds for } T, \bar{\boldsymbol{p}}, \boldsymbol{x} \\ & \text{(MB) holds for } \boldsymbol{x}. \end{aligned} \tag{SIP-cr}$$

Note that (SIP-cl) and (SIP-cr) are themselves SIPs with (P-TP) being their corresponding LLPs, which can be solved with the algorithm proposed by Blankenship and Falk (1976), c.f., Algorithm C.1 in Appendix C.1 or *B&F* in Section 3.3.1. If (the final lower bound of) the optimal objective value $f^{\text{SIP-cl}}$ (and $f^{\text{SIP-cr}}$) is less equal to $\varepsilon^{\text{a,SIP-cr}}$, $\bar{\boldsymbol{p}}$ fulfills the corresponding condition. If not, the optimal solution T^* and y^* of (SIP-cl) (and (SIP-cr)) are used to discretize (C-cl) (and (C-cr)).

Note that the algorithm by Mitsos (2011), i.e., *RRHS*, is not applicable for (SIP-cl), (SIP-cr), (SIP-sec) (to be introduced) because it requires an ε^{f}-optimal SIP-Slater Point, c.f. Definitions 6 and 8. However, an ε^{f}-optimal SIP-Slater Point does not exist in the considered cases because of the constraint (C-TP-SIP).

D.2.2 Gibbs Tangent Plane Below $\bar{G}^{\mathrm{M}}$ at the Boundaries

The Gibbs TPs constructed near $x \approx 0$ and $\approx= 1$ must lie below the $\bar{G}^{\mathrm{M}}$, c.f. Figure 5.1c in Section 5.2.2. This condition extended on $\mathcal{T}$ reads

$$\bar{G}^{\mathrm{M}}(T, \bar{\boldsymbol{p}}, x_{\mathrm{L}}) + \frac{\partial \bar{G}^{\mathrm{M}}}{\partial x}\bigg|_{T,\bar{\boldsymbol{p}},x_{\mathrm{L}}} (y - x_{\mathrm{L}}) \leq \bar{G}^{\mathrm{M}}(T, \bar{\boldsymbol{p}}, y),\ \forall T \in \mathcal{T},\ \forall y \in [x_{\mathrm{L}}, x_{\mathrm{U}}] \tag{C-tpl-SIP}$$

and

$$\bar{G}^{\mathrm{M}}(T,\boldsymbol{p},x_{\mathrm{U}})+\left.\frac{\partial\bar{G}^{\mathrm{M}}}{\partial x}\right|_{T,\bar{\boldsymbol{p}},x_{\mathrm{U}}}(y-x_{\mathrm{U}})\leq\bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},y),\ \forall T\in\mathcal{T},\ \forall y\in[x_{\mathrm{L}},x_{\mathrm{U}}].\quad\text{(C-tpr-SIP)}$$

These two conditions exclude parameter values that will predict additional spurious phase splits outside the true one (case E in Mitsos et al. (2009b)), e.g., parameter values producing a $\bar{G}^{\mathrm{M}}$ curve as depicted in Figure 5.1a in Section 5.2.2 would be excluded. Again, discretizing using index d yields

$$\bar{G}^{\mathrm{M}}\big(T^d,\bar{\boldsymbol{p}},x_{\mathrm{L}}\big)+\left.\frac{\partial\bar{G}^{\mathrm{M}}}{\partial x}\right|_{T^d,\bar{\boldsymbol{p}},x_{\mathrm{L}}}(y^d-x_{\mathrm{L}})\leq\bar{G}^{\mathrm{M}}\big(T^d,\bar{\boldsymbol{p}},y^d\big)\qquad\text{(C-tpl)}$$

and

$$\bar{G}^{\mathrm{M}}\big(T^d,\boldsymbol{p},x_{\mathrm{U}}\big)+\left.\frac{\partial\bar{G}^{\mathrm{M}}}{\partial x}\right|_{T^d,\bar{\boldsymbol{p}},x_{\mathrm{U}}}(y^d-x_{\mathrm{U}})\leq\bar{G}^{\mathrm{M}}\big(T^d,\bar{\boldsymbol{p}},y^d\big).\qquad\text{(C-tpr)}$$

Similarly, we can check if parameter values $\bar{\boldsymbol{p}}$ fulfill conditions (C-tpl-SIP) and (C-tpr-SIP) by solving the subproblems

$$f^{\text{P-tpl}}=\max_{T\in\mathcal{T},y\in\mathcal{X}}\ \bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},x_{\mathrm{L}})+\left.\tfrac{\partial\bar{G}^{\mathrm{M}}}{\partial x}\right|_{T,\bar{\boldsymbol{p}},x_{\mathrm{L}}}(y-x_{\mathrm{L}})-\bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},y)\qquad\text{(P-tpl)}$$

and

$$f^{\text{P-tpr}}=\max_{T\in\mathcal{T},y\in\mathcal{X}}\ \bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},x_{\mathrm{U}})+\left.\tfrac{\partial\bar{G}^{\mathrm{M}}}{\partial x}\right|_{T,\bar{\boldsymbol{p}},x_{\mathrm{U}}}(y-x_{\mathrm{U}})-\bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},y).\qquad\text{(P-tpr)}$$

Again, if (the final lower bound of) the optimal objective value of (P-tpl) and (P-tpr) is less equal to $\varepsilon^{\text{a,P-tp}}$, $\bar{\boldsymbol{p}}$ fulfills the corresponding conditions.

D.2.3 Secant Strictly Below $\bar{G}^{\mathrm{M}}$ in the Interior of the Phase Split

In order to exclude additional phases from the predicted phase split, Mitsos et al. (2009b) additionally require that the secant, which is constructed approximately at the phase split, lies in the interior of the predicted phase split strictly below the Gibbs surface (case C in Mitsos et al. (2009b)), c.f. Figure 5.1d in Section 5.2.2. As strict inequalities and open sets are difficult to handle numerically, a back-off $\delta_x>0$ is introduced and the strict inequality is relaxed to

$$\begin{aligned}\bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},x_1+\delta_x)+\frac{\bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},x_2-\delta_x)-\bar{G}^{\mathrm{M}}(T,\bar{\boldsymbol{p}},x_1+\delta_x)}{(x_2-\delta_x)-(x_1+\delta_x)}(y-(x_1+\delta_x))\\ \leq\bar{G}^{\mathrm{M}}(T,\boldsymbol{p},y),\ \forall T\in\mathcal{T},\ \forall y\in[x_1+\delta_x,x_2-\delta_x].\end{aligned}\qquad\text{(C-sec-GSIP)}$$

Similar to (C-c-GSIP), $\boldsymbol{x}$ depends on the temperature T and parameter values $\bar{\boldsymbol{p}}$. Reformulation, relaxation, and subsequent discretization, akin (C-c-GSIP), yields

$$\begin{aligned}&-\bar{G}^{\mathrm{M}}\big(T^d,\boldsymbol{p},y^d\big)+\bar{G}^{\mathrm{M}}\big(T^d,\bar{\boldsymbol{p}},x_1^d+\delta_x\big)\\&+\frac{\bar{G}^{\mathrm{M}}\big(T^d,\bar{\boldsymbol{p}},x_2^d-\delta_x\big)-\bar{G}^{\mathrm{M}}\big(T^d,\bar{\boldsymbol{p}},x_1^d+\delta_x\big)}{\big(x_2^d-\delta_x\big)-\big(x_1^d+\delta_x\big)}\big(y^d-\big(x_1^d+\delta_x\big)\big)\leq 0.\end{aligned}\qquad\text{(C-sec)}$$

Feasibility of parameter values $\bar{\boldsymbol{p}}$ in (C-sec-GSIP) can be checked by solving the subproblem

$$
\begin{aligned}
f^{\text{SIP-sec}} = \max_{\substack{T\in\mathcal{T},y\in\mathcal{X},\\ \boldsymbol{x}\in\mathcal{X}^2}} \quad & -\bar{G}^{\text{M}}(T,\bar{\boldsymbol{p}},y+\delta_x)+\bar{G}^{\text{M}}(T,\bar{\boldsymbol{p}},x_1+\delta_x) \\
& +\tfrac{\bar{G}^{\text{M}}(T,\bar{\boldsymbol{p}},x_2-\delta_x)-\bar{G}^{\text{M}}(T,\bar{\boldsymbol{p}},x_1+\delta_x)}{(x_2-\delta_x)-(x_1+\delta_x)}(y-(x_1+\delta_x)) \\
\text{s.t.} \quad & x_1+\delta_x \le y \\
& y \le x_2-\delta_x \\
& \text{(C-TP-SIP) holds for } T,\bar{\boldsymbol{p}},\boldsymbol{x} \\
& \text{(MB) holds for } \boldsymbol{x},
\end{aligned}
\tag{SIP-sec}
$$

which itself is an SIP with (P-TP) being its corresponding lower-level program. If (the final lower bound of) the optimal objective value $f^{\text{SIP-sec}}$ is less equal to a small $\varepsilon^{\text{a,SIP-sec}}$, $\bar{\boldsymbol{p}}$ fulfills the corresponding condition. Otherwise the optimal solution T^* and y^* of (SIP-sec) are used to discretize (C-sec).

D.3 Computational Setup

The used tolerances and settings are summarized in Tables D.1 to D.3. All other settings of MAiNGO, except for the output settings, have been left at their default values.

Table D.1: MAiNGO settings used in the numerical case study in Section 5.5 for (OED-ULP).

Setting	Value
`BAB_nodeSelection`	1
`LBP_linPoints`	5
`LBP_activateMoreScaling`	1000

Table D.2: MAiNGO settings used in the numerical case study in Section 5.5 for (OED-MLP) and (OED-AUX).

Setting	Value
`BAB_nodeSelection`	2

Table D.3: Settings used in the numerical case study in Section 5.5. Note that some settings are changed automatically during the solution.

Setting	Value
`abs_tol`	1.00×10^{-2}
`rel_tol`	1.00×10^{-2}
`abs_tol_ulp`	1.00×10^{-3}
`rel_tol_ulp`	1.00×10^{-3}
`abs_tol_mlp`	1.00×10^{-4}
`rel_tol_mlp`	1.00×10^{-4}

Table D.3: continued

Setting	Value
`abs_tol_aux`	1.00×10^{-4}
`rel_tol_aux`	1.00×10^{-4}
`abs_tol_gibbs`	1.00×10^{-6}
`rel_tol_gibbs`	1.00×10^{-6}
`feas_tol_gibbs`	1.00×10^{-5}
`abs_tol_secant`	1.00×10^{-5}
`rel_tol_secant`	1.00×10^{-5}
`feas_tol_secant`	1.00×10^{-4}
`abs_tol_convex`	1.00×10^{-5}
`rel_tol_convex`	1.00×10^{-5}
`feas_tol_convex`	1.00×10^{-4}
`abs_tol_tangent`	1.00×10^{-5}
`rel_tol_tangent`	1.00×10^{-5}
`feas_tol_tangent`	1.00×10^{-4}
`discr_all_ulp`	`false`
`discr_all_mlp`	`false`
`discr_all_aux`	`false`
`discr_all_gibbs`	`false`

Appendix E

Supplemental Material Regarding Adaptive Discretization-Based Algorithms for Semi-Infinite Programs with Unbounded Variables

Note that the used notations and (sub)problem formulations (introduced in Chapter 6) in Appendix E slightly differ compared to Chapter 3 and Appendix C.

E.1 Lower-Bounding Procedure of Mitsos (2011)

E.1.1 Lower Semi-Continuous Constraint Function g

Falk and Hoffman (1977) note that the lower-bounding procedure is still applicable if g is lower-semicontinuous on $\mathcal{X}$ for all $\boldsymbol{y} \in \mathcal{Y}$ and lower-semicontinuous in $\mathcal{X}$ for all $\boldsymbol{y} \in \mathcal{Y}$. However, upper-semicontinuity on $\mathcal{X}$ is not necessary.

Assumption 15 The function f is continuous on $\mathcal{X}$ and g is lower semi-continuous on $\mathcal{X}$ for all $\boldsymbol{y} \in \mathcal{Y}$.

Proposition 5 *Under Assumptions 7 to 9 and 15, the lower-bounding procedure converges.*

Proof. We first show that the first part of of the proof in Section 6.1.3 also holds under Proposition 5. Consider a *compact set of infeasible points* $\mathcal{K}^{\text{infeas}} \subseteq \Lambda(\mathcal{Y}^0)$. In the following, we restrict the iterations to be in this set, i.e., $\boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \mathcal{K}^{\text{infeas}}$. Recall that $g(\boldsymbol{x}^k, \boldsymbol{y}^{*,k}) \leq 0$ would imply that $\boldsymbol{x}^k$ is feasible, $\boldsymbol{x}^k$ is not a member of $\mathcal{K}^{\text{infeas}}$, and we have left $\mathcal{K}^{\text{infeas}}$. Due to Assumption 7, we have

$$\exists \mu > 0 : \ \forall \boldsymbol{x}^k \in \{\boldsymbol{x}^k\} \cap \mathcal{K}^{\text{infeas}} \big[g(\boldsymbol{x}^k, \boldsymbol{y}^k) \geq \alpha g^*(\boldsymbol{x}^k) \geq \mu > 0 \big]. \tag{E.1}$$

$g(\cdot, \boldsymbol{y}^k)$ is lower semi-continuous in $\mathcal{X}$ iff

$$\begin{aligned} \forall \mu > 0 \ \exists \delta_1 > 0 : \ \forall \boldsymbol{x}^k, \boldsymbol{x} \in \mathcal{X} \big[\big\| \boldsymbol{x}^k - \boldsymbol{x} \big\| < \delta_1 \big] \\ \Rightarrow g(\boldsymbol{x}^k, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k) < \mu. \end{aligned} \tag{E.2}$$

Therefore, it also holds

$$\begin{aligned}\forall \mu > 0\ \exists \delta_1 > 0 :\ \forall \boldsymbol{x}^k \in \left\{\boldsymbol{x}^k\right\} \cap \mathcal{K}^{\text{infeas}}, \boldsymbol{x} \in \mathcal{K}^{\text{infeas}} \left[\left\|\boldsymbol{x}^k - \boldsymbol{x}\right\| < \delta_1\right] \\ \Rightarrow g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) - g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) < \mu.\end{aligned} \tag{E.3}$$

Now, we consider the two cases for $\boldsymbol{x} \in \mathcal{K}^{\text{infeas}}$:

Case 1: $g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) - g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) \geq 0$:

$$\begin{aligned}&\Rightarrow && g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) - g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) < \mu \\ &\overset{\text{(E.1)}}{\Longrightarrow} && \mu - g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) \leq g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) - g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) < \mu \\ &\Rightarrow && 0 < g\left(\boldsymbol{x}, \boldsymbol{y}^k\right)\end{aligned} \tag{E.4}$$

Case 2: $g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) - g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) < 0$:

$$\begin{aligned}&\Rightarrow && g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) < g\left(\boldsymbol{x}, \boldsymbol{y}^k\right) \\ &\overset{\text{(E.1)}}{\Longrightarrow} && 0 < \mu \leq g\left(\boldsymbol{x}^k, \boldsymbol{y}^k\right) < g\left(\boldsymbol{x}, \boldsymbol{y}^k\right)\end{aligned} \tag{E.5}$$

Therefore, it holds

$$\begin{aligned}\exists \delta_1 > 0 :\ \forall \boldsymbol{x}^k \in \left\{\boldsymbol{x}^k\right\} \cap \mathcal{K}^{\text{infeas}}, \boldsymbol{x} \in \mathcal{K}^{\text{infeas}} \left[\left\|\boldsymbol{x}^k - \boldsymbol{x}\right\| < \delta_1\right] \\ \Rightarrow 0 < g\left(\boldsymbol{x}, \boldsymbol{y}^k\right).\end{aligned} \tag{E.6}$$

Note, the equivalence between (E.6) and (6.11). The rest of the proof in Section 6.1.3 is conceptually the same and therefore omitted. □

E.1.2 Additional Illustrative Examples

E.1.2.1 Illustrative Example for $\mathcal{X}$ Compact and $\mathcal{Y}$ Bounded and Not Closed

The same convergence issues as in (E1) (Section 6.2.1) arise when we consider (E1.1) with $\mathcal{X}$ compact and $\mathcal{Y}$ bounded but not closed.

$$\begin{aligned}f^* = \inf_{x \in \mathcal{X}}\ & -x \\ \text{s.t. } \forall \boldsymbol{y} \in \mathcal{Y} & \left[-\frac{1}{y_1}(y_2 - x)^2 + 1 \leq 0\right]\end{aligned} \tag{E1.1}$$

$$\begin{aligned}\text{with } \mathcal{X} &:= [0, 2] \\ \mathcal{Y} &:= (0, 0.5] \times [0, 2].\end{aligned}$$

The optimal solutions of (LLP^{unb}) are $\boldsymbol{y}^*(\bar{x}) = (\tilde{y}, \bar{x})^T$ with $\tilde{y} \in (0, 0.5]$ or $\boldsymbol{y}^*(\bar{x}) = (0, \tilde{y})^T$ with $\tilde{y} \in [0, 2]$. Starting with an empty discretization $\mathcal{Y}^0 = \emptyset$ and using $\boldsymbol{y}^{*,k} = \left(2^{-(k+1)}, x^k\right)^T$ for all iterations, the same issues arises.

In general, one could use the closure of the non-compact host sets. However, this is not possible here.

E.1.2.2 Illustrative Example For $\mathcal{X}$ Compact and $\mathcal{Y}$ Unbounded: Feasible SIP

Consider the feasible SIP with unbounded host sets.

$$\begin{aligned} f^* = \inf_{x\in\mathcal{X}} \; & -x \\ & \text{s.t.}\, \forall \boldsymbol{y} \in \mathcal{Y} \left[\begin{cases} -y_1^2(y_2-x)^2+1 \leq 0, & x \geq 1 \\ -y_1^2(y_2-x)^2+3x-2 \leq 0, & x<1 \end{cases} \right] \end{aligned} \tag{E2}$$

$$\begin{aligned} \text{with}\, \mathcal{X} &\coloneqq [0,2] \\ \mathcal{Y} &\coloneqq \mathbb{R}^2 \end{aligned}$$

Remark 3 The function g is continuous but not differentiable on $\mathcal{X}\times\mathcal{Y}$. This is not required according to Assumptions 5 to 7. The non-differentiability of g is of no relevance in this example, as we receive the same convergence properties when using the function

$$\begin{aligned} g(x,\boldsymbol{y}) &= \begin{cases} -y_1^2(y_2-x)^2+1 \leq 0, & x \geq 1 \\ -y_1^2(y_2-x)^2+3x^{2n}-2 \leq 0, & x<1 \end{cases} \\ &\text{with}\; n \in \mathbb{N}_{>0} \end{aligned} \tag{E.7}$$

belonging to the differentiability class C^n instead.

The optimal value function of (LLP$^{\text{unb}}$) of (E2) is

$$g^*(\bar{x}) = \begin{cases} 1, & x \geq 1 \\ 3\bar{x}-2, & x<1 \end{cases} \tag{E.8}$$

with the optimal value $\boldsymbol{y}^*(\bar{x}) = (\tilde{y},\bar{x})^T$ or $\boldsymbol{y}^*(\bar{x}) = (0,\tilde{y})^T$ with $\tilde{y}\in\mathbb{R}$. The feasible set is $\mathcal{X}^{\text{feas}} = \left[0,\frac{2}{3}\right]$. The optimal solution of (E2) is $x^* = \frac{2}{3}$ with $f^* = -\frac{2}{3}$. Using the same sequence of points as in (E1), we converge to the same infeasible point $x=1$ in the limit, c.f., Figure E.1. We do not converge to the globally optimal solution, again failing our expectations.

E.2 Upper-Bounding Procedure of Mitsos (2011)

E.2.1 Definitions and Assumptions of the Upper-Bounding Procedure of Mitsos (2011)

We first review the UBP which we adapt to the case of unbounded variables, c.f., for the compact case refer to (UBP) in Section 3.3.1.

The UBP with $\mathcal{Y}^{\text{UBD},k} \subsetneq \mathcal{Y}$ is

$$\begin{aligned} f^{\text{UBD},k} = \inf_{\boldsymbol{x}\in\mathcal{X}} \; & f(\boldsymbol{x}) \\ & \text{s.t.}\, \forall \boldsymbol{y} \in \mathcal{Y}^{\text{UBD},k} [g(\boldsymbol{x},\boldsymbol{y}) \leq -\varepsilon^{\text{r}}] \end{aligned} \tag{UBP$^{\text{unb}}$}$$

with some $\varepsilon^{\text{r}} > 0$. For convencience we define $g^{\text{UBD}}(\boldsymbol{x},\boldsymbol{y}) = g(\boldsymbol{x},\boldsymbol{y}) - \varepsilon^{\text{r}}$. In the following, we use the same Notation 1 to 3 as for the lower-bounding procedure. The proof of convergence of the upper-bounding procedure in Mitsos (2011) relies on the existence of an ε^{f}-optimal SIP-Slater point, c.f, Definition 8.

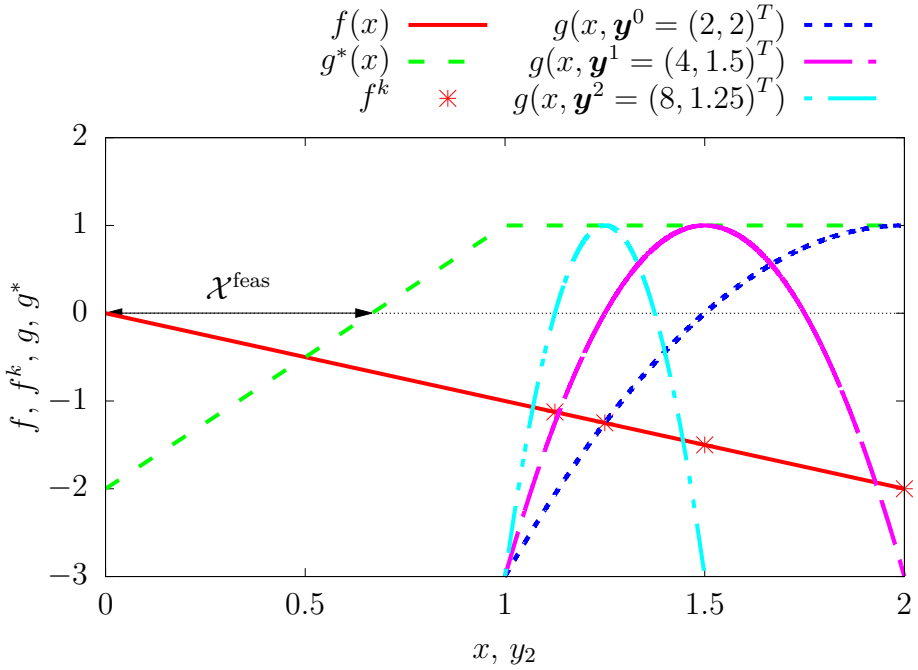


Figure E.1: Illustrative example (E2) for $\mathcal{X}$ compact and $\mathcal{Y}$ unbounded with parametric solutions of (LLP$^{\text{unb}}$). (E2) is feasible. Note: Adapted from Jungen et al. (2022) with minor formatting and typesetting changes, CC BY 4.0, © The Author(s) 2022.

Assumption 16 (*Existence of ε^{f}-Optimal SIP-Slater Point*) An ε^{f}-optimal SIP-Slater point $\boldsymbol{x}^{\text{S}}$ exists, c.f., Definition 8.

E.2.2 Illustrative Example for $\mathcal{X}$ Compact and $\mathcal{Y}$ Ubounded

Mitsos (2011) states

> "at each iteration of the upper-bounding procedure, there are three potential outcomes, namely (i) (UBP$^{\text{unb}}$) is infeasible, (ii) (UBP$^{\text{unb}}$) is feasible and furnishes an SIP-feasible point and (iii) (UBP$^{\text{unb}}$) is feasible but furnishes an SIP-infeasible point. In the former two cases the restriction parameter ε^{r} is reduced and in the latter case $\mathcal{Y}^{\text{UBD},k}$ is populated."

and shows that after a finite number of updates of ε^{r}, the upper-bounding procedure produces an SIP-feasible point, c.f, Definition 4 in Appendix B.1. We show that in the case of unbounded sets, the upper-bounding procedure may never produce an SIP-feasible point. Consider

$$\begin{aligned} f^* = \inf_{x \in \mathcal{X}} \; & -x \\ \text{s.t.} \; \forall \boldsymbol{y} \in \mathcal{Y} & \left[\begin{cases} -y_1^2 (y_2 - x)^2 + 0.5 \leq 0, & x \geq 1 \\ -y_1^2 (y_2 - x)^2 + 3x - 2.5 \leq 0, & x < 1 \end{cases} \right], \end{aligned} \tag{E2}$$

$$\begin{aligned} \text{with} \; \mathcal{X} &:= [0, 2] \\ \mathcal{Y} &:= \mathbb{R}^2, \end{aligned}$$

and choose $\varepsilon^{\mathrm{r}} = 0.5$ in (UBP$^{\mathrm{unb}}$). With the chosen ε^{r}, (UBP$^{\mathrm{unb}}$) of (E2) is equivalent to (LBP$^{\mathrm{unb}}$) of (E2). Figure E.2 depicts the objective function f, $f^{\mathrm{UBD},k}$, $g^{*,\mathrm{UBD}}$, g^* and the feasible set of (UBP$^{\mathrm{unb}}$) and (E2), i.e., and $\mathcal{X}^{\mathrm{UBD,feas}}$ $\mathcal{X}^{\mathrm{feas}}$, respectively.

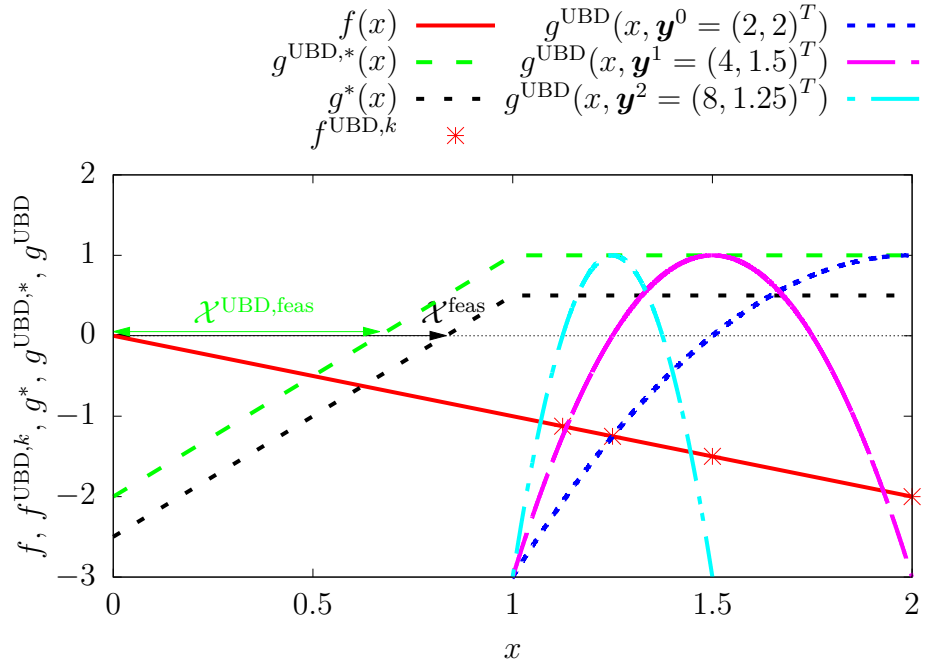


Figure E.2: Illustrative example (E2) for the upper-bounding procedure with $\mathcal{X}$ compact and $\mathcal{Y}$ unbounde. $\varepsilon^{\mathrm{r}} = 0.5$. Note: Adapted from Jungen et al. (2022) with minor formatting and typesetting changes, CC BY 4.0, © The Author(s) 2022.

Note that (LLP$^{\mathrm{unb}}$) solution point of (E2) is independent of ε^{r}. Using the same sequence of points as in (E2), the outcome is always (iii), i.e., no SIP-feasible point is furnished. Therefore, ε^{r} is never reduced, and the upper-bounding procedure converges to the infeasible point $x = 1$ in the limit.

E.2.3 Proof of Convergence of the Upper-Bounding Procedure in Mitsos (2011)

In Mitsos (2011), through the existence of an SIP-Slater point $\boldsymbol{x}^{\mathrm{S}}$, compactness of $\mathcal{Y}$, and continuity of g (c.f, Assumptions 5 and 6), there exists a $\varepsilon^{\mathrm{S}} > 0$ such that

$$\forall \boldsymbol{y} \in \mathcal{Y}\left[g\left(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\right) \leq -\varepsilon^{\mathrm{S}}\right]. \tag{E.9}$$

This also holds for the case of unbounded $\mathcal{X}$ and compact $\mathcal{Y}$ but does not hold for the case of unbounded $\mathcal{Y}$. For example, for $y \in \mathcal{Y} = [1, +\infty)$, $g(x, y) = -\frac{1}{y}$ there exists an SIP-Slater point but $\nexists \varepsilon^{\mathrm{S}} : \ \forall \boldsymbol{y} \in \mathcal{Y}\left[g\left(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\right) \leq \ -\varepsilon^{\mathrm{S}}\right])$. Therefore, for the case of unbounded $\mathcal{Y}$, we slightly adapt Assumption 16 to

Assumption 17 (*Existence of ε^{f}-Optimal and Strictly Feasible Point*) An ε^{f}-optimal and strictly feasible point $\boldsymbol{x}^{\mathrm{S}}$ exists such that there exists an $\varepsilon^{\mathrm{S}} > 0$ and $f\left(\boldsymbol{x}^{\mathrm{S}}\right) \leq f^* + \varepsilon^{\mathrm{f}}$ and $\forall \boldsymbol{y} \in \mathcal{Y}\left[g\left(\boldsymbol{x}^{\mathrm{S}}, \boldsymbol{y}\right) \leq -\varepsilon^{\mathrm{S}}\right]$.

Note that for compact $\mathcal{X}$ and $\mathcal{Y}$, Assumption 17 directly follows from Assumption 16. For the upper-bounding procedure, Assumption 8 has to be slightly stricter because the upper-bounding procedure visits points with $f(\boldsymbol{x}^k) > f^*$.

Assumption 18 It holds (a) $\mathrm{cl}(\Lambda(\mathcal{Y}^0)) \subseteq \mathcal{X}$ and (b) $\Lambda(\mathcal{Y}^0)$ is bounded with the set $\Lambda(\mathcal{Y}^0)$ defined as

$$\Lambda(\mathcal{Y}^0) = \mathcal{X}^{\text{infeas}}. \tag{E.10}$$

To extend the applicability to unbounded SIPs, we adapt the original Lemma to

Theorem 2 *Take any $\mathcal{Y}^{UBD,0} \subsetneq \mathcal{Y}$, any ε^{f} and any $r > 1$. Assume that Assumptions 7 to 10 and 17 hold. Then the upper-bounding procedure in Mitsos (2011) furnishes an SIP-feasible point $\boldsymbol{x}^*$ finitely, such that $f(\boldsymbol{x}^*) \leq f^* + \varepsilon^{\text{f}}$.*

The proof of Theorem 2 is, apart from slight changes, equivalent to the original proof in Mitsos (2011).

Proof. In each iteration one of the following cases hold

i) (UBP$^{\text{unb}}$) is infeasible

ii) (UBP$^{\text{unb}}$) is feasible and a SIP feasible point is furnished

iii) (UBP$^{\text{unb}}$) is feasible and a SIP infeasible point is furnished

In case i) and ii) ε^{r} is reduced. In case iii) $\mathcal{Y}^{\text{UBD},k}$ is populated with the solution point of the corresponding (LLP$^{\text{unb}}$).
We first show that a infinite sequence of infeasible (UBP$^{\text{unb}}$) is not possible. By Assumption 17 there exists a point $\boldsymbol{x}^{\text{S}}$ such that $f(\boldsymbol{x}^{\text{S}}) \leq f^* + \varepsilon^{\text{f}}$ and $\forall \boldsymbol{y} \in \mathcal{Y}\big[g(\boldsymbol{x}^{\text{S}}, \boldsymbol{y}) \leq -\varepsilon^{\text{S}}\big]$. This point is feasible in (UBP$^{\text{unb}}$) if $\varepsilon^{\text{r}} \leq \varepsilon^{\text{S}}$, regardless of $\mathcal{Y}^{\text{UBD},k}$. Therefore, (UBP$^{\text{unb}}$) is feasible if $\varepsilon^{\text{r}} \leq \varepsilon^{\text{S}}$. Because of $\varepsilon^{\text{r}} = \frac{\varepsilon^{\text{r},0}}{(\varepsilon^{\text{red}})^a}$ with a being the number of reductions, after $a = \lceil \log_{\varepsilon^{\text{red}}} \frac{\varepsilon^{\text{r},0}}{\varepsilon^{\text{S}}} \rceil$ reductions, (UBP$^{\text{unb}}$) becomes feasible.
For (UBP$^{\text{unb}}$) with $\varepsilon^{\text{r}} < \varepsilon^{\text{S}}$ follows $f(\boldsymbol{x}^k) \leq f(\boldsymbol{x}^{\text{S}}) \leq f^* + \varepsilon^{\text{f}}$. Hence, the solution point of (UBP$^{\text{unb}}$) $\boldsymbol{x}^k$ is a candidate point with ε^{f}-optimality. If $\boldsymbol{x}^k$ is SIP-feasible, the desired result holds.
It remains to show that an infinite sequence of iii) is not possible, which also means that ε^{r} is no longer updated. It therefore holds for all iterations $\varepsilon^{\text{r}} \geq \varepsilon^{\text{r,min}} \equiv \frac{\varepsilon^{\text{S}}}{\varepsilon^{\text{red}}}$. By construction of (UBP$^{\text{unb}}$) we have

$$\forall l, k: \; l > k \big[g(\boldsymbol{x}^l, \boldsymbol{y}^k) \leq -\varepsilon^{\text{r}} \leq -\varepsilon^{\text{r,min}}\big] \tag{E.11}$$

Due to Assumption 10, it also holds

$$\begin{aligned} \exists \delta_1 > 0: \; \forall \boldsymbol{x}^l \in \{\boldsymbol{x}^k\} \cap \mathcal{X}^{\text{infeas}}, \boldsymbol{x} \in \mathcal{X}^{\text{infeas}} \big[\|\boldsymbol{x}^l - \boldsymbol{x}\| < \delta_1\big] \\ \Rightarrow \big|g(\boldsymbol{x}^l, \boldsymbol{y}^k) - g(\boldsymbol{x}, \boldsymbol{y}^k)\big| < \frac{\varepsilon^{\text{r,min}}}{2}. \end{aligned} \tag{E.12}$$

Combining (E.11) and (E.12), we have

$$\forall l, k: l > k \left[g(\boldsymbol{x}, \boldsymbol{y}^k) < -\frac{\varepsilon^{\text{r,min}}}{2} < 0\right] \tag{E.13}$$

Due to Assumption 8, the limit point $\boldsymbol{x}^* \in \mathrm{cl}(\mathcal{X})^{\mathrm{infeas}}$ exists. From $\boldsymbol{x}^k \to \boldsymbol{x}^*$, we have

$$\forall \delta_1 \exists K : \forall l, k [l > k \geq K] \big[\big\| \boldsymbol{x}^l - \boldsymbol{x}^k \big\| < \delta_1 \big]. \tag{E.14}$$

Substituting $\boldsymbol{x} = \boldsymbol{x}^k$ in (E.13), we obtain

$$\exists K : g\big(\boldsymbol{x}^k, \boldsymbol{y}^k\big) < -\frac{\varepsilon^{\mathrm{r,min}}}{2} < 0. \tag{E.15}$$

Therefore, after a finite number of iterations K the point $\boldsymbol{x}^k$ is SIP-feasible which gives us the desired result. Combining the results above, the upper-bounding procedure furnishes a point $\boldsymbol{x}^*$ that satisfies $f(\boldsymbol{x}^*) \leq f^* + \varepsilon^{\mathrm{f}}$ after a finite number of iterations. □

Bibliography

D. S. Abrams and J. M. Prausnitz. Statistical thermodynamics of liquid mixtures: A new expression for the excess Gibbs energy of partly or completely miscible systems. *AIChE Journal*, 21(1):116–128, 1975. doi:10.1002/aic.690210115.

D. H. Ackley. *A connectionist machine for genetic hillclimbing*, volume 28 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, 1987. ISBN 978-0-89838-236-5.

C. S. Adjiman, I. P. Androulakis, C. D. Maranas, C. A. Floudas, C. S. Adjiman, and C. A. Floudas. A global optimization method, αBB, for process design. *Computers & Chemical Engineering*, 20:419–424, 1996. doi:10.1016/0098-1354(96)00080-4.

E. B. Andersen. Asymptotic properties of conditional maximum-likelihood estimators. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 32(2):283–301, 1970. doi:10.1111/j.2517-6161.1970.tb00842.x.

E. J. Anderson and A. S. Lewis. An extension of the simplex algorithm for semi-infinite linear programming. *Mathematical Programming*, 44(1-3):247–269, 1989. doi:10.1007/BF01587092.

D. O. Andreassen and G. A. Watson. Linear Chebyshev approximation without Chebyshev sets. *BIT. Numerical Mathematics*, 16(4):349–362, 1976. doi:10.1007/BF01932717.

S. P. Asprey and S. Macchietto. Designing robust optimal dynamic experiments. *Journal of Process Control*, 12(4):545–556, 2002. doi:10.1016/S0959-1524(01)00020-8.

N. Asprion, R. Böttcher, J. Mairhofer, M. Yliruka, J. Höller, J. Schwientek, C. Vanaret, and M. Bortz. Implementation and application of model-based design of experiments in a flowsheet simulator. *Journal of Chemical & Engineering Data*, 65(3):1135–1145, 2020. doi:10.1021/acs.jced.9b00494.

R. C. Aster and C. H. Thurber. *Parameter estimation and inverse problems*. Academic Press, Waltham MA, 2 edition, 2013. ISBN 978-0-12-385048-5.

A. C. Atkinson and V. V. Federov. The design of experiments for discriminating between two rival models. *Biometrika*, 62(1):57–70, 1975. doi:10.1093/biomet/62.1.57.

A. C. Atkinson, A. N. Donev, and R. Tobias. *Optimum experimental designs, with SAS*, volume 34 of *Oxford Statistical Science Series*. Oxford University Press, Oxford, 2007. ISBN 978-0-19-929660-6.

A. Auslender, A. Ferrer, M. A. Goberna, and M. A. López. Comparative study of RPSALG algorithm for convex semi-infinite programming. *Computational Optimization and Applications*, 60(1):59–87, 2015. doi:10.1007/s10589-014-9667-7.

S. Avraamidou and E. N. Pistikopoulos. B-POP: Bi-level parametric optimization toolbox. *Computers & Chemical Engineering*, 122:193–202, 2019. doi:10.1016/j.compchemeng.2018.07.007.

L. E. Baker, A. C. Pierce, and K. D. Luks. Gibbs energy analysis of phase equilibria. *Society of Petroleum Engineers Journal*, 22(05):731–742, 1982. doi:10.2118/9806-PA.

M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H.-T. Lin, editors, *NIPS '20: Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 21524–21538, Red Hook, NY, USA, 2020. Curran Associates, Inc. ISBN 978-1-7138-2954-6.

A. Barragán and J.-F. Camacho-Vallejo. An exact algorithm based on the Kuhn–Tucker conditions for solving linear generalized semi-infinite programming problems. *Journal of Mathematics*, 2022:1–14, 2022. doi:10.1155/2022/1765385.

I. Barrodale, L. M. Delves, and J. C. Mason. Linear Chebyshev approximation of complex-valued functions. *Mathematics of Computation*, 32(143):853, 1978. doi:10.2307/2006490.

Y. Beck, D. Bienstock, M. Schmidt, and J. Thürauf. On a computationally ill-behaved bilevel problem with a continuous and nonconvex lower level. *Journal of Optimization Theory and Applications*, 198(1):428–447, 2023. doi:10.1007/s10957-023-02238-9.

G. Belforte and P. Gay. Optimal worst case estimation for LPV–FIR models with bounded errors. In R. R. Bitmead and C. B. Schrader, editors, *Proceedings of the 39th IEEE Conference on Decision and Control*, volume 5, pages 4573–4577. IEEE Control Systems Society, 2000. doi:10.1109/CDC.2001.914636.

G. Belforte and P. Gay. Optimal worst case estimation for LPV–FIR models with bounded errors. *Systems & Control Letters*, 53(3):259–268, 2004. doi:10.1016/j.sysconle.2004.05.004.

G. Belforte and T. T. Tay. Optimal input design for worst-case system identification in $l_1/l_2/l_\infty$. *Systems & Control Letters*, 20(4):273–278, 1993. doi:10.1016/0167-6911(93)90003-O.

G. Belforte, B. Bona, and S. Frediani. Optimal sampling schedule with unknown but bounded measurement errors: Families of linear models. In A. H. Haddad and K. P. Polis, editors, *Proceedings of the 23rd IEEE Conference on Decision and Control*, IEEE Control Systems Society, pages 1554–1559, 1984. doi:10.1109/CDC.1984.272342.

G. Belforte, B. Bona, and S. Frediani. Optimal sampling schedule for parameter estimation of linear models with unknown but bounded measurement errors. *IEEE Transactions on Automatic Control*, 32(2):179–182, 1987. doi:10.1109/TAC.1987.1104535.

G. Belforte, B. Bona, and V. Cerone. Parameter estimation with set membership uncertainty: Nonlinear families of models. In *8th IFAC/IFOORS Symposium on Identification and System Parameter Estimation*, volume 21, pages 399–404, 1988. doi:10.1016/S1474-6670(17)54760-6.

G. Belforte, B. Bona, and V. Cerone. Parameter estimation algorithms for a set-membership description of uncertainty. *Automatica*, 26(5):887–898, 1990. doi:10.1016/0005-1098(90)90005-3.

A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations Research Letters*, 25(1):1–13, 1999. doi:10.1016/S0167-6377(99)00016-4.

A. Ben-Tal and A. Nemirovski. Robust optimization – methodology and applications. *Mathematical Programming*, 92(3):453–480, 2002. doi:10.1007/s101070100286.

B. Betro. An accelerated central cutting plane algorithm for linear semi-infinite programming. *Mathematical Programming*, 101(3):479–495, 2004. doi:10.1007/s10107-003-0492-5.

B. Beykal, S. Avraamidou, I. P. E. Pistikopoulos, M. Onel, and E. N. Pistikopoulos. DOMINO: Data-driven Optimization of bi-level Mixed-Integer NOnlinear Problems. *Journal of Global Optimization*, 78(1):1–36, 2020. doi:10.1007/s10898-020-00890-3.

B. Bhattacharjee, W. H. Green Jr., and P. I. Barton. Interval methods for semi-infinite programs. *Computational Optimization and Applications*, 30(1):63–93, 2005a. doi:10.1007/s10589-005-4556-8.

B. Bhattacharjee, P. Lemonidis, W. H. Green Jr., and P. I. Barton. Global solution of semi-infinite programs. *Mathematical Programming*, 103(2):283–307, 2005b. doi:10.1007/s10107-005-0583-6.

J. W. Blankenship and J. E. Falk. Infinitely constrained optimization problems. *Journal of Optimization Theory and Applications*, 19(2):261–281, 1976. doi:10.1007/BF00934096.

G. M. Bollas, P. I. Barton, and A. Mitsos. Bilevel optimization formulation for parameter estimation in vapor–liquid(–liquid) phase equilibrium problems. *Chemical Engineering Science*, 64(8):1768–1783, 2009. doi:10.1016/j.ces.2009.01.003.

H. P. Bolton, A. A. Groenwold, and J. A. Snyman. The application of a unified Bayesian stopping criterion in competing parallel algorithms for global optimization. *Computers & Mathematics with Applications*, 48(3-4):549–560, 2004. doi:10.1016/j.camwa.2003.09.030.

S. Bolusani, M. Besançon, K. Bestuzheva, A. Chmiela, J. Dionísio, T. Donkiewicz, J. van Doornmalen, L. Eifler, M. Ghannam, A. Gleixner, C. Graczyk, K. Halbig, I. Hedtke, A. Hoen, C. Hojny, R. van der Hulst, D. Kamp, T. Koch, K. Kofler, J. Lentz, J. Manns, G. Mexi, E. Mühmer, M. E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, M. Turner, S. Vigerske, D. Weninger, and L. Xu. The SCIP Optimization Suite 9.0: Technical report, 2024. URL `https://optimization-online.org/2024/02/the-scip-optimization-suite-9-0/`. accessed 2025/02/25.

I. Bomze, A. Horländer, and M. Schmidt. Mixed-integer bilevel optimization with nonconvex quadratic lower-level problems: Complexity and a solution method. *Journal of Global Optimization*, 2025. doi:10.1007/s10898-025-01522-4.

D. Bongartz, J. Najman, S. Sass, and A. Mitsos. MAiNGO — McCormick-based Algorithm for mixed-integer Nonlinear Global Optimization, 2018. URL `http://www.avt.rwth-aachen.de/global/show_document.asp?id=aaaaaaaaabclahw`. accessed 2023/07/20.

J. F. Bonnans and A. Shapiro. *Perturbation analysis of optimization problems*. Springer Series in Operations Research. Springer, New York, USA, 2000. ISBN 978-0387987057.

S. Borchers and R. Findeisen. Design of experiments for guaranteed parameter estimation in membership setting. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pages 2602–2607. IEEE Control Systems Society, 2011. doi:10.1109/CDC.2011.6161062.

S. Borchers, S. V. Raković, and R. Findeisen. Set membership parameter estimation and design of experiments using homothety. *IFAC Proceedings Volumes*, 44(1):9035–9040, 2011. doi:10.3182/20110828-6-IT-1002.03555.

G. E. Box and W. J. Hill. Discrimination among mechanistic models. *Technometrics*, 9(1): 57–71, 1967. doi:10.1080/00401706.1967.10490441.

E. Bradford, A. M. Schweidtmann, and A. A. Lapkin. Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm. *Journal of Global Optimization*, 71(2):407–438, 2018. doi:10.1007/s10898-018-0609-2.

A. D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(88):2879–2904, 2011.

M. J. Cánovas, M. A. López, J. Parra, and M. I. Todorov. Stability and well-posedness in linear semi-infinite programming. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 10(1):82–98, 1999. doi:10.1137/S1052623497319869.

C. Cartis, L. Roberts, and O. Sheridan-Methven. Escaping local minima with local derivative-free methods: a numerical investigation. *Optimization*, 71(8):2343–2373, 2022. doi:10.1080/02331934.2021.1883015.

A. Caspari. *Hierarchical Optimization Methods in Optimal Experimental Design for Guaranteed Parameter Estimation*. Master thesis, RWTH Aachen University, Germany, 2016.

M. Cerulli, A. Oustry, C. d'Ambrosio, and L. Liberti. Convergent algorithms for a class of convex semi-infinite programs. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 32(4):2493–2526, 2022. doi:10.1137/21M1431047.

B. Chachuat, B. Houska, R. Paulen, N. Peri'c, J. Rajyaguru, and M. E. Villanueva. Set-theoretic approaches in analysis, estimation and control of nonlinear systems. *IFAC-PapersOnLine*, 48(8):981–995, 2015. doi:10.1016/j.ifacol.2015.09.097.

W. G. Chapman, K. E. Gubbins, G. Jackson, and M. Radosz. New reference equation of state for associating liquids. *Industrial & Engineering Chemistry Research*, 29(8): 1709–1721, 1990. doi:10.1021/ie00104a021.

G. Chen, Z. Song, Z. Qi, and K. Sundmacher. Neural recommender system for the activity coefficient prediction and UNIFAC model extension of ionic liquid–solute systems. *AIChE Journal*, 67(4), 2021. doi:10.1002/aic.17171.

J. Chen, F. Luo, G. Li, and Z. Wang. Batch Bayesian optimization with adaptive batch acquisition functions via multi-objective optimization. *Swarm and Evolutionary Computation*, 79:101293, 2023. doi:10.1016/j.swevo.2023.101293.

Z. Chen. *Optimality Conditions of Semi-Infinite Programming and Generalized Semi-Infinite Programming*. Ph.D. thesis, The Hong Kong Polytechnic University, Hong Kong, 2013.

P. A. Clark and A. W. Westerberg. Bilevel programming for steady-state chemical process design — I. fundamentals and algorithms. *Computers & Chemical Engineering*, 14(1): 87–97, 1990. doi:10.1016/0098-1354(90)87007-C.

I. D. Coope and C. J. Price. Exact penalty function methods for nonlinear semi-infinite programming. In P. Pardalos, R. Horst, R. Reemtsen, and J.-J. Rückmann, editors, *Semi-Infinite Programming*, volume 25 of *Nonconvex Optimization and Its Applications*, pages 137–157. Springer US, Boston, MA, USA, 1998. ISBN 978-1-4419-4795-6.

I. D. Coope and G. A. Watson. A projected Lagrangian algorithm for semi-infinite programming. *Mathematical Programming*, 32(3):337–356, 1985. doi:10.1007/BF01582053.

E. Cramer, O. Stollenwerk, L. Kutschak, J. A. Paulson, and A. Mitsos. Bayesian optimization with mechanistic knowledge. In *8th Machine Learning and AI in Bio(Chemical) Engineering Conference*, Cambridge, United Kingdom, 2025. conference presentation.

A. R. Curtis and M. J. D. Powell. Necessary conditions for a minimax approximation. *The Computer Journal*, 8(4):358–361, 1966. doi:10.1093/comjnl/8.4.358.

W. Dai, S. Cremaschi, H. J. Subramani, and H. Gao. Estimation of data uncertainty in the absence of replicate experiments. *Chemical Engineering Research and Design*, 147: 187–199, 2019a. doi:10.1016/j.cherd.2019.05.007.

Z. Dai, H. Yu, B. K. H. Low, and P. Jaillet. Bayesian optimization meets Bayesian optimal stopping. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1496–1506. PMLR, 2019b.

G. de Ath, R. M. Everson, Rahat, Alma A. M., J. E. Fieldsend, and A. A. M. Rahat. Greed is good: Exploration and exploitation trade-offs in Bayesian optimisation. *ACM Transactions on Evolutionary Learning and Optimization*, 1(1):1–22, 2021. doi:10.1145/3425501.

D. Dechambre, L. Wolff, C. Pauls, and A. Bardow. Optimal experimental design for the characterization of liquid–liquid equilibria. *Industrial & Engineering Chemistry Research*, 53(50):19620–19627, 2014. doi:10.1021/ie5035573.

DECHEMA e.V. DECHEMA Data Preparation Package (DPP), 2024. URL `https://i-systems.dechema.de/MyDECHEMA/DPP-Tutorials.htm`. accessed 2025/06/09.

L. Denis-Vidal, C. Jauberthie, and M. Kieffer. Optimal experiment design for bounded-error estimation of nonlinear models. In C. Canudas de Wit and R. Sepulchre, editors, *Proceedings of the 58th IEEE Conference on Decision and Control*, pages 4147–4154, 2019. doi:10.1109/CDC40024.2019.9030003.

H. Dette, V. B. Melas, and P. Shpilev. T-optimal designs for discrimination between two polynomial models. *Annals of Statistics*, 40(1):188–205, 2012. doi:10.1214/11-AOS956.

M. Diehl, B. Houska, O. Stein, and P. Steuermann. A lifting method for generalized semi-infinite programs based on lower level Wolfe duality. *Computational Optimization and Applications*, 54(1):189–210, 2013. doi:10.1007/s10589-012-9489-4.

L. C. W. Dixon and G. P. Szegö. The global optimization problem: An introduction. In L. C. W. Dixon and G. P. Szegö, editors, *Towards Global Optimisation 2*, pages 1–15. North-Holland Pub. Co., Amsterdam and New York, 1978. ISBN 0444851712.

H. Djelassi. *Discretization-Based Algorithms for the Global Solution of Hierarchical Programs*. Dissertation, RWTH Aachen University, Aachen, Germany, 2020.

H. Djelassi and A. Mitsos. A hybrid discretization algorithm with guaranteed feasibility for the global solution of semi-infinite programs. *Journal of Global Optimization*, 68(2): 227–253, 2017. doi:10.1007/s10898-016-0476-7.

H. Djelassi and A. Mitsos. Global solution of semi-infinite programs with existence constraints. *Journal of Optimization Theory and Applications*, 188(3):863–881, 2021. doi:10.1007/s10957-021-01813-2.

H. Djelassi, S. Fliscounakis, A. Mitsos, and P. Panciatici. Hierarchical programming for worst-case analysis of power grids. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7, 2018. doi:10.23919/PSCC.2018.8444136.

H. Djelassi, M. Glass, and A. Mitsos. Discretization-based algorithms for generalized semi-infinite and bilevel programs with coupling equality constraints. *Journal of Global Optimization*, 75(2):341–392, 2019. doi:10.1007/s10898-019-00764-3.

H. Djelassi, A. Mitsos, and O. Stein. Recent advances in nonconvex semi-infinite programming: Applications and algorithms. *EURO Journal on Computational Optimization*, 9 (5):100006, 2021. doi:10.1016/j.ejco.2021.100006.

Q. Dong, R. D. Chirico, X. Yan, X. Hong, and M. Frenkel. Uncertainty reporting for experimental thermodynamic properties. *Journal of Chemical & Engineering Data*, 50 (2):546–550, 2005. doi:10.1021/je049673d.

B. P. Duarte, A. C. Atkinson, J. F. Granjo, and N. M. Oliveira. Optimal design of experiments for liquid–liquid equilibria characterization via semidefinite programming. *Processes*, 7(11):834, 2019. doi:10.3390/pr7110834.

B. P. M. Duarte, W. K. Wong, and A. C. Atkinson. A semi-infinite programming based algorithm for determining T–optimum designs for model discrimination. *Journal of Multivariate Analysis*, 135:11–24, 2015. doi:10.1016/j.jmva.2014.11.006.

D. Eriksson, M. Pearce, J. Gardner, R. D. Turner, and M. Poloczek. Scalable global optimization via local Bayesian optimization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, and E. B. Fox, editors, *NIPS'19: Proceedings of the 33rd International Conference on Neural Information Processing Systems*, volume 32, Red Hook, NY, USA, 2019. Curran Associates, Inc. ISBN 9781713807933.

P. T. Eubank and K. R. Hall. Equal area rule and algorithm for determining phase compositions. *AIChE Journal*, 41(4):924–927, 1995. doi:10.1002/aic.690410419.

P. T. Eubank, A. E. Elhassan, M. A. Barrufet, and W. B. Whiting. Area method for prediction of fluid-phase equilibria. *Industrial & Engineering Chemistry Research*, 31(3): 942–949, 1992. doi:10.1021/ie00003a041.

Fair Isaac Corporation. FICO Xpress-Optimizer, reference manual, 2023. URL `https://www.fico.com/fico-xpress-optimization/docs/`. accessed 2024/11/17.

J. E. Falk and K. Hoffman. A nonconvex max-min problem. *Naval Research Logistics Quarterly*, 24(3):441–450, 1977. doi:10.1002/nav.3800240307.

S.-C. Fang, C.-J. Lin, and S.-Y. Wu. On solving convex quadratic semi-infinite programming problems. *Optimization*, 31(2):107–125, 1994. doi:10.1080/02331939408844009.

M. C. Ferris and A. B. Philpott. An interior point algorithm for semi-infinite linear programming. *Mathematical Programming*, 43(1-3):257–276, 1989. doi:10.1007/BF01582293.

M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. A new general-purpose algorithm for mixed-integer bilevel linear programs. *Operations Research*, 65(6):1615–1637, 2017. doi:10.1287/opre.2017.1650.

L. Fleitmann, J. Pyschik, L. W. M. Wolff, and A. Bardow. Optimal physical property data for process simulations by optimal experimental design. In M. Türkay and R. Gani, editors, *31st European Symposium on Computer Aided Process Engineering*, volume 50 of *Computer Aided Chemical Engineering*, pages 851–857. Elsevier BV, 2021. doi:10.1016/B978-0-323-88506-5.50133-9.

L. Fleitmann, J. Pyschik, L. Wolff, J. Schilling, and A. Bardow. Optimal experimental design of physical property measurements for optimal chemical process simulations. *Fluid Phase Equilibria*, 557:113420, 2022. doi:10.1016/j.fluid.2022.113420.

C. A. Floudas and C. E. Gounaris. A review of recent advances in global optimization. *Journal of Global Optimization*, 45(1):3–38, 2009. doi:10.1007/s10898-008-9332-8.

C. A. Floudas and O. Stein. The adaptive convexification algorithm: A feasible point method for semi-infinite programming. *SIAM Journal on Optimization*, 18(4):1187–1208, 2008. doi:10.1137/060657741.

G. Franceschini and S. Macchietto. Model-based design of experiments for parameter precision: State of the art. *Chemical Engineering Science*, 63(19):4846–4872, 2008. doi:10.1016/j.ces.2007.11.034.

P. I. Frazier. Bayesian optimization. In E. Gel and L. Ntaimo, editors, *Tutorials in Operations Research*, pages 255–278, Cantonsville MD, 2018. INFORMS. doi:10.1287/educ.2018.0188.

GAMS Development Corporation. General algebraic modeling system (GAMS), 2019. URL `http://www.gams.com/`. accessed 2025/08/14.

W. Gan, Z. Ji, and Y. Liang. Acquisition functions in Bayesian optimization. In *2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE)*, pages 129–135, 2021. doi:10.1109/ICBASE53849.2021.00032.

R. Garnett. *Bayesian Optimization*. Cambridge University Press, Cambridge, United Kingdom, 2023. ISBN 9781108348973.

A. Georgiou. *Data-Driven Algorithms for Accelerating Discovery and Optimization in Complex Dynamical Systems*. Ph.D. thesis, Johns Hopkins University, Baltimore, MD, USA, 2025.

A. Georgiou, D. Jungen, L. Kaven, V. Hunstig, C. Frangakis, I. Kevrekidis, and A. Mitsos. Deterministic global optimization of the acquisition function in bayesian optimization: To do or not to do? *submitted to Journal of Global Optimization*, 2025a.

A. Georgiou, D. Jungen, L. Kaven, V. Hunstig, C. Frangakis, I. Kevrekidis, and A. Mitsos. Deterministic global optimization of the acquisition function in bayesian optimization: To do or not to do? *arXiv*, 2025b. arXiv:2503.03625.

K. Glashoff and S.-Å. Gustafson. *Linear optimization and approximation: An introduction to the theoretical analysis and numerical treatment of semi-infinite programs*, volume 45 of *Applied Mathematical Sciences*. Springer New York, New York, NY, USA, 1 edition, 1983. ISBN 978-1-4612-1142-6.

M. Glass and A. Mitsos. Thermodynamic analysis of formulations to discriminate multiple roots of cubic equations of state in process models. *Computers & Chemical Engineering*, 106:407–420, 2017. doi:10.1016/j.compchemeng.2017.06.023.

M. Glass and A. Mitsos. Parameter estimation in reactive systems subject to sufficient criteria for thermodynamic stability. *Chemical Engineering Science*, 197:420–431, 2019. doi:10.1016/j.ces.2018.08.035.

M. Glass, M. Aigner, J. Viell, A. Jupke, and A. Mitsos. Liquid-liquid equilibrium of 2-methyltetrahydrofuran/water over wide temperature range: Measurements and rigorous regression. *Fluid Phase Equilibria*, 433:212–225, 2017. doi:10.1016/j.fluid.2016.11.004.

M. Glass, H. Djelassi, and A. Mitsos. Parameter estimation for cubic equations of state models subject to sufficient criteria for thermodynamic stability. *Chemical Engineering Science*, 192:981–992, 2018. doi:10.1016/j.ces.2018.08.033.

A. Gleixner, G. Hendel, G. Gamrath, T. Achterberg, M. Bastubbe, T. Berthold, P. M. Christophel, K. Jarck, T. Koch, J. Linderoth, M. Lübbecke, H. D. Mittelmann, D. Ozyurt, T. K. Ralphs, D. Salvagnin, and Y. Shinano. MIPLIB 2017: Data-driven compilation of the 6th mixed-integer programming library. *Mathematical Programming Computation*, 2021. doi:10.1007/s12532-020-00194-3.

M. A. Goberna and M. A. López. *Linear semi-infinite optimization*. Wiley series in mathematical methods in practice. Wiley, Chichester, 1998. ISBN 978-0471970408.

M. Goerigk, J. Kurtz, M. Schmidt, and J. Thürauf. Connections between robust and bilevel optimization. *Open Journal of Mathematical Optimization*, 6:1–17, 2025. doi:10.5802/ojmo.38.

S. Görner. *Ein Hybridverfahren zur Lösung nichtlinearer semi-infiniter Optimierungsprobleme*. Dissertation, Technische Universität Berlin, Berlin, 1997.

A. R. Gottu Mukkula and R. Paulen. Optimal design of dynamic experiments for guaranteed parameter estimation. In D. Abramovitch and G. Chiu, editors, *2016 American Control Conference*, pages 1826–1831. IEEE, 2016a. doi:10.1109/ACC.2016.7525184.

A. R. Gottu Mukkula and R. Paulen. Optimal dynamic experiment design for guaranteed parameter estimation. In Z. Kravanja and M. Bogataj, editors, *26th European Symposium on Computer Aided Process Engineering*, volume 38, pages 757–762. Elsevier BV, 2016b. doi:10.1016/B978-0-444-63428-3.50131-4.

A. R. Gottu Mukkula and R. Paulen. Model-based design of optimal experiments for nonlinear systems in the context of guaranteed parameter estimation. *Computers & Chemical Engineering*, 99:198–213, 2017a. doi:10.1016/j.compchemeng.2017.01.029.

A. R. Gottu Mukkula and R. Paulen. Robust model-based design of experiments for guaranteed parameter estimation. In A. Espuña, M. Graells, and L. Puigjaner, editors, *27th European Symposium on Computer Aided Process Engineering*, volume 40 of *Computer Aided Chemical Engineering*, pages 1639–1644. Elsevier, 2017b. doi:10.1016/B978-0-444-63965-3.50275-0.

A. R. Gottu Mukkula and R. Paulen. Robust design of optimal experiments considering consecutive re-designs. *IFAC-PapersOnLine*, 55(7):13–18, 2022. doi:10.1016/j.ifacol.2022.07.415.

A. R. Gottu Mukkula, M. Mateáš, M. Fikar, and R. Paulen. Robust multi-stage model-based design of optimal experiments for nonlinear estimation. *Computers & Chemical Engineering*, 155:107499, 2021. doi:10.1016/j.compchemeng.2021.107499.

J. Gross and G. Sadowski. Perturbed-Chain SAFT: An equation of state based on a perturbation theory for chain molecules. *Industrial & Engineering Chemistry Research*, 40(4):1244–1260, 2001. doi:10.1021/ie0003887.

J. Gross and J. Vrabec. An equation–of–state contribution for polar components: Dipolar molecules. *AIChE Journal*, 52(3):1194–1204, 2006. doi:10.1002/aic.10683.

I. E. Grossmann and R. W. H. Sargent. Optimum design of chemical plants with uncertain parameters. *American Institute of Chemical Engineers Journal*, 24(6):1021–1028, 1978. doi:10.1002/aic.690240612.

F. Guerra-Vázquez, J.-J. Rückmann, O. Stein, and G. Still. Generalized semi-infinite programming: A tutorial. *Journal of Computational and Applied Mathematics*, 217(2): 394–419, 2008. doi:10.1016/j.cam.2007.02.012.

Z. H. Gümüş. Reactive distillation column design with vapor/liquid/liquid equilibria. *Computers & Chemical Engineering*, 21(1-2):983–988, 1997. doi:10.1016/S0098-1354(97)00177-4.

H. Günzel, H. T. Jongen, and O. Stein. On the closure of the feasible set in generalized semi-infinite programming. *Central European Journal of Operations Research*, 15(3): 271–280, 2007. doi:10.1007/s10100-007-0030-2.

F. Guo and X. Sun. Semidefinite programming relaxations for linear semi-infinite polynomial programming. *Pacific Journal of Optimization*, 16(3):395–418, 2020.

F. Guo and M. Zhang. An SDP method for fractional semi-infinite programming problems with SOS-convex polynomials. *Optimization Letters*, 2023. doi:10.1007/s11590-023-01974-1.

L. Gurobi Optimization. Gurobi Optimizer Reference Manual, 2025. URL `https://www.gurobi.com`. accessed 2025/02/21.

S.-Å. Gustafson. On numerical analysis in semi-infinite programming. In A. V. Balakrishnan, M. Thoma, and R. Hettich, editors, *Semi-Infinite Programming*, volume 15 of *Lecture Notes in Control and Information Sciences*, pages 51–65. Springer Berlin Heidelberg, 1979. ISBN 3-540-09479-2.

S.-Å. Gustafson and K. O. Kortanek. Numerical treatment of a class of semi-infinite programming problems. *Naval Research Logistics Quarterly*, 20(3):477–504, 1973. doi:10.1002/nav.3800200310.

N. Hansen. The CMA evolution strategy: A comparing review. In J. Kacprzyk, J. A. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea, and J. A. Lozano, editors, *Towards a New Evolutionary Computation*, volume 192 of *Studies in Fuzziness and Soft Computing*, pages 75–102. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. doi:10.1007/3-540-32494-1_4.

S. M. Harwood. A note on generalized semi-infinite program bounding methods, 2019.

S. M. Harwood and P. I. Barton. How to solve a design centering problem. *Mathematical Methods of Operations Research*, 86(1):215–254, 2017. doi:10.1007/s00186-017-0591-3.

S. M. Harwood, D. J. Papageorgiou, and F. Trespalacios. A note on semi-infinite program bounding methods. *Optimization Letters*, 15(4):1485–1490, 2021. doi:10.1007/s11590-020-01638-4.

J. Hasenauer, S. Waldherr, K. Wagner, and F. Allgöwer. Parameter identification, experimental design and model falsification for biological network models using semidefinite programming. *IET systems biology*, 4(2):119–130, 2010. doi:10.1049/iet-syb.2009.0030.

R. Hettich. Chebyshev approximation by H-polynomials: A numerical method. *Journal of Approximation Theory*, 17(1):97–106, 1976a. doi:10.1016/0021-9045(76)90114-3.

R. Hettich. A Newton-method for nonlinear Chebyshev approximation. In R. Schaback and K. Scherer, editors, *Approximation Theory*, volume 556, pages 222–236, Berlin, Heidelberg, 1976b. Springer. doi:10.1007/BFb0087410.

R. Hettich. A comparison of some numerical methods for semi-infinite programming. In A. V. Balakrishnan, M. Thoma, and R. Hettich, editors, *Semi-Infinite Programming*, volume 15 of *Lecture Notes in Control and Information Sciences*, pages 112–125. Springer Berlin Heidelberg, 1979. ISBN 3-540-09479-2.

R. Hettich. An implementation of a discretization method for semi-infinite programming. *Mathematical Programming*, 34(3):354–361, 1986. doi:10.1007/BF01582235.

R. Hettich and W. van Honstede. On quadratically convergent methods for semi-infinite programming. In A. V. Balakrishnan, M. Thoma, and R. Hettich, editors, *Semi-Infinite Programming*, volume 15 of *Lecture Notes in Control and Information Sciences*, pages 97–111. Springer Berlin Heidelberg, 1979. ISBN 3-540-09479-2.

R. Hettich and P. Zencke. *Numerische Methoden der Approximation und semi-infiniten Optimierung*. Teubner Studienbücher: Mathematik. Teubner and Vieweg+Teubner Verlag, Stuttgart, 1982. ISBN 978-3-519-02063-9.

International Business Machines Corporation. IBM ILOG CPLEX v22.1.1, 2022.

H. Ishibashi, M. Karasuyama, I. Takeuchi, and H. Hino. A stopping criterion for Bayesian optimization by the gap of expected minimum simple regrets. In F. Ruiz, J. Dy, and J.-W. van de Meent, editors, *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 6463–6497. PMLR, 2023.

M. Jamil and X. S. Yang. A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2):150–194, 2013. doi:10.1504/IJMMNO.2013.055204.

K. Jänich. *Topologie*. Springer-Lehrbuch. Springer, Berlin, 8 edition, 2008. ISBN 978-3-540-21393-2. doi:10.1007/978-3-540-26828-4.

F. Jirasek, R. Bamler, S. Fellenz, M. Bortz, M. Kloft, S. Mandt, and H. Hasse. Making thermodynamic models of mixtures predictive by machine learning: Matrix completion of pair interactions. *Chemical Science*, 13(17):4854–4862, 2022. doi:10.1039/D1SC07210B.

D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993. doi:10.1007/BF00941892.

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998. doi:10.1023/a:1008306431147.

H. T. Jongen, J.-J. Rückmann, and O. Stein. Generalized semi-infinite optimization: A first order optimality condition and examples. *Mathematical Programming*, 83(1-3):145–158, 1998. doi:10.1007/BF02680555.

D. Jungen and A. Mitsos. An improved oracle adaption for bilevel programs. In F. Manenti and G. V. Reklaitis, editors, *Proceedings of the 34th European Symposium on Computer*

Aided Process Engineering / 15th International Symposium on Process Systems Engineering (ESCAPE-34/PSE-24), volume 53 of *Computer Aided Chemical Engineering*, pages 3301–3306. Elsevier, 2024a. doi:10.1016/B978-0-443-28824-1.50551-2.

D. Jungen and A. Mitsos. An improved oracle adaption for bilevel programs. In *34th European Symposium on Computer Aided Process Engineering / 15th International Symposium on Process Systems Engineering (ESCAPE-34/PSE-24)*, Florence, Italy, 2024b. conference presentation.

D. Jungen, H. Djelassi, and A. Mitsos. Adaptive discretization-based algorithms for semi-infinite programs with unbounded variables. *Mathematical Methods of Operations Research*, 96(1):83–112, 2022. doi:10.1007/s00186-022-00792-y.

D. Jungen, A. Zingler, H. Djelassi, and A. Mitsos. libDIPS — Discretization-based semi-Infinite and bilevel Programming Solvers. *Optimization Online*, 2023a. URL `https://optimization-online.org/?p=24914`. accessed 2025/08/20.

D. Jungen, A. Zingler, H. Djelassi, and A. Mitsos. libDIPS — Discretization-based semi-Infinite and bilevel Programming Solvers. *submitted to Mathematical Programming and Computation*, 2023b.

D. Jungen, A. Zingler, H. Djelassi, and A. Mitsos. Open-source software for the global optimization of bilevel programs and (generalized) semi-infinite programs. In *International Conference on Operations Research 2023*, Hamburg, Germany, 2023c. conference presentation.

D. Jungen, A. Zingler, H. Djelassi, and A. Mitsos. Global optimization of bilevel programs and (generalized) semi-infinite programs — an open-source platform. In *SIAM Conference on Optimization (OP23)*, Seattle, WA, USA, 2023d. conference presentation.

D. Jungen, J.-F. Laub, and A. Mitsos. Optimal experimental design for guaranteed parameter estimation: Application to a liquid-liquid equilibrium system with guaranteed thermodynamics embedded. In *13th International Conference "Fuel Science - From Production to Propulsion"*, Aachen, Germany, 2025. poster presentation.

D. Jungen, J.-F. Laub, and A. Mitsos. Optimal experimental design for guaranteed parameter estimation with rigorous formulation of the embedded liquid-liquid equilibrium. *Chemical Engineering Science*, page 123437, 2026. doi:10.1016/j.ces.2026.123437.

K. Kandasamy, J. Schneider, and B. Póczos. High dimensional Bayesian optimisation and bandits via additive models. In F. Bach and D. Blei, editors, *ICML'15: Proceedings of the 32nd International Conference on International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 295–304, Lille, France, 2015. PMLR.

N. Kanzi. Lagrange multiplier rules for non-differentiable DC generalized semi-infinite programming problems. *Journal of Global Optimization*, 56(2):417–430, 2013. doi:10.1007/s10898-011-9828-5.

K. Kawaguchi, L. P. Kaelbling, and T. Lozano-Pérez. Bayesian optimization with exponential convergence. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *NIPS'15: Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2*, pages 2809–2817, Cambridge, MA, USA, 2015. MIT Press.

A. Khajavirad and N. V. Sahinidis. A hybrid LP/NLP paradigm for global optimization relaxations. *Mathematical Programming Computation*, 10(3):383–421, 2018. doi:10.1007/s12532-018-0138-5.

J. Kim and S. Choi. On local optimizers of acquisition functions in Bayesian optimization. In F. Hutter, K. Kersting, J. Lijffijt, and I. Valera, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 675–690. Springer International Publishing, Cham, 2021. doi:10.1007/978-3-030-67661-2_40.

A. Klamt. Conductor-like screening model for real solvents: A new approach to the quantitative calculation of solvation phenomena. *The Journal of Physical Chemistry*, 99 (7):2224–2235, 1995. doi:10.1021/j100007a062.

R. Kleinberg, Y. Li, and Y. Yuan. An alternative view: When does SGD escape local minima? *arXiv*, 2018. arXiv:1802.06175v2.

P.-M. Kleniati and C. S. Adjiman. Branch-and-sandwich: a deterministic global optimization algorithm for optimistic bilevel programming problems. Part II: Convergence analysis and numerical results. *Journal of Global Optimization*, 60(3):459–481, 2014. doi:10.1007/s10898-013-0120-8.

P.-M. Kleniati and C. S. Adjiman. A generalization of the branch-and-sandwich algorithm: From continuous to mixed-integer nonlinear bilevel problems. *Computers & Chemical Engineering*, 72:373–386, 2015. doi:10.1016/j.compchemeng.2014.06.004.

G. M. Kontogeorgis, R. Dohrn, I. G. Economou, J.-C. de Hemptinne, A. ten Kate, S. Kuitunen, M. Mooijer, L. F. Žilnik, and V. Vesovic. Industrial requirements for thermodynamic and transport properties: 2020. *Industrial & Engineering Chemistry Research*, 60(13):4987–5013, 2021. doi:10.1021/acs.iecr.0c05356.

K. O. Kortanek and H. No. A central cutting plane algorithm for convex semi-infinite programming problems. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 3(4):901–918, 1993. doi:10.1137/0803047.

O. I. Kostyukova and T. V. Tchemisova. Sufficient optimality conditions for convex semi-infinite programming. *Optimization Methods and Software*, 25(2):279–297, 2010. doi:10.1080/10556780902992803.

K.-H. Küfer, O. Stein, and A. Winterfeld. Semi-infinite optimization meets industry: A deterministic approach to gemstone cutting. *SIAM News*, 41, 2008.

A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497, 1960. doi:10.2307/1910129.

P. Lemonidis. *Global Optimization Algorithms for Semi-Infinite and Generalized Semi-Infinite Programs*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2008.

T. Leon and E. Vercher. A purification algorithm for semi-infinite programming. *European Journal of Operational Research*, 57(3):412–420, 1992. doi:10.1016/0377-2217(92)90353-B.

C. Li, D. R. de Celis Leal, S. Rana, S. Gupta, A. Sutti, S. Greenhill, T. Slezak, M. Height, and S. Venkatesh. Rapid Bayesian optimisation for synthesis of short polymer fiber materials. *Scientific Reports*, 7(1), 2017. doi:10.1038/s41598-017-05723-0.

D.-H. Li, L. Qi, J. Tam, and S.-Y. Wu. A smoothing Newton method for semi-infinite programming. *Journal of Global Optimization*, 30(2-3):169–194, 2004. doi:10.1007/s10898-004-8266-z.

S. Li, K. Li, and W. Li. "Why Not Looking backward?" a robust two-step method to automatically terminate Bayesian optimization. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *NIPS '23: Proceedings of the 37th International Conference on Neural Information Processing Systems*, volume 36, pages 43435–43446, Red Hook, NY, USA, 2023. Curran Associates, Inc.

S. J. Li. *Semi-infinite programming and semi-definite optimization problems*. Ph.D. thesis, Hong Kong Polytechnic University, 2003.

C.-J. Lin, S.-C. Fang, and S.-Y. Wu. A dual affine scaling based algorithm for solving linear semi-infinite programming problems. In P. Pardalos, R. Horst, D.-Z. Du, and J. Sun, editors, *Advances in Optimization and Approximation*, volume 1 of *Nonconvex Optimization and Its Applications*, pages 217–234. Springer US, Boston, MA, USA, 1994. ISBN 978-1-4613-3631-0.

W. Liu and C. Wang. A smoothing Levenberg–Marquardt method for generalized semi-infinite programming. *Computational and Applied Mathematics*, 32(1):89–105, 2013. doi:10.1007/s40314-013-0013-y.

Y. Liu, K. L. Teo, and S. Ito. A dual parameterization approach to linear-quadratic semi-infinite programming problems. *Optimization Methods and Software*, 10(3):471–495, 1999. doi:10.1080/10556789908805725.

C. G. Lo Bianco and A. Piazzi. A hybrid algorithm for infinitely constrained optimization. *International Journal of Systems Science*, 32(1):91–102, 2001. doi:10.1080/00207720121051.

M. López and G. Still. Semi-infinite programming. *European Journal of Operational Research*, 180(2):491–518, 2007. doi:10.1016/j.ejor.2006.08.045.

S. Maitra. Adaptive Bayesian optimization algorithm for unpredictable business environments. In *ISMSI '24: Proceedings of the 2024 8th International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence*, pages 78–85, New York, NY, USA, 2024. Association for Computing Machinery (ACM). doi:10.1145/3665065.3665078.

A. Makarova, H. Shen, V. Perrone, A. Klein, J. B. Faddoul, A. Krause, M. Seeger, and C. Archambeau. Automatic termination for hyperparameter optimization. In I. Guyon, M. Lindauer, M. van der Schaar, F. Hutter, and R. Garnett, editors, *Proceedings of the First International Conference on Automated Machine Learning*, volume 188:7 of *Proceedings of Machine Learning Research*, pages 1–21. PMLR, 2022. URL `https://proceedings.mlr.press/v188/makarova22a.html`.

A. Marcilla, M. M. Olaya, M. D. Serrano, and J. A. Reyes-Labarta. Aspects to be considered for the development of a correlation algorithm for condensed phase equilibrium data of ternary systems. *Industrial & Engineering Chemistry Research*, 49(20):10100–10110, 2010. doi:10.1021/ie1010383.

A. Marcilla, J. A. Reyes-Labarta, and M. M. Olaya. Should we trust all the published LLE correlation parameters in phase equilibria? Necessity of their assessment prior to publication. *Fluid Phase Equilibria*, 433:243–252, 2017. doi:10.1016/j.fluid.2016.11.009.

A. Marcilla, M. Martínez-Rodríguez, and M. M. Del Olaya. The reliability of published LLE correlation parameters in phase equilibria: A continuing challenge. *Journal of Chemical & Engineering Data*, 2025. doi:10.1021/acs.jced.5c00107.

A. Marcilla Gomis. G^E models and algorithms for condensed phase equilibrium data regression in ternary systems: Limitations and proposals. *The Open Thermodynamics Journal*, 5(1):48–62, 2011. doi:10.2174/1874396X01105010048.

A. Marendet, A. Goldsztejn, G. Chabert, and C. Jermann. A standard branch-and-bound approach for nonlinear semi-infinite problems. *European Journal of Operational Research*, 282(2):438–452, 2020. doi:10.1016/j.ejor.2019.10.025.

S. Markidis. The old and the new: Can physics-informed deep-learning replace traditional linear solvers? *arXiv*, 2021. arXiv:2103.09655v2.

A. Marousi and V. M. Charitopoulos. Global and robust optimisation for non-convex quadratic programs. *arXiv*, 2025. arXiv:2503.07310.

S. W. Marvel and C. M. Williams. Set membership experimental design for biological systems. *BMC systems biology*, 6:21, 2012. doi:10.1186/1752-0509-6-21.

G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I — convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976. doi:10.1007/BF01580665.

M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239, 1979. doi:10.2307/1268522.

S. Mehrotra and D. Papp. A cutting surface algorithm for semi-infinite convex programming with an application to moment robust optimization. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 24(4):1670–1697, 2014. doi:10.1137/130925013.

M. L. Michelsen. The isothermal flash problem. Part I. Stability. *Fluid Phase Equilibria*, 9 (1):1–19, 1982a. doi:10.1016/0378-3812(82)85001-2.

M. L. Michelsen. The isothermal flash problem. Part II. Phase-split calculation. *Fluid Phase Equilibria*, 9(1):21–40, 1982b. doi:10.1016/0378-3812(82)85002-4.

M. Milanese and G. Belforte. Estimation theory and uncertainty intervals evaluation in presence of unknown but bounded errors: Linear families of models and estimators. *IEEE Transactions on Automatic Control*, 27(2):408–414, 1982. doi:10.1109/TAC.1982.1102926.

M. Milanese and A. Vicino. Optimal estimation theory for dynamic systems with set membership uncertainty. *Automatica*, 27(6):997–1009, 1991. doi:10.1016/0005-1098(91)90134-N.

R. Misener and C. A. Floudas. ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 59(2):503–526, 2014. doi:10.1007/s10898-014-0166-2.

A. Mitsos. *Man-Portable Power Generation Devices: Product Design and Supporting Algorithms*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006.

A. Mitsos. Test set of semi-infinite programs, 2009. URL `https://www.avt.rwth-aachen.de/cms/AVT/Forschung/Systemverfahrenstechnik/~kpdo/A-Test-Set-of-Semi-Infinite-Programs/?lidx=1`. accessed 2023/08/17.

A. Mitsos. Global solution of nonlinear mixed-integer bilevel programs. *Journal of Global Optimization*, 47(4):557–582, 2010. doi:10.1007/s10898-009-9479-y.

A. Mitsos. Global optimization of semi-infinite programs via restriction of the right-hand side. *Optimization*, 60(10-11):1291–1308, 2011. doi:10.1080/02331934.2010.527970.

A. Mitsos and P. I. Barton. A dual extremum principle in thermodynamics. *AIChE Journal*, 53(8):2131–2147, 2007. doi:10.1002/aic.11230.

A. Mitsos and A. Tsoukalas. Global optimization of generalized semi-infinite programs via restriction of the right hand side. *Journal of Global Optimization*, 61(1):1–17, 2015. doi:10.1007/s10898-014-0146-6.

A. Mitsos, P. Lemonidis, and P. I. Barton. Global solution of bilevel programs with a nonconvex inner program. *Journal of Global Optimization*, 42(4):475–513, 2008a. doi:10.1007/s10898-007-9260-z.

A. Mitsos, P. Lemonidis, C. K. Lee, and P. I. Barton. Relaxation-based bounds for semi-infinite programs. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 19(1):77–113, 2008b. doi:10.1137/060674685.

A. Mitsos, G. M. Bollas, and P. I. Barton. Model and parameter identification in phase equilibria. In J. Jeżowski and J. Thullie, editors, *19th European Symposium on Computer Aided Process Engineering*, volume 26 of *Computer Aided Chemical Engineering*, pages 597–601. Elsevier, 2009a. doi:10.1016/S1570-7946(09)70100-2.

A. Mitsos, G. M. Bollas, and P. I. Barton. Bilevel optimization formulation for parameter estimation in liquid–liquid phase equilibrium problems. *Chemical Engineering Science*, 64(3):548–559, 2009b. doi:10.1016/j.ces.2008.09.034.

A. Mitsos, M. Glass, and T. Hoffmann. Bilevel optimization algorithm for rigorous & robust parameter estimation in thermodynamics: BOARPET, 2018. URL `https://www.avt.rwth-aachen.de/cms/avt/forschung/sonstiges/software/~kvkz/boarpet/`. accessed 2025/06/19.

J. Mockus. *Bayesian Approach to Global Optimization*, volume 37 of *Mathematics and its Applications*. Springer Netherlands, Dordrecht, 1989. ISBN 978-94-009-0909-0.

K. Müller and L. D. Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theoretica Chimica Acta*, 53(1):75–93, 1979. doi:10.1007/BF00547608.

D. A. Muñoz and W. Marquardt. Robust control design of a class of nonlinear input- and state-constrained systems. *Annual Reviews in Control*, 37(2):232–245, 2013. doi:10.1016/j.arcontrol.2013.09.003.

D. A. Muñoz and W. Marquardt. Design of robust input-constrained feedback controllers for nonlinear systems. *IFAC-PapersOnLine*, 51(18):732–737, 2018. doi:10.1016/j.ifacol.2018.09.276.

J. Najman, D. Bongartz, and A. Mitsos. Relaxations of thermodynamic property and costing models in process engineering. *Computers & Chemical Engineering*, 130:106571, 2019. doi:10.1016/j.compchemeng.2019.106571.

V. Nguyen, S. Gupta, S. Rana, C. Li, and S. Venkatesh. Regret for expected improvement over the best-observed value and stopping condition. In M.-L. Zhang and Y.-K. Noh, editors, *Proceedings of the Ninth Asian Conference on Machine Learning*, volume 77 of *Proceedings of Machine Learning Research*, pages 279–294, Yonsei University, Seoul, Republic of Korea, 2017. PMLR.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer series in operations research. Springer, New York, 1999. ISBN 0-387-98793-2.

J. P. Norton. Identification and application of bounded-parameter models. *Automatica*, 23 (4):497–507, 1987. doi:10.1016/0005-1098(87)90079-3.

C. Oh, E. Gavves, and M. Welling. BOCK: Bayesian optimization with cylindrical kernels. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3868–3877. PMLR, 2018.

M. M. Olaya, J. A. Reyes-Labarta, R. Velasco, I. Ibarra, and A. Marcilla. Modelling liquid–liquid equilibria for island type ternary systems. *Fluid Phase Equilibria*, 265(1-2): 184–191, 2008. doi:10.1016/j.fluid.2007.12.010.

Osborne and G. A. Watson. A note on singular minimax approximation problems. *Journal of Mathematical Analysis and Applications*, 25(3):692–700, 1969. doi:10.1016/0022-247X(69)90266-2.

L.-P. Pang and Q. Wu. A feasible proximal bundle algorithm with convexification for nonsmooth, nonconvex semi-infinite programming. *Numerical Algorithms*, 90(1):387–422, 2022. doi:10.1007/s11075-021-01192-9.

L.-P. Pang, J. Lv, and J.-H. Wang. Constrained incremental bundle method with partial inexact oracle for nonsmooth convex semi-infinite programming problems. *Computational Optimization and Applications*, 64(2):433–465, 2016. doi:10.1007/s10589-015-9810-0.

T. P. Papalexopoulos, C. Tjandraatmadja, R. Anderson, J. P. Vielma, and D. Belanger. Constrained discrete black-box optimization using mixed-integer programming. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 17295–17322, 2022.

R. Paulavičius and C. S. Adjiman. BASBLib — a library of bilevel test problems. *Computers & Chemical Engineering*, 132:106609, 2020. doi:10.5281/zenodo.3266835.

A. Pfennig. *Thermodynamik der Gemische*. Springer, Berlin and New York, 2004. ISBN 3-540-02776-9.

E. Polak, L. Qi, and D. Sun. First-order algorithms for generalized semi-infinite min-max problems. *Computational Optimization and Applications*, 13(1/3):137–161, 1999. doi:10.1023/A:1008660924636.

M. J. D. Powell. Karmarkar's algorithm: A view from nonlinear programming, 1989. URL `https://ir.canterbury.ac.nz/items/50cd806f-63c4-456b-a66e-132f78653b64`. accessed 2023/08/17.

C. J. Price. *Non-linear semi-infinite programming*. Ph.D. thesis, University of Canterbury, New Zealand, 1992.

C. J. Price and I. D. Coope. Numerical experiments in semi-infinite programming. *Computational Optimization and Applications*, 6(2):169–189, 1996. doi:10.1007/BF00249645.

L. Pronzato. Optimal experimental design and some related control problems. *Automatica*, 44(2):303–325, 2008. doi:10.1016/j.automatica.2007.05.016.

L. Pronzato and É. Walter. Experiment design in a bounded-error context: Comparison with *D*-optimality. *Automatica*, 25(3):383–391, 1989. doi:10.1016/0005-1098(89)90006-X.

L. Pronzato and É. Walter. Experiment design for bounded-error models. *Mathematics and Computers in Simulation*, 32(5-6):571–584, 1990. doi:10.1016/0378-4754(90)90013-9.

L. Qi, S.-Y. Wu, and G. Zhou. Semismooth Newton methods for solving semi-infinite programming problems. *Journal of Global Optimization*, 27(2/3):215–232, 2003. doi:10.1023/A:1024814401713.

S. Qin, S. Jiang, J. Li, P. Balaprakash, R. C. van Lehn, and V. M. Zavala. Capturing molecular interactions in graph neural networks: A case study in multi-component phase equilibrium. *Digital Discovery*, 2(1):138–151, 2023. doi:10.1039/D2DD00045H.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 2006. ISBN 0-262-18253-X.

O. Redlich and A. T. Kister. Algebraic representation of thermodynamic properties and the classification of solutions. *Industrial & Engineering Chemistry*, 40(2):345–348, 1948. doi:10.1021/ie50458a036.

R. Reemtsen and J.-J. Rückmann, editors. *Semi-Infinite Programming*, volume 25 of *Nonconvex Optimization and Its Applications*. Springer US, Boston, MA, USA, 1998. ISBN 978-1-4419-4795-6. doi:10.1007/978-1-4757-2868-2.

E. I. Remez. *General computational methods of Chebyshev approximation: The problems with linear real parameters*. Translation series, AEC-tr-4491. U.S. Atomic Energy Comission. Division of Technical Information, Oak Ridge, TN, USA, 1962.

J. Ren and D. Sweet. Optimal initialization of batch Bayesian optimization. *arXiv*, 2024. arXiv:2404.17997v1.

H. Renon and J. M. Prausnitz. Local compositions in thermodynamic excess functions for liquid mixtures. *AIChE Journal*, 14(1):135–144, 1968. doi:10.1002/aic.690140124.

J. A. Reyes-Labarta, M. M. Olaya, R. Velasco, M. D. Serrano, and A. Marcilla. Correlation of the liquid–liquid equilibrium data for specific ternary systems with one or two partially miscible binary subsystems. *Fluid Phase Equilibria*, 278(1-2):9–14, 2009. doi:10.1016/j.fluid.2008.12.002.

A. H. G. Rinnooy Kan and G. T. Timmer. Stochastic global optimization methods part I: Clustering methods. *Mathematical Programming*, 39(1):27–56, 1987. doi:10.1007/BF02592070.

J. G. Rittig, K. Ben Hicham, A. M. Schweidtmann, M. Dahmen, and A. Mitsos. Graph neural networks for temperature-dependent activity coefficient prediction of solutes in ionic liquids. *Computers & Chemical Engineering*, 171:108153, 2023. doi:10.1016/j.compchemeng.2023.108153.

J. O. Royset, E. Polak, and A. Kiureghian. Adaptive approximations and exact penalization for the solution of generalized semi-infinite min-max problems. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 14 (1):1–34, 2003. doi:10.1137/S1052623402406777.

J.-J. Rückmann and J. Alfredo Gómez. On generalized semi-infinite programming. *Top / Sociedad Espanñla de Estadistica e Investigación Operativa*, 14(1):1–32, 2006. doi:10.1007/BF02578994.

J.-J. Rückmann and A. Shapiro. First-order optimality conditions in generalized semi-infinite programming. *Journal of Optimization Theory and Applications*, 101(3):677–691, 1999. doi:10.1023/A:1021746305759.

J.-J. Rückmann and A. Shapiro. Second-order optimality conditions in generalized semi-infinite programming. *Set-Valued and Variational Analysis*, 9(1/2):169–186, 2001. doi:10.1023/A:1011239607220.

J.-J. Rückmann and A. Shapiro. Augmented Lagrangians in semi-infinite programming. *Mathematical Programming*, 116(1-2):499–512, 2009. doi:10.1007/s10107-007-0115-7.

J.-J. Rückmann and O. Stein. On linear and linearized generalized semi-infinite optimization problems. *Annals of Operations Research*, 101(1/4):191–208, 2001. doi:10.1023/A:1010972524021.

E. Rudnick-Cohen, J. W. Herrmann, and S. Azarm. Non-convex feasibility robust optimization via scenario generation and local refinement. *Journal of Mechanical Design*, 142(5), 2020. doi:10.1115/1.4044918.

E. I. Sanchez Medina, S. Linke, M. Stoll, and K. Sundmacher. Graph neural networks for the prediction of infinite dilution activity coefficients. *Digital Discovery*, 1(3):216–225, 2022. doi:10.1039/D1DD00037C.

E. I. Sanchez Medina, S. Linke, M. Stoll, and K. Sundmacher. Gibbs–Helmholtz graph neural network: Capturing the temperature dependency of activity coefficients at infinite dilution. *Digital Discovery*, 2(3):781–798, 2023. doi:10.1039/D2DD00142J.

M. Sapkowski and T. Hofman. Problems and limitations in the calculation of liquid-liquid equilibrium. *Fluid Phase Equilibria*, 571:113823, 2023. doi:10.1016/j.fluid.2023.113823.

K. Schittkowski. Solving nonlinear programming problems with very many constraints. *Optimization*, 25(2-3):179–196, 1992. doi:10.1080/02331939208843818.

A. M. Schweidtmann and A. Mitsos. Deterministic global optimization with artificial neural networks embedded. *Journal of Optimization Theory and Applications*, 180(3):925–948, 2019. doi:10.1007/s10957-018-1396-0.

A. M. Schweidtmann, D. Bongartz, D. Grothe, T. Kerkenhoff, X. Lin, J. Najman, and A. Mitsos. Deterministic global optimization with Gaussian processes embedded. *Mathematical Programming Computation*, 13(3):553–581, 2021. doi:10.1007/s12532-021-00204-y.

F. C. Schweppe. Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE Transactions on Automatic Control*, 13(1):22–28, 1968. doi:10.1109/TAC.1968.1098790.

J. Schwientek. *Modellierung und Lösung parametrischer Packungsprobleme mittels semi-infiniter Optimierung: Angewandt auf die Verwertung von Edelsteinen*. Fraunhofer-Verl., Stuttgart, 2013. ISBN 978-3839605660.

J. Schwientek, T. Seidel, and K.-H. Küfer. A transformation-based discretization method for solving general semi-infinite optimization problems. *Mathematical Methods of Operations Research*, 93(1):83–114, 2021. doi:10.1007/s00186-020-00724-8.

T. Seidel. *Solving Semi-infinite Optimization Problems with Quadratic Rate of Convergence*. Dissertation, Fraunhofer-Institut für Techno- und Wirtschaftsmathematik ITWM, Kaiserslautern, 2020.

T. Seidel and K.-H. Küfer. An adaptive discretization method solving semi-infinite optimization problems with quadratic rate of convergence. *Optimization*, 71(8):2211–2239, 2022. doi:10.1080/02331934.2020.1804566.

T. Seidel and J. Schwientek. GSIPLib&Gen: A library and generator of general semi-infinite programming test problems: Version 1.0, 2017. URL `https://www.itwm.fraunhofer.de/en/departments/optimization/products-and-services/gsip-lib-and-gen.html`. accessed 2023/04/19.

A. G. W. Selassie. *A Coarse Solution of Generalized Semi-Infinite Optimization Problems via Robust Analysis of Marginal Functions and Global Optimization*. Dissertation, Technischen Universitat Ilmenau, 2004.

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2016. doi:10.1109/JPROC.2015.2494218.

A. Sinha, P. Malo, A. Frantsev, and K. Deb. Finding optimal strategies in a multi-period multi-leader–follower Stackelberg game using an evolutionary algorithm. *Computers & Operations Research*, 41:374–385, 2014. doi:10.1016/j.cor.2013.07.010.

A. Sinha, P. Malo, and K. Deb. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2018. doi:10.1109/TEVC.2017.2712906.

I. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. doi:10.1016/0041-5553(67)90144-9.

X. Song, Q. Zhang, C. Lee, E. Fertig, T.-K. Huang, L. Belenki, G. Kochanski, S. Ariafar, S. Vasudevan, S. Perel, and D. Golovin. The Vizier Gaussian process bandit algorithm. *arXiv*, 2024. arXiv:2408.11527v3.

J. M. Sørensen, T. Magnussen, P. Rasmussen, and A. Fredenslund. Liquid–liquid equilibrium data: Their retrieval, correlation and prediction: Part II: Correlation. *Fluid Phase Equilibria*, 3(1):47–82, 1979. doi:10.1016/0378-3812(79)80027-8.

J. Spjøtvold, P. Tøndel, and T. A. Johansen. A method for obtaing continuos solutions to multiparametric linear programs. *IFAC proceedings volumes*, 38(1):253–258, 2005. doi:10.3182/20050703-6-CZ-1902.00903.

N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger. Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, 2012. doi:10.1109/TIT.2011.2182033.

O. Stein. *Bi-Level Strategies in Semi-Infinite Programming*, volume 71 of *Nonconvex Optimization and Its Applications*. Springer US, Boston, MA, 2003. ISBN 978-1-4613-4817-7.

O. Stein and P. Steuermann. The adaptive convexification algorithm for semi-infinite programming with arbitrary index sets. *Mathematical Programming*, 136(1):183–207, 2012. doi:10.1007/s10107-012-0556-5.

O. Stein and G. Still. On generalized semi-infinite optimization and bilevel optimization. *European Journal of Operational Research*, 142(3):444–462, 2002. doi:10.1016/S0377-2217(01)00307-1.

O. Stein and A. Winterfeld. Feasible method for generalized semi-infinite programming. *Journal of Optimization Theory and Applications*, 146(2):419–443, 2010. doi:10.1007/s10957-010-9674-5.

G. Still. Generalized semi-infinite programming: Theory and methods. *European Journal of Operational Research*, 119(2):301–313, 1999. doi:10.1016/S0377-2217(99)00132-0.

G. Still. Generalized semi-infinite programming: Numerical aspects. *Optimization*, 49(3): 223–242, 2001a. doi:10.1080/02331930108844531.

G. Still. Discretization in semi-infinite programming: The rate of convergence. *Mathematical Programming*, 91(1):53–69, 2001b. doi:10.1007/s101070100239.

G. Still. Optimization problems with infinitely many constraints. *Buletinul ştiinţific al Universitatii Baia Mare, Seria B, Fascicola matematică-informatică*, 18(2):343–354, 2002.

R. L. Streit and A. H. Nuttall. A general Chebyshev complex function approximation procedure and an application to beamforming. *The Journal of the Acoustical Society of America*, 72(1):181–190, 1982. doi:10.1121/1.388002.

K. Su, C. Xu, and L. Ren. Filter trust region method for nonlinear semi-infinite programming problem. *Mathematical problems in engineering*, 2018:1–9, 2018. doi:10.1155/2018/3921592.

R. E. Swaney and I. E. Grossmann. An index for operational flexibility in chemical process design. Part I: Formulation and theory. *AIChE Journal*, 31(4):621–630, 1985. doi:10.1002/aic.690310412.

S. Tahernejad, T. K. Ralphs, and S. T. DeNegre. A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation. *Mathematical Programming Computation*, 12(4):529–568, 2020. doi:10.1007/s12532-020-00183-6.

Y. Tanaka, M. Fukushima, and T. Ibaraki. A globally convergent SQP method for semi-infinite nonlinear optimization. *Journal of Computational and Applied Mathematics*, 23 (2):141–153, 1988a. doi:10.1016/0377-0427(88)90276-2.

Y. Tanaka, M. Fukushima, and T. Ibaraki. A comparative study of several semi-infinite nonlinear programming algorithms. *European Journal of Operational Research*, 36(1): 92–100, 1988b. doi:10.1016/0377-2217(88)90010-0.

W.-T. Tang, A. Kudva, C. Tsay, and J. Paulson. Scalable global optimization of Gaussian processes using a specialized branch-and-bound algorithm. In *2024 AIChE Annual Meeting*, 2024.

M. Tawarmalani and N. V. Sahinidis. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99(3): 563–591, 2004. doi:10.1007/s10107-003-0467-6.

K. L. Teo, X. Q. Yang, and L. S. Jennings. Computational discretization algorithms for functional inequality constrained optimization. *Annals of Operations Research*, 98(1/4): 215–234, 2000. doi:10.1023/A:1019260508329.

A. Tezel Özturan. Solving generalized semi-infinite programming problems with a trust region method. *Acta Physica Polonica: A*, 128(2B):B–93–B–97, 2015. doi:10.12693/APhysPolA.128.B-93.

A. Thebelt, C. Tsay, R. M. Lee, N. Sudermann-Merx, D. Walz, T. Tranter, and R. Misener. Multi-objective constrained optimization for energy applications via tree ensembles. *Applied Energy*, 306:118061, 2022. doi:10.1016/j.apenergy.2021.118061.

R. Tichatschke and V. Nebeling. A cutting-plane method for quadratic semi infinite programming problems. *Optimization*, 19(6):803–817, 1988. doi:10.1080/02331938808843393.

M. J. Todd. Interior-point algorithms for semi-infinite programming. *Mathematical Programming*, 65(1-3):217–245, 1994. doi:10.1007/BF01581697.

A. Tsoukalas and B. Rustem. A feasible point adaptation of the Blankenship and Falk algorithm for semi-infinite programming. *Optimization Letters*, 5(4):705–716, 2011. doi:10.1007/s11590-010-0236-4.

A. Tsoukalas, B. Rustem, and E. N. Pistikopoulos. A global optimization algorithm for generalized semi-infinite, continuous minimax with coupled constraints and bi-level problems. *Journal of Global Optimization*, 44(2):235–250, 2009. doi:10.1007/s10898-008-9321-y.

T. Tsuboka and T. Katayama. Modified Wilson equation for vapor-liquid and liquid-liquid equilibria. *Journal of Chemical Engineering of Japan*, 8(3):181–187, 1975. doi:10.1252/jcej.8.181.

E. M. Turan, J. Jäschke, and R. Kannan. Bounding-focused discretization methods for the global optimization of nonconvex semi-infinite programs. *Computational Optimization and Applications*, 2025. doi:10.1007/s10589-025-00710-y.

N. van Nhu, M. Singh, and K. Leonhard. Quantum mechanically based estimation of perturbed-chain polar statistical associating fluid theory parameters for analyzing their physical significance and predicting properties. *The Journal of Physical Chemistry B*, 112(18):5693–5701, 2008. doi:10.1021/jp7105742.

A. I. F. Vaz, E. Fernandes, and M. Gomes. NSIPS: Nonlinear Semi-Infinite Programming Solver, 2004a.

A. I. F. Vaz, E. M. G. P. Fernandes, and M. P. S. F. Gomes. SIPAMPL. *ACM Transactions on Mathematical Software*, 30(1):47–61, 2004b. doi:10.1145/974781.974784.

F. G. Vázquez and J.-J. Rückmann. Extensions of the Kuhn–Tucker constraint qualification to generalized semi-infinite programming. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 15(3):926–937, 2005. doi:10.1137/S1052623403431500.

A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1): 25–57, 2006. doi:10.1007/s10107-004-0559-y.

É. Walter and H. Piet-Lahanier. Estimation of parameter bounds from bounded-error data: A survey. *Mathematics and Computers in Simulation*, 32(5-6):449–468, 1990. doi:10.1016/0378-4754(90)90002-Z.

O. Walz, H. Djelassi, A. Caspari, and A. Mitsos. Bounded-error optimal experimental design via global solution of constrained min–max program. *Computers & Chemical Engineering*, 111:92–101, 2018. doi:10.1016/j.compchemeng.2017.12.016.

O. Walz, H. Djelassi, and A. Mitsos. Optimal experimental design for optimal process design: A trilevel optimization formulation. *AIChE Journal*, 66(1):307, 2020. doi:10.1002/aic.16788.

X. Wang, Y. Jin, S. Schmitt, and M. Olhofer. Recent advances in Bayesian optimization. *ACM Computing Surveys*, 55(13s):1–36, 2023. doi:10.1145/3582078.

Z. Wang, C. Gehring, P. Kohli, and S. Jegelka. Batched large-scale Bayesian optimization in high-dimensional spaces. In A. Storkey and F. Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 745–754. PMLR, 2018.

Z. Wang, G. Dahl, K. Swersky, C. Lee, Z. Nado, J. Gilmer, J. Snoek, and Z. Ghahramani. Pre-trained Gaussian processes for Bayesian optimization. *Journal of Machine Learning Research*, 25(212):1–83, 2024.

G. A. Watson. A multiple exchange algorithm for multivariate Chebyshev approximation. *SIAM Journal on Numerical Analysis*, 12(1):46–52, 1975. doi:10.1137/0712004.

G. A. Watson. Numerical experiments with globally convergent methods for semi-infinite programming problems. In A. V. Fiacco and K. O. Kortanek, editors, *Semi-Infinite Programming and Applications*, Lecture Notes in Economics and Mathematical Systems, pages 193–205. Springer Berlin Heidelberg, Berlin, Heidelberg, 1983. ISBN 978-3-642-46477-5.

J. S. Welsh and C. R. Rojas. A scenario based approach to robust experiment design. In É. Walter, editor, *15th IFAC Symposium on System Identification*, volume 42, pages 186–191. Elsevier Science Direct - IFAC PapersOnline, 2009. doi:10.3182/20090706-3-FR-2004.00031.

W. Wiesemann, A. Tsoukalas, P.-M. Kleniati, and B. Rustem. Pessimistic bilevel optimization. *SIAM Journal on Optimization*, 23(1):353–380, 2013. doi:10.1137/120864015.

M. E. Wilhelm and M. D. Stuber. EAGO.jl: easy advanced global optimization in Julia. *Optimization Methods and Software*, 37(2):425–450, 2022. doi:10.1080/10556788.2020.1786566.

J. T. Wilson. Stopping Bayesian optimization with probabilistic regret bounds. *arXiv*, 2024. arXiv:2402.16811v2.

J. T. Wilson, F. Hutter, and M. P. Deisenroth. Maximizing acquisition functions for Bayesian optimization. In S. Bengio, H. M. Wallach, L. Hugo, K. Grauman, and N. Cesa-Bianchi, editors, *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 9906–9917, Red Hook, NY, USA, 2018. Curran Associates, Inc.

B. Winter, C. Winter, J. Schilling, and A. Bardow. A smile is all you need: Predicting limiting activity coefficients from SMILES with natural language processing. *Digital Discovery*, 1(6):859–869, 2022. doi:10.1039/D2DD00058J.

A. Winterfeld. Application of general semi-infinite programming to lapidary cutting problems. *European Journal of Operational Research*, 191(3):838–854, 2008. doi:10.1016/j.ejor.2007.01.057.

K. Wu, K. Kim, R. Garnett, and J. R. Gardner. The behavior and convergence of local Bayesian optimization. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *NIPS '23: Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 73497–73523, Red Hook, NY, USA, 2023. Curran Associates, Inc.

S.-Y. Wu, D.-H. Li, L. Qi, and G. Zhou. An iterative method for solving KKT system of the semi-infinite programming. *Optimization Methods and Software*, 20(6):629–643, 2005. doi:10.1080/10556780500094739.

Y. Xie, S. Zhang, J. Paulson, and C. Tsay. Global optimization of Gaussian process acquisition functions using a piecewise-linear kernel approximation. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL `https://openreview.net/forum?id=IZv4W76V3T`. accessed 2025/02/18.

W. Xu, M. Adachi, C. Jones, and M. A. Osborne. Principled Bayesian optimization in collaboration with human experts. In *NeurIPS: '24 The Thirty-eighth Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 2024.

Y. Xu, W. Sun, and L. Qi. On solving a class of linear semi-infinite programming by SDP method. *Optimization*, pages 1–14, 2013. doi:10.1080/02331934.2013.793325.

C. Yan, H. Du, E. Kang, D. Mi, H. Liu, and Y. You. AVEI-BO: An efficient Bayesian optimization using adaptively varied expected improvement. *Structural and Multidisciplinary Optimization*, 65(6), 2022. doi:10.1007/s00158-022-03256-3.

C. Yu, K. Lay Teo, L. Zhang, and Y. Bai. A new exact penalty function method for continuous inequality constrained optimization problems. *Journal of Industrial and Management Optimization*, 6(4):895–910, 2010. doi:10.3934/jimo.2010.6.895.

S. Žaković and B. Rustem. Semi-infinite programming and applications to minimax problems. *Annals of Operations Research*, 124(1-4):81–110, 2003. doi:10.1023/B:ANOR.0000004764.76984.30.

L. Zhang, S.-Y. Wu, and M. A. López. A new exchange method for convex semi-infinite programming. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 20(6):2959–2977, 2010. doi:10.1137/090767133.

J. L. Zhou and A. L. Tits. An SQP algorithm for finely discretized continuous minimax problems and other minimax problems with many objective functions. *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, 6 (2):461–487, 1996. doi:10.1137/0806025.

S. Zhou and A. B. Zemkoho. BiOpt: Bilevel optimization toolboxes, 2021. URL `https://biopt.github.io/`. accessed 2024/11/29.

S. Zhou, A. B. Zemkoho, and A. Tin. BOLIB: Bilevel Optimization LIBrary of Test Problems. In S. Dempe and A. B. Zemkoho, editors, *Bilevel Optimization*, volume 161 of *Springer Optimization and Its Applications*, pages 563–580. Springer, Cham, Switzerland, 2020. ISBN 978-3-030-52119-6.

Y. Zhou, J. Yang, H. Zhang, Y. Liang, and V. Tarokh. SGD converges to global minimum in deep learning via star-convex path. *arXiv*, 2019. arXiv:1901.00451v1.

C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 23(4):550–560, 1997. doi:10.1145/279232.279236.

A. Zingler. *Adaptive discretization methods for the global solution of semi-infinite optimization problems*. Dissertation, RWTH Aachen University, Aachen, Germany, 2025.

A. Zingler and A. Mitsos. Derivative-enhanced lower-bounding in adaptive discretization for the global solution of semi-infinite optimization problems. *Optimization*, pages 1–34, 2025. doi:10.1080/02331934.2025.2571756.

A. Zingler, D. Jungen, H. Djelassi, and A. Mitsos. libALE — a library for algebraic logical expression trees, 2022. URL `https://git.rwth-aachen.de/avt.svt/public/libale`.

A. Zingler, S. Fliscounakis, P. Panciatici, and A. Mitsos. Optimizing flexibility in power systems by maximizing the region of manageable uncertainties. *Optimization and Engineering*, pages 1–23, 2025a. doi:10.1007/s11081-025-09958-z.

A. Zingler, A. W. Lipow, and A. Mitsos. Discretization algorithms for generalized semi-infinite programs with coupling equality constraints under local solution stability. *Journal of Global Optimization*, 2025b. doi:10.1007/s10898-025-01515-3.