

**MIXED-INTEGER OPTIMIZATION FOR
NANOMATERIAL DESIGN AND OPTIMIZATION
UNDER UNCERTAINTY FOR NONLINEAR PROCESS
MODELS**

Submitted in partial fulfillment of the requirements for

the degree of

DOCTOR OF PHILOSOPHY

in

Chemical Engineering

NATALIE MALKA ISENBERG

B.S., Chemical Engineering, University of Pittsburgh, Pittsburgh, PA

Carnegie Mellon University
Pittsburgh, PA

December, 2021

To my parents, Ariella and Jeffrey

ACKNOWLEDGMENTS

My time spent in the Chemical Engineering department at Carnegie Mellon University (CMU) has challenged me intellectually and given me countless opportunities to learn and grow as a scientist. This would not have been possible without the exemplary community of researchers, mentors, and friends at CMU.

Firstly, I would like to extend my gratitude to my academic advisor, Chrysanthos Gounaris. Chrysanthos has taught me how to be an effective communicator and problem solver, and I learned to be a more detail-oriented person through our time working together. I am fortunate to have worked with someone as inventive and driven as Chrysanthos. And most importantly, I am thankful for his compassion whenever I struggled in this process.

Next, I would like to thank my thesis committee members: Professor Nikolaos Sahinidis, Professor Zachary Ulissi, Professor Fatma Kılınç-Karzan, Professor Debangsu Bhattacharyya, and Dr. John D. Sirola, for their time and thoughtful feedback on my thesis work.

I am also very grateful for the financial support I received from the Institute for the Design of Advanced Energy Systems (IDAES) during my studies as well as support from the Department of Energy, Office of Science Graduate Student Research Fellowship to participate in an internship at the Sandia National Laboratories in Albuquerque, New Mexico, and support from the National Science Foundation under grant CMMI 1634594.

I would also like to thank a group of outstanding collaborators with whom I had the pleasure of working over the course of my thesis work, including Dr. John Sirola, Professor Debangsu Bhattacharyya, Paul Akula, Dr. David Miller, Dr. John Eslick, Dr. Michael Taylor, Zihao Yan, Prof. Giannis Mpourmpakis, and Dr. Christopher Hanselman and Xiangyu Yin.

I would like to thank my undergraduate research mentor at the University of Pittsburgh, Professor Goetz Vesper, who mentored me through my first opportunity in hands-on research and encouraged me to pursue a research career.

While at CMU, I worked with outstanding individuals whom I also consider dear friends. From my own cohort of graduate students, I want to thank Nicholas Golio for being a good friend and confidant. From our research group, I cannot thank Anirudh Subramanyam, Nikos Lappas, Christopher Hanselman, and Akang Wang enough for mentoring me in my first years as a doctoral student. These individuals set a golden standard

for me in my work which I will always strive to meet. And for the rest of the research group: Hua Wang, Aliakbar Izadkhah, Xiangyu Yin, William Strahl, Ilayda Akkor and Jason Sherman, I am so glad we got the chance to work together and share coffees and white board discussions.

In addition to groupmates, I was very fortunate to be at CMU the same time as many other wonderful people in the process systems engineering (PSE) group: Saif, David B., David T., Marissa, Yixin, Devin, Carlos, and many others, thank you for the brainstorming sessions and other good times. Overall, working in the PSE group at CMU provided an amazing learning experience where I got to meet and work beside some of the most talented folks I have ever met.

I need to thank my partner of eight years, Connor, for all of his support during this process. Connor provided sanity checks, encouragement, and was often my first resource for working through problems (especially if they were programming related).

Finally, I would like to thank my family. My parents, Ariella and Jeffrey, who are so over-the-top supportive of me, they would be proud of me for remembering to tie my shoes on any given day. My siblings, Ben and Danielle, who are my best friends. My family in Israel: Chana, Clara, Yossi, and Nava and everyone else who have been constantly asking me when I will be done with school for the past five years. To my cousin James, my uncles Israel and Shlomo, my grandparents Jean & Abe, and Malka & Emil. I wish you were here to share this with me. And I thank you all for setting me on my current path.

*Natalie Malka Isenberg
Pittsburgh, PA
September 17, 2021*

ABSTRACT

In the first part of this work, we consider small nanoparticles, a.k.a. *nanoclusters*, of transition metals. Transition metal nanoclusters have been studied extensively for a wide range of applications due to their highly tunable properties dependent on size, structure, and composition. For these small particles, there has been considerable effort towards theoretically predicting what is the most energetically favorable arrangement of atoms. To that end, we develop a computational framework that couples density-functional theory calculations with mathematical optimization modeling to identify highly stable, mono-metallic transition metal nanoclusters at various sizes.

Next, we devise a novel computational framework for the robust optimization of highly nonlinear, non-convex models that possess uncertain data. The proposed method is a generalization of a robust cutting-set algorithm that can handle models containing irremovable equality constraints, as is often the case with models in the process systems engineering domain. Additionally, we accommodate general forms of decision rules to facilitate recourse in second-stage degrees of freedom. Our proposed approach is demonstrated on three process flowsheet models, including a relatively complex model for amine-based CO₂ capture. Finally, we propose an open-source robust optimization solver implementation of our cutting-set approach called PyROS. PyROS is a Python-based robust optimization meta-solver for solving non-convex, two-stage optimization models using adjustable robust optimization. The PyROS solver enables facile robust optimization tasks given a deterministic model and description of uncertainty. With each of the applications presented here, we illustrate that mathematical optimization modeling and algorithms can be effectively utilized to address open problems in engineering.

PUBLICATIONS

The following papers related to the work presented in this thesis have been published, submitted, or are in preparation to be submitted in peer-reviewed journals:

1. **N. M. Isenberg**, M. G. Taylor, Z. Yan, C. L. Hanselman, G. Mpourmpakis and C. E. Gounaris. "Identification of optimally stable nanocluster geometries via mathematical optimization and density-functional theory." *Molecular Systems Design & Engineering* 5 (1) (2020), pp. 232-244.
2. **N. M. Isenberg**, P. Akula, J. C. Eslick, D. Bhattacharyya, D. C. Miller, and C. E. Gounaris. "A generalized cutting-set approach for nonlinear robust optimization in process systems engineering." *AIChE Journal* 67 (5) (2021), e17175.
3. **N. M. Isenberg**, J. D. Sirola, and C. E. Gounaris. "PyROS: a Pyomo Robust Optimization Solver for nonlinear robust optimization." Manuscript in preparation.

CONTENTS

1	INTRODUCTION	1
1.1	Nanocluster Structure Elucidation	1
1.2	Thesis Aims for Optimal Nanocluster Design	2
1.3	Robust Optimization for Nonlinear Problems	3
1.4	Thesis Aims for Nonlinear Robust Optimization	5
2	OPTIMAL DESIGN OF TRANSITION METAL NANOCCLUSERS	7
2.1	Introduction	7
2.2	Nanocluster Geometry Optimization	7
2.3	Mathematical Optimization-based Design	9
2.3.1	Optimization Model	10
2.3.2	Concave Objective Function	11
2.3.3	Symmetry Breaking	14
2.3.4	Improving Numerical Tractability	14
2.4	Optimal Designs	16
2.5	Conclusions	17
2.6	Notation	19
2.7	Appendix	20
3	METAL-SPECIFIC NANOCCLUSER COHESIVE ENERGY	22
3.1	Introduction	22
3.2	Metal-specific Nanocluster Geometry Optimization	22
3.3	Metal-specific Optimal Designs	25
3.4	Conclusions	27
3.5	Notation	30
3.6	Appendix	31
4	MODELING NANOCCLUSER GEOMETRY RELAXATIONS	33
4.1	Introduction	33
4.2	Cohesive Energy and Relaxed Structures	34
4.3	MILP Modeling for Relaxed Nanoclusters	34
4.4	Conclusions	36
4.5	Notation	37
5	THE GENERALIZED ROBUST CUTTING-SET ALGORITHM	38
5.1	Introduction	38
5.2	The Robust Counterpart to a Process Design Formulation	39
5.2.1	The Generalized Robust Cutting-Set Algorithm	42

5.2.2	Decision Rules	46
5.3	Implementation Details	47
5.3.1	Solving Master Problems	48
5.3.2	Separation Approach	49
5.3.3	Decision Rules Polishing	50
5.4	Evaluation of Robust Solution Quality	52
5.5	Conclusions	53
5.6	Appendix	54
5.6.1	Convergence Proof	54
5.7	Notation	56
6	NONLINEAR ROBUST OPTIMIZATION CASE STUDIES	57
6.1	Introduction	57
6.2	Case Study I: Reactor-Separator	58
6.2.1	Case Study I Results	59
6.3	Case Study II: Reactor-Heater	63
6.3.1	Case Study II Results	64
6.4	Case Study III: MEA-solvent CO ₂ Separation Flowsheet	67
6.4.1	Case Study III Results	69
6.5	Discussion on Choosing Form of Recourse Policy	74
6.6	Conclusions	76
6.7	Appendix	77
6.7.1	Reactor Separator Model	78
6.7.2	Reactor Heater Model	83
6.7.3	CO ₂ Capture Flowsheet Model	86
7	PYROS: THE PYOMO ROBUST OPTIMIZATION SOLVER	94
7.1	Introduction	94
7.2	PyROS Methodology	96
7.2.1	Polynomial Coefficient Matching	97
7.2.2	PyROS Separation Procedure	97
7.3	PyROS Solver Interface	98
7.3.1	Uncertainty Sets	99
7.3.2	PyROS Options	103
7.3.3	Calling PyROS	105
7.4	Tractability and Performance	110
7.5	Conclusions	112
7.6	Appendix	113
7.6.1	Construction of Benchmark Problems	113
7.6.2	PyROS Termination Conditions	115
7.6.3	Pyomo Subsolver Statuses in PyROS	115
8	CONCLUSIONS AND FUTURE WORK	118
8.1	Contributions	118

8.2 Future Directions	120
---------------------------------	-----

BIBLIOGRAPHY	124
--------------	-----

LIST OF TABLES

Table 2.1	Representative optimally-cohesive nanocluster geometries, as predicted by the MILP-model maximizing the SRB cohesive energy.	17
Table 3.1	Comparison of a few optimal nanocluster structures as determined by the SRB function (first row) and the new optimal structures determined by the corrected models for different metals (second row).	26
Table 6.1	Optimal values of first-stage variables and costs for the reactor-separator model.	59
Table 6.2	Optimal values of second-stage variables and costs, under the nominal realization of uncertainty, for the reactor-separator model.	60
Table 6.3	Expected values and standard deviations of second-stage variables and costs for the reactor-separator model.	60
Table 6.4	Total number of iterations and CPU time spent within the GRCS algorithm when addressing the reactor-separator model. The total time includes the time to execute the algorithm and subordinate solver calls. The percentage of time spent on master and separation problems only includes the total execution time for the respective subordinate solvers.	61
Table 6.5	Optimal values of first-stage variables and costs for the reactor-heater model.	64
Table 6.6	Optimal values of second-stage variables and costs, under the nominal realization of uncertainty, for the reactor-heater model.	65
Table 6.7	Expected values and standard deviations of second-stage variables and costs for the reactor-heater model.	65
Table 6.8	Total number of iterations and CPU time spent within the GRCS algorithm when addressing the reactor-heater model. The total time includes the time to execute the algorithm and subordinate solver calls. The percentage of time spent on master and separation problems only includes the total execution time for the respective subordinate solvers.	66

Table 6.9	Optimal values of first-stage variables and costs for the CO ₂ capture flowsheet model.	70
Table 6.10	Optimal values of second-stage control and other key variables, as well as total and second-stage costs, evaluated under the nominal realization of uncertainty, for the CO ₂ capture flowsheet model. .	72
Table 6.11	Expected values and standard deviations of second-stage control and other key second-stage variables, and second-stage costs, for the CO ₂ capture flowsheet model.	72
Table 6.12	Total number of iterations and CPU time spent within the GRCS algorithm when addressing the CO ₂ capture flowsheet model. The total time includes the time to execute the algorithm and subordinate solver calls. The percentage of time spent on master and separation problems only includes the total execution time for the respective subordinate solvers.	73
Table 6.13	Evolution of robust feasibility for the reactor-separator design across different recourse policies. The [†] annotations refer to non-robust solutions that happened to remain feasible under all chosen realization samples; in these cases, we instead report the magnitude of violations, as identified by the respective separation problems.	82
Table 6.14	Evolution of robust feasibility for the reactor-heater design across different recourse policies. The [†] annotations refer to non-robust solutions that happened to remain feasible under all chosen realization samples; in these cases, we instead report the magnitude of violations, as identified by the respective separation problems.	85
Table 6.15	Evolution of robust feasibility for the CO ₂ capture flowsheet across different recourse policies. The [†] annotations refer to non-robust solutions that happened to remain feasible under all chosen realization samples; in these cases, we instead report the magnitude of the violation, as identified by the respective separation problem.	93
Table 7.1	Capabilities of robust optimization tools for handling nonlinear uncertain optimization problems. .	95

Table 7.2	Tabulated information regarding pre-implemented uncertainty set classes in PyROS, including uncertainty set name, mathematical representation as a constraint, and inferred bounds.	102
Table 7.3	Details regarding models used to derive robust optimization test problems for benchmarking PyROS.	110
Table 7.4	Example instance information for derived benchmark problems from base problem <i>s381</i>	111
Table 7.5	Overall performance statistics (statuses at termination and average time and iterations) for benchmark problems solved via PyROS.	112
Table 7.6	Robust worst-case objective values ζ^* for 5-D box uncertainty sets for a 353 problem instance.	112
Table 7.7	Uncertainty set information to generate the uncertainty sets used in the PyROS benchmarking study.	114
Table 7.8	PyROS Return Statuses.	115
Table 7.9	Mapping Pyomo sub-solver termination conditions to PyROS master and separation problem actions.	116

LIST OF FIGURES

Figure 2.1	Square Root Bond-cutting model for cohesive energy (dimensionless) of an FCC atom i , plotted against CN_i . Also shown are the secant lines used to exactly represent the evaluations of cohesive energy at integral CN_i values.	12
Figure 2.2	Best solutions and best upper bounds at termination for $N = 4 - 100$, using the MILP model without (2.2a) and with (2.2b) algorithmic enhancements. . .	18
Figure 2.3	Most cohesive structure according to the SRB cohesive energy function for every N	21
Figure 3.1	Parity plots between cohesive energies calculated by the SRB model (y-axis), with and without metal-specific corrections, and by DFT (x-axis) for various metals.	28
Figure 3.2	Metal-specific corrections to the original SRB model for cohesive energy, identified via constrained regression based on DFT predicted values.	29
Figure 3.3	New optima at various N , as determined by the Cu-corrected function for cohesive energy.	31
Figure 3.4	New optima at various N , as determined by the Au-corrected function for cohesive energy.	31
Figure 3.5	New optima at various N , as determined by the Ag-corrected function for cohesive energy.	32
Figure 3.6	New optima at various N , as determined by the Pd-corrected function for cohesive energy.	32
Figure 3.7	New optima at various N , as determined by the Pt-corrected function for cohesive energy.	32
Figure 5.1	The generalized robust cutting-set algorithm.	44
Figure 5.2	Implementation details of the generalized robust cutting-set algorithm.	48
Figure 6.1	Flowsheet (a) and reaction mechanism (b) representing the reactor-separator system considered in Section 6.2, as adapted from Grossmann and Sargent [46].	58
Figure 6.2	Evolution during the GRCS algorithm of the robust feasibility of the reactor-separator designs using the static approximation policy.	62

Figure 6.3	Flowsheet representing the reactor-heater system considered in Section 6.3, as adapted from Halemane and Grossmann [50]	63
Figure 6.4	Evolution during the GRCS algorithm of the robust feasibility of the reactor-heater designs using the static approximation policy.	66
Figure 6.5	Flowsheet representing the MEA-based CO ₂ capture flowsheet considered in Section 6.4, as adapted from Mores et al. [87].	67
Figure 6.6	Evolution during the GRCS algorithm of the robust feasibility of the MEA-based CO ₂ capture flowsheet using the static approximation policy.	74
Figure 6.7	Evolution during the GRCS algorithm of the robust feasibility of the reactor-separator designs using affine decision rules.	80
Figure 6.8	Evolution during the GRCS algorithm of the robust feasibility of the reactor-separator designs using quadratic decision rules.	81
Figure 6.9	Evolution during the GRCS algorithm of the robust feasibility of the reactor-heater designs using affine decision rules.	85
Figure 6.10	Evolution during the GRCS algorithm of the robust feasibility of the reactor-heater designs using quadratic decision rules.	85
Figure 6.11	Evolution during the GRCS algorithm of the robust feasibility of the MEA-based CO ₂ capture flowsheet using affine decision rules.	92
Figure 8.1	Illustration of how initialization of master problems (MP_{k+1}) in the GRCS can be improved based on information from previous master (MP_k) and separation problems (SP_k).	122

INTRODUCTION

Small transition metal nanoclusters possess properties that are highly dependent on size, shape, and composition. Optimizing these material parameters can lead to drastically improved material performance for application in catalysis,[2, 9, 63, 113] electronics,[57] and biological systems.[29] One key research question in the study of small transition metal nanoclusters is to identify the most stable morphology for a nanocluster of exactly N metal atoms.[6] While it is possible to determine stable sizes of small nanoclusters experimentally by measuring the frequency in which those sizes appear during synthesis, morphological trends in small clusters are difficult to elucidate since small particles cannot be observed in high enough resolution to discern specific atomic arrangements.[79] Therefore, understanding small nanocluster morphology requires complimentary theoretical calculations and predictions.

1.1 NANOCUSTER STRUCTURE ELUCIDATION

In order to determine the most stable structure for a nanocluster, one must identify the configuration of atoms with the lowest total energy, as assessed with some empirical or semi-empirical function, or some *ab initio* calculation for the potential energy.[7]

One approach is to use a meta-heuristic type algorithm for identifying a putative minimum of the potential energy surface (PES) to identify a low potential energy arrangement for a N -atom nanocluster. The selection of a PES for a metallic nanocluster presents another heuristic choice, as there are many variations among PES functions. Examples of PES functions utilized in the literature include the n -body Gupta[40, 84] potential, Morse potential[32, 99], or Lennard-Jones pair potentials.[31, 100, 126] The Gupta potential is known to provide the most accurate predictions for metallic clusters because it is derived from the second-moment approximation of the electron density of states in the tight-binding model, unlike the Morse and Lennard-Jones potentials which are simpler pairwise additive potentials. Density-functional theory (DFT) based methods have also been

used as a means for predicting nanocluster structures at energy minima for both mono- and bi-metallic systems.[16, 127] Utilizing DFT for certain transition metals can capture energy phenomena related to relativistic effects at small sizes, as is the case with small gold nanoclusters.[48]

The task of identifying a potential energy minimum given a potential energy representation becomes an optimization problem in the continuous space of interatomic distances to compute attractive and repulsive forces. Solving these optimization problems is challenging, as determining the global minimum energy structure for a nanocluster of N atoms is a highly combinatorial problem that is a member of the NP-hard complexity class of computational problems [135]. This means that, in principle, one might be required to evaluate the energy of each possible arrangement of atoms, which is generally an intractable task. Despite such challenges, sophisticated meta-heuristic search algorithms have been employed to search for minimum energy mono- and bi-metallic nanocluster morphologies.[30, 54] Approaches of this type include genetic algorithms,[28] basin-hopping,[8] and simulated annealing.[38] *Ab initio* molecular dynamics simulations have also been applied to minimizing potential energy surfaces for nanoclusters. [124] While the reported structures are in general highly stable, and may indeed be ground state geometries, they are not provably optimal against the stability metric used because meta-heuristic search algorithms use arbitrary termination criteria that lack guarantees of searching the entire solution space. Without a proof of optimality upon completion, such algorithms might converge to a *local* minimum, as opposed to the true, *global* minimum.

And while a *good* locally optimal solution may be sufficient for further computational studies, the local optimum may be entirely morphologically different than the global optimum. Therefore, it is important to consider methods for identifying the true globally optimal solution so that we can learn trends in morphology among highly stable nanoclusters. This is what motivates the approach proposed in this thesis for mathematical optimization and modeling of transition metal nanoclusters. Such a treatment can provide a proof of global optimality, which is a novel feature of the approach.

1.2 THESIS AIMS FOR OPTIMAL NANOCUSTER DESIGN

In this thesis, we propose a complementary approach for optimal nanocluster design that is a mathematical optimization-based framework. In this framework, we formulate a mixed-integer linear programming (MILP) model for determining minimum energy structures of three-dimensional, mono-metallic nanoclusters. The distinctive feature of our approach is that, when solved to algorithmic termination by an appropriate MILP numerical

solver, the model returns a low energy nanocluster that is guaranteed to be globally optimal up to the accuracy of the energy functional used and the flexibility afforded by the explicitly encoded lattice. Next, we evaluate the optimal predicted structures via density-functional theory (DFT) for their *true* cohesive energies. These DFT predicted cohesive energies are then used to further improve structure-function predictions for use in optimization. This work-flow is extensible to other systems, and has been used by Yin et al. [141] to identify optimally cohesive bimetallic transition metal nanocluster structures.

The contributions of the present work are three-fold. First, we use rigorous mathematical modeling and optimization to identify highly cohesive mono-metallic nanocluster geometries at various sizes. Next, we use density-functional theory cohesive energy predictions to regress metal-specific models for nanocluster cohesive energy. And finally, we conduct a comprehensive computational study to identify sequences of minimum energy structures unique to different metals and for a wide range of sizes (number of atoms).

The relevant chapters of this thesis are organized as follows: In Chapter 2, we present a mathematical modeling approach for identifying optimal N -atom mono-metallic nanoclusters via cohesive energy. In Chapter 3, we build a work-flow between mathematical optimization and density-functional theory for devising metal-specific cohesive energy functions. Finally, Chapter 8 provides a summary of the key contributions of the work presented in these chapters, as well as future research directions.

1.3 ROBUST OPTIMIZATION FOR NONLINEAR PROBLEMS

Data in mathematical optimization models are often subject to some level of uncertainty. The latter can originate from measurement errors and the use of empirical data, economic stochasticity (e.g., market prices), or variations in the process environment (e.g., feedstock quality). For chemical process models, uncertainty most often originates from a lack of knowledge regarding underlying physical properties, such as thermodynamic and kinetic properties, or constants associated with the prevailing heat and mass transport phenomena. In the context of process systems engineering, where critical design and control decisions are made by solving models that are subject to such uncertainties, it is especially important to understand the effects of parametric uncertainties on the performance of the chosen solutions, and if significant, to mitigate uncertainty during the optimization phase so that any resulting design is safely and robustly implemented.

Due to the ubiquitous existence of uncertainty in process systems engineering models, there exists a breadth of literature in developing and

applying risk-averse optimization approaches. Early work in the field introduced two-stage nonlinear programming formulations with bounded uncertain parameters[46], two-stage stochastic programming approaches[1, 95], chance-constrained optimization[132], and flexibility analysis formulations and algorithms[45, 111][44] for handling process systems engineering design under uncertainty. More recently, work by Li and Grossmann [75] proposed a novel algorithmic approach for solving convex, nonlinear stochastic programming problems with mixed-integer recourse with applications in batch plant design and planning with uncertainties in demands and prices. Kelley, Baldick, and Baldea [65] developed a framework to account for uncertainty via chance-constraints in dynamically-constrained scheduling problems, demonstrating this methodology on a complex air separation unit. Finally, Wang et al. [128] devised an approach for handling parameter uncertainty in solid-liquid batch reactors wherein worst-case values for parameters in the uncertainty space are iteratively added as scenarios to the optimal control problem.

In the chemical process design context, optimization models typically possess complex nonlinearities, including many non-convexities originating from physical and chemical equations. These nonlinearities can be in terms of both decision variables and uncertain parameters in the model, meaning that traditional duality-based reformulation methods in the RO literature may lead to either overly conservative or non-robust solutions due to violations of certain underlying assumptions. To address this, Bertsimas, Nohadani, and Teo [14] proposed a local search algorithm for identifying robust feasible solutions to uncertain optimization problems with non-convex inequality constraints. Additionally, there have been recent advances in the development of novel methods and applications of RO methods to nonlinear process systems engineering models, including general nonlinear programming robust counterpart formulations[148], robust counterparts with local linearization of nonlinear uncertain constraints and a novel sampling algorithm[143], application to the pooling problem utilizing a cutting-plane solution algorithm[133], application to water treatment network operation[62], robust counterpart derivation for the synthesis of fuel refineries under cost uncertainty[81], and design and operation of process systems with resilience to disruptive events[43], to name but a few.

We acknowledge that there remains a practical need to develop general RO approaches that can identify robust solutions in nonlinear, non-convex process models that consist mostly of equality constraints or state equations, which cannot be readily simplified or solved out of the formulation. Such constraints ubiquitously arise in process design models due to the extensive use of empirical property correlations and the presence of recycle streams in process flowsheets, among other reasons. For these classes of

problems, there is currently no RO solution approach that guarantees robust solutions against the entire uncertainty space. To that end, we propose an extension to the robust-cutting plane method proposed by Mutapcic and Boyd [91], which we refer to as the generalized robust cutting-set (GRCS) approach [59]. We aim for the latter to be capable of certifying fully robust solutions to non-convex optimization problems with a large contingent of equality constraints, as well as be valid for models with nonlinearities and non-convexities from both the decision variables and uncertain parameters.

The GRCS algorithm has two key features. First, it handles equality constraints systematically and without reformulation. To achieve this, the algorithm sequentially hedges against realizations of parametric uncertainties by maintaining copies of state variables and equations for each added uncertain parameter realization to ensure the feasibility of the master problem state equations. Second, it uses general decision rules, as applied in the area of adjustable robust optimization[10], in order to handle control variables, which can be thought of as second-stage variables in process design contexts. We also present *PyROS*, an open-source, Python-based implementation of a robust optimization solver for nonlinear models. PyROS utilizes the GRCS algorithm to automatically identify robust solutions to general uncertain, nonlinear optimization models. PyROS features an interface for specifying uncertainty sets, first- and second-stage variables, and decision rule relationships. Through PyROS, we provide a novel capability to easily study the effects of uncertainty and the nature of robust solutions for general nonlinear models.

1.4 THESIS AIMS FOR NONLINEAR ROBUST OPTIMIZATION

The key contributions of this thesis are presented here. First, we provide a formal robust optimization framework in the context of complex, highly non-convex, equality-constrained process design models via the GRCS algorithm. Second, we demonstrate the effective use of nonlinear decision rule functions in decreasing the adaptivity gap in solving the two-stage problem, i.e., increasing second-stage flexibility to approach true two-stage optimality. We illustrate the tractability of our proposed approach on a number of case studies, including a complex equation-oriented flowsheet model for an amine solvent-based carbon capture process. Finally, we present PyROS, a robust optimization solver capability within the Python algebraic modeling language, Pyomo, which implements our novel RO approach.

The relevant chapters of this thesis are organized as follows: In Chapter 5, we explain the details of our proposed RO approach, including the problem formulation, solution algorithm, and implementation details. We then

showcase several case studies in Chapter 6 to illustrate the performance of the GRCS algorithm on real process systems models. Next, the GRCS algorithm-based solver PyROS is introduced and discussed in Chapter 7. Finally, we conclude with some remarks on contributions and future work in Chapter 8.

OPTIMAL DESIGN OF TRANSITION METAL NANOCLUSTERS

2.1 INTRODUCTION

It is well known that the physical and chemical behaviors of small nanoparticles are governed by size, geometry, as well as composition or the presence of a support. For example, small supported gold nanoparticles (Au_n , $n \leq 20$) were shown by Sanchez et al. [103] to display enhanced catalytic activity for low temperature oxidation of CO. Furthermore, it was determined that this enhanced catalytic behavior began at nanoclusters where $n \geq 8$. Additional research by Cai, Guo, and Liu [18] has shown that the arrangement of supported Au atoms in either a single 2-D layer or 3-D nanoclusters has a direct impact on catalytic activity for CO oxidation. This illustrates the impact that tunable properties, such as size and geometry, can have on nanocluster performance. This strongly motivates efforts to theoretically predict and study trends in structure and functionality for small metallic nanoclusters.

In this chapter, we will consider the problem of identifying the optimal arrangement of N atoms in mono-metallic transition metal nanoclusters. We will devise a mixed-integer linear program (MILP) for identifying the optimal placement of N atoms against an objective of maximizing the stability of the nanocluster. Solving the resulting MILP models at various N values retrieves the proven optimal arrangement of atoms for the selected objective function of minimizing nanocluster energy.

2.2 NANOCLUSTER GEOMETRY OPTIMIZATION

The geometry of a minimum energy nanocluster of a given size is assumed in this chapter to be the one that attains the maximum cohesive energy (E_{coh}). The cohesive energy is chosen as a good proxy for the particle's overall stability because it measures the cumulative strength of interatomic bonding between atoms. Cohesive energy is an important, size-dependent

property of nanomaterials, and can be used to understand thermal stability properties such as melting point and vacancy formation energy. [119, 151]

As a first pass, our mathematical model utilizes an analytical cohesive energy function first proposed by Tománek, Mukherjee, and Bennemann [115], which stipulates that the contribution of each atom to the total cohesive energy of a particle in a metallic nanocluster depends only on the square-root of its coordination number (CN), i.e., the number of neighbors surrounding this atom within the lattice. It has been shown that square-root coordination number-based models of cohesive energy provide good agreement with predictions of cohesive energy for transition metals when compared to more costly DFT predictions.[83]

Previous work has successfully utilized CN as a catalytic site descriptor to design transition metal surfaces via mixed-integer linear programming techniques.[51] The work presented in this chapter extends such techniques towards the design of three-dimensional nanoclusters. Notably, the rigorous optimality guarantees afforded to us by the MILP-based approach often allow us to identify unintuitive and previously unconsidered designs that complement the breadth of existing results in the identification of low-energy small nanoclusters. It should be noted, however, that the new approach only seeks structures on a predefined, discrete lattice. This means that only structures that conform to the chosen lattice can be identified as optimal, highlighting the need for the user to provide a lattice input that can accommodate reasonable expectations about the geometry of highly cohesive structures.

The cohesive energy, E_{coh} , of a material represents the energetic benefit imparted when neutral metal atoms come together from infinite separation to form a crystalline solid. It has been shown that the moment expansion method for determining the electron density-of-states can accurately describe cohesion in transition metals.[25, 110] Based on this result, a transition metal atom contributes to the cohesiveness of a nanocluster proportionally to the square root of its coordination number.[25, 110, 115] Therefore, the average (per atom) cohesive energy of a transition metal nanocluster can be represented as a function of the coordination numbers of all its N atoms according to Equation 2.1.

$$E_{coh} = \frac{E_{coh}^{BULK}}{N} \sum_{i=1}^N \sqrt{\frac{CN_i}{CN_{max}}} + E_R \quad (2.1)$$

In the above equation, CN_i refers to the coordination number attained by the i^{th} atom, CN_{max} is an integer parameter specifying the maximum attainable coordination number for a given crystal lattice, E_{coh}^{BULK} is the cohesive energy of the bulk material, and E_R is a residual energy term. The residual term E_R represents repulsive interactions between atoms in a nanocluster at non-equilibrium interatomic distances. Whereas this term is

especially prevalent at small sizes N , [34] there generally exists no closed-form representation for it. Hence, we shall initially neglect it by assuming $E_R=0$. This model with no residual energy term is also referred to as the *Square Root Bond-cutting* (SRB) model for cohesive energy. [140] Importantly, the SRB model is MILP-representable via standard modeling methods described in the following sections, opening up interesting possibilities for its inclusion as the basis of a tractable optimization model for nanocluster design.

From this point onwards, when we refer to cohesive energy, we will be referring to its *dimensionless* form, which is the above defined quantity (E_{coh}) normalized to (divided by) the value of E_{coh}^{BULK} , and which can thus attain values between 0 and 1, irrespective of the identity of the material involved.

2.3 MATHEMATICAL OPTIMIZATION-BASED DESIGN

We shall now propose a mathematical optimization modeling framework for determining the minimum cohesive energy structures of three-dimensional, mono-metallic nanoclusters. In this framework, sites on a crystal lattice are indexed via the set $i \in I$. We refer to this set as a *canvas*, as it constitutes the space wherein an allotment of N atoms can be placed to design the nanoclusters. For each lattice site i , we introduce a binary design variable, Y_i , to indicate the presence or not of an atom on this site. If $Y_i = 1$, an atom exists at canvas location i , while if $Y_i = 0$, the canvas site is devoid of an atom. Using this framework, it is possible to represent any nanocluster design as a collection of “0/1” values for all design variables in the canvas.

The size and shape of the canvas should be carefully selected by the modeler. For example, if one wishes to design a face-centered cubic (FCC) nanocluster with $N = 100$ atoms, a possible canvas to use would be a cuboctahedral geometry and 561 lattice sites (i.e., 5 shells of a perfect cuboctahedron). However, one should keep in mind that the difficulty of solving the nanocluster optimization model depends upon the size of the canvas (degrees of freedom) in relation to how much of the canvas should be occupied (size of nanocluster), and that there exists a trade-off between numerical tractability and flexibility to accommodate any conceivable nanocluster design of a particular size N . Finally, it should be noted that, although we focus this study on FCC nanoclusters, the concept of a canvas, and thus our proposed optimization model, can be easily extended for the design of nanoclusters with any crystalline geometry.

2.3.1 Optimization Model

Given degrees of freedom Y_i to indicate placement of atoms as well as auxiliary variables CN_i to encode the coordination number at every canvas location $i \in I$,ⁱⁱ the basic optimization model to identify maximally cohesive transition-metal nanoclusters is given below in Equations 2.2 through 2.10.

$$\max_{Y_i, CN_i} \quad \frac{1}{N\sqrt{CN_{\max}}} \sum_{i \in I} \sqrt{CN_i} \quad (2.2)$$

$$\text{s.t.} \quad \sum_{i \in I} Y_i = N \quad (2.3)$$

$$\{Y_i = 1\} \Rightarrow \left\{ CN_i \leq \sum_{j \in L_i} Y_j \right\} \forall i \in I \quad (2.4)$$

$$\{Y_i = 1\} \Rightarrow \{CN_i \geq CN_{\min}\} \quad \forall i \in I \quad (2.5)$$

$$\{Y_i = 0\} \Rightarrow \{CN_i \leq 0\} \quad \forall i \in I \quad (2.6)$$

$$0 \leq CN_i \leq CN_{\max} \quad \forall i \in I \quad (2.7)$$

$$Y_i \in \{0, 1\} \quad \forall i \in I \quad (2.8)$$

$$\text{All atoms are connected} \quad (2.9)$$

$$\text{Nanoclusters are non-hollow} \quad (2.10)$$

The model's objective function, Equation 2.2, consists of the (dimensionless) SRB cohesive energy function, which we seek to maximize. Equation 2.3 defines the nanocluster's size (number of atoms), where N is an integer parameter of the model to be provided as a constant. For occupied canvas locations, Equations 2.4 set the auxiliary variables CN_i to their applicable values,ⁱⁱⁱ where the sets L_i have been defined to represent the neighboring sites to each location i . At the same time, Equations 2.5 ensure that all atoms adhere to some minimum value, CN_{\min} , which is provided to avoid low-coordinated, unrealistic atom placement. Equations 2.6 enforce that, if no atom is placed at a location i , then the corresponding CN_i variable attains the value of 0, and hence, prohibit unoccupied locations from contributing to the objective function. Note that the implication constraints 2.4 through 2.6 can be transformed to standard linear equations using well-known MILP modeling techniques, such as the so-called *big-M* reformulation, which is what we used in our implementation.

Equations 2.7 declare the non-negativity of the coordination number variables, as well as enforce applicable upper bounds on their possible val-

ii In this context, the "coordination number of an unoccupied location is regarded to be equal to 0.

iii We remark that coordination numbers are defined here via \leq inequality constraints (as opposed to strict equalities) at the interest of yielding an MILP model with tighter LP relaxations. Due to the direct maximization of variables CN_i in the objective function, the coordination number evaluations will be exact at any optimal solution.

ues. Here, the upper bound of CN_{\max} is chosen as the maximum achievable coordination number in a given canvas, as determined by the applicable lattice. Finally, Equation 2.8 explicitly enforces the integrality constraint for the binary variables Y_i .^{iv} We remark that, for the FCC geometry used in this study, we use constant values $CN_{\min} = 3$ and $CN_{\max} = 12$, meaning that any given atom is allowed to have at least three and at most twelve nearest neighbors. For other crystalline geometries, other appropriate values should be used (e.g., for body-centered cubic, $CN_{\max}=8$).

There also exist two additional requirements on our nanocluster designs, namely those of *connectivity* and *non-hollowness*. The purpose of requiring connectedness is to avoid presumed solutions where the N atoms have been divided into two or more smaller nanoclusters. The requirement for non-hollowness is imposed to avoid nanocluster designs that feature void enclosed volume. Because it is not straightforward to represent such requirements as explicit constraints on our model’s decision variables, we are only presenting them conceptually in Equations 2.9 and 2.10, respectively. These two constraints are enforced dynamically during the solution procedure via a *lazy-constraint interface*, which is available in modern MILP solvers. Omitting many details at the interest of brevity, the main idea is to inspect every design as soon as it is returned by the numerical solver, and if found to be either disconnected or hollow, to add an *integer cut* constraint to the model so as to explicitly render this specific design infeasible, eliminating the possibility that this design persists as the final optimal solution identified by the framework.

2.3.2 Concave Objective Function

We remark that the objective function is a non-linear, concave function in variables CN_i . Whereas at first glance this equation appears incompatible with an MILP model, we can reformulate it into an MILP-representable form due to the special mathematical structure of the model, namely the integrality of variables CN_i and the fact that we seek to maximize such a concave function. More specifically, we introduce a new set of auxiliary variables, CNR_i , to represent the square root value of the coordination number at each canvas location $i \in I$, adding also the following bound definitions.

$$0 \leq CNR_i \leq \sqrt{CN_{\max}} \quad \forall i \in I \quad (2.11)$$

We can now choose to model the square root of the coordination number not as a smooth function, rather as a set of secant lines passing through

^{iv} Note how the integrality of variables CN_i need not be explicitly declared, as it is implied by the integrality of variables Y_i .

points on the curve $\sqrt{CN_i}$ at integer values of CN_i , as shown in Figure 2.1 for the case of an FCC lattice. Note how this approximation of the square root function is exact at all locations of interest, namely the integer values of CN . The secant-line definition of CNR_i is then imposed in the model via Equations 2.12, where α_ℓ and β_ℓ are appropriate constants to represent the slope and intercept, respectively, of each consecutive secant line ℓ .

$$CNR_i \leq \alpha_\ell CN_i + \beta_\ell \quad \forall i \in I, \forall \ell \in \{1, 2, \dots, CN_{\max}\} \quad (2.12)$$

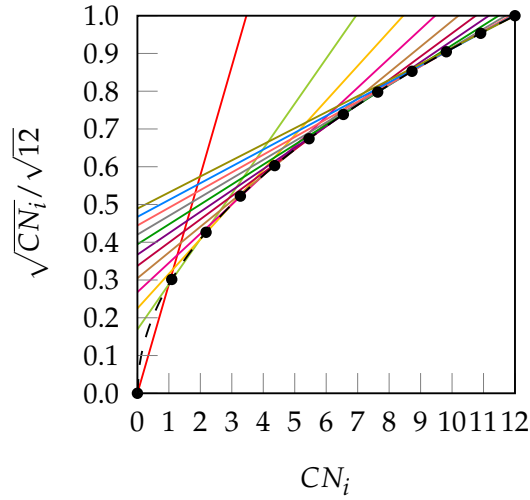


Figure 2.1: Square Root Bond-cutting model for cohesive energy (dimensionless) of an FCC atom i , plotted against CN_i . Also shown are the secant lines used to exactly represent the evaluations of cohesive energy at integral CN_i values.

Finally, the objective function is then replaced with Equation 2.13. Note that, because we are maximizing the cohesive energy, the optimizer has the incentive to choose the exact value of the applicable (intersection of) secant lines, as it is the maximally attainable value permitted by the inequalities 2.12. Hence, this substitution models the SRB cohesive energy not only in a linear form, but also exactly (i.e., without approximation error).

$$\frac{1}{N\sqrt{CN_{\max}}} \sum_{i \in I} CNR_i \quad (2.13)$$

2.3.2.1 Opportunities for Second-Order Conic Programming Reformulations

The cohesive energy objective function is effectively a maximization of a sum of square roots of integer variables, CN_i . Because the square-root

function is concave and non-decreasing, maximization of such an objective leads to a convex optimization problem. Furthermore, such a formulation can be represented as an second-order conic program (SOCP). By definition, an SOCP is an optimization model in which a linear objective function is minimized over the intersection of an affine set and the product of second order (i.e. quadratic) cones. SOCP problems are convex optimization problems that are known to be solveable in polynomial time via interior point methods. Although we have not verified the performance of applying the SOCP modeling approach, we provide an overview of how to apply SOCP to the nanocluster design problem.

As shown below in Equations 2.14–2.15, we first transform a simplified square-root coordination number objective via an epigraph reformulation using the scalar auxiliary variables $t_i \in \mathbb{R}_+ \ \forall i \in I$.

$$\max_{t_i, CN_i} \quad \sum_{i \in I} t_i \quad (2.14)$$

$$\text{s.t.} \quad \sqrt{CN_i} \geq t_i \quad \forall i \in I \quad (2.15)$$

We then focus on transforming the constraint in 2.15 into a second-order conic constraint. This is done by the transformation shown in 2.16.

$$\sqrt{CN_i} \geq t_i \Leftrightarrow CN_i \geq t_i^2 \Leftrightarrow \left\| \begin{array}{c} 1 - CN_i \\ 2t_i \end{array} \right\|_2 \leq 1 + CN_i \quad \forall i \in I \quad (2.16)$$

Although the integrality of CN_i variables is not explicitly enforced in our optimization formulation shown above, the presence of binary Y_i variables means the resulting formulation with second-order conic constraints leads to a mixed-integer second-order conic program (MISOCP). The area of mixed-integer conic programming is a relatively newer field of research with applications in areas such as portfolio optimization and network design and operation.[11] To solve these MISOCP problems, a branch-and-cut algorithm is often employed along with an interior point solver amenable to solving SOCPs. Additional work has been done to improve the solution algorithms for MISOCP. For example, convex hull inequalities have been derived to describing disjunctive conic sets[66, 67] as well as rounding cuts to improve branch-and-cut performance.[3] Solving the nanocluster geometry optimization via the MISOCP approach would provide an interesting tractability comparison to an MILP approach that has been tested through our work.

2.3.3 Symmetry Breaking

Due to the highly symmetric nature of crystallographic spaces, there exist many isomorphically equivalent ways to represent the same nanocluster in a canvas, by means of rotation, translation and reflection operations. More specifically, the FCC lattice is close-packed and has two-, three- and four-fold axes of symmetry. Symmetry of this form makes the MILP model more difficult to solve to optimality due to the large number of equivalent, feasible solutions. In order to mitigate this effect, Equations 2.17 and 2.18 were added to the model as symmetry-breaking constraints. These constraints aim to eliminate some isomorphic solutions from the design space, while guaranteeing that at least one representative solution remains feasible in the resulting model, and hence, that at least one isomorphic equivalent of the optimal nanocluster is accessible from the design space induced by the model.

$$\sum_{i \in I_s^+} Y_i - \sum_{i \in I_s^-} Y_i \geq 0 \quad \forall s \in \{1, 2, 3\} \quad (2.17)$$

$$\sum_{i \in I_s^+} Y_i - \sum_{i \in I_s^-} Y_i \leq \sum_{i \in I_s^0} Y_i \quad \forall s \in \{1, 2, 3\} \quad (2.18)$$

The sets I_s^+ , I_s^- and I_s^0 in the symmetry-breaking constraints represent suitable partitions of the canvas, as dictated by three intersecting crystallographic planes, s . Any lattice site i in the canvas can be viewed as either being “above” plane s , $i \in I_s^+$, “below” plane s , $i \in I_s^-$, or “on” plane s , $i \in I_s^0$. By restricting the distribution of atoms in the canvas to be approximately balanced, many isomorphically equivalent solutions are removed from the set of feasible solutions.

2.3.4 Improving Numerical Tractability

The mathematical optimization model presented in the previous section can be addressed by any off-the-shelf MILP solver. However, the latter being a form of numerical software, it is subject to numerical tractability issues when applied on models that feature large feasible spaces, such as those that arise when we use large values of N . In order to improve solution performance at all N values we wish to consider in this study, we choose to apply our model sequentially, increasing the value of N one at a time. As we do so, we adapt the canvas for optimizing the N -atom nanocluster based on the shape of the optimal $(N - 1)$ -atom design. In addition, recognizing that in large clusters there is a significant amount of bulk atom sites, we fix certain binary variables in central locations of the canvas, again being informed by optimal solutions preceding in the

sequence. Below we elaborate further on these algorithmic enhancements to our framework.

2.3.4.1 *Adaptively select the canvas size and shape*

In order to ensure that the MILP solver has enough degrees of freedom to enumerate and identify the optimal nanocluster at a given size, a sufficiently large canvas must be used. Ideally, this size of the canvas should be as large as possible, so that the MILP solver has access to a design space that is guaranteed to include the optimal nanocluster geometry. However, the tractability of the problem scales inversely with the size of the canvas. In order to alleviate this issue, the shape and size of the canvas is determined by the optimal solution of the $(N - 1)$ -atom nanocluster. Starting with $N = 4$, where the optimal solution is easily determined (e.g., using the unenhanced framework) to be a tetrahedron,^v all following canvases can be constructed by taking the $(N - 1)$ -atom optimal nanocluster and expanding it by two complete shells around that particle. We have empirically determined that this procedure leads to sufficiently large design spaces, as maximizing cohesive energy will tend toward centralized, roughly spherical shapes. It also ensures that the canvas is not excessively large at smaller values of N .^{vi}

2.3.4.2 *Fixing select atom positions*

Another enhancement we have applied in order to improve our framework's numerical tractability is the fixing (to the value of 1) of certain Y_i variables based on optimal solutions at smaller N values. This decreases the complexity of the optimization problem by decreasing its degrees of freedom (number of the unfixed binary decision variables). Physically, this forces some lattice positions to be occupied by an atom in all solutions considered in the design space.

The algorithm for selecting which atoms to fix is as follows. If $N \geq m$, consider the set of the previous m optimal nanoclusters, $O = \{N - 1, N - 2, \dots, N - m\}$. Enumerate all rotations of each of the nanoclusters in O that satisfy the symmetry-breaking constraints of Equations 2.17 and 2.18, and denote the set of these transformed nanoclusters as O^* . The atoms that will be fixed at lattice locations i are those that appear in all nanoclusters in the set O^* . In other words, if a particular atom is present in the m previous optimal solutions, we expect it to arise again in the current solution. In

^v The design problem is technically infeasible for values of $N \leq 3$, due to the requirement for minimum coordination equal to 3 for all atoms.

^{vi} In practice, canvases designed this way will not be regular cuboctahedra, though this poses no concern in terms of defining the optimization model, which can be cast for any irregularly shaped canvas I .

this work, $m = 6$ was used at all N sizes, because that setting was found to provide sufficiently conservative sets of lattice locations to fix, while also significantly improving the tractability at all N .

2.4 OPTIMAL DESIGNS






The nanocluster optimization model was solved with and without the numerical enhancements of canvas sizing and atom fixing. In the model instances solved without enhancements, the canvas was taken to be the 561 lattice site cuboctahedron. Resulting cohesive energies at consecutive sizes N are shown in Figures 2.2a and 2.2b. Each model was solved using the MILP solver CPLEX 12.8,[26] using a one hour time limit and four threads in parallel mode. Additionally, each run was provided with an initial solution via the *MIP start* feature of this solver. The initial solution used at a given N was generated by taking the $(N - 1)$ -atom nanocluster solution and attaching on its surface a single atom in the most favorable (out of all feasible options) position.

First, it should be noted that, as N increases, the best integer solutions asymptotically approach the value of 1; that is, the optimal cohesive energies approach the bulk cohesive energy of the material, which is consistent with the expected behavior of the SRB function. However, there are some instances where the cohesive energy does not trend monotonically as N increases. This has been observed in the literature and can be explained via the concept of magic number effects.[36]

It is also clear that, without enhancements, the solver fails to prove the optimality of its best identified designs (red dots) for cases as low as $N = 10$. This is likely due to poor LP relaxations that are observed while integrality is relaxed, as the mass of all N atoms is diffused across all sites in the canvas. It is also clear that, without enhancements, the solver fails to prove the optimality of its best identified designs (red dots) for cases as low as $N = 10$.

On the other hand, once the proposed enhancements are enabled, the performance of the MILP solver drastically improves, and the solver is able to close the optimality gap in all cases except the regimes $N = 54 - 64$ and $N = 70 - 80$, where some very small gaps remain. This can be explained by inspecting how the set of central fixed atoms evolves over N . In the ranges where the upper and lower bound are not equal at termination, there are approximately ten fewer fixed binary variables, when compared to the following and preceding sequences. This increase in free binary variables decreases the tractability of these instances, as compared to those with fewer degrees of freedom, leading to non-zero, yet small, gaps after the imposed time limit.

Table 2.1: Representative optimally-cohesive nanocluster geometries, as predicted by the MILP-model maximizing the SRB cohesive energy.

				
$N=10$	$N=20$	$N=30$	$N=40$	$N=50$






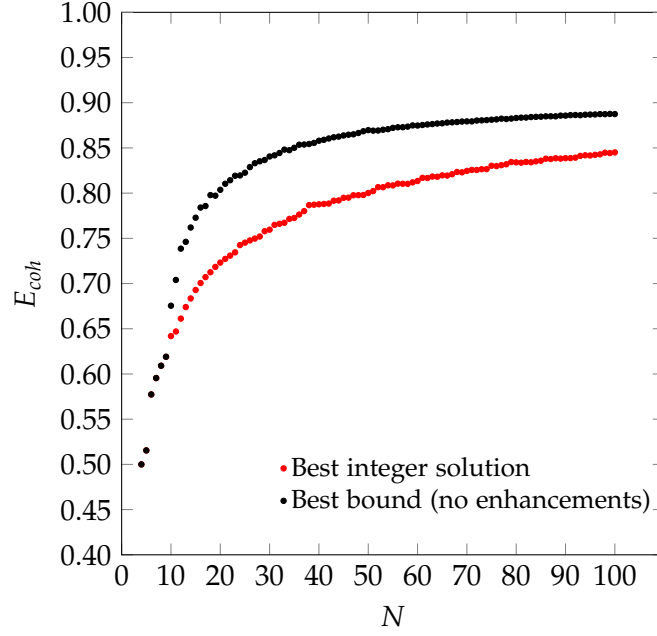
				
$N=60$	$N=70$	$N=80$	$N=90$	$N=100$

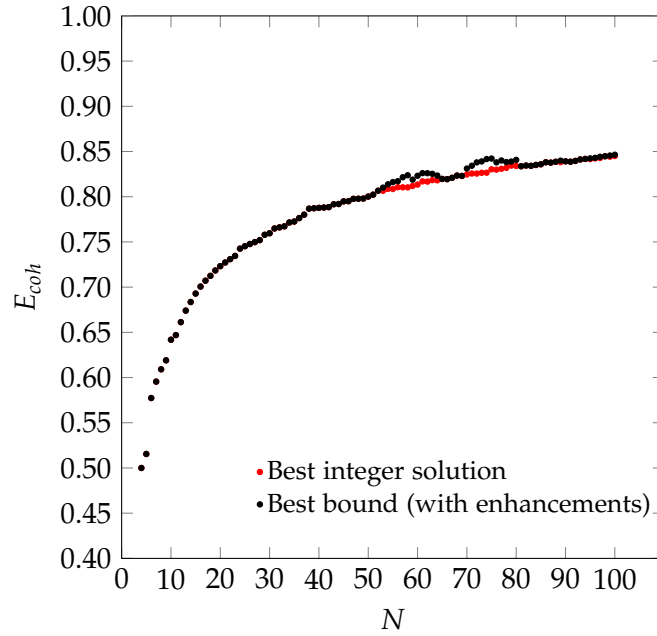
Table 2.1 shows some representative optimal solutions. We observe that these generally possess a somewhat octahedral shape, in accordance with empirical expectation. Furthermore, it is worthy to note that there was never a case when, by solver termination, the unenhanced framework had identified an integral solution that was better than the one having been identified by the enhanced framework. To this end, we believe that the algorithmic enhancements do not cause optimal solutions to be eliminated from the design spaces, and hence, they are welcome to adopt moving forward inasmuch as they improve tractability without any deterioration in solution optimality. Schematics of all optimal structures are plotted in Appendix 2.7.

2.5 CONCLUSIONS

In this chapter, we have identified a suitable representation of the cohesive energy of a transition metal nanoparticle for use in a mixed-integer linear optimization problem. We then propose an MILP model for identifying optimally cohesive nanocluster structures where the size of the nanocluster, N , is a parameter. We solve this optimization problem with and without numerical efficiencies to identify proven optimal structures. As is expected, because of the implicit definition of the FCC canvas, optimal solutions at various sizes are fragmented octahedra. The accuracy of the results acquired via the MILP modeling framework presented here is highly dependent on the form of energy function utilized in the objective. Therefore, there is a need to identify coordination-based cohesive energy functions which more accurately capture metal-specific behaviors.



(a)



(b)

Figure 2.2: Best solutions and best upper bounds at termination for $N = 4 - 100$, using the MILP model without (2.2a) and with (2.2b) algorithmic enhancements.

2.6 NOTATION

Indices

i	canvas (lattice) location
j	neighboring canvas (lattice) location
s	symmetry plane
ℓ	secant line index

Sets

I	canvas (lattice) locations
L_i	neighboring canvas (lattice) locations to location i
I_s^+	locations strictly above the symmetry plane s
I_s^-	locations strictly below the symmetry plane s
I_s^0	locations laying directly on symmetry plane s

Binary Variables

Y_i	presence of an atom at canvas (lattice) location i
-------	--

Continuous Variables

CN_i	coordination number of atom at location i
CNR_i	square root of the coordination number of atom at location i

Parameters

CN_{\max}	maximum possible coordination number (corresponding to crystal lattice bulk coordination number)
CN_{\min}	minimum allowed coordination number for an atom in the nanocluster
N	number of atoms
α_ℓ	slope of the secant line defining CNR_i
β_ℓ	intercept of the secant line defining CNR_i

2.7 APPENDIX

Optimal Nanoclusters

We present here the detailed list of nanocluster designs identified via our optimization framework. Figure [2.3](#) depicts the optimal clusters identified by maximizing the SRB function for cohesive energy.

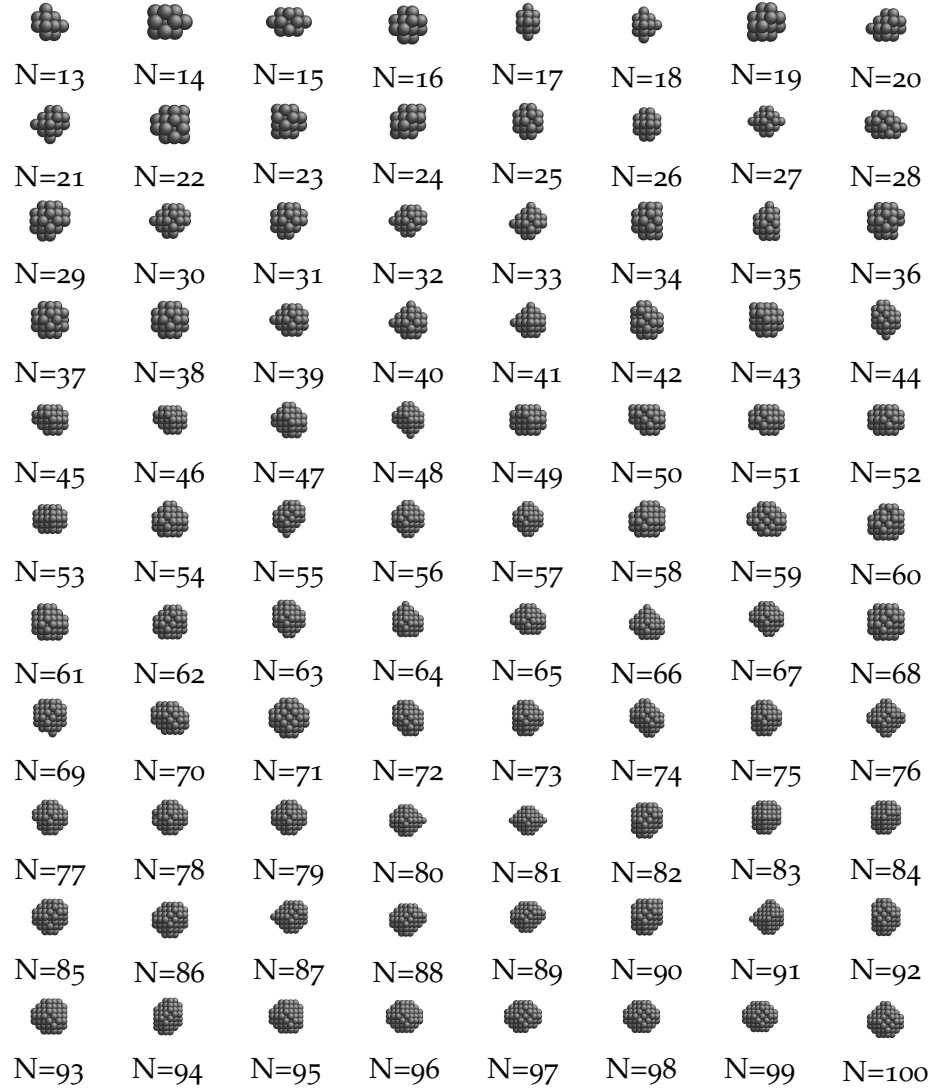


Figure 2.3: Most cohesive structure according to the SRB cohesive energy function for every N .

METAL-SPECIFIC NANOCLUSTER COHESIVE ENERGY

3.1 INTRODUCTION

The model of cohesive energy utilized in Chapter 2 does not reference atomic species, meaning the cohesive energy determined by the square root bond-cutting (SRB) model is agnostic to chemistry. For this reason, the SRB model for cohesive energy is known to be an approximation for the true cohesive energy of small metallic clusters. Firstly, the SRB cohesive energy model neglects the residual (repulsive) energy term for non-ideal interatomic interactions proposed by Tománek, Mukherjee, and Bennemann [115]. Secondly, the SRB dimensionless cohesive energy model has no metal-specific considerations, essentially assuming that all metals will form bonds in the same way, which is a poor assumption for small nanoclusters where quantum effects are prominent. Thus, there exists a need to develop more accurate, metal-specific, yet still MILP-representable, formulas for cohesive energy that correct the SRB model.

3.2 METAL-SPECIFIC NANOCLUSTER GEOMETRY OPTIMIZATION

The workflow for identifying metal-specific square root bond-cutting models for cohesive energy optimization is outlined in the following section. First, the enhanced optimization framework, in its original form using the SRB model as its objective, was utilized to identify a set of highly cohesive nanocluster geometries at a range of sizes. For this, we used the *solution pool* of the CPLEX solver, which allows us to identify the k -best integral solutions to an MILP model at a computational cost that is only marginally higher than that of a standard run to identify just the (one) optimal solution. More specifically, for sizes $N = 13 - 25$, the best $k = 10$ nanocluster geometries were collected, with an additional data point at $N = 20$ to include a perfect tetrahedron. For $N = 26 - 40$, we collected the best $k = 3$ nanocluster geometries, while for $N = 41 - 100$, only the optimal nanocluster was collected. This led to a total of 236 highly cohesive nanoclusters, with most of the data in the $N = 13 - 40$ size regime. With

all values of N , a four hour time limit was imposed to collect such solution pools.

The identified optimal and near-optimal structures were then evaluated with density function theory (DFT) for their “true” cohesive energies in the context of five transition metals of interest, namely silver (Ag), gold (Au), copper (Cu), palladium (Pd) and platinum (Pt). These metals were chosen due to their array of applications in catalysis and alternative energy applications.[21, 68, 142] All DFT calculations were performed in the *CP2K* computational package [58] with the PBE functional [94], DZVP basis set [118], and GTH pseudopotentials.[41] These methods have been successfully used in the past to evaluate the energetics of metal clusters.[140] Note that we used the MILP-predicted optimal clusters as input structures for the DFT calculations, with their interatomic distances being set to those in the bulk. The bulk cohesive energy, as calculated by PBE, was used as a consistent energy reference for our comparisons with DFT. Single-point energy evaluations were performed on the interatomic scaled monometallic clusters.[60]

Parity plots between the energies predicted by the SRB cohesive energy function and by DFT are shown in Figure 3.1 (black dots). From the parity plots of Figure 3.1, it is clear that the SRB cohesive energy is over-estimating the cohesive energy in all cases. This is likely due to (metal) group-dependent effects such as stresses and surface relaxations,[70, 82] which are not considered by the metal-agnostic SRB function and which may be the source of the prediction errors observed. We remark that, in our DFT calculations, metal-specific stresses are present because we do not relax the nanoclusters and rather constrain the atoms to sit on perfect lattices with set bulk interatomic spacings. The latter can deviate from DFT-calculated spacings and can also change in a metal-dependent fashion at the nanoscale.

The degree of error varies across metals. In fact, the SRB function estimates the cohesive energy of the Group 11 metals (Au, Ag, Cu) significantly better than the Group 10 metals (Pd, Pt). In general, the approximation is poorer at lower cluster sizes, and better at larger sizes. Indeed, the SRB approximation is known to increase in accuracy for all metals as nanoclusters approach bulk material size. Finally, it should be noted that the outlier in each plot of parity is the $N = 20$ tetrahedron. This structure is known to exhibit special quantum effects that enhance its stability in DFT calculations for Group 10 metals like Au.[76, 97] Hence, it is not surprising that the SRB model, which ignores such effects, does a poor job at predicting its cohesive energy.

Using the parity plots from the previous analysis as a guide, metal-specific corrective terms in the coordination-dependent function for cohesive energy can now be identified using a constrained regression process.

The focus has been on deriving corrections that are MILP-representable, so that they can be embedded in our MILP-based nanocluster design framework. Furthermore, it is desirable for these corrections to reference quantities already encoded in our optimization formulation (e.g., the coordination numbers CN_i), so as not to further increase its complexity. To achieve this, a linear regression was performed. The general form for the corrected dimensionless cohesive energy models (E_{coh}^m , where the superscript m stands in for a specific metal) is shown in Equation 3.1. The basis functions—or features—we used to determine the regression coefficients γ_k^m are the fractions, fCN_k , of atoms in the nanocluster that attain a specific coordination number k (between the admissible values CN_{\min} and CN_{\max}), as per the definition of Equation 3.2. Conceptually, this selected functional form of E_{coh}^m captures metal-specific cohesive energies via an “offset” from the SRB model prediction.

$$E_{coh}^m = E_{coh}^{SRB} + \sum_{k=CN_{\min}}^{CN_{\max}} \gamma_k^m fCN_k \quad (3.1)$$

$$fCN_k := \frac{1}{N} \left(\sum_{i=1}^N \mathbb{1}_{\{CN_i=k\}} \right) \quad \forall k \in \{CN_{\min}, \dots, CN_{\max}\} \quad (3.2)$$

The constrained regression optimization model used to identify coefficients for the metal-specific cohesive energy functions is shown in Equations 3.3–3.6. This regression model, which is parameterized by the metal type m , was applied separately for each specific investigated in this study. The objective function (Eqn. 3.3) represents the sum of squared errors for each of the $n = 236$ nanocluster structures used as input data (see Section 3.2 for further details as to the origin of these structures). The error for a particular nanocluster structure j is calculated as the difference between its DFT-predicted cohesive energy and its evaluation of energy E_{coh}^m from Equation 3.1. In order to ensure the same linearization methods can be used on the new objective function in the context of the MILP optimization model (see discussion in Section 2.3.2), constraints for concavity of the new E_{coh}^m functions are added to the regression via Equations 3.4. Additionally, a set of hierarchical constraints (Eqns. 3.5) is added to enforce that the cohesive energy functions increase monotonically with CN , while the assertion in Equation 3.6 enforces that no correction is required for the contribution of bulk atoms, which always equals one (in dimensionless

terms). Together, these constraints ensure that no individual atom's energy contribution surpasses that of the bulk.

$$\min_{\gamma_k^m} \sum_{j=1}^n \left(E_{coh,j}^{DFT} - \left(E_{coh,j}^{SRB} + \sum_{k=3}^{12} \gamma_k^m fCN_{k,j} \right) \right)^2 \quad (3.3)$$

$$\text{s.t.} \quad \sqrt{\frac{k}{12}} + \gamma_k^m \geq \left(\frac{\sqrt{\frac{k+1}{12}} + \gamma_{k+1}^m + \sqrt{\frac{k-1}{12}} + \gamma_{k-1}^m}{2} \right) \quad \forall k = \{4, \dots, 11\} \quad (3.4)$$

$$\sqrt{\frac{k}{12}} + \gamma_k \leq \sqrt{\frac{k+1}{12}} + \gamma_{k+1} \quad \forall k = \{3, \dots, 11\} \quad (3.5)$$

$$\gamma_{12}^m = 0 \quad (3.6)$$

After solving the regression model above, we obtain metal-specific variations of the cohesive energy from the SRB model, as shown in Figure 3.2. The most dramatic shifts in the per-atom energy contributions are exhibited for the case of Pd, which reflects the fact that the SRB function over-estimates the DFT-predicted energies the most. Interestingly, for the cases of Pd, Cu and Au, there are consistent decreases in energetic contributions from all CN values, while the Pt and Ag models promote contributions (i.e., impart cohesive energy greater than the SRB model) from highly-coordinated atoms. We note that these results may, in part, be due to the fact that transition metals are well-known to possess metal-dependent nanoscale strain,[7, 82] which could be captured here through the metal-specific deviations from the SRB function.

3.3 METAL-SPECIFIC OPTIMAL DESIGNS

Using the collected DFT data and methods outlined previously, metal-specific functions for nanocluster cohesive energy were regressed to predict nanocluster cohesive energies with greater accuracy. The same set of highly cohesive nanoclusters is evaluated with the metal-specific cohesive energy models and plotted against the DFT predictions in Figure 3.1. It is clear that the new cohesive energy function predicts the cohesive energies of these nanoclusters much better than the original SRB function.

In fact, because of the shifts in per-CN cohesive energy contributions in the surrogate models for each metal m , E_{coh}^m , it is possible that the optimal, most cohesive structures at a certain size N might change from what was determined previously using merely the SRB function. In order to design transition metal nanoclusters that are guaranteed to possess maximal cohesive energies against the more accurate, DFT-based evaluation, the

surrogate cohesive energy functions were embedded in the formulation of our mathematical optimization model. More specifically, the original objective function (Eqn. 2.2) of the optimization model was replaced with the surrogate models by simply shifting the set-points of the secant lines (Fig. 2.1). Apart from this change in the objective function, the rest of the optimization model remained unchanged. Examples of new optima determined this way are shown in Table 3.1. New optima identified for all sizes N and metals m we considered in this study are depicted in the Appendix 3.6.











	N=19	N=22	N=36	N=42	N=83
SRB					
Metal-specific					
	Pd ₁₉	Ag ₂₂	Au ₃₆	Cu ₄₂	Pt ₈₃

Table 3.1: Comparison of a few optimal nanocluster structures as determined by the SRB function (first row) and the new optimal structures determined by the corrected models for different metals (second row).

It should be noted that the nature of the canvas used in this study (FCC lattice) means that only FCC structures are identified here as optimally cohesive. This can be a valid assumption for certain regimes, such as in the case of smaller Pd clusters that have been shown to trend toward FCC structures over non-crystalline alternatives.[37] Furthermore, the modified octahedral shape of many of the optimal nanoclusters predicted by our framework at intermediate sizes are in agreement with some previously reported results.[47, 137, 144] However, we should acknowledge that non-FCC structures featuring decahedral[104] or icosahedral[7, 56] geometries can also arise in transition metal nanoclusters. This is due to the fact that such 5-fold symmetric structures maximize the coordination of surface atoms, which is especially favored at small N where the strain induced by this surface packing is not prohibitive. Whereas the current study did not search over the space of non-FCC geometries, and thus could not obtain results reflecting the above cases, our framework could be modified to do so via the use of appropriately defined canvases that can accommodate all reasonably expected possibilities regarding non-FCC placements, including 5-fold symmetric lattices. In addition, we

could introduce additional terms to our objective function of cohesive energy so as to reproduce special cases of enhanced stability that arise due magic number phenomena via electronic shell closures[36] (e.g., the $N = 20$ Au tetrahedron[70, 76]) or relativistic effects[96] (e.g., planar Au nanoclusters[4, 56, 136, 138]). In any case, it should be highlighted that, once optimal FCC structures are obtained under the current model setup, it is advisable to subject them to local energetic relaxation using any appropriate, sophisticated functional of choice, in order to determine the precise off-lattice placement, whenever applicable.

3.4 CONCLUSIONS

In this chapter, we have identified metal-specific corrective factors for the square root bond-cutting model for several transition metals. We accomplished this by evaluating a suite of SRB-proven optimal nanoclusters with density-functional theory to determine the DFT-predicted cohesive energies. We utilized the DFT predictions for each transition metal studied in a sum-squared errors constrained regression to identify a convex, metal-specific cohesive energy function. These metal-specific cohesive energies were utilized in a MILP model to identify several new metal-specific optimal geometries at various sizes. This chapter has shown that there is a benefit to pairing rigorous optimization approaches with complex, computation chemistry methods when studying nanoclusters.

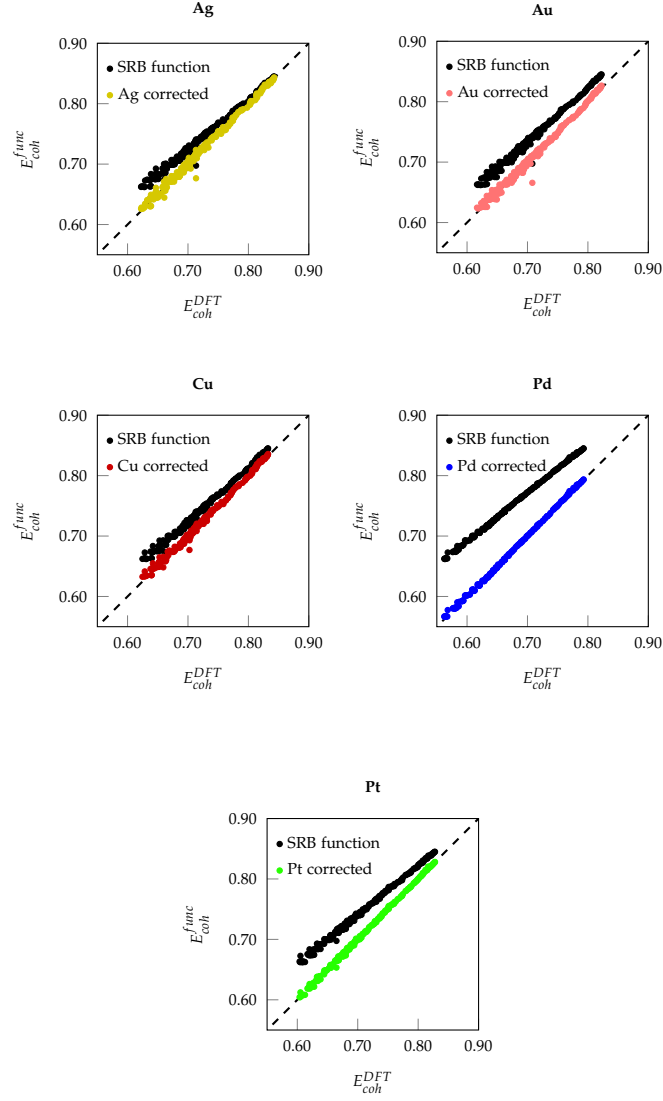


Figure 3.1: Parity plots between cohesive energies calculated by the SRB model (y-axis), with and without metal-specific corrections, and by DFT (x-axis) for various metals.

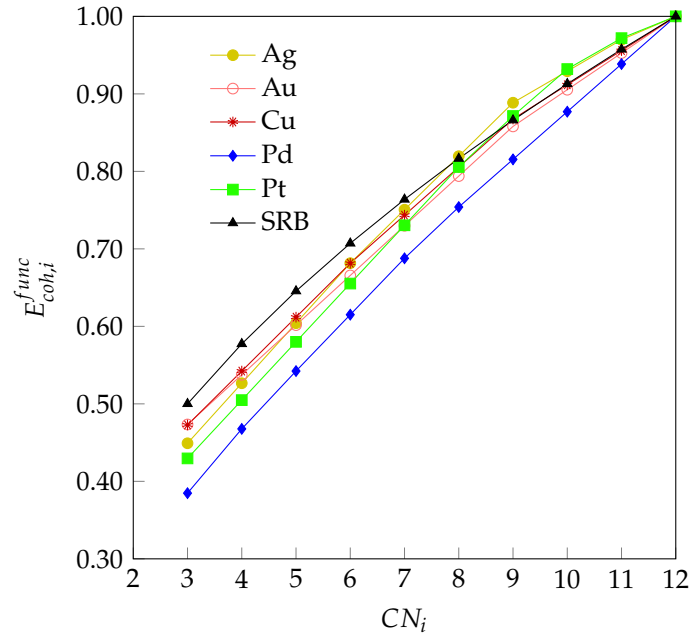


Figure 3.2: Metal-specific corrections to the original SRB model for cohesive energy, identified via constrained regression based on DFT predicted values.

3.5 NOTATION

Indices

n	number of data points used for SSE regression
m	metal type
k	coordination number breakpoints for the secant lines

Continuous Variables

E_{coh}^m	metal-specific cohesive energy
fCN_k	fraction of total atoms in nanocluster with coordination number k
γ_k^m	regression coefficient for secant line at coordination number k for metal m
$E_{coh,j}^{SRB}$	cohesive energy predicted by SRB optimization for nanocluster j
$E_{coh,j}^{DFT}$	cohesive energy predicted by DFT for nanocluster j

Parameters

CN_{\max}	maximum possible coordination number (corresponding to crystal lattice bulk coordination number)
CN_{\min}	minimum allowed coordination number for an atom in the nanocluster
N	number of atoms

3.6 APPENDIX

Optimal Nanoclusters

Figures 3.3 through 3.7 show optima identified by maximizing the metal-specific functions (E_{coh}^m) of cohesive energy. Note that, at the interest of space, we only present the structures that differ from the ones predicted by the SRB function in Figure 2.3.

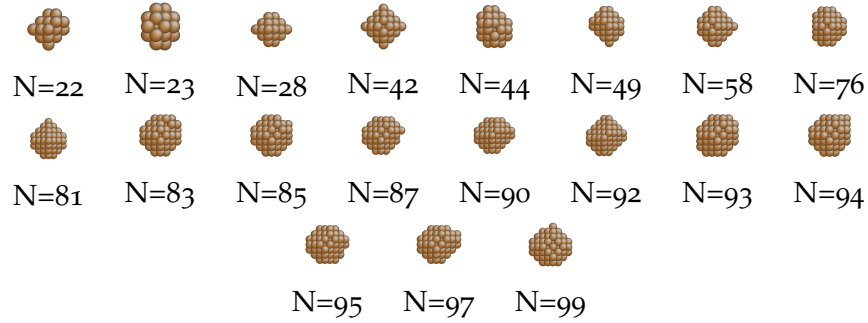


Figure 3.3: New optima at various N , as determined by the Cu-corrected function for cohesive energy.

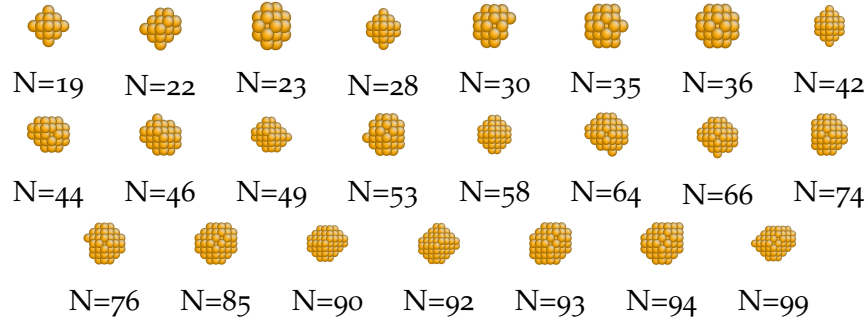


Figure 3.4: New optima at various N , as determined by the Au-corrected function for cohesive energy.

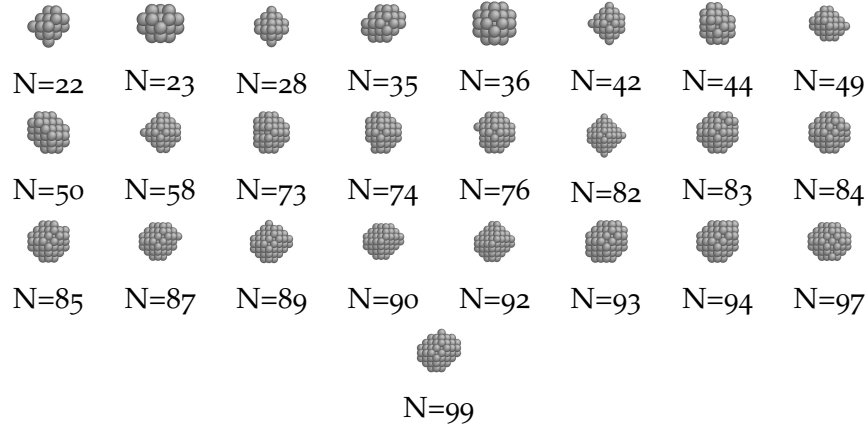


Figure 3.5: New optima at various N , as determined by the Ag-corrected function for cohesive energy.

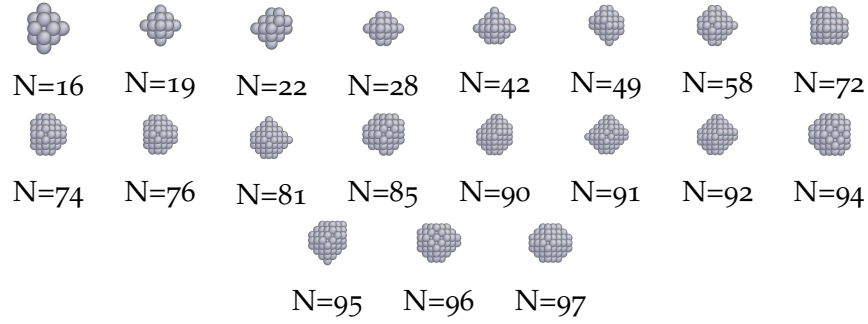


Figure 3.6: New optima at various N , as determined by the Pd-corrected function for cohesive energy.

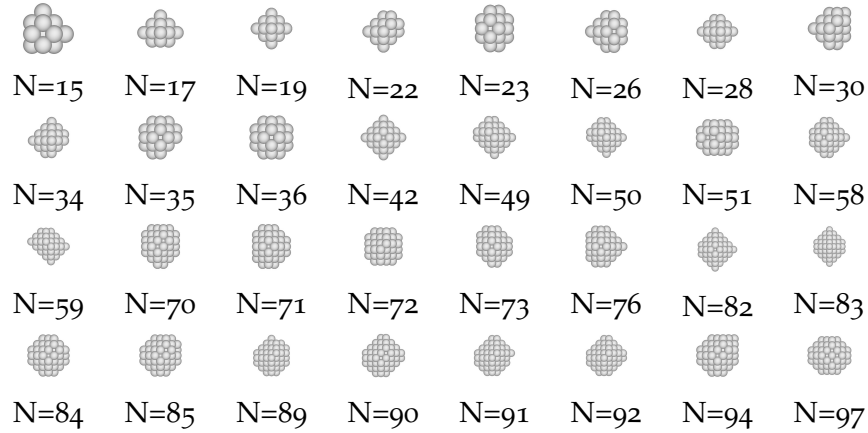


Figure 3.7: New optima at various N , as determined by the Pt-corrected function for cohesive energy.

MODELING NANOCUSTER GEOMETRY RELAXATIONS

4.1 INTRODUCTION

At very small sizes of nanoclusters, the effect of surface strain is more prominent and has an impact on nanocluster geometry.[7] This leads to non-FCC or non-crystalline structures for certain metal nanoclusters at small N . Such structures need not obey crystallographic translational symmetry, and may take on 5-fold icosahedral or decahedral symmetries. The transition from non-crystalline geometry to bulk FCC geometry has been studied in the literature both experimentally[69] and theoretically[130] and has been shown to occur at different sizes for different metals.

In general, our nanocluster design framework is not restricted to a particular crystallographic lattice type, but does assume *a priori* knowledge of a uniform crystal geometry for specifying the canvas. In the work presented thus far, our MILP optimization framework has focused on purely FCC structures by utilizing an FCC canvas, which can be a valid assumption for certain transition metals at certain sizes. For example, a study by Futschek, Marsman, and Hafner [37] found that $N=11, 12$, and 13 Pd clusters take on FCC structures over the non-crystalline or relaxed alternatives. And in general, at small N , non-crystalline, icosahedral structures maximize the coordination numbers of surface atoms, but the strain induced by this surface packing increases with the size of the nanocluster and makes them unfavorable at larger N . In principle, the selection of the canvas used in the MILP optimization can aide in this task. Simply selecting an icosahedral or decahedral canvas would lead to the identification of optimal 5-fold symmetric structures at given sizes. Another more pressing question is how to account for off-lattice or relaxed structures. These have been observed in predictions by Darby et al. [27] in small mono-metallic gold nanoclusters, and highly stable amorphous Au_{55} structures were found by Garzon and Posada-Amarillas [39] and Häkkinen et al. [49]. Modeling the space to describe amorphous structures in three dimensions would prove to be intractable in the MILP modeling framework considered here, as

the canvas would need to effectively represent all of \mathbb{R}^3 . Therefore, in this chapter, we propose a modeling approach and MILP model for optimizing nanocluster cohesive energy for off-lattice, relaxed FCC structures.

4.2 COHESIVE ENERGY AND RELAXED STRUCTURES

To encode relaxed crystal structures, modifications to our MILP models can be made to optimize over “*sub-lattices*” of a crystallographic lattice, which can be viewed as a finer discretization of space. The sub-lattice is a set of additional points that lie off of crystal lattice sites which are feasible atom locations. This modeling approach will require the usage of an effective coordination number ECN_i to capture attractive *and* repulsive cohesive energy contributions based on inter-atomic distances, as these distances can be non-ideal in the proposed sub-lattice model. The effective coordination number has been used by Shandiz [106] when calculating the cohesive energy of a nanoparticle in order to account for different bond lengths. Their proposed functional relationship for effective coordination number ECN_i for a given atom i is shown in Equation 4.3. In the proposed model, the average cohesive energy is a function of the effective coordination number, which depends on the energy change due to the relaxed bond lengths. Effective coordination number is determined by multiplying a parameter c_i raised to the $-m$ power by the ideal site coordination number (i.e. nearest neighbor count, CN_i). Here, c_i represents the reduced bond length parameter for the relaxed bond and m is also an adjustable parameter that varies according to the nature of the bond.

$$ECN_i = c_i^{-m} CN_i \quad (4.1)$$

We note that the original derivation for the square-root relationship for cohesive energy by Tománek, Mukherjee, and Bennemann [115] utilizes that effective coordination number, which was taken as the true coordination number in our original model due to the assumption of equilibrium distances.

4.3 MILP MODELING FOR RELAXED NANOCLOUDS

In the model considered in Equations 4.2–4.14, we will assume we have a set of what we will call *on-lattice* canvas locations I which represent a crystal lattice, such as a close-packed FCC lattice. In addition, we define an *off-lattice*, or sub-lattice, set K_i , per on-lattice location $i \in I$ which represents the possible non-ideal lattice positions for location i . We note that $i \in K_i$, meaning the ideal on-lattice point is a part of the sub-lattice. Therefore, the binary decision of placing an atom at location i will now be tied to

a decision of which sub-lattice site $k \in K_i$ to occupy. The placement of an atom at an ideal lattice location is indicated by the binary variable P_i , while the actual sub-lattice location is determined by the binary variable Q_{ik} .

The model we will consider is concerned with defining bonds between first-neighbors. These bonds are defined between the sub-lattice locations $k \in K_i, l \in K_j$ for the ideal lattice locations $i, j \in I, i \neq j$ and are defined by the binary variable X_{ijkl} . Finally, we must consider how the variables described here relate to the cohesive energy.

In this model, we will consider an effective coordination number of each occupied lattice position, ECN_i , which accounts for the impact that non-ideal inter-atomic distances has on cohesive energy. Because our sub-lattice is still a finite set of points, the ECN_i variable also has discrete values which depend on the number and location of sub-lattice positions K_i . Similar to the model shown in Chapter 2, we can use a secant line representation for the square-root of ECN_i via the auxiliary variable E_i .

$$\max_{P_i, Q_{ik}, X_{ijkl}, ECN_i, E_i} \frac{1}{N\sqrt{ECN_{max}}} \sum_{i \in I} E_i \quad (4.2)$$

$$\text{s.t. } \sum_{k \in K_i} Q_{ik} = P_i \quad \forall i \in I \quad (4.3)$$

$$\{X_{ijkl}\} \Rightarrow \{Q_{ik}\} \quad \forall i \in I, \forall j \in I, i \neq j, \forall k \in K_i, \forall l \in K_j \quad (4.4)$$

$$\{X_{ijkl}\} \Rightarrow \{Q_{jl}\} \quad \forall i \in I, \forall j \in I, i \neq j, \forall k \in K_i, \forall l \in K_j \quad (4.5)$$

$$\{P_i\} \Rightarrow \left\{ ECN_i = \sum_{j \in I} \sum_{k \in K_i} \sum_{l \in K_j} \alpha_{ijkl} X_{ijkl} \right\} \quad \forall i \in I \quad (4.6)$$

$$\{\neg P_i\} \Rightarrow \{ECN_i = 0\} \quad \forall i \in I \quad (4.7)$$

$$E_i \leq \beta_s ECN_i + \gamma_s \quad \forall i \in I, \forall s \in S \quad (4.8)$$

$$\sum_{i \in I} P_i = N \quad (4.9)$$

$$P_i \in \{0, 1\} \quad \forall i \in I \quad (4.10)$$

$$Q_{ik} \in \{0, 1\} \quad \forall i \in I, \forall k \in K_i \quad (4.11)$$

$$X_{ijkl} \in \{0, 1\} \quad \forall i \in I, \forall j \in I, i \neq j, \forall k \in K_i, \forall l \in K_j \quad (4.12)$$

$$ECN_i \in [0, ECN_{max}] \quad \forall i \in I \quad (4.13)$$

$$E_i \in \mathbb{R} \quad \forall i \in I \quad (4.14)$$

The objective function 4.2 aims to maximize the average dimensionless cohesive energy of the nanocluster. The constraints 4.3 – 4.7 encode implication logic for linking binary decisions to the value of the effective coordination number at each canvas location. The constraint in 4.8 encodes the secant lines representing the cohesive energy at each ECN_i , while constraint 4.9 restricts the size of the nanocluster to N . The parameter ECN_{max} represents the upper limit to effective coordination number given the sub-lattice choice per lattice location i .

The remaining research questions for this MILP model are with respect to how to determine the scaling α_{ijkl} parameters, and how to select sub-lattices. In our bond-centric MILP model, we are grouping the bond length deviation parameter shown in Equation 4.1 into the α_{ijkl} constant. The purpose of the scaling parameter α_{ijkl} is to capture any cohesive energy contributions due to non-ideal bond lengths caused by either repulsive energy contributions or surface relaxations. In a relaxed nanocluster, the atoms may occupy off-lattice, or non-equilibrium, positions. Under these conditions, a repulsive energy contribution is required to apply the square-root bond cutting model utilized in previous chapters.[90, 115] The remaining research effort for this MILP model would be to identify a α_{ijkl} for different transition metals, possibly via hard-core potentials. How to select sub-lattice geometries is also an open question for this modeling approach. There is an opportunity to try different, regular arrangements of off-lattice locations, or more irregular patterns.

4.4 CONCLUSIONS

In this chapter, we address the ongoing need to develop an MILP modeling framework for tractable off-lattice nanocluster geometry optimization. We pose a preliminary MILP model for achieving this goal that relies on the definition of a *sub-lattice* canvas. In this canvas, we allow a finite number of off-lattice locations for atoms in the nanocluster. This treatment requires the definition of an effective coordination number that accounts for the effect of non-ideal inter-atomic distances on cohesive energy. There is a remaining need to derive meaningful scaling parameters for the effective coordination number that are consistent with the repulsive effects predicted by existing hard-core potentials.

4.5 NOTATION

Indices

i	on-lattice canvas location
j	neighboring on-lattice canvas location
k	off-lattice canvas location
l	neighboring off-lattice canvas location
s	secant line index

Sets

I	on-lattice canvas locations
L_i	neighboring on-lattice canvas locations to location i
K_i	off-lattice canvas locations for location i
K_j	off-lattice canvas locations for neighboring location j

Binary Variables

P_i	presence of an atom at on-lattice location i
Q_{ik}	presence of an atom at on-lattice location i at sub-lattice location k
X_{ijkl}	presence of a bond between atom i at sub-lattice locations k and atom j at sub-lattice location l

Continuous Variables

ECN_i	effective coordination number of atom at location i
E_i	square-root of the scaled, effective coordination number of atom at location i

Parameters

ECN_{\max}	maximum possible effective coordination number
N	number of atoms
α_s	slope of the secant line defining E_i
β_s	intercept of the secant line defining E_i
α_{ijkl}	repulsive energy scaling factor for effective coordination number

THE GENERALIZED ROBUST CUTTING-SET ALGORITHM

5.1 INTRODUCTION

Within the field of mathematical programming, robust optimization (RO) is a well-established approach for formulating and solving risk-averse models. The vast majority of the RO literature investigates its application on linear and convex models. Specifically, linear and convex RO and adjustable robust optimization (ARO)[10], wherein a subset of variables become “wait-and-see” decisions via functional dependence on the uncertainty, have had much success in identifying robust solutions to a variety of problems such as process scheduling[71–73, 107, 146], model-predictive control[114, 147], vehicle routing[109], project scheduling[15], industrial steam system optimization[150], and resilient network design[80], among many other settings. It has also been shown by Zhang, Grossmann, and Lima [145] that there are theoretical similarities between robust optimization and flexibility analysis when applied to linear systems, which highlights the fact that optimization under uncertainty has long been an area of focus within chemical engineering that has led to the development of novel methodologies.

A less widely utilized RO solution approach is the robust cutting-set algorithm (RCS), which was first proposed by Mutapcic and Boyd [91] as an adaptation of Kelley’s cutting-set approach[64] for application to inequality-only constrained robust optimization problems. In the RCS algorithm, the robust counterpart is solved by iterating between two subproblems, namely the *master* and the *separation* problems. In the master subproblem, optimal designs are identified that are robust against a carefully chosen finite set of uncertain parameter realizations. Then, given a master subproblem solution, the separation subproblem is solved to identify violating parameter realizations that are to be added back to the master problem, with the process repeating until no more violations can be found. This algorithmic solution approach is generally applicable to any continuous optimization problem, so long as the model possesses only

inequality constraints and/or equality constraints that can be reformulated via direct state-variable elimination

In this chapter, we aim to extend the aforementioned developments in robust optimization for process systems engineering by proposing a generalized robust cutting-set algorithm (GRCS). The generalization refers to the ability to handle general, two-stage nonlinear optimization problems, where nonlinearities may appear in the uncertain parameters or first- and second-stage variables. The resulting subproblem formulations and algorithmic procedure for applying the GRCS are presented here.

5.2 THE ROBUST COUNTERPART TO A PROCESS DESIGN FORMULATION

We begin the derivation of the robust counterpart with the definition of variables, parameters, and function mappings. For a process optimization model, we define *design* variables $x \in \mathcal{X} \subseteq \mathbb{R}^m$, *control* variables $z \in \mathbb{R}^n$, *state* variables $y \in \mathbb{R}^a$, and all potentially uncertain input data $q \in \mathbb{R}^w$. The domain of the design variables, $x \in \mathcal{X}$, is defined here abstractly to represent non-uncertain constraints involving just these variables. Most often, this domain incorporates the applicable variable bounds. The design variables (e.g., equipment sizes) are also referred to as first-stage variables, as they have to be committed upon before the true realization of the parameters q is known. In contrast, the control variables (e.g., flowrates, as manipulated via a valve) are also referred to as second-stage variables, as in principle their values can be adjusted after this realization is known. Finally, the state variables y are those second-stage variables that do not constitute degrees of freedom, rather they depend on the values of x , z , and q .

In process design optimization models, the objective function considered is most often an economic one, such as some net present value or some equivalent annual cost (or profit). Its general form is shown in Equation 5.1, where we split the objective into two parts, first-stage (i.e., investment) costs, $f_1(x)$, where $f_1 : \mathbb{R}^m \mapsto \mathbb{R}$, and second-stage (i.e., operational) costs, $f_2(x, z, y, q)$, where $f_2 : \mathbb{R}^{m+n+a} \mapsto \mathbb{R}$. Here, ζ represents the objective function value, which is to be minimized.

$$\zeta = f_1(x) + f_2(x, z, y, q) \quad (5.1)$$

Note how, by definition, the first-stage costs depend exclusively on the design variables and have no dependence on uncertain parameters. At the same time, both first- and second-stage variables are allowed to inform second-stage costs, which represents the most general case. For example, one may choose to not explicitly model the design of a pump, thus eliminating the freedom to control a given flowrate. However, the

state variable for that flowrate, which is evaluated at a particular design and control setting, could still induce an operational cost. There is also clear motivation to allow first-stage design variables to effect second-stage costs. For example, the height of the distillation column is chosen at the first stage, yet this height factors into calculating power consumption and the corresponding fluid pumping costs at the second stage.

Given q^0 to be a specific realization of the input data q , the deterministic formulation of a generic process design model is shown in Equations 5.2a–5.2c. Constraints 5.2b correspond to a set of inequality constraints $g_i : \mathbb{R}^{m+n+a} \mapsto \mathbb{R}$, $i \in \mathcal{I}$, to which we will be referring to as *performance constraints*. The latter typically express desirable levels of system performance metrics, such as product yields, utilities usage, or safety thresholds, but a few examples. Explicit bounds on the z variables are also considered as part of the g_i constraints, for notational convenience. The equality constraints 5.2c, $h_j : \mathbb{R}^{m+n+a} \mapsto \mathbb{R}$, $j \in \mathcal{J}$, are a system of state equations that define the state variables y . Note that we pose no restriction on the nature of the objective or constraints, meaning they may be linear or nonlinear (convex or non-convex) in either the model variables, parameters, or both.

$$\min_{\substack{x \in \mathcal{X}, \\ z \in \mathbb{R}^n, y \in \mathbb{R}^a}} f_1(x) + f_2(x, z, y; q^0) \quad (5.2a)$$

$$\text{s.t.} \quad g_i(x, z, y; q^0) \leq 0 \quad \forall i \in \mathcal{I} \quad (5.2b)$$

$$h_j(x, z, y; q^0) = 0 \quad \forall j \in \mathcal{J} \quad (5.2c)$$

The above formulation is deterministic because it only considers a single value for each parameter in the model. However, if the input parameter data are indeed uncertain, and if that uncertainty is properly characterized, then the above deterministic model can be used as the basis to casting a robust optimization model. More specifically, let us postulate that the uncertain data q may attain values from within a general uncertainty set, $\mathcal{Q} \subset \mathbb{R}^w$. The form of the (often multidimensional) uncertainty set is chosen by the modeler in each case, usually with the help of a suitable parameter estimation method. Typically, the “shape” of the set is such that it captures known correlations among the uncertain parameters, while the “size” of the set is tuned to reflect a desirable confidence interval for their realization. In principle, the uncertainty sets in our framework can take on any form, e.g., a continuous convex or non-convex set, or a disjoint set (e.g., a set of discrete points representing scenarios). Without loss of generality, however, in the remainder of this thesis we will limit our attention to uncertainty sets that are continuous and compact. Convexity is not a necessary property for our sets.

Given uncertain parameters and an associated uncertainty set, $q \in \mathcal{Q}$, Equations 5.3a–5.3c represent the robust counterpart formulation of the

previously shown deterministic model. This robust counterpart constitutes a two-stage, min-max-min formulation, where decisions x are taken *before* the realized value of the uncertain parameters is known, while decisions z are taken *after* this is the case. Due to their nature as state variables, values for variables y are also chosen *after* the realization of the uncertainty. We assume that all state variables are non-trivial, meaning they are coupled to uncertain parameters q or second-stage variables z and cannot be simply solved out of the model equations.

$$\min_{x \in \mathcal{X}} \max_{q \in \mathcal{Q}} \min_{z \in \mathbb{R}^n, y \in \mathbb{R}^a} f_1(x) + f_2(x, z, y, q) \quad (5.3a)$$

$$\text{s.t.} \quad g_i(x, z, y, q) \leq 0 \quad \forall i \in \mathcal{I} \quad (5.3b)$$

$$h_j(x, z, y, q) = 0 \quad \forall j \in \mathcal{J} \quad (5.3c)$$

In its current form, the robust counterpart allows total flexibility for the control variables within the inner minimization problem. To simplify the two-stage robust counterpart, we employ decision rules (DR), which have been ubiquitously used in the area of adjustable robust optimization (ARO) to convert second-stage variables into a function of the uncertain parameters q and new first-stage decision variables d (the parameterization of the decision rules themselves). ARO with affine decision rules, i.e. an affine relationship between uncertain parameters and second-stage variables, was first proposed by Ben-Tal et al. [10]. In process systems engineering applications, affine decision rules have also been used to solve two-stage robust optimization problems in the contexts of water treatment networks[62] and steel-making processes[98]. ARO with generalized affine decision rules for mixed-integer linear optimization models has also been recently demonstrated by Avraamidou and Pistikopoulos [5] via a multi-parametric programming approach.

We note that the motivation for strictly affine decision rules to preserve linear model structure does not apply here, as most chemical process models possess nonlinearities. Here, we present a general functional relationship for decision rules, as shown in Equations 5.4, to highlight the fact that the proposed approach can admit any general, nonlinear decision rule function. In this equation, each control variable z_ℓ , $\ell \in \{1, \dots, n\}$, has a functional dependence on the uncertain parameters q and the corresponding first-stage variables, d_ℓ . More specifically, $d_\ell \in \mathbb{R}^p$, $\forall \ell \in \{1, \dots, n\}$, are mutually exclusive subvectors of d that are only referenced in the specific decision rule function $v_\ell : \mathbb{R}^{p+w} \mapsto \mathbb{R}$ associated with each z_ℓ . In Section 5.2.2, we will specify the general form of Equation 5.4 to consider constant, affine, and quadratic decision rules, which we will later employ in our computational studies.

$$z_\ell = v_\ell(d_\ell, q) \quad \forall \ell \in \{1, \dots, n\} \quad (5.4)$$

The application of this general decision rules relationship modifies the robust counterpart to formulation RC , which is shown in Equations 5.5a–5.5e. Note how the functional dependence of the control policy on the realization of uncertainty, as expressed via the decision rules, is chosen at the first stage. Furthermore, an auxiliary epigraph variable $\zeta \in \mathbb{R}$ has been incorporated to push the objective function to the set of constraints, for convenience. It is assumed from this point forward that any given values of x , z and q map to a unique value of y . Under this assumption, y are simply evaluated in the maximization step.ⁱ Furthermore, the inner minimization problem possesses no degrees of freedom and merely evaluates the (robust) feasibility and (worst-case) objective value resulting from our first-stage decisions.

$$(RC) : \min_{\substack{x \in \mathcal{X}, \\ d_\ell \in \mathbb{R}^p \forall \ell}} \max_{\substack{q \in \mathcal{Q}, \\ z \in \mathbb{R}^n, y \in \mathbb{R}^a}} \min_{\zeta \in \mathbb{R}} \zeta \quad (5.5a)$$

$$\text{s.t. } \zeta \geq f_1(x) + f_2(x, z, y, q) \quad (5.5b)$$

$$g_i(x, z, y, q) \leq 0 \quad \forall i \in \mathcal{I} \quad (5.5c)$$

$$\text{s.t. } h_j(x, z, y, q) = 0 \quad \forall j \in \mathcal{J} \quad (5.5d)$$

$$z_\ell = v_\ell(d_\ell, q) \quad \forall \ell \in \{1, \dots, n\} \quad (5.5e)$$

The above model enforces the robust feasibility of the overall design under the postulated control policy, which suffices to qualify the design as robust. However, we remark that the decision maker need not commit *a priori* to following this policy, as it may lead to overly restrictive control actions in light of certain realizations. In practice, the recourse actions to be followed will be determined *a posteriori* by solving an optimal control (or operational) optimization problem after the uncertainty parameters have revealed their true values for the operating period of interest. We elaborate further on these issues in Section 5.4, where we calculate expected second-stage variable values to *a posteriori* assess the overall performance and cost of the robust designs.

5.2.1 The Generalized Robust Cutting-Set Algorithm

In this section, we devise a cutting-set based solution approach to address formulation RC . The generalized robust cutting-set algorithm (GRCS) defines certain subproblems: a master problem and a set of separation problems, one for each constraint 5.5b and 5.5c. As this is an iterative

ⁱ This unique mapping from (x, z, q) to y is a reasonable assumption for most process models. If this assumption does not hold, however, the offending elements of vector y should be regarded as flexible, second-stage variables and handled in the same way as the z variables.

solution approach, the master and separation problems are solved in alternating fashion until converging to the robust solution. To that end, these problems are denoted and indexed as MP_k , SP_i^{perf} , $\forall i \in \mathcal{I}$, and SP^{obj} . Here, k is the iteration index, i is used for the separation problems to denote which performance constraint g_i (out of a total of $|\mathcal{I}|$ such constraints) the problem refers to, while the superscript “obj” implies that the last problem is associated with the epigraph constraint 5.5b. A flowchart representing the algorithm is shown in Figure 5.1.

The initial master problem, MP_0 , is initialized to be the deterministic model, defined for some nominal value of the uncertain parameters, q^0 . In each iteration, the master problem MP_k is solved and requires that the solution is feasible against all realizations q^k , $k \in \mathcal{K}$, that have been identified during the GRCS algorithm’s progression. In the separation problems SP_i^{perf} , we search for new realizations of uncertainty that render the master problem solution infeasible, leading to a constraint violation in one or more of the g_i inequality constraints. Then, the parameter realization that corresponds to the largest relative constraint violation, q^* , is identified. If a violation is indeed identified in the current iteration of the algorithm, a full copy of the second stage variables is defined and all constraints (i.e., objective epigraph, performance constraints, state equations, decision rules), instantiated for the offending realization q^* , are added back to the master problem using these variables (along with the original, common set of first stage variables). Conversely, if there are no violations identified in SP_i^{perf} , for any $i \in \mathcal{I}$, when solved to global optimality, then the current design x^* is deemed robust feasible. At that point, the separation effort could switch focus to solving problem SP^{obj} so as to identify realizations q^* that yield worse objective value than the one currently at hand, in order to reach worst-case optimality.

The formulations and solution approaches for these subproblems will be explained in more detail in the following sections, but before we do so, it is important to discuss convergence properties. We start by noting that Mutapcic and Boyd [91] provide a proof of convergence for their original cutting-set method. In that proof, it is argued that convexity of the model is not required to prove convergence, but it is assumed to ensure tractable subproblem solving steps. Hence, by following a similar proof as in the above reference, we can guarantee convergence in terms of total number of iterations for the GRCS algorithm presented here, as long as any and all master and separation subproblems that arise are tractable. This is shown in an adapted proof in Appendix 5.6. Of course, the latter is an assumption that may or may not hold in the context of process design models of interest to this work, given that the subproblems generated by our algorithm will be non-convex, in general. Indeed, the presence of non-linear irremovable state equations h_j , $j \in \mathcal{J}$, in the proposed master and

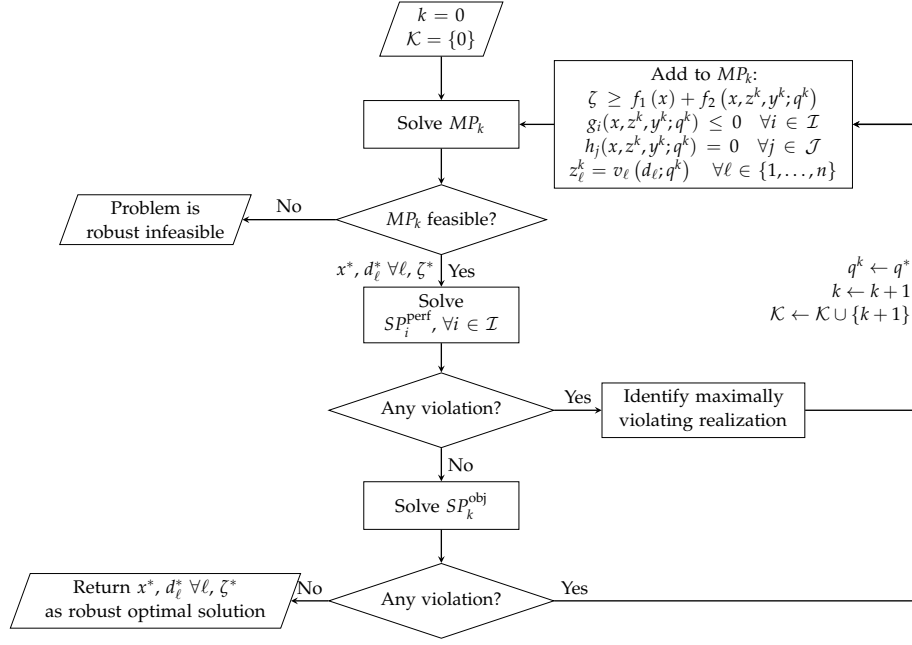


Figure 5.1: The generalized robust cutting-set algorithm.

separation problem formulations will generally preempt the non-convexity of the subproblems. From this perspective, if a pathological subproblem is generated during the progression of the GRCS algorithm such that the subordinate (local or global) nonlinear programming solver cannot converge to an optimal solution, then the overall algorithm will also stall. Despite this possibility, however, we should mention that we have had empirical success in using the GRCS algorithm to solve various rather complex process systems models in reasonable time scales, as we later demonstrate in Chapter 6.

5.2.1.1 The Master Problem

The general form of the master problem is shown in Equations 5.6a–5.6e. In this formulation, control variables, state variables and uncertain

parameters are indexed over a set \mathcal{K} , which is the set of iterations of the GRCS algorithm.

$$(MP_k) : \min_{\substack{x \in \mathcal{X}, \zeta \in \mathbb{R} \\ d_\ell \in \mathbb{R}^p \forall \ell, \\ z^k \in \mathbb{R}^n, y^k \in \mathbb{R}^a \forall k}} \zeta \quad (5.6a)$$

$$\text{s.t.} \quad \zeta \geq f_1(x) + f_2(x, z^k, y^k; q^k) \quad \forall k \in \mathcal{K} \quad (5.6b)$$

$$g_i(x, z^k, y^k; q^k) \leq 0 \quad \forall k \in \mathcal{K}, \forall i \in \mathcal{I} \quad (5.6c)$$

$$h_j(x, z^k, y^k; q^k) = 0 \quad \forall k \in \mathcal{K}, \forall j \in \mathcal{J} \quad (5.6d)$$

$$z_\ell^k = v_\ell(d_\ell; q^k) \quad \forall k \in \mathcal{K}, \forall \ell \in \{1, \dots, n\} \quad (5.6e)$$

Note how the constraints in formulation MP_k are cast for all iterations $k \in \mathcal{K}$, meaning that the number of constraints in the master problem increases by $(1 + |\mathcal{I}| + |\mathcal{J}| + n)$ in each iteration. Furthermore, note how a separate set of state variables y^k has been defined for each uncertain realization q^k that is explicitly referenced in this formulation, which is necessary due to the implicit dependence of variables y on uncertain parameters q in the robust counterpart. Similarly, separate sets of second-stage variables are also considered to reflect the fact that different values (z^k) for the control action might have to be chosen, if a different realization of the uncertain parameters (q^k) prevails. We remark that, unlike the case of implicit state variables, the introduction of separate copies of the control variables in MP_k is not strictly necessary, as in the actual implementation, one may simply substitute variables z_ℓ^k out of the formulation using Equations 5.6e.

5.2.1.2 The Separation Problems

Each separation problem SP_i^{perf} seeks to identify, if one exists, a realization of the uncertain parameters belonging to the uncertainty set, i.e., $q \in \mathcal{Q}$, which renders the optimal design (and associated control policy) from the most recently solved master problem, $MP_{|\mathcal{K}|-1}$, infeasible with respect to performance constraint g_i . The general formulation for these separation problems is shown in Equations 5.7a–5.7c. A strictly positive optimal objective value in this problem reveals that there exists a realization of the uncertain parameters (the separation problem's optimal solution itself) that makes the master problem solution, (x^*, d^*) , infeasible with respect to constraint g_i . Conversely, a non-positive globally optimal value is a certificate that the master solution will satisfy constraint g_i for all realizations

in the uncertainty set. Note that a separation problem of this type must be solved successively for each performance constraint g_i , $i \in \mathcal{I}$, in order to verify the overall robustness of the master solution.

$$(SP_i^{\text{perf}}) : \max_{\substack{q \in \mathcal{Q}, \\ z \in \mathbb{R}^n, y \in \mathbb{R}^a}} g_i(z, y, q; x^*) \quad (5.7a)$$

$$\text{s.t.} \quad h_j(z, y, q; x^*) = 0 \quad \forall j \in \mathcal{J} \quad (5.7b)$$

$$z_\ell = v_\ell(q; d_\ell^*) \quad \forall \ell \in \{1, \dots, n\} \quad (5.7c)$$

In a similar fashion, the separation problem SP^{obj} seeks to identify, if one exists, a realization of the uncertain parameters that would lead the most recently identified design (and associated control policy) to a worse objective value than the one the master problem solution suggested. The general formulation for this separation problem is shown in Equations 5.8a–5.8c. The optimal objective value of this problem being strictly positive signifies that there exists a realization of the uncertain parameters (the separation problem’s optimal solution itself) that would have evaluated the master problem solution, (x^*, d^*) , to a worse objective, and hence, the solution has not been properly adjudicated in terms of its worst-case performance. Conversely, a non-positive globally optimal value is a certificate that this has occurred.

$$(SP^{\text{obj}}) : \max_{\substack{q \in \mathcal{Q}, \\ z \in \mathbb{R}^n, y \in \mathbb{R}^a}} f_1(x^*) + f_2(z, y, q; x^*) - \zeta^* \quad (5.8a)$$

$$\text{s.t.} \quad h_j(z, y, q; x^*) = 0 \quad \forall j \in \mathcal{J} \quad (5.8b)$$

$$z_\ell = v_\ell(q; d_\ell^*) \quad \forall \ell \in \{1, \dots, n\} \quad (5.8c)$$

We highlight that the above formulations are parameterized over the set of first-stage design variables and decision rules, which are fixed to the optimal values from the previous master problem solution, x^* and d^* . At the same time, the uncertain parameters q constitute here decision variables that are free to take on any value in the uncertainty set \mathcal{Q} . This form of the separation problem differs from the original cutting-plane algorithm proposed by Mutapcic and Boyd [91] due to the necessity of carrying through the block of state equations h_j (Equations 5.7b and 5.8b) to evaluate the state variables y that are associated with the solution of the separation problem. Additionally, decision rule relationships (Equations 5.7c and 5.8c) must also be included in the formulation so as order to evaluate the control actions in the context of the separation problem’s optimal solution.

5.2.2 Decision Rules

In this section, we specify the form of the decision rule functions v_ℓ we consider in this study. More specifically, we consider three forms

that we refer to as the static approximation, affine decision rules, and quadratic decision rules. We highlight that Bertsimas, Iancu, and Parrilo [12] have explored polynomial decision rule relationships, including cubic ones, for linear dynamical systems affected by uncertainty. To the best of our knowledge, however, this work is the first to apply nonlinear decision rules in the context of nonlinear optimization models such as those arising in process design applications. For convenience, we will partition the variables d_ℓ into intercepts, d_ℓ^0 , coefficients for linear terms d_ℓ^1 , and coefficients for quadratic terms d_ℓ^2 to include both squares and bilinear terms.

$$v_\ell(d_\ell^0) = d_\ell^0 \quad \forall \ell \in \{1, \dots, n\} \quad (5.9)$$

$$v_\ell(d_\ell^0, d_\ell^1, q) = d_\ell^0 + \sum_{r=1}^w d_{\ell,r}^1 q_r \quad \forall \ell \in \{1, \dots, n\} \quad (5.10)$$

$$v_\ell(d_\ell^0, d_\ell^1, d_\ell^2, q) = d_\ell^0 + \sum_{r=1}^w d_{\ell,r}^1 q_r + \sum_{r=1}^w \sum_{s=r}^w d_{\ell,r,s}^2 q_r q_s \quad \forall \ell \in \{1, \dots, n\} \quad (5.11)$$

The form of the static approximation is shown in Equations 5.9. In this case, each control variable is chosen to a single value, and there is no flexibility in the control to respond to the uncertainty, once the latter is revealed. Effectively, the control variables are treated as first-stage decision variables. The affine decision rules are presented in Equations 5.10, and they constitute an affine relationship between the (first-stage) decision rule variables, d , and the uncertain parameters, q . Finally, we show the form for the quadratic decision rules in Equations 5.11, which constitutes a more general, nonlinear function that features also square and bilinear terms in the uncertain parameters. The modeler selects whether to utilize the static approximation, affine or quadratic decision rules in each case. Typically, this selection will be based on experience with the specific application of interest, in terms of the trade-off between computational tractability and extent of (and desire to reduce) the two-stage adaptivity gaps that arise.

5.3 IMPLEMENTATION DETAILS

In this section, we outline various important details regarding our implementation of the proposed GRCS algorithm. For many aspects of the algorithm, we recognize that there exist other options for implementation, and we elaborate on which options we selected in each case. Those choices have been largely informed by their effect on overall tractability, as assessed via the computational case studies we conducted to address various complex process models. Those studies are presented later in Chapter 6. A modified and more detailed flowchart illustrating the implementation details outlined in this section is presented in Figure 5.2.

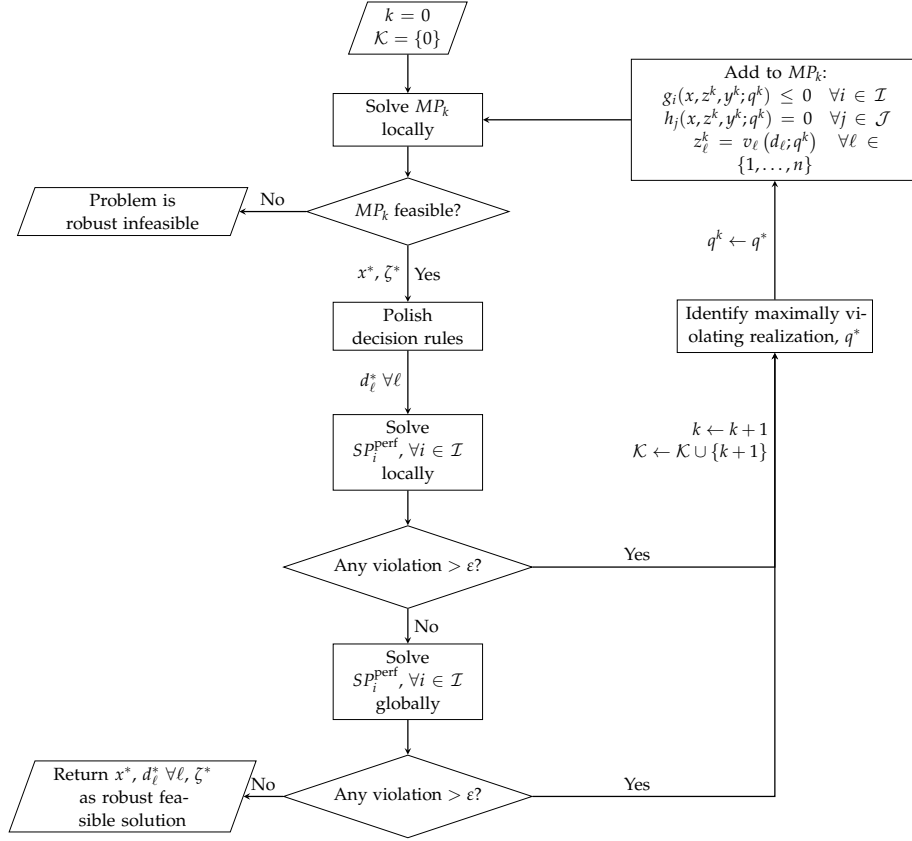


Figure 5.2: Implementation details of the generalized robust cutting-set algorithm.

5.3.1 Solving Master Problems

We first note that, in our implementation of the GRCS algorithm, we choose to solve all master problems MP_k locally, using a local nonlinear programming (NLP) solver. This is done given the relative inability of today's global NLP solvers to solve to zero gap the complex process design models we are interested in addressing in this study, even in their deterministic version. This shortfall is of course exacerbated by the fact that the master problem size increases in each iteration. The choice to solve master problems locally comes at the expense of not being able to assert traditional *robust optimality* of the final solutions, rather only their *robust feasibility*. We note, however, that the GRCS algorithm could in principle prove robust optimality, provided all master problems are solved globally at each iteration.

In light of the above, it becomes superfluous to seek to solve problems SP^{obj} at the interest of separating realizations that lead to worst-case objectives. Therefore, in this work, we only separate problems SP_i^{perf} , which suffices to guarantee robust feasibility of the final solutions. Furthermore,

we target to identify master problem solutions that perform best in the *nominal case*, i.e., $q \leftarrow q^0$. This is done by setting the set of realizations associated with constraints 5.6b to only include q^0 ; that is, $\mathcal{K} \leftarrow \{0\}$ for (only) constraints 5.6b when solving every iteration of the master problem. The choice of the nominal values of the uncertain parameters, q^0 , is in principle left for the modeler. Here, we consider q^0 as the most likely realization of q , as determined in expectation, which is also often used in process systems contexts for “deterministic” optimization. In addition to solving master problems targeting to minimize nominal costs, care can be taken to limit the variance of second-stage costs, e.g., by including *p-robust constraints*[108] to limit their increase in other, non-nominal scenarios.

5.3.2 Separation Approach

In order to guarantee the robust feasibility of the final solution determined by the GRCS algorithm, the separation problems must be solved to global optimality for each and every performance constraint. This ensures that there are no realizations of the uncertain parameters (within the uncertainty set) that render the final robust design infeasible. Unlike master problems, the size and dimensionality of separation problems are much smaller, making them tractable for global optimization in this setting. Regardless, the repeated execution of global optimization runs might still add up to significant computational burden. To that end, we choose in our implementation to first solve each separation problem locally. If violating parameters can be identified in this manner, the algorithm can proceed with its next iteration, defining the new master problem based on the local solution of a separation problem. However, whenever the local search returns no violation for each and every performance constraint, we do proceed to solve the separation problem using a global solver, in order to assert robust feasibility. This protocol reduces the overall number of expensive calls to a global optimization solver, and has been found to improve overall algorithm performance.

We now offer some remarks regarding the selection of which violating realization is chosen to iterate the GRCS algorithm when more than one performance constraints g_i exist in a model (i.e., $|\mathcal{I}| > 1$). In such a case, there may be different violating uncertainty realizations identified in separation. Whereas any and all of them could be chosen as critical scenarios against which to explicitly insure feasibility in the subsequent master problem iteration, we recognize that choosing more than one scenario would contribute to rapid increase of the master problem size beyond what is absolutely necessary. Therefore, in our implementation, we choose to only add a single violation, which is selected as follows. Let $\mathcal{I}^V \subseteq \mathcal{I}$ be the subset of performance constraints that can be violated in the

context of the most recent master problem solution. More specifically, given $(z^{i,*}, y^{i,*}, q^{i,*})$ as the optimal solution of problem SP_i^{perf} , we consider $i \in \mathcal{I}^V$ when $g_i(z^{i,*}, y^{i,*}, q^{i,*}; x^*) > \varepsilon$, where $\varepsilon \in \mathbb{R}_+$ is a small tolerance. Once the realizations $q^{j,*}$ that maximally violate each of the constraints $j \in \mathcal{I}^V$ have been determined, we generate a matrix of dimensions $|\mathcal{I}^V| \times |\mathcal{I}^V|$, where each entry $e_{i,j}$ represents the violation of constraint g_i associated with row i under the uncertain parameter realization $q^{j,*}$ associated with column j ; that is, $e_{i,j} := \max\{g_i(z^{i,*}, y^{i,*}, q^{j,*}; x^*), 0\}$. The entries are normalized by dividing each of them with the largest value in their row, leading to the maximum entry in each row being equal to one. The maximally violating realization to be added back to the next iteration of the master problem, q^* , is then picked as the one associated with the column possessing the largest sum of entries; that is, $q^* = q^{\gamma,*}$, where $\gamma := \arg \max_{j \in \mathcal{I}^V} \{\sum_{i \in \mathcal{I}^V} e_{i,j}\}$.

5.3.3 Decision Rules Polishing

It is a well-known fact that the use of decision rules often leads to solution degeneracy. In particular, when one solves the master problem at a given iteration k , there may exist many equivalently optimal combinations of decision rule coefficients d that satisfy Equations 5.6e. To that end, when using affine and quadratic decision rules, we augment our implementation with an additional post-processing step after solving the master problem, which we refer to as the *decision rules polishing* step. The purpose of this step is to comb through the set of equivalently optimal decision rules and to judiciously pick a specific, desirable policy. In our context, we consider decision rule coefficient values, $d_\ell \in \mathbb{R}^p$, $\ell \in \{1, \dots, n\}$, to be more desirable, if their relative magnitude is collectively smaller. To achieve this, we solve the auxiliary optimization problem shown in Equations 5.12a–

5.12h as soon as an optimal solution x^* with optimal objective value ζ^* is obtained from the master problem in each iteration k .

$$\min_{\substack{d_\ell \in \mathbb{R}^p \forall \ell, \\ \tau_\ell^0 \in \mathbb{R}_+, \tau_\ell^1 \in \mathbb{R}_+^w, \tau_\ell^2 \in \mathbb{R}_+^{w \times w} \forall \ell, \\ z^k \in \mathbb{R}^n, y^k \in \mathbb{R}^a \forall k}} \sum_{l=1}^n \left(\tau_l^0 + \sum_{r=1}^w \tau_{\ell,r}^1 + \sum_{r=1}^w \sum_{s=r}^w \tau_{\ell,r,s}^2 \right) \quad (5.12a)$$

$$\text{s.t.} \quad \zeta^* \geq f_1(x^*) + f_2(z^0, y^0; x^*, q^0) \quad (5.12b)$$

$$g_i(z^k, y^k; x^*, q^k) \leq 0 \quad \forall k \in \mathcal{K}, \forall i \in \mathcal{I} \quad (5.12c)$$

$$h_j(z^k, y^k; x^*, q^k) = 0 \quad \forall k \in \mathcal{K}, \forall j \in \mathcal{J} \quad (5.12d)$$

$$z_\ell^k = v_\ell(d_\ell; q^k) \quad \forall k \in \mathcal{K}, \forall \ell \in \{1, \dots, n\} \quad (5.12e)$$

$$-\tau_\ell^0 \leq d_\ell^0 \leq +\tau_\ell^0 \quad \forall \ell \in \{1, \dots, n\} \quad (5.12f)$$

$$-\tau_{\ell,r}^1 \leq d_{\ell,r}^1 q_r^0 \leq +\tau_{\ell,r}^1 \quad \forall \ell \in \{1, \dots, n\}, \forall r \in \{1, \dots, w\} \quad (5.12g)$$

$$-\tau_{\ell,r,s}^2 \leq d_{\ell,r,s}^2 q_r^0 q_s^0 \leq +\tau_{\ell,r,s}^2 \quad \forall \ell \in \{1, \dots, n\}, \forall r,s \in \{1, \dots, w\}: \{s \geq r\} \quad (5.12h)$$

In the above formulation, the objective function 5.12a minimizes the L_1 -norm of the vector of terms appearing in the applicable decision rule function, where τ are non-negative auxiliary variables introduced to represent the absolute values of each and every such term. Here, we focus the presentation on the case of quadratic decision rules as per Equations 5.11, noting that a reduced version of this formulation applicable for the case of affine decision rules can be obtained by simply fixing all τ_ℓ^2 variables to zero.ⁱⁱ We remark that the objective of the above formulation focuses on the decision rules evaluated under the nominal realization of uncertainty q^0 , corresponding to control actions z^0 , but other selections (e.g., an average of all realizations q^k) could be readily used instead. Constraint 5.12b is added to ensure that we are searching over the set of optimal decision rule policies, i.e., those that induce the same objective value as the master problem solution,ⁱⁱⁱ while constraints 5.12c–5.12e are added to ensure that

ii In the case of a quadratic decision rule function, it holds by construction that $p := 1 + w + w(w + 1)/2$.

iii Since ζ^* is the minimal objective value of the master problem, requiring that the objective of the polishing problem attains a value no greater than ζ^* is equivalent to requiring that its solution corresponds to one of the equivalently optimal solutions of the master problem.

we are searching over decision rule policies that remain feasible for the original master problem. Finally, constraints 5.12f–5.12h achieve the desired definition for variables τ as the absolute values of the corresponding decision rule function terms.

5.4 EVALUATION OF ROBUST SOLUTION QUALITY

The GRCS is designed to address primarily two-stage decision problems, wherein recourse decisions are permitted to adapt to undesirable deviations in performance caused by parameter uncertainty. These recourse decisions are represented in this presentation by the variables z , where for modeling convenience, we chose to limit them to values attained via decision rule relationships. In principle, however, an operator making decisions in the second stage is not beholden to the optimal decision rule functions and has the ability to respond to the actual realization of the uncertainty in an unrestricted fashion. Therefore, from a practical perspective, it is important to quantify the range of possible values for second-stage decision variables and the corresponding expectation and variance of the second-stage costs they might induce. These metrics allow the modeler to assess the accuracy of the decision rule approximation for a given recourse action, and to explicitly build confidence regarding the overall economic performance of the robust design at hand. In this section, we illustrate how to compute the expected operating costs and expected control variable values, as well as the associated variances, for a given robust feasible design x^* .

To compute these expected values, we solve a set of optimization problems where the control variables are free to take on any feasible value. We start with the deterministic formulation in Equations 5.2a–5.2c and fix the first-stage design variables to the robust design, $x \leftarrow x^*$. Then, we assign a randomly sampled value to the uncertain parameters, $q \leftarrow q_s$, resulting in the model shown in Equations 5.13a–5.13c.

$$\min_{z \in \mathbb{R}^n, y \in \mathbb{R}^d} f_2(z, y; x^*, q_s) \quad (5.13a)$$

$$\text{s.t.} \quad g_i(z, y; x^*, q_s) \leq 0 \quad \forall i \in \mathcal{I} \quad (5.13b)$$

$$h_j(z, y; x^*, q_s) = 0 \quad \forall j \in \mathcal{J} \quad (5.13c)$$

This optimization model is then solved for a set \mathcal{S} of uncertainty scenarios, q_s , $s \in \mathcal{S}$, which have been suitably defined to this purpose.^{iv} In order to determine desirable metrics about their distribution, the optimal solutions, (z^*, y^*) , and optimal second-stage costs, $f_2(z^*, y^*; x^*, q_s)$, are

iv We remark that the chosen scenarios may be samples from within the uncertainty set, i.e., $q_s \in \mathcal{Q}$, or may be out-of-sample scenarios.

thus recorded under each scenario.^v More specifically, given probabilities p_s for all sampled scenarios $s \in \mathcal{S}$, Equations 5.14 and 5.15 are used to compute the expected operating costs and their standard deviation. The expected second-stage costs may be considered when comparing robust feasible designs obtained via the GRCS algorithm to any deterministically optimal design, in order to elucidate the overall cost increase for insuring design robustness.

$$\mathbb{E}[f_2] = \sum_{s \in \mathcal{S}} p_s f_2(z^*, y^*; x^*, q_s) \quad (5.14)$$

$$\sigma[f_2] = \sqrt{\sum_{s \in \mathcal{S}} p_s (f_2(z^*, y^*; x^*, q_s) - \mathbb{E}[f_2])^2} \quad (5.15)$$

Using similar formulas, the above-described after-the-fact analysis is also useful to understand the range of control variable actions, namely $\mathbb{E}[z_\ell]$ and $\sigma[z_\ell]$ for all $\ell \in \{1, \dots, n\}$, that shall be required to achieve robust feasibility in the context of a given design. We demonstrate such analyses in the computational case studies we present in the next section.

5.5 CONCLUSIONS

In this chapter we have outlined modeling and algorithmic approaches for applying robust optimization to general two-stage nonlinear problems. We generalize the robust cutting-set algorithm and the corresponding master and separation subproblems to accommodate uncertain optimization problems with nonlinear or non-convex constraints, which include irremovable state equations. We consider a single-stage robust counterpart with general nonlinear decision rules, but focus only on constant, affine, and quadratic relationships between uncertain parameters and first-stage variables. Given the decision rule approximation, we provide practical schemes for mitigating degeneracy in decision rules via a polishing formulation, as well as a scheme for retrieving expected second-stage costs under fully-adaptive recourse variables. We have shown that for general nonlinear problems which are common in process engineering, robust optimization is a viable approach for identifying risk-averse solutions.

^v We remark that the first-stage costs would not vary depending on the scenario q_s , since the design variables, x^* , are kept fixed in this analysis.

5.6 APPENDIX

5.6.1 Convergence Proof

This proof is based on the resulting proof found in Mutapcic and Boyd [91] which relies on ideas from the cutting-set method proof in Kelley [64].

Remark 1. We propose that in the case that second-stage decision variables z are static, they can be considered part of the first-stage variable vector x . We assume that in the case that the z variables are adaptive, they reside in the group of state variables y , and additional decision rule variables d are then part of first-stage variables x , while the decision rule functions become part of the set of state equations h_j . This covers all cases for handling second-stage variables in the GRCS algorithm.

Assumption 1. We assume that the state variables y are uniquely defined via corresponding state equations h_j by a selection of x and q (or just q in the case of z variables).

Assumption 2. We concatenate the vectors x and y to a single vector $v = (x, y)$. We assume that the set of feasible solutions to the nominal problem, S_{nom} is bounded. Additionally, we assume inequality constraints g_i are uniformly Lipschitz continuous in both state and decision variables, v , on the set S_{nom} . This implies that there exists a constant C such that the following holds:

$$|g_i(v_1, q) - g_i(v_2, q)| \leq C \|v_1 - v_2\| \quad (5.16)$$

for all inequality constraints, $i = 1, \dots, m$, all pairs of variable vectors, $v_1, v_2 \in S_{nom}$, and all uncertain parameter realizations, $q \in \mathcal{Q}$.

Assumption 3. We assume one cannot guarantee a globally optimal solution is obtainable for all iterations k of the separation problems, SP_k . As is the case in our implementation of the GRCS, we instead assume that an local solutions are obtained at all intermediate iterations. At the final separation problem, we assume a global optimal solution is obtainable to confirm there no longer exists a realization of uncertain parameters $q \in \mathcal{Q}$ which makes our current optimal solution v^{k*} infeasible. Also, we do not assume the inequality constraint functions g_i are convex in v for each $q \in \mathcal{Q}$, and thus, the global optimization step may be intractable.

Proof. We take S_{master}^k to be the set of feasible solutions to the k th master problem MP_k . The vector v^{k*} represents the optimal solution to MP_k .

If for $k = 1, \dots, K$ the GRCS has not terminated, this proof will provide an upper bound on how large the total number of iterations can be with respect to the robustness tolerance TOL and the Lipschitz constant, C .

Let k be an index for a violating uncertain parameter realization q^k identified in separation problem SP_k and then added to the set \mathcal{K} . This q^k satisfies the following, given the previous master problem solution v^{k*} :

$$g_i(v^{k*}, q^k) = \text{Violation}(v^{k*}) > TOL, \quad (5.17)$$

All following solutions in the algorithmic progression, $v^s \in \mathcal{S}_{master}^s$ with $s > k$, must be feasible for $g_i(v^s, q^k) \leq 0$, since this constraint is explicitly enforced in the s th master problem.

In other words, this means that the following is true:

$$g_i(v^s, q^k) \leq 0 \text{ for } s > k, \quad (5.18)$$

We can then combine (5.17) and (5.18) to show that for $k < s$,

$$g_i(v^{k*}, q^k) - g_i(v^s, q^k) > TOL \quad (5.19)$$

By also considering the Lipschitz condition from (5.16), we can also show

$$\|v^{k*} - v^s\| > \frac{TOL}{C} \quad (5.20)$$

for $k < s$. The relationship in (5.20) states that the ratio TOL/C is a lower bound on the distance between any two solution vectors in v^1, \dots, v^K .

Next, we will use an argument based on volumes to show how this lower bound on solution distances relates to K , the total number of iterations. We select B_k to be the ball of radius TOL/C with its center at v^{k*} for all k . By conclusions drawn via (5.20), we know these balls do not intersect because TOL is a small, non-zero constant and thus the distance $\|v^{k*} - v^s\|$ must be non-zero. Therefore, the total volume represented by these K balls is K times the volume of a single ball. This quantity is $K\beta_n(TOL/C)^n$, where β_n is the volume of the unit ball in \mathbb{R}^n , i.e. $\beta_n = \frac{\pi^{n/2}}{\Gamma(n/2+1)}R^n$ where $R = 1$ is the radius and Γ is Euler's gamma function.

We define another ball B to be the ball with a radius R that contains the set of solutions to the nominal problem, \mathcal{S}_{nom} . Then the balls B_1, \dots, B_K are contained in the ball \tilde{B} , which is B with radius $R + TOL/C$. Thus, we know that the total volume of the balls B_1, \dots, B_K must be less than the volume of \tilde{B} , which is $\beta_n(R + TOL/C)^n$. Therefore, we can show

$$K\beta_n(TOL/C)^n \leq \beta_n(R + TOL/C)^n \quad (5.21)$$

which we can simplify to determine the bound

$$K \leq \left(\frac{RC}{TOL} + 1 \right)^n \quad (5.22)$$

The right hand side gives an upper bound on the number of iterations before the GRCS terminates, which is related to the robust feasibility tolerance, TOL , the radius of the ball enclosing the feasible set, R , the dimensionality of the ball containing the feasible set, n , and the Lipschitz constant, C .

□

5.7 NOTATION

Indices

i	inequality constraints
j	equality constraints
k	iterations of the GRCS
l	second-stage variables
s	uncertain parameter scenarios

Sets

\mathcal{I}	set of inequality (performance) constraints
\mathcal{J}	set of equality (state) equations
\mathcal{K}	set of GRCS iterations
\mathcal{Q}	uncertainty set
\mathcal{S}	finite set of uncertain parameter scenarios

Continuous Variables

$x \in \mathcal{X} \subseteq \mathbb{R}^m$	first-stage (design) variables
$z \in \mathbb{R}^n$	second-stage (control) variables
$y \in \mathbb{R}^a$	state variables
$d \in \mathbb{R}^p$	decision rule variables
$\tau_\ell^0 \in \mathbb{R}_+, \tau_\ell^1 \in \mathbb{R}_+^w, \tau_\ell^2 \in \mathbb{R}_+^{w \times w} \forall \ell$	decision rule polishing variables
$q \in \mathcal{Q}$	uncertain parameters
$q^0 \in \mathbb{R}^w$	nominal uncertain parameter values
$\zeta \in \mathbb{R}$	objective function value

Functions

g	inequality constraints
h	equality constraints
f_1	first-stage costs
f_2	second-stage costs
v	decision rule function
$\mathbb{E}(\cdot)$	expectation of
$\sigma(\cdot)$	standard deviation of

NONLINEAR ROBUST OPTIMIZATION CASE STUDIES

6.1 INTRODUCTION

We present a set of three case studies to illustrate the performance of the GRCS algorithm in identifying robust feasible solutions to nonlinear process optimization problems. In Case Study I, we focus on a reactor-separator process. In Case Study II, we study a reactor-heater process. Finally, in Case Study III, we consider a highly complex, high-fidelity flowsheet model for amine solvent-based CO₂ separation from flue gas. All models were implemented using Pyomo[52, 53] and tools from the Institute for the Design of Advanced Energy Systems (IDAES) integrated framework[85, 93]. We used IPOPT[125] paired with the HSL MA27[33] linear solver as the local optimization solver. Given the built-in flexibility of our implementation, the global solver Antigone[86] was utilized to confirm robust feasibility in Case Study I, while the global solver BARON[112] was employed for Case Studies II and III. Each case study was solved on a desktop computer featuring four Intel i7-6700 3.4 GHz processors and 16 GB RAM. All solvers were configured with an optimality tolerance of 1×10^{-6} , while the tolerance for identifying normalized violations of performance constraints via solving separation problems was set to 1×10^{-4} . In Case Studies I and II, we showcase the capabilities of all three decision rule types, as the overall model sizes were amenable to the addition of more complex and nonlinear decision rule functions. In the final case study, we only consider the static approximation and affine decision rules policies.

In all cases, we use the static approximation solutions to MP_1 and $SP_{1,i} \forall i \in \mathcal{I}$ to initialize the MP_1 and $SP_{1,i} \forall i \in \mathcal{I}$ under the affine and quadratic decision rules cases. This is because the static approximation MP_1 and $SP_{1,i} \forall i \in \mathcal{I}$ solutions are feasible and best-known solutions in the affine and quadratic cases.

6.2 CASE STUDY I: REACTOR-SEPARATOR

The flowsheet illustrating the reactor-separator system is shown in Figure 6.1a, and has been previously studied in Grossmann and Sargent [46], Rooney and Biegler [101], and Yuan, Li, and Huang [143]. In this design problem, we are modeling the isothermal, liquid-phase conversion of reactant A into a desired product C via a set of four first-order chemical reactions, which are outlined in Figure 6.1b. Products D and E are undesirable side-products in the reaction network.

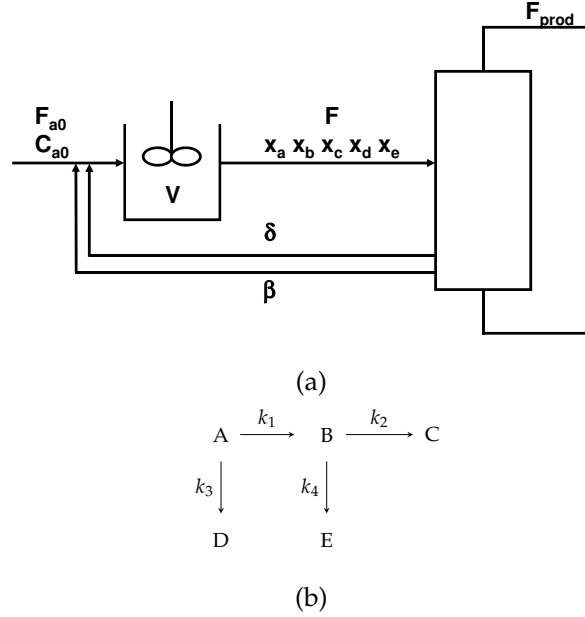


Figure 6.1: Flowsheet (a) and reaction mechanism (b) representing the reactor-separator system considered in Section 6.2, as adapted from Grossmann and Sargent [46].

Using this reaction network information and the process flowsheet, we can cast the deterministic model shown in Section 6.7.1.1 of the Appendix. The objective function in Equation 6.1a corresponds to the equivalent annualized cost using a capital recovery factor (CRF) of 0.09, which corresponds to an operating lifetime of 25 years under an 8% annual interest rate [87]. The inequality constraint of Equation 6.1h states that feasible designs must yield at least $40 \frac{\text{mol}}{\text{hr}}$ of product C . An additional inequality constraint, Equation 6.1i, limits the amount of byproduct D recycled. Both of these inequalities are considered performance constraints that are subject to separation steps in the GRCS algorithm. All other constraints are either state equations or second-stage variable bounds, the latter of which are also subject to separation.

In summary, the deterministic reactor-separator model consists of nine decision variables, six equality constraints, two inequality constraints (excluding variable bounds), and twelve parameters of which four will be considered in this work as being uncertain. More specifically, the first-stage decision variable is the volume of the reactor, V , while the second-stage control variables are the recycle ratios δ and β , which represent the fractions of A and B , and D and E recycled, respectively. The mole fractions (x_a, \dots, x_e) as well as the flowrate F constitute state variables. For this study, the known parameters are the inlet concentration of the reactant $C_{a0} = 10 \frac{\text{mol}}{\text{m}^3}$ and the inlet flowrate of the reactant $F_{a0} = 100 \frac{\text{mol}}{\text{hr}}$. The uncertain data are the reaction rate constants k_i , $i = \{1, 2, 3, 4\}$, which are correlated in a four-dimensional ellipsoidal uncertainty set. The original data for describing this set (mean, standard deviation) can be found in Rooney and Biegler [101] and are also reported in Section 6.7.1.3 of the Appendix, for convenience. Thus, for the application of the GRCS algorithm, $x = (V)$, $z = (\delta, \beta)$, all other variables are state variables y , while $q = (k_1, k_2, k_3, k_4)$.

6.2.1 Case Study I Results

Results for the optimal first-stage variables and costs for the deterministic and robust feasible cases are shown in Table 6.1. The reactor volume, V , identified in the robust solutions is larger than that of the deterministic solution, in order to insure against the postulated uncertainty. This is likely due to the fact that feasible robust solutions must permit sufficient production of product C and sufficient recycle of byproduct D across the range of values for k_i , $i = \{1, 2, 3, 4\}$, which in turn requires increased reactor sizes than the deterministic case. However, it is important to note that, as recourse flexibility increases via the use of more involved decision rules, the reactor volume as well as the first-stage cost decreases.

	Deterministic	Static Approx.	Affine DR	Quadratic DR
$V \text{ (m}^3\text{)}$	103.18	105.50	103.56	103.20
$f_1(x^*) \text{ (\$/yr)}$	9,973.32	10,426.31	10,047.29	9,978.13

Table 6.1: Optimal values of first-stage variables and costs for the reactor-separator model.

The optimal values for second-stage variables and costs for the deterministic and robust feasible cases are shown in Table 6.2. When we compare the values of the control variables δ and β in the deterministic case against the robust feasible values corresponding to the nominal realization of the uncertainty, we see that the values for the recycle ratio δ increases in the latter case. This leads to an increase in nominal second-stage costs

when compared to the deterministic solution. However, if we consider the expected second-stage variable values and costs in Table 6.3, they more closely match the deterministic case solution. In particular, full-flexibility in second-stage control has a significant economic benefit over more restrictive decision rule policies. Interestingly, if we pay attention to the expected robust feasible objective values, representing the total expected cost $\mathbb{E}[\zeta] = f_1(x^*) + \mathbb{E}[f_2]$, the affine and quadratic decision rules solutions result in overall lower costs than the static approximation policy, even when considering the associated variance.

	Deterministic	Static Approx.	Affine DR	Quadratic DR
$\delta \left(\frac{\text{mol A} + \text{mol B}}{\text{total mol}} \right)$	0.46	0.68	0.66	0.64
$\beta \left(\frac{\text{mol D} + \text{mol E}}{\text{total mol}} \right)$	0.016	0.017	0.017	0.018
$f_2(z^*, y^*; q^0)$ (\$/yr)	8,843.27	13,250.84	12,881.24	12,614.63

Table 6.2: Optimal values of second-stage variables and costs, under the nominal realization of uncertainty, for the reactor-separator model.

	Static Approx.	Affine DR	Quadratic DR
$\mathbb{E}[\delta] \pm \sigma[\delta] \left(\frac{\text{mol A} + \text{mol B}}{\text{total mol}} \right)$	0.45 ± 0.07	0.46 ± 0.07	0.46 ± 0.07
$\mathbb{E}[\beta] \pm \sigma[\beta] \left(\frac{\text{mol D} + \text{mol E}}{\text{total mol}} \right)$	0.016 ± 0.0004	0.016 ± 0.0004	0.016 ± 0.0004
$\mathbb{E}[f_2] \pm \sigma[f_2]$ (\$/yr)	$8,395.78 \pm 1,240.87$	$8,758.91 \pm 1,245.64$	$8,829.64 \pm 1,246.56$
$\mathbb{E}[\zeta] \pm \sigma[\zeta]$ (\$/yr)	$18,822.10 \pm 1,240.87$	$18,806.20 \pm 1,245.64$	$18,805.77 \pm 1,246.56$

Table 6.3: Expected values and standard deviations of second-stage variables and costs for the reactor-separator model.

A noteworthy observation in this case study is the fact that $\mathbb{E}[\zeta]_{SA} > \mathbb{E}[\zeta]_{ADR} > \mathbb{E}[\zeta]_{QDR}$. This means that, for this system, there is an economic benefit in utilizing more flexible, nonlinear decision rules over the traditional affine or inflexible cases. We also note that, when compared to the deterministic optimal equivalent annual cost, $\zeta_{det} = 18,816.59$ (\$/yr), the robust feasible designs incur in expectation a comparable cost. Indeed, $\mathbb{E}[\zeta]/\zeta_{det} \approx 1$ and $\sigma[f_2]/\mathbb{E}[\zeta] \approx 0.066$ for all three recourse policies, indicating that all robust designs are likely to yield actual costs that do not significantly deviate from those of the deterministic design.

The total number of iterations of the GRCS and the total CPU time for completing the GRCS algorithm are shown in Table 6.4. This table also reports the fraction of total time spent on master and separation subproblems via calls to subordinate solvers, as well as the residual fraction spent on code overhead, which primarily includes time for Pyomo model manipulations. Clearly, for this case study, which features a high-dimensional uncertainty set and multiple inequality constraints, the number of itera-

tions increases with affine and quadratic decision rules. This hints to the likelihood of decreased tractability when more flexible decision rules are used in conjunction with complex models, though the relatively small size of the underlying model in this case study caused no computational issues. Whereas the total times are generally small, a significant portion is spent in solving separation problems. This is due to the fact that there are multiple such problems to be solved in each iteration, as well as the fact that a number of calls to global solvers are required to confirm robustness.

	Static Approx.	Affine DR	Quadratic DR
# of GRCS iterations	4	9	9
Total CPU time (s)	10.3	11.4	33.2
% spent on master problems	0.1	0.2	0.1
% spent on separation	81.8	75.9	91.2
% overhead time	18.1	23.9	8.7

Table 6.4: Total number of iterations and CPU time spent within the GRCS algorithm when addressing the reactor-separator model. The total time includes the time to execute the algorithm and subordinate solver calls. The percentage of time spent on master and separation problems only includes the total execution time for the respective subordinate solvers.

Additionally, we plot in Figure 6.2 the GRCS progression towards robust feasibility at each iteration of the GRCS for the static approximation recourse policy. The trajectories of the algorithm under affine and quadratic decision rule policies are deferred to the Appendix (see Figures 6.7 and 6.8). Each plot depicts the performance of designs at a given iteration k , as determined from the master problem MP_k . The plots are generated by evaluating each optimal design via the same set \mathcal{S} of 200 uniformly-sampled realizations $q_s \in \mathcal{Q}$, $s \in \mathcal{S}$. Feasibility of each design at a realization q_s is determined via one of the two constraints in Equations 6.1h and 6.1i. We remark that, although the constraints for bounds on control variables δ and β were included in the set of constraints subject to separation, they never led to a violation across the range of values within the uncertainty set, and are hence not referenced in this analysis. The constraint for which feasibility is shown in a given plot is determined by which constraint yielded a maximum violation in the separation problem $SP_{k,i}$.

For each point in the plots, the solution is either feasible and represented as a blue dot, or is infeasible and represented as a red dot. All points shown as a green triangle are the realizations at which the current design is explic-

itly robust against.ⁱ All points represented as yellow crosses are violating points identified in solving $SP_{k,i}$ at the current iteration k . Constraints under these realizations will thus be appended in subsequent iterations of the master problem. We also note that, because the uncertainty set considered in the reactor-separator model is four-dimensional, we merely project to the two dimensions with the largest variance, namely k_3 and k_4 , and the black dotted lines represent the projection of the boundary of the uncertainty set in these two dimensions.

The plots reveal two regions in which each of the inequality constraints are active and lead to constraint violations. The main region of infeasibility for constraint 6.1h is in the upper-right end of the uncertainty set, while the region where violations for constraint 6.1i occur in the lower-left end. The trajectory towards robustness against each performance constraint is easily seen over progressive iterations of the algorithm for each recourse policy case, as the region of red infeasible sample points becomes blue. The amount of infeasibility that persists in each iteration is quantified in Table 6.13 of the Appendix.

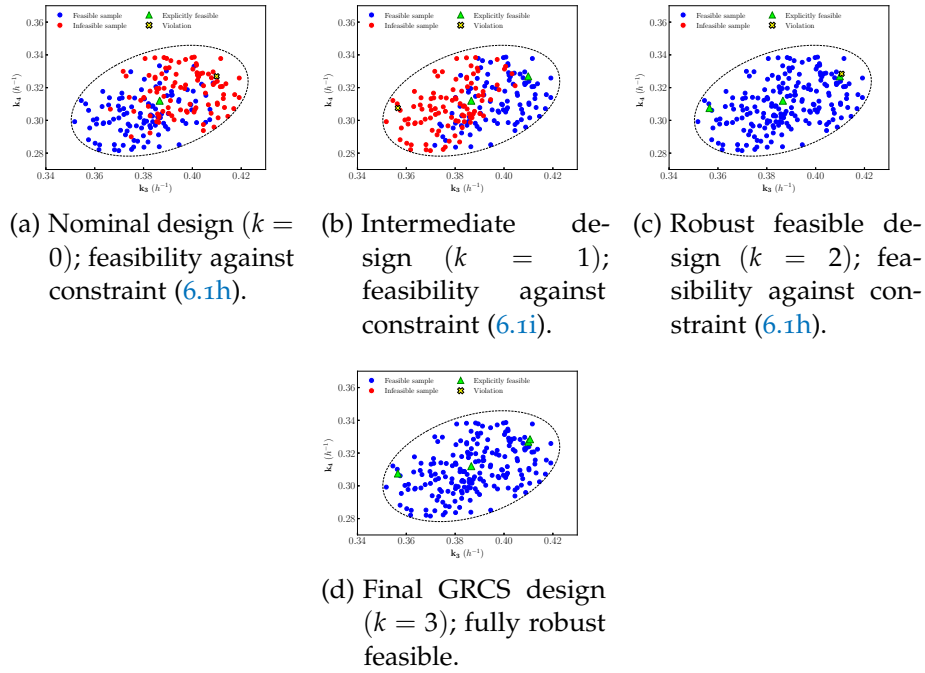


Figure 6.2: Evolution during the GRCS algorithm of the robust feasibility of the reactor-separator designs using the static approximation policy.

ⁱ For example, the nominal realization q^0 is used as early as the first master problem to initialize the GRCS algorithm, and hence every master problem solution is explicitly robust against the nominal point.

6.3 CASE STUDY II: REACTOR-HEATER

In this case study, we consider the flowsheet shown in Figure 6.3, which represents a reactor-heater system previously studied in Halemane and Grossmann [50], Varvarezos, Biegler, and Grossmann [120], Rooney and Biegler [102], and Yuan, Li, and Huang [143]. This system consists of a reactor and heat exchanger with a single first-order, exothermic reaction to convert reactant A to product B . The task is to identify a design that achieves the target of 90% conversion of reactant A , while minimizing equivalent annual cost.

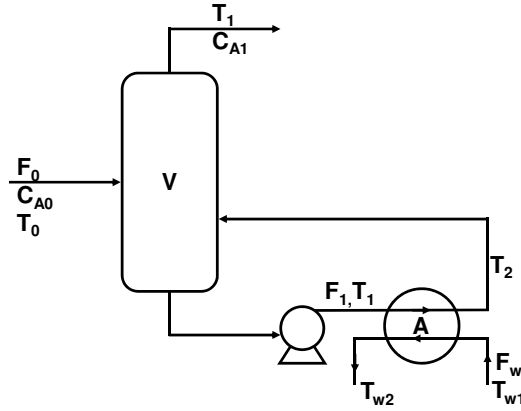


Figure 6.3: Flowsheet representing the reactor-heater system considered in Section 6.3, as adapted from Halemane and Grossmann [50]

The deterministic reactor-heater model consists of ten decision variables, five equality constraints, four inequality constraints (excluding variable bounds), and ten parameters of which two will be considered uncertain. The complete NLP model for the reactor-heater flowsheet, as well as a table of pertinent data, are shown in Sections 6.7.2.1 and 6.7.2.2 of the Appendix, respectively. The objective function in Equation 6.2a seeks to minimize the equivalent annual cost, including annualized capital cost and yearly operating cost terms taken from Varvarezos, Biegler, and Grossmann [120]. The set of state equations \mathcal{J} includes Equations 6.2b–6.2f, while the set of performance constraints \mathcal{I} includes Equations 6.2g–6.2n as well as the bounds on second-stage variables in Equations 6.2o and 6.2p.

In this optimization model, the size of the reactor V and the area of the heat exchanger A are the first-stage decision variables, while the two second-stage variables are the flowrates F_1 and F_w dictating utility usage. The remaining variables, x_A , T_1 , T_2 , T_{w1} , T_{w2} , and ΔT_{ln} ⁱⁱ constitute state variables. The uncertainty considered in this example lies in the Arrhenius

ii Note that we made use of the Underwood formula [117] to approximate ΔT_{ln} , which was found to be helpful in avoiding numerical issues with the NLP solvers.

rate constant for the reaction taking place in the reactor, k_0 , and overall heat transfer coefficient in the heat exchanger, U . Thus, $x = (V, A)$, $z = (F_1, F_w)$, all other variables are state variables y , and $q = (k_0, U)$. For this study, we postulate that the two aforementioned uncertain parameters are independent to each other, and therefore utilize a simple two-dimensional box uncertainty set. The data for the nominal values and uncertainty deviations of k_0 and U can be found in Section 6.7.2.3 of the Appendix.

6.3.1 Case Study II Results

Results for the optimal first-stage variables and costs for the deterministic and robust feasible cases are shown in Table 6.5. In this case study, the GRCS algorithm returned the same robust solution for the affine and quadratic decision rules. This outcome is possible due to the fact that a feasible solution with affine decision rules is also a feasible solution under quadratic decision rules, and it just so happens that, for this particular model and uncertainty set, greater flexibility via quadratic decision rules does not improve upon the robust feasible solution. This observation alludes to the possibility that the extra computational burden to consider more involved recourse policies might not always yield a payoff. To that end, the modeler has a critical role to play in judiciously selecting the form of decision rule to be employed in the context of each particular model. In Section 6.5, we provide additional remarks on the selection of recourse policy.

When comparing the deterministic first-stage variables to the robust solutions, we see that there is an increase in reactor volume and heat exchanger area for robust solutions. The robust designs must satisfy the constraint regarding production of product A for all uncertain parameter values within the uncertainty set, and this is accomplished via larger capacity facilitated by larger equipment.

	Deterministic	Static Approx.	Affine DR	Quadratic DR
$V \text{ (m}^3\text{)}$	4.43	4.98	4.94	4.94
$A \text{ (m}^2\text{)}$	9.70	9.97	9.92	9.92
$f_1(x^*) \text{ (\$/yr)}$	5,374.66	5,596.62	5,575.26	5,575.26

Table 6.5: Optimal values of first-stage variables and costs for the reactor-heater model.

The optimal values for second-stage variables and costs for the deterministic and robust feasible cases are shown in Table 6.6, where we note that these do not change substantially under the case of the nominal realization. When we compare the values of the control variables, F_1 and F_w , in the deterministic case against their robust feasible values, we notice

that the values of these flowrates increase, which leads to an increase in second-stage costs. This can be explained by the fact the reaction producing product A is exothermic, requiring increased circulation through the cooler in the robust designs. In regards to expected second-stage variable values and costs shown in Table 6.7, we observe that these values are actually greater than the nominal values in the previous table. This may be an indicator of the optimizer taking advantage of the nominal second-stage cost, $f_2(x, z, y; q^0)$, in the objective. In the case of the nominal uncertain parameter realization, q^0 , the robust feasible design requires slightly lower flowrates, and correspondingly a lower second-stage cost. However, in expectation, these flowrates will be higher, as reported in Table 6.7. It is also interesting to note that decision rules seem to help alleviate some of the conservativeness of the static approximation solution, as they strictly improve in terms of equivalent annual cost, both nominal and expected.

	Deterministic	Static Approx.	Affine DR	Quadratic DR
F_1 (kmol/hr)	94.19	95.77	95.69	95.69
F_w (kmol/hr)	1,754.75	1,782.49	1,784.21	1,784.21
$f_2(z^*, y^*; q^0)$ (\$/yr)	4,107.47	4,175.15	4,177.78	4,177.78

Table 6.6: Optimal values of second-stage variables and costs, under the nominal realization of uncertainty, for the reactor-heater model.

	Static Approx.	Affine DR	Quadratic DR
$\mathbb{E}[F_1] \pm \sigma[F_1]$ (kmol/hr)	96.98 \pm 9.82	96.80 \pm 9.63	96.80 \pm 9.63
$\mathbb{E}[F_w] \pm \sigma[F_w]$ (kmol/hr)	1,809.47 \pm 97.89	1,798.57 \pm 94.64	1,798.57 \pm 94.64
$\mathbb{E}[f_2] \pm \sigma[f_2]$ (\$/yr)	4,236.47 \pm 264.11	4,214.15 \pm 256.51	4,214.15 \pm 256.51
$\mathbb{E}[\zeta] \pm \sigma[\zeta]$ (\$/yr)	9,833.10 \pm 264.11	9,789.41 \pm 256.51	9,789.41 \pm 256.51

Table 6.7: Expected values and standard deviations of second-stage variables and costs for the reactor-heater model.

The total number of iterations of the GRCS algorithm, its total CPU time and the fraction of total time spent on various portions of the algorithm are shown in Table 6.8. The number of iterations and CPU times are relatively small, due to the small scale and good tractability of the deterministic optimization problem. As in the previous example, the majority of the GRCS algorithm's time is spent solving separation problems. The algorithmic overhead tasks also consumed a significant fraction of the total CPU time, noting though that the very small total CPU times do not encourage general insights.

In Figure 6.4, we show the progression of feasibility over each iteration of the GRCS for the case of the static approximation recourse policy. Similar plots for the cases of affine and quadratic decision rule policies are

	Static Approx.	Affine DR	Quadratic DR
# of GRCS iterations	3	3	3
Total CPU time (s)	1.3	2.9	4.0
% spent on master problems	0.5	0.3	0.4
% spent on separation	49.0	66.5	66.0
% overhead time	50.5	33.2	33.6

Table 6.8: Total number of iterations and CPU time spent within the GRCS algorithm when addressing the reactor-heater model. The total time includes the time to execute the algorithm and subordinate solver calls. The percentage of time spent on master and separation problems only includes the total execution time for the respective subordinate solvers.

provided in the Appendix (see 6.9 and 6.10). In these plots, the designs from each iteration of the master problem are evaluated across a set \mathcal{S} of 200 uniformly-sampled realizations $q_s \in \mathcal{Q}$, $s \in \mathcal{S}$. The feasibility of a design under a realization q_s is defined against the performance constraint in Equation 5.2n, as this was the only inequality constraint in the formulation that led to violations in the separation problems solved. In other words, for points shown in blue, the design leads to $x_A \geq 0.9$, while for points shown in red, $x_A < 0.9$. Figure 6.4a, which is a plot depicting the feasibility of the nominal deterministic design under various uncertain parameter realizations, clearly shows a horizontal division in the uncertainty set between a region of feasibility and a region of infeasibility. More specifically, while the nominal realization, $q^0 = (U^0, k_0^0)$, is included in the region of feasibility, most points with $k_0 < k_0^0$ render the design infeasible. Interestingly, the first violating parameter realization identified in separation, q^1 , is a non-vertex point on the border of the otherwise polyhedral uncertainty set. The amount of infeasibility that persists in each iteration is quantified in Table 6.14 of the Appendix.

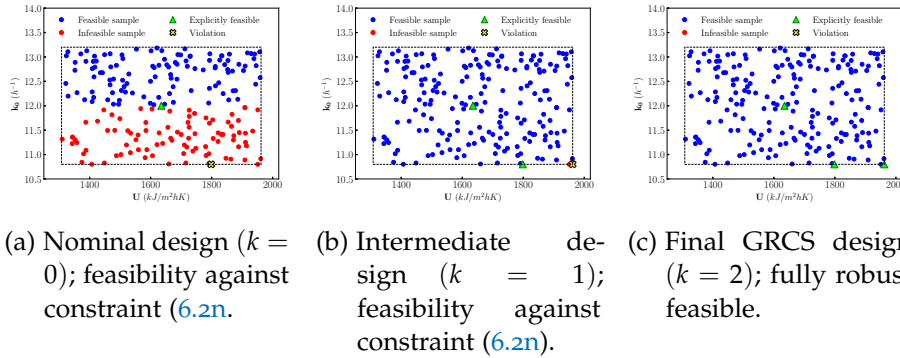


Figure 6.4: Evolution during the GRCS algorithm of the robust feasibility of the reactor-heater designs using the static approximation policy.

6.4 CASE STUDY III: MEA-SOLVENT CO₂ SEPARATION FLOWSHEET

The final example studied here is a complete equation-oriented model for post-combustion CO₂ capture using a monoethanolamine (MEA) solvent. The complete flowsheet for the process is shown in Figure 6.5. The key units in this process include the absorber and regenerator (stripper) columns, the cross heat exchanger, and the reboiler and condenser. The electrolyte non-random two-liquid (e-NRTL) model[55] was used to represent the vapor-liquid equilibrium, while an enhancement factor[131] is used to characterize the effect of liquid phase reaction on the mass transfer rate between the liquid and gas phases. The latter was based on a two-film model, wherein mass transfer occurs via molecular diffusion through a stagnant film of a given thickness and the bulk phase is well mixed[149]. In addition, our process model includes rigorous submodels to describe gas-liquid reactions, surface tension, diffusivity, and heat transfer. The mass and energy balances in the column models are differential equations, as the concentrations of vapor and liquid components as well as the temperature varies along the length of the columns. Some representative model equations are shown in Section 6.7.3.3 of the Appendix and have first been presented in Chinen et al. [24] for a solvent-based carbon capture system. Notably, the simulation model of the flowsheet, featuring zero degrees of freedom and 30 finite elements to discretize the differential equations, is a system of 4,334 variables and 4,327 constraints.

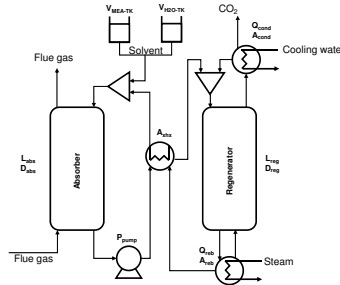


Figure 6.5: Flowsheet representing the MEA-based CO₂ capture flowsheet considered in Section 6.4, as adapted from Mores et al. [87].

In our study, we seek to design a process for treating 1,000 mol/s of flue gas, with the main performance constraint calling for the process to achieve 85% CO₂ capture. Therefore, we augmented the simulation model to include degrees of freedom for investments to install capacity for the various utilities and an overall economic objective to minimize the equivalent annual cost (EAC). We adapted the straightforward algebraic formulation of Mores et al. [87] and utilize simple capital cost correlations from Towler and Sinnott [116] and Seider, Seader, and Lewin [105] to

demonstrate relative impact on EAC of our robust optimization approach. Note that we do not account for uncertainty in costs and that the use of more rigorous costing methods are outside the scope of this work. The full set of equations defining the EAC function can be found in Section 6.7.3.2 of the Appendix.

After the above augmentations, the deterministic optimization model we used consists of 4,342 decision variables, 4,327 equality constraints, and 13 inequality constraints (excluding variable bounds). The main first-stage design variables correspond to the sizing of major equipment, namely the diameters of the columns, D_{abs} and D_{reb} , the heights of the columns, L_{abs} and L_{reg} , and the total surface area of the cross heat exchanger, A_{xhx} . There are also several first-stage variables introduced in the model to properly capture the investment costs for the installed capacities of necessary utilities. These include the power to pump CO₂-rich solvent from the absorber bottom to the inlet of the regenerator, P_{pump} , volumes of storage tanks for the hold-up of H₂O and MEA solvent to supply to the make-up mixer, V_{MEA-TK} and V_{H_2O-TK} , as well as the effective surface areas of the reboiler and condenser, A_{reb} and A_{con} , respectively. In summary, $x=(D_{abs}, D_{reg}, L_{abs}, L_{reg}, A_{xhx}, P_{pump}, V_{MEA-TK}, V_{H_2O-TK}, A_{reb}, A_{con})$. The second-stage control variables are the heat duties supplied to the reboiler and condenser, thus $z = (Q_{reb}, Q_{cond})$. All remaining variables are considered to be state variables y .

Potential sources of uncertainty have been studied in the literature[24, 88], while the optimization under uncertainty of the absorber column unit for MEA-based post-combustion carbon capture has been studied also in [20]. Here, we expand upon previous works by studying the entire process flowsheet, presenting solutions obtained via our novel algorithm. More specifically, we postulate an ellipsoidal uncertainty set representing the 95% confidence interval around two parameters, $q = (b_1, b_2)$, used to calculate equilibrium constants $K_{eq,r}$, $r \in \{1, 2\}$ via Equations 6.3m in Section 6.7.3.3 of the Appendix. Here, the index r refers to two solvent-phase reactions considered in the kinetic model[89]. We stress that parameters b_1 and b_2 participate non-linearly in the set of equations defining the equilibrium constants, while the latter are further non-linearly related to the overall mass transfer coefficient between the vapor and liquid phases. As we shall show later, this leads to interesting regions of feasibility of the deterministic design within the uncertainty set. Details regarding the uncertainty set used, including the mean and covariance matrix values, are provided in Section 6.7.3.4 of the Appendix.

There are several inequality constraints in the model representing requirements on performance. These include a constraint on the fraction of CO₂ captured, i.e., $\eta_{CO_2} \geq 0.85$, where $\eta_{CO_2} = \frac{y_{CO_2}^{in} - y_{CO_2}^{out}}{y_{CO_2}^{in}}$ and y_{CO_2} is the

gas phase mole fraction of CO₂. In addition, explicit constraints on the superficial vapor velocities at the bottom of the absorber and the top of the regenerator are imposed, in order to prevent flooding. These constraints can be found in Section 6.7.3.3 of the Appendix (Equations 6.3n–6.3r).

Additional inequality constraints of interest are the bounds on the control variables, namely the reboiler and condenser duties. The latter include trivial “zero” bounds to limit their sign (Equations 6.3u and 6.3v) as well as induced bounds due to the availability of steam (Equation 6.3w) and cooling water (Equation 6.3x), respectively. In the case of affine decision rules, these bounds must be included in the set of performance constraints to be separated against. All other equations in the model represent state equations, in which the uncertain parameters participate nonlinearly and implicitly. Additionally, we note that the flue gas flowrate at the bottom of the absorber is fixed to $1,000 \frac{\text{mol}}{\text{s}}$, while the solvent flowrate at the top of the absorber is fixed to $3.64 \frac{\text{kmol}}{\text{s}}$. The solvent’s temperature at this inlet is calculated as a state variable in the model. The solvent make-up is set to be 27.5% wt. MEA at the outlet of the make-up mixer.

6.4.1 Case Study III Results

For this case study, we only consider the static approximation and affine decision rules as our two recourse policies. Results under quadratic recourse are not presented due to the fact that the subordinate nonlinear optimization solver failed to converge while attempting to solve a subproblem generated at an intermediate iteration of the GRCS algorithm. This is an example of how the overall performance of the GRCS algorithm depends on the ability of the employed subordinate solvers to identify optimal solutions to subproblems at each and every iteration and in reasonable computational times.

The optimal values for first-stage variables and costs for the deterministic and robust feasible solutions are shown in Table 6.9. Our deterministic design is consistent with general trends established in optimal designs previously presented in the literature[35, 87], such as the fact that the absorber height is greater than the regenerator column, as well as a drum-like regenerator column wherein $L/D \approx 1$. One possible motivation for this is to drive down capital and utility costs associated with pumping the solvent to the top of a higher regenerator column, at the expense of increased reboiler duty. This may also be motivated by the presence of explicit constraints to limit the superficial vapor velocity within a fraction of the flooding velocity at the regenerator bottom, since an increase in diameter corresponds to a decrease in velocity for a given volumetric flowrate.

We begin by comparing the deterministic solution to the robust solutions. First, we notice that the robust solutions (static approximation and affine decision rules) feature decreased heights of both columns as well as an increased diameter of the regenerator column, as compared to the deterministic solution. As a result of the reboiler column height decrease in the robust solutions, there is also a corresponding decrease in the required pump power. Additionally, there is an increase in heat transfer surface area in the reboiler and condenser. This can be attributed to the increase in reboiler and condenser heat duties, contributing to achieving robust feasibility. This also relates to the decrease in the robust values of cross heat exchanger surface area, since more heat transfer load between streams can be taken on by the larger condenser and reboiler.

Next, we consider notable differences between the robust solutions determined via the static approximation and affine DR policy. First, we see that the optimal surface areas for the heat exchanger, reboiler and condenser are smaller in magnitude in the case of an affine decision rule policy, when compared to the static approximation. Because the affine decision rules allow for more flexible heat transfer policies in the second-stage, there is less need to invest in larger equipment, such as heat transfer surface areas, in the first-stage. This decrease in heat transfer surface areas in the affine case leads to a corresponding increase in the regenerator column height, which is necessary for achieving sufficient CO₂ regeneration at the implied lower reboiler and condenser duties. In spite of the above differences, the first-stage capital costs appear to be very similar in all three solutions.

	Deterministic	Static Approx.	Affine DR
L_{abs} (m)	7.57	6.00	6.93
D_{abs} (m)	4.95	4.96	4.96
L_{reg} (m)	4.00	3.00	3.52
D_{reg} (m)	3.44	4.04	4.00
A_{xhx} (m ²)	4,734	3,928	3,764
P_{pump} (kW)	4.44	3.67	4.28
V_{MEA-TK} (m ³)	2.25	3.46	3.48
V_{H_2O-TK} (m ³)	11,541	13,300	13,486
A_{reb} (m ²)	179.4	813.4	797.4
A_{con} (m ²)	191.8	2,116.8	2,034.0
$f_1(x^*)$ (MM\$/yr)	2.06	2.07	2.09

Table 6.9: Optimal values of first-stage variables and costs for the CO₂ capture flowsheet model.

Table 6.10 presents information related to the second-stage variables and costs, under the scenario of nominal uncertainty realization. In this

case, the robust reboiler heat duty, Q_{reb} , is much higher than the optimal value in the deterministic case, as is also the flowrate of solvent supplied to the regenerator. This is illustrating a trade-off wherein the shorter, more drum-like regenerator columns require higher reboiler duties to achieve the lower lean loading values. The decrease in column height also allows for larger flowrates of solvent due to a decreased pumping requirement. As expected, robustness comes at a noticeable increase in total cost, which is necessary for the design to possess enough flexibility to remain operational (feasible) under a wide range of scenarios (uncertainty set). It is important to note, however, that the use of affine decision rules leads to a robust feasible solution with a slightly lower nominal EAC, when compared to the static approximation policy, which demonstrates reduction of the two-stage adaptivity gap. In addition to the optimal deterministic second-stage variables and costs, Table 6.10 also shows the optimal values for several state variables of interest. These include the CO₂ capture, denoted as η_{CO_2} , the lean loading calculated as $\alpha = \frac{x_{CO_2}}{x_{MEA}}$, where x_{CO_2} and x_{MEA} are liquid phase mole fractions, as well as several key flowrates related to the pumping utility ($F_{reg,liq}^{in}$) and the solvent make-up supplied by the mixer at the top of the absorber ($F_{mix,MEA}^{in}$ and F_{mix,H_2O}^{in}). We see that the nominal capture for the robust solutions, in both the affine decision rule and static approximation cases, is larger than the deterministic case. To achieve feasibility against all uncertain parameter realizations, there must be a degree of “over-design” when considered in light of a subset of realizations in the uncertainty set. This means that, for some realizations, and in particular the nominal realization, the robust design will exceed the requirements of the performance constraints. Additionally, we see that lean loading values are lower in the robust solutions, when compared to the deterministic case, due to the increase in the robust reboiler duties.

The expected values and standard deviations for the second-stage costs and variables are shown in Table 6.11. We note that the expected second-stage costs are significantly lower than their values when the nominal realization prevails. This can be attributed to the fact that the expected values are determined without a particular decision rule policy in place, instead being calculated in light of unrestricted recourse potential. By simply allowing for more flexibility in second-stage recourse, there is a broader range of operating points available, which can decrease the second-stage costs. Unlike in the previous case studies, the affine decision rules solution does not result in lower overall expected costs when compared to the static approximation solution, albeit these costs show less variation. Note that the trade-offs between first-stage investment costs and second-stage operating costs are more complex in this system, and since the economic objective function only considers the nominal scenario, we do

not get a guarantee that the designs under the more flexible recourse policy will be lower in expectation.

In terms of control variables, Table 6.11 reveals that the need for reboiler duty is generally more stable than the need for condenser duty. Indeed, as different scenarios were sampled, the latter ranged quite a bit more on a relative scale, leading to a more skewed distribution with an elongated tail. Overall, the range of values that these duties would have to attain under the various sampled scenarios is larger in the design stemming from the affine decision rules policy, which can be explained by the fact that this design was explicitly chosen to perform under a wider range of control variable values.

	Deterministic	Static Approx.	Affine DR
Q_{reb} (MW)	18.14	41.10	39.05
Q_{con} (MW)	-4.54	-25.18	-22.89
η_{CO_2} $\left(\frac{mol CO_2}{total mol}\right)$	0.85	0.93	0.96
α $\left(\frac{mol CO_2}{mol MEA}\right)$	0.22	0.17	0.17
$F_{reg,liq}^{in}$ (kg/s)	84.84	92.10	91.22
$F_{mix,MEA}^{in}$ (kg/s)	0.03	0.04	0.04
F_{mix,H_2O}^{in} (kg/s)	4.44	4.85	5.00
$f_2(z^*, y^*; q^0)$ (MM\$/yr)	5.19	8.83	8.67
ζ^* (MM\$/yr)	7.25	10.90	10.76

Table 6.10: Optimal values of second-stage control and other key variables, as well as total and second-stage costs, evaluated under the nominal realization of uncertainty, for the CO₂ capture flowsheet model.

	Static Approx.	Affine DR
$\mathbb{E}[Q_{reb}] \pm \sigma[Q_{reb}]$ (MW)	18.57 ± 0.55	19.27 ± 4.42
$\mathbb{E}[Q_{con}] \pm \sigma[Q_{con}]$ (MW)	-0.935 ± 1.36	-1.86 ± 4.87
$\mathbb{E}[\eta_{CO_2}] \pm \sigma[\eta_{CO_2}]$	0.85 ± 0.00	0.86 ± 0.03
$\mathbb{E}[\alpha] \pm \sigma[\alpha]$ $\left(\frac{mol CO_2}{mol MEA}\right)$	0.24 ± 0.03	0.24 ± 0.03
$\mathbb{E}[F_{reg,liq}^{in}] \pm \sigma[F_{reg,liq}^{in}]$ (kg/s)	83.63 ± 0.51	83.87 ± 1.52
$\mathbb{E}[F_{mix,MEA}^{in}] \pm \sigma[F_{mix,MEA}^{in}]$ (kg/s)	0.03 ± 0.00	0.03 ± 0.00
$\mathbb{E}[F_{mix,H_2O}^{in}] \pm \sigma[F_{mix,H_2O}^{in}]$ (kg/s)	5.89 ± 0.34	5.79 ± 0.27
$\mathbb{E}[f_2] \pm \sigma[f_2]$ (MM\$/yr)	5.51 ± 1.38	5.63 ± 0.78

Table 6.11: Expected values and standard deviations of second-stage control and other key second-stage variables, and second-stage costs, for the CO₂ capture flowsheet model.

The iterations count and algorithm execution times for the CO₂ capture flowsheet study are shown in Table 6.12. When compared to the previous

case studies, the total CPU times are much higher due to the larger model size and complexity. For reference, the deterministic version of the CO₂ capture flowsheet model already requires 31 seconds of CPU time to optimize locally. Table 6.12 also reveals that, despite the number of iterations being the same in both cases, the total CPU time for the affine decision rules is roughly 50% higher than the static approximation case, alluding to the fact that the complexity of the GRCS is directly related to the complexity of the decision rule relationship used. Additionally, we see the same trend as in previous case studies wherein the percentage of time spent solving separation problems is significantly larger than that spent solving the master problems. We also see that there is significant overhead time spent outside of subordinate solver optimization calls. In particular, due to the larger models involved in this case study, there are significantly more constraints and variables that need to be copied at each iteration as the master problem formulation evolves, leading to increased times spent towards Pyomo model building.

	Static Approx.	Affine DR
# of GRCS iterations	5	5
Total CPU time (s)	1,030.0	1,543.4
% spent on master problems	19.4	16.2
% spent on separation	48.6	51.2
% overhead time	32.0	32.6

Table 6.12: Total number of iterations and CPU time spent within the GRCS algorithm when addressing the CO₂ capture flowsheet model. The total time includes the time to execute the algorithm and subordinate solver calls. The percentage of time spent on master and separation problems only includes the total execution time for the respective subordinate solvers.

Figure 6.6 presents the progression of feasibility over each iteration of the GRCS under the static approximation policy. The progression under affine decision rules can be found in the Appendix (Figure 6.11). As before, designs are evaluated in light of 200 uniformly-sampled realizations $q_s \in \mathcal{Q}$, $s \in \mathcal{S}$. Here, feasibility of the designs primarily refers to the CO₂ capture constraint and the upper bound constraining the superficial vapor velocity in the absorber, as these were the only inequality constraints to lead to violations in the separation step. The nominal design performance shown in Figure 6.6a reveals that there exist two disjoint regions within the uncertainty set that render the nominal solution infeasible, which is indicative of the nonlinear manner in which the uncertain parameters affect design feasibility. Regardless, after a few iterations of the GRCS algorithm, the design becomes robust feasible across the totality of the

uncertainty set. The amount of infeasibility that persists in each iteration is quantified in Table 6.15 of the Appendix. It is also worth noting that, in this example, all of the violating parameter realizations added to master problems were points from the boundary of the ellipsoidal uncertainty set.

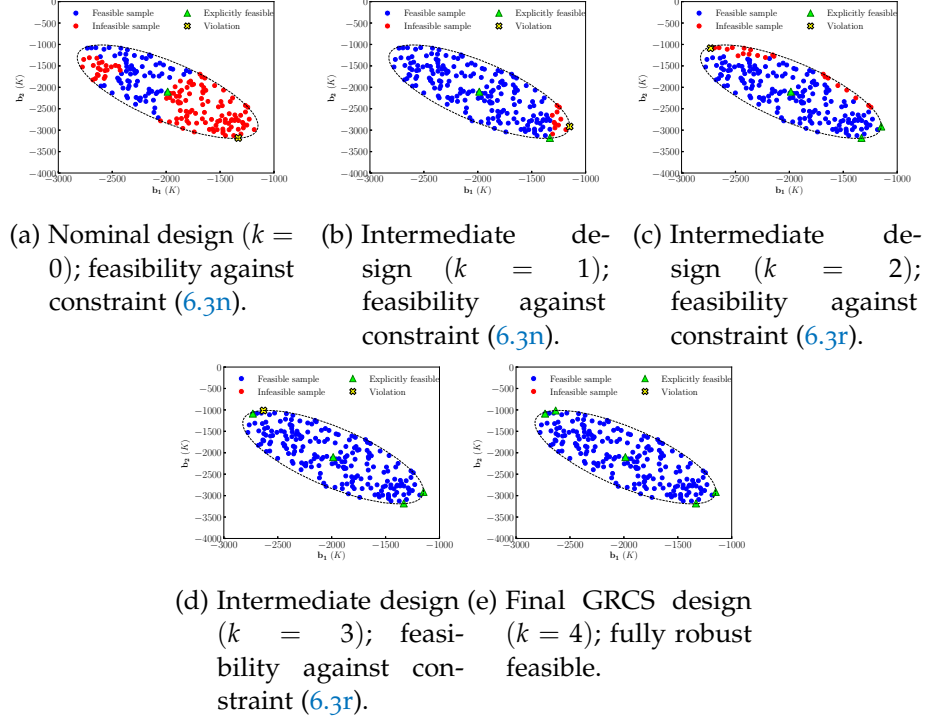


Figure 6.6: Evolution during the GRCS algorithm of the robust feasibility of the MEA-based CO₂ capture flowsheet using the static approximation policy.

6.5 DISCUSSION ON CHOOSING FORM OF RECOURSE POLICY

In this section, we summarize some empirical findings and conclusions pertaining to the performance of the different recourse policies, hoping to inform a future modeler that wishes to address other process design models via our GRCS algorithm. First, we note that, in each of the examples investigated, the costs of designs determined via affine and quadratic decision rules are relatively close to each other, while they both differ more significantly from the deterministic and static approximation costs. From this perspective, our results clearly support the incentive to choose some kind of decision rule (i.e., affine or quadratic) over the static approximation recourse policy.

The choice to deviate from affine decision rules and invoke quadratic rules was less clearly motivated in our investigations. For example, the

results for Case Study I showed a small, yet strict, improvement in expectation (i.e., $\mathbb{E}[\zeta]_{ADR} > \mathbb{E}[\zeta]_{QDR}$), while in Case Study II, the two equally-robust solutions were the same. For a general model, there is no a-priori indicator to predict whether the invocation of quadratic decision rules will or will not improve the results derived with affine decision rules. Certainly, as one considers hierarchies of decision rules (e.g., going from affine to quadratic to cubic, or even, to quartic polynomials), diminishing returns are expected due to the closure of the two-stage adaptivity gap, and the modeler has an incentive to empirically investigate that. This in fact strengthens the need for an algorithm like GRCS, which can consider different decision rule functions in a modular manner, enabling the modeler to perform such investigations.

One should also keep in mind that the GRCS algorithm also admits non-polynomial rules, which cannot be hierarchically compared to those tested here; those non-polynomial rules may prove to perform particularly well in certain models. We should also highlight that there generally exist opportunities to employ the more involved decision rule forms only for a carefully select subset of second-stage decisions, leaving the remaining ones to be decided based on simpler recourse policies (e.g., static approximation). Whereas at the interest of brevity such strategy was not explored in this work, anyone wishing to apply the GRCS algorithm presented here can readily do so.

Finally, any potential savings in terms of second-stage costs have to be viewed in light of the computational burden associated with choosing more involved decision rule functions, as the complexity of solving the resulting optimization subproblems is expected to increase, in general. For example, in cases when variables z and y as well as parameters q participate linearly in the deterministic model, and the uncertainty set is polyhedral, quadratic decision rules may not be preferable, since they will change the class of the underlying separation problems from linear to nonlinear. In contrast, if the deterministic model is already nonlinear, as is often the case in the process systems engineering context, then quadratic decision rules are a plausible option. Whether or not the additional nonlinearities from the quadratic decision rules lead to exceedingly difficult numerical issues when paired with a particular model and solver would need to be explored in each case. Eventually, the practitioner must empirically determine the trade-off between solution quality and tractability, recognizing that employing higher order polynomial decision rules will come at a computational cost, in general.

6.6 CONCLUSIONS

In this chapter, we applied the generalized robust-cutting set (GRCS) algorithm for identifying robust feasible solutions to three two-stage, nonlinear process systems models under uncertainty. Through these case studies, we showed that flexibility in recourse is inversely related to total costs, notably demonstrating the use of nonlinear decision rules towards this. To conclude, our work builds upon existing literature by applying general, model-agnostic robust optimization methodologies to process systems engineering models. With this capability, practitioners of process design under uncertainty can readily identify risk-averse solutions that are explicitly robust against user-defined uncertainty sets.

6.7 APPENDIX

This Appendix includes detailed model formulations, certain parameter values, data to generate the postulated uncertainty sets for the uncertain parameters, as well as various figures and tables with detailed results, for the Case Studies presented in Chapter 6. More specifically, information for Case Studies I, II and III can be found in Sections 6.7.1, 6.7.2 and 6.7.3, respectively.

6.7.1 Reactor Separator Model

6.7.1.1 Model Formulation

The complete mathematical model for the reactor-separator case study is presented in Equations 6.1a–6.1k.

$$\min_{\substack{V, \delta, \beta, F \\ x_a, x_b, x_c, x_d, x_e}} CRF(c_1 V^2) + c_2 c_3 F [\delta (x_a + x_b) + \beta (x_D + x_E)] \quad (6.1a)$$

$$\text{s.t. } F_{a0} - x_a F (1 - \delta) - C_{a0} x_a V (k_1 + k_3) = 0 \quad (6.1b)$$

$$- x_b F (1 - \delta) + C_{a0} V x_a k_1 - C_{a0} V x_b (k_2 + k_4) = 0 \quad (6.1c)$$

$$- x_c F + C_{a0} V x_b k_2 = 0 \quad (6.1d)$$

$$- x_d F (1 - \beta) + C_{a0} V x_a k_3 = 0 \quad (6.1e)$$

$$- x_e F (1 - \beta) + C_{a0} V x_b k_4 = 0 \quad (6.1f)$$

$$x_a + x_b + x_c + x_d + x_e = 1 \quad (6.1g)$$

$$F x_c \geq \chi \quad (6.1h)$$

$$F x_d \beta \geq \omega \quad (6.1i)$$

$$0 \leq \delta \leq 1 \quad (6.1j)$$

$$0 \leq \beta \leq 1 \quad (6.1k)$$

6.7.1.2 Certain Parameter Data

The following table includes the certain parameter data for the reactor-separator case study.

Parameter	Value
C_{a0}	10 ($\frac{\text{mol}}{\text{m}^3}$)
F_{a0}	100 ($\frac{\text{mol}}{\text{hr}}$)
CRF	0.09 ($\frac{1}{\text{yr}}$)
χ	40 ($\frac{\text{mol } c}{\text{hr}}$)
ω	0.4 ($\frac{\text{mol } d}{\text{hr}}$)
c_1	10 $\$/\text{m}^3$
c_2	0.125 $\$/\text{mol}$
c_3	8,760 ($\frac{\text{hr}}{\text{yr}}$)

6.7.1.3 Uncertain Parameter Data (Ellipsoidal Set)

The relevant data required to construct the applicable uncertainty set is presented in the below table.

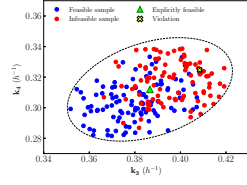
Parameter (hr ⁻¹)	Mean	Std. Deviation
k_1	0.9945	0.02075
k_2	0.5047	0.01340
k_3	0.3866	0.01176
k_4	0.3120	0.01099

$$\text{cov}_{k_1, k_1, k_3, k_4} = \begin{pmatrix} 4.3056 \times 10^{-4} & -2.9751 \times 10^{-7} & 5.4880 \times 10^{-5} & 8.2415 \times 10^{-5} \\ -2.9751 \times 10^{-7} & 1.7956 \times 10^{-4} & 2.8223 \times 10^{-5} & 8.5252 \times 10^{-5} \\ 5.4880 \times 10^{-5} & 2.8223 \times 10^{-5} & 1.3830 \times 10^{-4} & 4.5623 \times 10^{-5} \\ 8.2415 \times 10^{-5} & 8.5252 \times 10^{-5} & 4.5623 \times 10^{-5} & 1.2078 \times 10^{-4} \end{pmatrix}$$

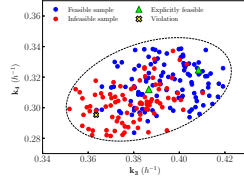
6.7.1.4 Results

In Figures 6.7 and 6.8, we plot the algorithmic progression towards robust feasibility at each iteration of the GRCS under the affine and quadratic decision rule policies. These plots can be interpreted in the same manner as Figure 6.2 in the chapter corresponding to the static approximation case.

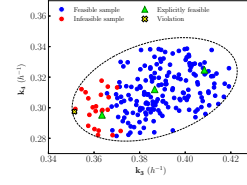
Furthermore, results that quantify how infeasibility is progressively diminished along each solution trajectory are provided in Table 6.13. For each recourse policy, we report the number of points from the sample set \mathcal{S} that lead to infeasibility of the provisional design against each performance constraint and at each iteration. We also report the average infeasibility across all sampled realizations.



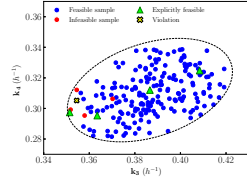
(a) Nominal design
($k = 0$); feasibility
against constraint
(6.1h).



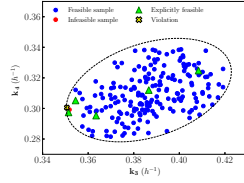
(b) Intermediate de-
sign
($k = 1$); feasibility
against constraint
(6.1i).



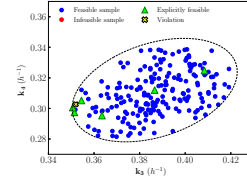
(c) Intermediate de-
sign
($k = 2$); feasibility
against constraint
(6.1i).



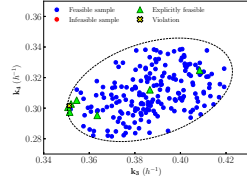
(d) Intermediate de-
sign
($k = 3$); feasibility
against constraint
(6.1i).



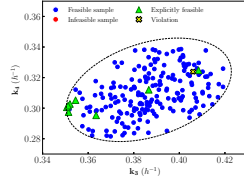
(e) Intermediate de-
sign
($k = 4$); feasibility
against constraint
(6.1i).



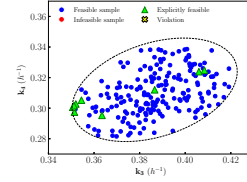
(f) Intermediate design
($k = 5$); feasibility
against constraint
(6.1i).



(g) Intermediate de-
sign
($k = 6$); feasibility
against constraint
(6.1i).

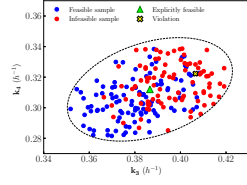


(h) Intermediate de-
sign
($k = 7$); feasibility
against constraint
(6.1h).

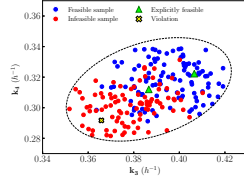


(i) Final GRCS design
($k = 8$); fully robust
feasible.

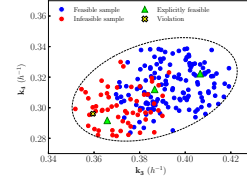
Figure 6.7: Evolution during the GRCS algorithm of the robust feasibility of the reactor-separator designs using affine decision rules.



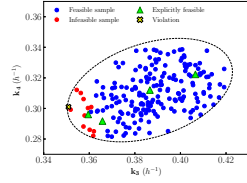
(a) Nominal design
($k = 0$); feasibility
against constraint
(6.1h).



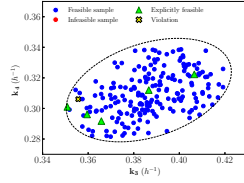
(b) Intermediate de-
sign
($k = 1$); feasibility
against constraint
(6.1i).



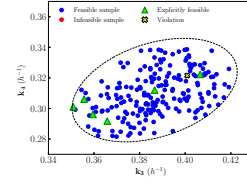
(c) Intermediate de-
sign
($k = 2$); feasibility
against constraint
(6.1i).



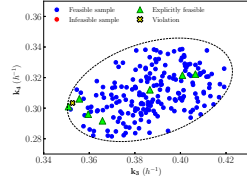
(d) Intermediate de-
sign
($k = 3$); feasibility
against constraint
(6.1i).



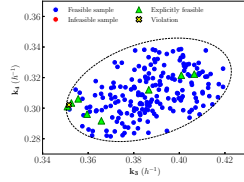
(e) Intermediate de-
sign
($k = 4$); feasibility
against constraint
(6.1i).



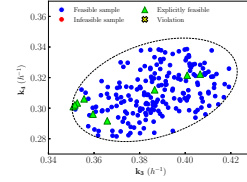
(f) Intermediate design
($k = 5$); feasibility
against constraint
(6.1h).



(g) Intermediate de-
sign
($k = 6$); feasibility
against constraint
(6.1i).



(h) Intermediate de-
sign
($k = 7$); feasibility
against constraint
(6.1i).



(i) Final GRCS design
($k = 8$); fully robust
feasible.

Figure 6.8: Evolution during the GRCS algorithm of the robust feasibility of the reactor-separator designs using quadratic decision rules.

k	Static Approximation				Affine DR				Quadratic DR			
	# infeas. of (6.1h)	Avg. infeas. of (6.1h)	# infeas. of (6.1i)	Avg. infeas. of (6.1i)	# infeas. of (6.1h)	Avg. infeas. of (6.1h)	# infeas. of (6.1i)	Avg. infeas. of (6.1i)	# infeas. of (6.1h)	Avg. infeas. of (6.1h)	# infeas. of (6.1i)	Avg. infeas. of (6.1i)
0	96	2.65×10^{-1}	98	4.65×10^{-3}	96	2.65×10^{-1}	98	4.65×10^{-3}	96	2.65×10^{-1}	98	4.65×10^{-3}
1	0	—	98	4.65×10^{-3}	0	—	90	8.77×10^{-3}	0	—	93	1.25×10^{-2}
2	0	$2.68 \times 10^{-3\dagger}$	0	—	0	—	22	7.45×10^{-4}	0	—	58	3.69×10^{-3}
3	0	—	0	—	0	—	4	6.70×10^{-5}	0	—	11	2.88×10^{-4}
4					0	—	1	4.48×10^{-6}	0	—	0	$2.76 \times 10^{-3\dagger}$
5					0	—	0	$5.73 \times 10^{-4\dagger}$	0	$2.21 \times 10^{-2\dagger}$	0	—
6					0	—	0	$1.43 \times 10^{-4\dagger}$	0	—	0	$6.78 \times 10^{-4\dagger}$
7					0	$4.91 \times 10^{-3\dagger}$	0	—	0	—	0	$1.69 \times 10^{-4\dagger}$
8					0	—	0	—	0	—	0	—

Table 6.13: Evolution of robust feasibility for the reactor-separator design across different recourse policies. The † annotations refer to non-robust solutions that happened to remain feasible under all chosen realization samples; in these cases, we instead report the magnitude of violations, as identified by the respective separation problems.

6.7.2 Reactor Heater Model

6.7.2.1 Model Formulation

The complete mathematical model for the reactor-heater case study is presented in Equations 6.2a–6.2p.

$$\min_{\substack{V, A, F_w, F_1 \\ x_A, T_1, T_2, T_{w1}, T_{w2}, \Delta T_{ln}}} 0.3 (2304V^{0.7} + 2912A^{0.6}) + 8760(2.2e^{-4}F_w + 8.82e^{-4}F_1) \quad (6.2a)$$

$$\text{s.t. } F_0x_A - k_0 \exp(-E/RT_1) C_{A0} (1 - x_A) V = 0 \quad (6.2b)$$

$$F_0c_p (T_0 - T_1) - F_1c_p (T_1 - T_2) + (-\Delta H_R) F_0x_A = 0 \quad (6.2c)$$

$$F_1c_p (T_1 - T_2) = AU\Delta T_{ln} \quad (6.2d)$$

$$\Delta T_{ln} = \frac{(T_1 - T_{w2}) - (T_2 - T_{w1})}{\ln((T_1 - T_{w2}) / (T_2 - T_{w1}))} \quad (6.2e)$$

$$F_1c_p (T_1 - T_2) = F_wc_{pw} (T_{w2} - T_{w1}) \quad (6.2f)$$

$$311 \leq T_1 \leq 389 \quad (6.2g)$$

$$311 \leq T_2 \leq 389 \quad (6.2h)$$

$$300 \leq T_{w2} \leq 380 \quad (6.2i)$$

$$T_1 - T_2 \geq 0 \quad (6.2j)$$

$$T_{w2} - T_{w1} \geq 0 \quad (6.2k)$$

$$T_1 - T_{w2} \geq 11.1 \quad (6.2l)$$

$$T_2 - T_{w1} \geq 11.1 \quad (6.2m)$$

$$x_A \geq 0.9 \quad (6.2n)$$

$$0 \leq F_w \leq 5,000 \quad (6.2o)$$

$$0 \leq F_1 \leq 5,000 \quad (6.2p)$$

6.7.2.2 Certain Parameter Data

The following table includes the certain parameter data for the reactor-heater case study.

Parameter	Value
C_{A0}	32.04 (kmol/m ³)
T_0	333.0 (K)
T_{w1}	300.0 (K)
E/R	555.6 (K)
$-\Delta H_R$	23,260.0 (kJ/kmol)
c_p	167.4 (kJ/kg K)
c_{pw}	4.184 (kJ/kg K)
F_0	45.36 (kmol/hr)

6.7.2.3 Uncertain Parameter Data (Box Set)

The relevant data required to construct the applicable uncertainty set is presented in the below table.

Parameter	Nominal Value	\pm Deviation
U (kJ/m ² h K)	1,635.0	20%
k_0 (hr ⁻¹)	12.0	10%

6.7.2.4 Results

In Figures 6.9 and 6.10, we plot the algorithmic progression towards robust feasibility at each iteration of the GRCS under the affine and quadratic decision rule policies. These plots can be interpreted in the same manner as Figure 6.4 in the chapter corresponding to the static approximation case.

Furthermore, results that quantify how infeasibility is progressively diminished along each solution trajectory are provided in Table 6.13. For each recourse policy, we report the number of points from the sample set \mathcal{S} that lead to infeasibility of the provisional design against each performance constraint and at each iteration. We also report the average infeasibility across all sampled realizations.

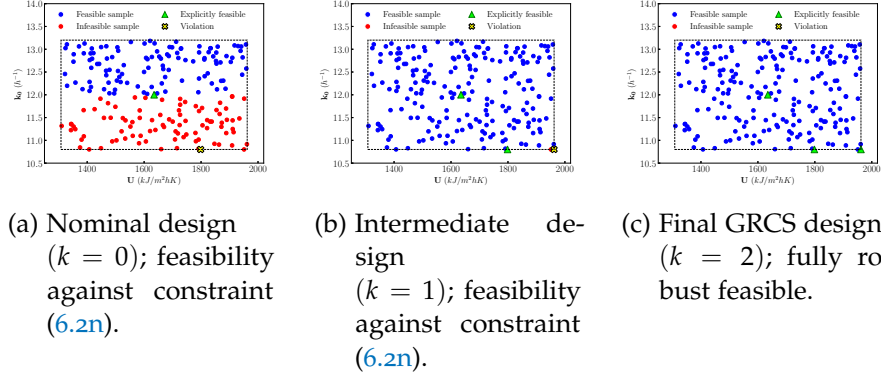


Figure 6.9: Evolution during the GRCS algorithm of the robust feasibility of the reactor-heater designs using affine decision rules.

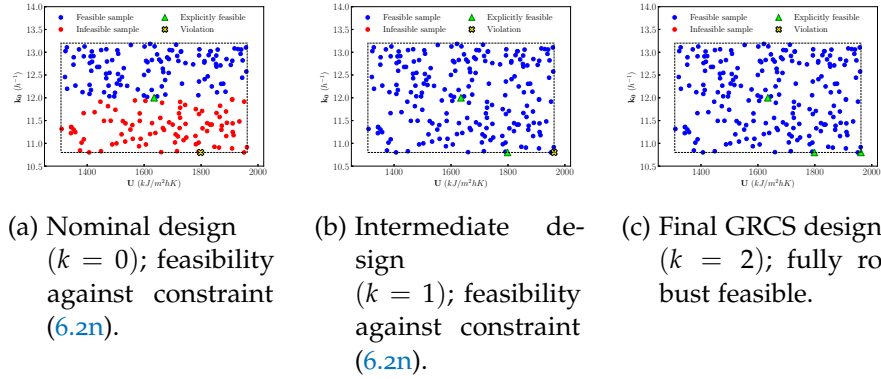


Figure 6.10: Evolution during the GRCS algorithm of the robust feasibility of the reactor-heater designs using quadratic decision rules.

	Static Approximation		Affine DR		Quadratic DR	
k	# infeas. of (6.2n)	Average infeas. of (6.2n)	# infeas. of (6.2n)	Average infeas. of (6.2n)	# infeas. of (6.2n)	Average infeas. of (6.2n)
0	94	2.59×10^{-3}	94	2.59×10^{-3}	94	2.59×10^{-3}
1	1	3.19×10^{-4}	1	3.78×10^{-4}	0	$2.73 \times 10^{-3\dagger}$
2	0	—	0	—	0	—

Table 6.14: Evolution of robust feasibility for the reactor-heater design across different recourse policies. The † annotations refer to non-robust solutions that happened to remain feasible under all chosen realization samples; in these cases, we instead report the magnitude of violations, as identified by the respective separation problems.

6.7.3 *CO₂ Capture Flowsheet Model*6.7.3.1 *Model Notation**Sets*

\mathcal{V}	Vapor phase components $\{MEA, CO_2, N_2, H_2O\}$
\mathcal{L}	Liquid phase components $\{MEA, CO_2, H_2O\}$

Variables

$C_{i,v}$	Molar concentration of species i in vapor phase (mol/m ³)
u_v	Vapor phase velocity (m/s)
$N_{i,v}$	Net specific molar transfer rate of species i from vapor to liquid phase (mol/m ³ s)
$r_{i,v}$	Rate of generation of species i in vapor phase (mol/m ³ s)
ϵ_V	Vapor fraction
$C_{i,l}$	Molar concentration of species i in liquid phase (mol/m ³)
u_l	Liquid phase velocity (m/s)
$N_{i,l}$	Net specific molar transfer rate of species i from liquid to vapor phase (mol/m ³ s)
$r_{i,l}$	Rate of generation of species i in liquid phase (mol/m ³ s)
ϵ_L	Liquid fraction
$K_{i,v}$	Overall mass transfer coefficient in vapor phase of species i (m/s)
$C_{tot,v}$	Total vapor phase concentration (mol/m ³)
$N_{tot,v}$	Total net specific molar transfer rate of species i from vapor to liquid phase (mol/m ³ s)
y_i	Mole fraction of species i vapor phase
$y_i^{*,eq}$	Equilibrium mole fraction of species i vapor phase
a_e	Specific vapor-liquid interfacial area (m ² /m ³)
t	Time (s)
z	Length of column (m)

6.7.3.2 *Economic Objective Function*

The objective of the MEA-based CO₂ capture flowsheet model is presented in this section.

$$\zeta = ACC + TOMC$$

$$ACC = CRF \cdot TPC$$

$$CRF = \frac{i(i+1)^n}{(i+1)^n - 1}$$

$$TPC = \sum_k PC_k \quad k \in \{abs, reg, xhx, pump, MEA, H_2O, reb, con\}$$

$$TOMC = \lambda_1 TPC + \lambda_2 UMUC$$

$$UMUC = \sum_i UC_i \quad i \in \{pump, reb, con, MEA, H_2O\}$$

where: $i = 8\%$, $n = 25$ yr, $\lambda_1 = 0.3863$, and $\lambda_2 = 1.05$.

CAPITAL COSTS:

- Absorber and Regenerator Columns:

$$PC_i = PC_{vessel,i} + PC_{packing,i} \quad i \in \{abs, reg\}$$

$$PC_{vessel,i} = PC_{vessel,i}^0 \left(\frac{A_{vessel,i}}{A_{vessel,i}^0} \right)^{0.6} \quad i \in \{abs, reg\}$$

$$A_{vessel,i} = \pi D_i L_i + \frac{1}{2} \pi D_i^2 \quad i \in \{abs, reg\}$$

$$PC_{vessel,i}^0 = \exp \left(7.2756 + 0.18255 \ln (W_i^0) + 0.02297 \ln (W_i^0)^2 \right) \quad i \in \{abs, reg\}$$

$$W_i^0 = (0.08 \text{ ft}) \rho_{steel} A_{vessel,i}^0 \quad i \in \{abs, reg\}$$

$$PC_{packing,i} = PC_{packing,i}^0 \left(\frac{V_i}{V_{packing,i}^0} \right)^{0.6} \quad i \in \{abs, reg\}$$

$$V_{vessel,i} = \frac{1}{4} \pi D_i^2 L_i \quad i \in \{abs, reg\}$$

$$PC_{packing,i}^0 = V_{packing,i}^0 C_p \quad i \in \{abs, reg\}$$

where: W_i^0 in lbs, $\rho_{steel} = 490 \text{ lb/ft}^3$, $C_p = 250 \text{ \$/ft}^3$, while $A_{vessel,i}^0 = 2,518.76 \text{ ft}^2$ and $V_{packing,i}^0 = 6,427 \text{ ft}^3$, for i being both absorber and regenerator.

- Heat Exchanger (Floating Head Type):

$$PC_{xhx}^0 = \exp \left(11.667 - 0.8709 \ln (A_{xhx}^0) + 0.09005 \ln (A_{xhx}^0)^2 \right)$$

$$PC_{xhx} = PC_{xhx}^0 \left(\frac{A_{xhx}}{A_{xhx}^0} \right)^{0.6}$$

where: $A_{xhx}^0 = 9,687 \text{ ft}^2$.

- Pump:

$$PC_{pump}^0 = a + b \left(P_{pump}^0 \right)^n$$

$$PC_{pump} = PC_{pump}^0 \left(\frac{P_{pump}}{P_{pump}^0} \right)^{0.6}$$

where: $a = 580,000$, $b = 20,000$, $n = 0.6$, and $P_{pump}^0 = 250$ kW.

- MEA and H₂O Storage Tanks:

$$PC_{tank,i}^0 = a + b \left(V_{tank,i}^0 \right)^n \quad i \in \{MEA, H_2O\}$$

$$V_{tank,i} = F_{mix,i}^{in} MW_i t_i / \rho_i \quad i \in \{MEA, H_2O\}$$

$$PC_{tank,i} = PC_{tank,i}^0 \left(\frac{V_{tank,i}}{V_{tank,i}^0} \right)^{0.6} \quad i \in \{MEA, H_2O\}$$

where: $a = 113,000$, $b = 3,250$, $n = 0.65$, $MW_{MEA} = 0.06108$ kg/-mol, $MW_{H_2O} = 0.01802$ kg/mol, $\rho_{MEA} = 1,010$ kg/m³, $\rho_{H_2O} = 997$ kg/m³, $t_{MEA} = 1$ day and $t_{H_2O} = 30$ days, while $V_{tank,MEA}^0 = V_{tank,H_2O}^0 = 100$ m³.

- Reboiler (Kettle Vaporizer Type):

$$PC_{reb}^0 = \exp \left(11.967 - 0.8709 \ln (A_{reb}^0) + 0.09005 \ln (A_{reb}^0)^2 \right)$$

$$PC_{reb} = PC_{reb}^0 \left(\frac{A_{reb}}{A_{reb}^0} \right)^{0.6}$$

$$A_{reb} = \frac{Q_{reb}}{U_{reb} \Delta T}$$

where: $A_{reb}^0 = 1,076$ ft², and $U_{reb} = 1,360.3 \frac{W}{m^2 K}$.

- Condenser (Floating Head Type):

$$PC_{con}^0 = \exp \left(11.667 - 0.8709 \ln (A_{con}^0) + 0.09005 \ln (A_{con}^0)^2 \right)$$

$$PC_{con} = PC_{con}^0 \left(\frac{A_{con}}{A_{con}^0} \right)^{0.6}$$

$$A_{con} = \frac{Q_{con}}{U_{con} \Delta T}$$

where: $A_{con}^0 = 9,687$ ft², and $U_{con} = 320.2 \frac{W}{m^2 K}$.

UTILITY COSTS:

- Electricity (Pump):

$$UC_{pump} = P_{pump} P_{elec}^0$$

where: $P_{elec}^0 = 0.06$ \$/kWh.

- Steam (Reboiler):

$$UC_{reb} = F_{st} P_{st}^0$$

$$F_{st} = \frac{Q_{reb}}{\Delta H_{st}^{vap}}$$

where: $P_{st}^0 = 14.5$ \$/ton, and $\Delta H_{st}^{vap} = 2,114.3$ MJ/ton.

- Cooling Water (Condenser):

$$UC_{con} = F_{cw} P_{cw}^0$$

$$F_{cw} = \frac{-Q_{con}}{C_{p,cw} \Delta T_{cw}}$$

where: $P_{cw}^0 = 0.0329$ \$/ton, $C_{p,cw} = 4.184$ MJ/ton K and $\Delta T_{cw} = 15$ K.

- Make-up Streams (MEA and H₂O Hold-up):

$$UC_i = F_{mix,i}^{in} MW_i P_i^0 \quad i \in \{MEA, H_2O\}$$

where: $P_{MEA}^0 = 1,250$ \$/ton, $P_{H_2O}^0 = 0.04$ \$/ton, $MW_{MEA} = 0.06108$ kg/mol, and $MW_{H_2O} = 0.01802$ kg/mol.

6.7.3.3 Model Equations

Some representative constraints of the MEA-based CO₂ capture flowsheet model is presented in this section, focusing mainly on equations to model the absorber and regenerator columns. Although the mole balance equations are presented here with dynamic terms in time, the present study was performed at steady-state conditions.

MOLE BALANCE EQUATIONS:

$$\epsilon_V \frac{\partial C_{i,V}}{\partial t} = -\frac{\partial(C_{i,V}u_V)}{\partial z} - N_{i,V} + r_{i,V} \quad \forall i \in \mathcal{V} \quad (6.3a)$$

$$r_{i,V} = 0 \quad \forall i \in \mathcal{V} \quad (6.3b)$$

$$\frac{\partial C_{i,V}}{\partial t} = 0 \quad \forall i \in \mathcal{V} \quad (6.3c)$$

$$\epsilon_L \frac{\partial C_{i,L}}{\partial t} = \frac{\partial(C_{i,L}u_L)}{\partial z} + N_{i,L} + r_{i,L} \quad \forall i \in \mathcal{L} \quad (6.3d)$$

$$r_{i,L} = 0 \quad \forall i \in \mathcal{L} \quad (6.3e)$$

$$\epsilon_L = 1 - \epsilon_V \quad (6.3f)$$

$$\frac{\partial C_{i,L}}{\partial t} = 0 \quad \forall i \in \mathcal{L} \quad (6.3g)$$

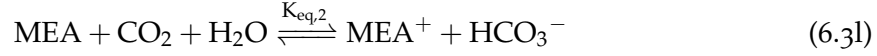
$$(6.3h)$$

MASS TRANSFER EQUATIONS:

$$N_{i,V} = K_{i,V} a_e C_{tot,V} (y_i - y_i^{*,eq}) \quad \forall i \in \mathcal{V} \quad (6.3i)$$

$$N_{i,V} = N_{i,L} \quad \forall i \in \mathcal{L} \quad (6.3j)$$

REACTION KINETICS MODEL:



$$K_{eq,i} = \exp \left(a_i + b_i \ln(T_L) + c_i T_L^{-1} \right) \quad \forall i \in \{1, 2\} \quad (6.3m)$$

where: $a_1 = 198.9$, $b_1 = -1,986$, $c_1 = -32.71$, $a_2 = 167.0$, $b_2 = -2,102$, and $c_2 = -26.51$.

PERFORMANCE CONSTRAINTS:

$$\eta_{\text{CO}_2} \geq 0.85 \quad (6.3n)$$

$$u_{v,abs}^{bottom} \geq 0.5u_{f,abs}^{bottom} \quad (6.3o)$$

$$u_{v,abs}^{bottom} \leq 0.8u_{f,abs}^{bottom} \quad (6.3p)$$

$$u_{v,reg}^{top} \geq 0.5u_{f,reg}^{top} \quad (6.3q)$$

$$u_{v,reg}^{top} \leq 0.8u_{f,reg'}^{top} \quad (6.3r)$$

where:

$$u_{f,j}^i = \left[\left(\frac{g\epsilon^3}{a} \right) \left(\frac{\rho_L}{\rho_V} \right) \left(\frac{\mu_L}{\mu_w} \right)^{-0.2} \exp(-4H^{0.25}) \right]^{0.5} \quad \forall i \in \{top, bottom\}, \forall j \in \{abs, reg\}$$

$$H = \frac{F_L}{F_V} \left(\frac{\rho_V}{\rho_L} \right)^{0.5}$$

$$\epsilon = 0.97, a = 250 \text{ m}^2/\text{m}^3, g = 9.8 \text{ m/s}^2$$

DECISION VARIABLE BOUNDS:

$$3.0 \text{ m} \leq L_j \leq 50.0 \text{ m} \quad \forall j \in \{abs, reg\} \quad (6.3s)$$

$$0.5 \text{ m} \leq D_j \leq 20.0 \text{ m} \quad \forall j \in \{abs, reg\} \quad (6.3t)$$

$$Q_{reb} \geq 0 \quad (6.3u)$$

$$Q_{con} \leq 0 \quad (6.3v)$$

UTILITY AVAILABILITY BOUNDS:

$$F_{st} \leq 0.020 \text{ ton/s} \quad (6.3w)$$

$$F_{cw} \leq 0.420 \text{ ton/s} \quad (6.3x)$$

6.7.3.4 Uncertain Parameter Data (Ellipsoidal Set)

The relevant data required to construct the applicable uncertainty set is presented in the below table.

Parameter	Mean
b_1 (K)	-1,986.0
b_2 (K)	-2,102.0

$$\text{cov}_{b_1, b_2} = \begin{pmatrix} +1.173 & -1.112 \\ -1.112 & +1.983 \end{pmatrix} \times 10^6$$

6.7.3.5 Results

In Figure 6.11, we plot the algorithmic progression towards robust feasibility at each iteration of the GRCS under the affine decision rule policy. These plots can be interpreted in the same manner as Figure 6.6 in the chapter corresponding to the static approximation case.

Furthermore, results that quantify how infeasibility is progressively diminished along each solution trajectory are provided in Table 6.15. For each recourse policy, we report the number of points from the sample set \mathcal{S} that lead to infeasibility of the provisional design against each performance constraint and at each iteration. We also report the average infeasibility across all sampled realizations.

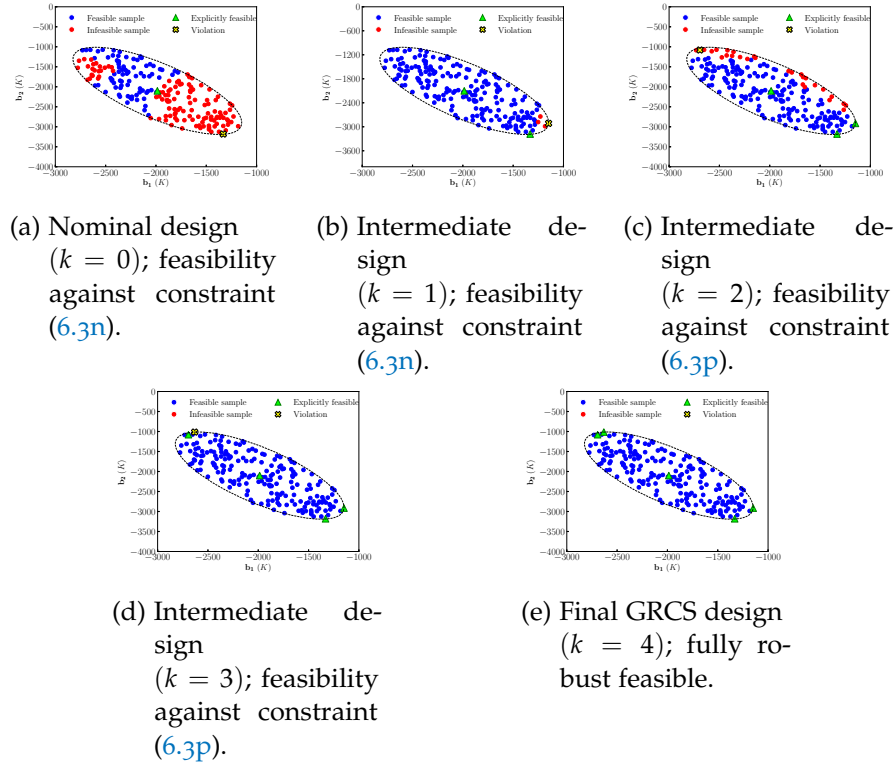


Figure 6.11: Evolution during the GRCS algorithm of the robust feasibility of the MEA-based CO₂ capture flowsheet using affine decision rules.

k	Static Approximation				Affine DR			
	# infeas. of (6.3n)	Average infeas. of (6.3n)	# infeas. of (6.3p)	Average infeas. of (6.3p)	# infeas. of (6.3n)	Average infeas. of (6.3n)	# infeas. of (6.3p)	Average infeas. of (6.3p)
0	120	3.51×10^{-2}	93	2.96×10^{-3}	120	3.51×10^{-2}	93	2.96×10^{-3}
1	17	3.00×10^{-3}	47	2.74×10^{-4}	4	6.00×10^{-4}	36	3.90×10^{-4}
2	0	—	19	1.31×10^{-4}	0	—	27	1.31×10^{-4}
3	0	—	0	$1.24 \times 10^{-3\dagger}$	0	—	0	$1.36 \times 10^{-3\dagger}$
4	0	—	0	—	0	—	0	—

Table 6.15: Evolution of robust feasibility for the CO₂ capture flowsheet across different recourse policies. The [†] annotations refer to non-robust solutions that happened to remain feasible under all chosen realization samples; in these cases, we instead report the magnitude of the violation, as identified by the respective separation problem.

PYROS: THE PYOMO ROBUST OPTIMIZATION SOLVER

7.1 INTRODUCTION

The robust optimization literature has begun addressing increasingly complex and practical problems of interest. In medical logistics applications, robust optimization formulations have been devised and solved for allocating medical evacuation assets [61] and ambulance deployment [13]. In the area of renewable energy, Xiong, Jirutitijaroen, and Singh [139] propose a two-stage RO formulation for the unit commitment problem under uncertainty from renewable wind energy resources. Correspondingly, this expansion in relevant applications for RO methods has created an interest in expanding the available software tools for automatic robust optimization. To date, there have been a number of software tools developed for automatically applying robust optimization methods for solving uncertain optimization problems. Table 7.1 summarizes the type of uncertain optimization problems handled by the available tools for automatic RO, with a specific focus on the classes of non-linear and non-convex problems handled by each tool.

The *AML* column refers to the programming language or algebraic modeling language through which the tool is callable. The *Convex* and *Non-convex NLP* columns denote whether or not the tool can solve convex or non-convex deterministic non-linear optimization models under uncertainty. The *Non-convex in parameters* column indicates whether the tool can handle the case of nonlinear (and/or non-convex) participation of uncertain parameters within the model constraints. The *Two-stage* and *Multi-stage* columns indicate whether the tool can handle two- or multi-stage robust optimization problems. The column *Uncertain equalities* indicates whether uncertain parameters may participate in equality constraints within the uncertain optimization problem. Finally, the column *Open source* signifies whether the software is available without a license (excluding any potentially required subordinate solver licenses).

The AIMMS Robust Optimization extension is an add-on to AIMMS¹ for automatically casting and solving robust counterparts for deterministic

Tool	AML	Convex NLP	Non-convex NLP	Non-convex in parameters	Two-stage	Multi-stage	Non-fixed Recourse	Uncertain equalities	Open source
AIMMS RO	AIMMS				✓	✓			
JuMPeR	Julia				✓				✓
ROC++	C++				✓	✓			✓
ROME	MATLAB				✓	✓			✓
ROModel	Pyomo	✓	✓		✓				✓
SIPAMPL	AMPL	✓							✓
YALMIP	MATLAB	✓							✓
PyROS	Pyomo	✓	✓	✓	✓		✓	✓	✓

Table 7.1: Capabilities of robust optimization tools for handling nonlinear uncertain optimization problems.

LPs or MIPs. There are three explicitly supported uncertainty set types: box, ellipsoidal, and a convex hull of supplied finite set of scenarios. Additionally, AIMMS supports two- and multi-stage recourse via linear decision rules.

JuMPeR² is a robust optimization toolbox implemented in the Julia programming language with the JuMP AML. Similar to AIMMS RO, JuMPeR can solve deterministic LP or MIP models under uncertainty via robust optimization. JuMPeR also allows the user to select from two RO solution approaches – the robust counterpart formulation or a robust cutting-plane algorithm. The JuMPeR package also provides pre-implemented uncertainty set classes for polyhedral sets, ellipsoidal sets, and an interface for custom, user-defined sets.

ROC++³, published by Vayanos, Jin, and Elissaios [122], provides a modeling framework for RO in C++. ROC++ can solve uncertain LP and MIP models under two-stage or multi-stage recourse. Recourse is handled in ROC++ via constant, piecewise constant, or linear decision rules, as well as finite adaptability [121]. In order to specify uncertainty sets on a robust optimization model in ROC++, the user must specify the explicit constraints representing the set. The uncertainty sets supported in ROC++ include ellipsoidal and polyhedral sets. Furthermore, ROC++ provides support for endogenous uncertainty.

The MATLAB modeling language ROME⁴ can also be used to represent and solve multi-stage uncertain optimization problems via robust counterpart reformulation and linear decision rules to a deterministic equivalent for solving via commercial solvers [42]. A related tool, RSOME⁵ which has recently been made available via Python, utilizes robust stochastic optimization to solve the same class of problems, and even supports distributionally robust optimization [22] [23].

YALMIP⁶ was a MATLAB toolbox initially devised for solving semidefinite programs, but has been extended to robust optimization. YALMIP can solve single-stage convex deterministic optimization models under uncertainty and supports polyhedral, ellipsoidal, and other conic uncertainty sets [78] [77].

SIPAMPL⁷ solves models in AMPL via a native semi-infinite programming solver. SIPAMPL does not support recourse and thus is suitable for single-stage robust optimization [123].

Recently, ROModel⁸ [134] has been proposed for modeling and solving two-stage robust optimization problems via linear decision rules and both reformulation-based and cutting-plane based solution approaches. ROModel provides interfaces for defining polyhedral, ellipsoidal, and custom user-defined uncertainty sets. ROModel is implemented in Python via the AML Pyomo.

In this work, we propose PyROS, the Pyomo Robust Optimization Solver. PyROS is a robust optimization solver in Python and Pyomo for solving general nonlinear robust optimization problems. The existing software in RO is able to handle a many classes of robust optimization problems, but PyROS aims to expand capabilities for solving nonlinear robust optimization problems.

The contributions made through PyROS are outlined here. First, PyROS is the first automatic robust optimization tool for handling general, nonlinear (convex or non-convex) uncertain optimization problems. PyROS is capable of solving nonlinear models containing implicitly defined state variables and state equations without any required reformulation from the user or within the underlying algorithm. PyROS operates entirely on the deterministic optimization model provided by the user and solves the robust optimization problem with user-provided data regarding uncertainty in the deterministic model parameters.

Secondly, in all of the tools noted above, *fixed recourse* is required. This means that the coefficients of the adjustable variables may not be potentially uncertain. This is not a requirement within PyROS due to the underlying GRCS algorithm utilized. The authors are not aware of any other RO software that supports such problems, making PyROS a novel extension to existing capabilities in the field.

Additionally, for solving two-stage problems, PyROS implements constant, affine, and quadratic decision rules for handling recourse variables. To date, there is no explicit support for quadratic decision rules in the existing software tools.

7.2 PYROS METHODOLOGY

The PyROS solver employs the GRCS as the underlying robust optimization scheme. The advantage of this approach is that it requires no reformulation of the deterministic model and can thus be applied to a broad range of deterministic model types. The generalized robust cutting-set algorithm (GRCS) is a meta-algorithm for determining robust solutions to general two-stage, nonlinear programming problems. The theory is

explained in detail in Chapter 5. Here, we will reiterate the GRCS methodology in Section 5.2.1, as well as explain additions or modifications to the GRCS made in PyROS. We explain the addition of coefficient matching capabilities in PyROS for handling general uncertain equality constraints in Section 7.2.1. We also outline modified algorithmic procedures for identification of worst-case objective and discrete scenario uncertainty sets in separation in Section 7.2.2.

7.2.1 Polynomial Coefficient Matching

We note that any equality constraints of the form $h(x, z, q) = 0 \forall q \in \mathcal{Q}$ may also be part of the canonical uncertain optimization problem, D . These uncertain equality constraints are not explicit state variable definitions, but instead restrict the feasible values of the x and z variables due to the robustness requirement on the constraint.

In the case that these uncertain equality constraints are polynomial in uncertain parameters q , a *matching of polynomial coefficients* scheme can be used to reduce the uncertain equality to a set of certain equalities $\tilde{h}(x) = 0$. First, the decision rule approximation is applied to achieve the following substitution: $z_\ell \leftarrow v_\ell(d_\ell, q) \forall \ell \in \{1, \dots, n\}$. Because the d_ℓ variables are first-stage and can be grouped with x in this context, this reduces the uncertain equality $h(x, z, q) = 0$ to $h(x, q) = 0$. Then, polynomial coefficients of q are grouped and set to zero to create the constraints $\tilde{h}(x) = 0$. Thus, the original uncertain $h(x, z, q) = 0 \forall q \in \mathcal{Q}$ constraints are replaced with the certain $\tilde{h}(x) = 0$ constraints in the master problem formulation as part of the definition of variables x and are therefore part of \mathcal{X} . The uncertain equalities of this form are also correspondingly omitted from the separation problem formulation.

7.2.2 PyROS Separation Procedure

The key algorithmic feature of the separation problem is with regards to worst-case violation identification. At each iteration of the PyROS solution algorithm, a single violating parameter realization is identified to be added to the following master problem. Although multiple violations could be added in principle, this is done to limit the rate of growth of the master problem in constraints and state variables. The worst-case violation selection scheme in PyROS is explained here.

We identify a set \mathcal{I}^V such that $\mathcal{I}^V \subseteq \mathcal{I}$ represents the subset of inequality constraints that lead to violations in the context of the most recent master problem solution. A constraint i is considered part of the set \mathcal{I}^V if $\frac{g_i(z^{i,*}, y^{i,*}, q^{i,*}, x^*)}{\max\{1, |g_i(z^0, y^0, q^0, x^*)|\}} > \varepsilon$, where $\varepsilon \in \mathbb{R}_+$ is a small tolerance and

$(z^{i,*}, y^{i,*}, q^{i,*})$ is the optimal solution to the separation problem SP_i . The ratio used to determine constraint violations here normalizes against the absolute value of the nominal scenario for each constraint.

Once the realizations $q^{j,*}$ that maximally violate each of the constraints $j \in \mathcal{I}^V$ have been determined, we generate a matrix of dimensions $|\mathcal{I}^V| \times |\mathcal{I}^V|$, where each entry $e_{i,j}$ represents the violation of constraint g_i associated with row i under the uncertain parameter realization $q^{j,*}$ associated with column j ; that is, $e_{i,j} := \max\left\{\frac{g_i(z^{i,*}, y^{i,*}, q^{j,*}, x^*)}{\max\{1, |g_i(z^0, y^0, q^0, x^*)|\}}, 0\right\}$. The violating realization that is added back to the next iteration of the master problem, q^* , is chosen to have the largest sum across the columns of this resulting scaled violations matrix, meaning it leads to the largest magnitude net constraint violation across the constraints in the set \mathcal{I}^V .

In the case that the provided uncertainty set is a set of discrete scenarios, either via a `DiscreteScenarioSet` or `IntersectionSet` object, PyROS implements a custom separation protocol. In this case, PyROS solves the formulation in Equations 7.1a-7.1b. In this separation scheme, the uncertainty set is a finite set of scenarios, Q_D and the uncertain parameter realizations in the set are $q^s \in Q_D$, $s = \{1, \dots, S\}$. The discrete scenario separation routine will then loop over each scenario s not already accounted for in the master problem (i.e. $s \notin \mathcal{K}$) and solve the resulting separation problem.

For all $i \in \mathcal{I}$:

For all $s \in \{1, \dots, S\}$ if $s \notin \mathcal{K}$:

$$(SP_i^s) : \max_{z \in \mathbb{R}^n, y \in \mathbb{R}^a} g_i(z, y; q^s, x^*) \quad (7.1a)$$

$$\text{s.t.} \quad h_j(z, y; q^s, x^*) = 0 \quad \forall j \in \mathcal{J} \quad (7.1b)$$

$$z_\ell = v_\ell(q^s; d_\ell^*) \quad \forall \ell \in \{1, \dots, n\} \quad (7.1c)$$

In this case, the separation problem reduces to simply evaluating the state variables y and second-stage degrees of freedom z given each scenario in the uncertainty set. Then, the worst-case realization is identified via the standard separation procedure outlined previously.

7.3 PYROS SOLVER INTERFACE

PyROS is implemented in Pyomo, a Python based algebraic modeling language [17, 52]. Pyomo contains many advanced, contributed packages which extend the AML capabilities. For example, `pyomo.dae` [92] is a framework in Pyomo for representing optimization models containing differential and algebraic variables, and `pyomo.sp` [129] is a package within Pyomo for solving stochastic programming problems. With the inclusion of PyROS, Pyomo now contains a generic cutting-set based solver capability for non-linear robust optimization problems.

One of the novel utilities of the PyROS solver capability within Pyomo is that it is easily callable on any existing deterministic Pyomo optimization model and requires few additional solver arguments. This allows for a seamless transition for Pyomo users from their existing models to robust optimization capabilities.

PyROS is designed to require several specifications from the user, which are as follows:

- The deterministic optimization model
- List of first-stage (“design”) degree of freedom variables
- List of second-stage (“control”) degree of freedom variables
- List of parameters to be considered uncertain
- The uncertainty set
- Subordinate local and global NLP optimization solvers

Of the list of required arguments, the uncertainty set and subordinate NLP solvers are the only additional modeling components that the user must instantiate. The subordinate solvers are required for solving the master and separation subproblems within the algorithm. We note that in all cases, a global NLP solver must be provided to certify robustness (feasibility or optimality) before termination. Additionally, any free model variables not specified as first- or second-stage when calling the PyROS solver is assumed to be state variables. How the uncertainty sets are specified for PyROS is further explained in Section 7.3.1.

7.3.1 *Uncertainty Sets*

The key modeling component of PyROS are the uncertainty set classes. These classes represent data containers which encapsulate pertinent data for defining a mathematical uncertainty set. PyROS includes several pre-implemented uncertainty set classes that represent uncertainty set types corresponding to common sets in the robust optimization literature. See Table 7.2 for the comprehensive list of these sets, their mathematical representations, and the derived inferred bounds on the uncertain parameters defining the set. In PyROS, these inferred bounds on the uncertain parameters become part of the separation problem in the definition of $q \in \mathcal{Q}$.

In the following sections, we offer a brief explanation for each of the available uncertainty set types in PyROS.

7.3.1.1 Box Sets

The box uncertainty set (BoxSet) represents a n -dimensional hyper-rectangle. We refer to this as a box set because the set is devised such that each uncertain parameter $q \in \mathbb{R}^n$ may attain values from within box bounds, i.e. lower and upper bounds, $q^\ell \in \mathbb{R}^n, q^u \in \mathbb{R}^n$.

7.3.1.2 Cardinality Sets

The cardinality, or gamma, uncertainty set (CardinalitySet) states that uncertain parameters $q \in \mathbb{R}^n$ may deviate from their nominal value $q^0 \in \mathbb{R}^n$ by no more than the positive deviations $\hat{q} \in \mathbb{R}_+^n$, and the total amount of positive deviations is bounded by parameter Γ . When one selects $\Gamma = 0$, the uncertainty set reduces to a singleton set containing only the nominal point, q^0 . When $\Gamma = n$, then the uncertainty set becomes an n -dimensional hyper-rectangle.

7.3.1.3 Budget Sets

The budget set (BudgetSet) in PyROS is the intersection of the positive orthant with $L \in \mathbb{N}$ budget constraints. The budget constraints are specified via parameters $b_\ell \in \mathbb{R}_+^L$, the budget constraint upper bounds, and B_ℓ , the set representing uncertain parameter participation in a given budget ℓ .

7.3.1.4 Factor Model Sets

The factor model set (FactorModelSet) states that uncertain parameters q are in the set defined by additive disturbances $\Psi\zeta$ about the nominal realization, q^0 . The disturbances, which are a linear combination of the independent factors ζ , are in part defined by the matrix $\Psi \in \mathbb{R}_+^{n \times F}$, and we choose to only permit positive disturbances here. Similar to the cardinality set, the parameter β bounds the number of independent factors which can attain their extreme values via the upper bound of βF . Factor model sets are often proposed such that $F \ll n$, making the uncertainty set lower dimensionality in the space of the variables ζ .

7.3.1.5 Polyhedral Sets

The general polyhedral set (PolyhedralSet) is provided to allow more for the specification of polyhedra not representable in the other specialized polyhedral sets (e.g. box, gamma, budget, and factor model). The user must simply specify the coefficient matrix $A \in \mathbb{R}^{m \times n}$ and right-hand side vector, $b \in \mathbb{R}^m$. The linear constraints defining the polyhedral set must lead to a closed and bounded set containing the nominal point, q^0 , to ensure non-emptiness.

7.3.1.6 Ellipsoidal Sets

We provide two constructors for ellipsoidal uncertainty sets in the case that the ellipsoid is axes-parallel (`AxisAlignedEllipsoidalSet`) or more general ellipsoidal sets (`EllipsoidalSet`). The ellipsoidal uncertainty sets represent the case that uncertain parameters are believed to be random variables coming from a multivariate normal distribution with mean q^0 and covariance Σ . In the case of an axes-parallel ellipsoid, the covariance is simply a diagonal matrix of variances in each dimension.

7.3.1.7 Discrete Sets

The discrete sets (`DiscreteScenarioSet`) in PyROS represent a set of finitely many scenarios, S , from which the uncertain parameters may attain values. This uncertainty set may be constructed using previous observations for the uncertain parameters.

Each of the outlined uncertainty set classes are derived from the abstract base class called `UncertaintySet`. PyROS provides advanced uncertainty set capabilities via the `UncertaintySet` abstract base class and the `IntersectionSet` class. With the abstract base class, users can easily translate a mathematical representation of any convex, compact set for use in robust optimization. The `IntersectionSet` class allows users to pose an uncertainty set that is the (non-empty) intersection of any finite number of PyROS `UncertaintySet`-derived objects. If the proposed set intersection is indeed empty, PyROS will throw an error directing the user to ensure a non-empty intersection. Additionally, the PyROS `UncertaintySet` provides an abstract method for user-defined uncertainty sets. This allows users to easily translate a mathematical representation of a closed and bounded set to an `UncertaintySet` object.

PyROS Object	Input Data	Set Representation	Interval Enclosure for Parameters $q_i, i \in \{1 \dots n\}$
BoxSet	$q^l \in \mathbb{R}^n, q^u \in \mathbb{R}^n : q^l \leq q^u$	$Q_X = \{q \in \mathbb{R}^n : q^l \leq q \leq q^u\}$	$q_i^l \leq q_i \leq q_i^u$
CardinalitySet	$\Gamma \in [0, n], q \in \mathbb{R}_+^n, q^0 \in \mathbb{R}^n$	$Q_C = \left\{ q \in \mathbb{R}^n : q = q^0 + (q \circ \xi) \text{ for some } \xi \in \Xi_C \right\}$ $\Xi_C = \left\{ \xi \in [0, 1]^n : \sum_{i=1}^n \xi_i \leq \Gamma \right\}$	$q_i^0 \leq q_i \leq q_i^0 + \min\{\Gamma, 1\} q_i$
BudgetSet	$b_\ell \in \mathbb{R}_+^L, B_\ell \subset \mathbb{N}, L \in \mathbb{R}$	$Q_B = \left\{ q \in \mathbb{R}_+^n : \sum_{i \in B_\ell} q_i \leq b_\ell \forall \ell \in \{1, \dots, L\} \right\}$	$0 \leq q_i \leq \min_{\ell \in \{1, \dots, L\}: i \in B_\ell} \{b_\ell\}$
FactorModelSet	$\beta \in [0, 1], \Psi \in \mathbb{R}_+^{n \times F}, q^0 \in \mathbb{R}^n$	$Q_F = \left\{ q \in \mathbb{R}^n : q = q^0 + \Psi \xi \text{ for some } \xi \in \Xi_F \right\}$ $\Xi_F = \left\{ \xi \in [-1, 1]^F : \sum_{j=1}^F \xi_j \leq \beta F \right\}$	$q_i^0 - \sum_{j=1}^{\lfloor \beta F \rfloor} \Psi_{ifj} + (\beta F - \lfloor \beta F \rfloor) \Psi_{if\lfloor \beta F \rfloor + 1} \leq q_i \leq q_i^0 + \sum_{j=1}^{\lfloor \beta F \rfloor} \Psi_{ifj} + (\beta F - \lfloor \beta F \rfloor) \Psi_{if\lfloor \beta F \rfloor + 1}$
PolyhedralSet	$A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, q^0 \in \mathbb{R}^n : Aq^0 \leq b$	$Q_P = \{q \in \mathbb{R}^n : Aq \leq b\}$	Numerically determined by solving LPs
AxisAlignedEllipsoidalSet	$\alpha \in \mathbb{R}_+^n, q^0 \in \mathbb{R}^n$	$Q_A = \left\{ q \in \mathbb{R}^n : \sum_{i=1, \{a_i > 0\}} \left(\frac{q_i - q_i^0}{\alpha_i} \right)^2 \leq 1, \quad q_i = q_i^0 \quad \forall i : \{a_i = 0\} \right\}$	$q_i^0 - \alpha_i \leq q_i \leq q_i^0 + \alpha_i$
EllipsoidalSet	$P \in \mathbb{S}_+^{n \times n}, s \in \mathbb{R}_+, q^0 \in \mathbb{R}^n$	$Q_E = \{q \in \mathbb{R}^n : (q - q^0)^\top \Sigma^{-1} (q - q^0) \leq s\}$	$q_i^0 - s(\Sigma_{ii})^{1/2} \leq q_i \leq q_i^0 + s(\Sigma_{ii})^{1/2}$
UncertaintySet	$m \in \mathbb{N}_+, g_i : \mathbb{R}^n \mapsto \mathbb{R} \forall i \in \{1 \dots m\}, q^0 \in \mathbb{R}^n : g_i(q^0) \leq 0 \forall i \in \{1 \dots m\}$	$Q_U = \{q \in \mathbb{R}^n : g_i(q) \leq 0 \quad \forall i \in \{1, \dots, m\}\}$	Numerically determined by solving NLPs globally
DiscreteScenarioSet	$S \in \mathbb{N}, q^s \in \mathbb{R}^n$	$Q_D = \{q^s : s = 0, \dots, S\}$	$\min_{s \in \{0, \dots, S\}} \{q_i^s\} \leq q_i \leq \max_{s \in \{0, \dots, S\}} \{q_i^s\}$
IntersectionSet	$Q_i \subset \mathbb{R}^n \quad \forall i \in \{1, \dots, m\}$	$Q_I = \left\{ q \in \mathbb{R}^n : q \in \bigcap_{i \in \{1, \dots, m\}} Q_i \right\}$	Numerically determined.

Table 7.2: Tabulated information regarding pre-implemented uncertainty set classes in PyROS, including uncertainty set name, mathematical representation as a constraint, and inferred bounds.

- i By ensuring a user-supplied point, such as the nominal point q^0 , is in the set, we can guarantee non-emptiness of the set.

7.3.2 PyROS Options

In addition to the solver arguments required by PyROS, there are additional user options which can be specified via keyword arguments or via an options dictionary. Some of the key user options are outlined in this section.

7.3.2.1 Objective Focus

PyROS currently supports two options for robust optimization objective function targets specified via the option `objective_focus`. This option is set by selecting one of the `ObjectiveFocus` types, outlined below:

<code>ObjectiveFocus.nominal</code>	The nominal objective is the sum of first- and (nominal) second-stage objective function terms, shown in 5.6b for the nominal uncertain parameter realization, q^0 . Selecting nominal objective can only lead to a proven robust feasible solution.
<code>ObjectiveFocus.worst_case</code>	The worst-case objective is shown in 5.5b . Selecting worst-case objective guarantees robust optimality.

The `objective_focus` option is closely related to another option, `solve_masters_globally`. In the case that the selected objective function focus is `ObjectiveFocus.worst_case`, then selecting `solve_masters_globally = True` can provide guarantees of robust optimality in the case that the deterministic problem leads to non-convex master problems. We note that for sufficiently large problems, solving the master problem globally at each iteration may lead to algorithmic slow down. Therefore, the default setting for `solve_masters_globally` in PyROS is `False`. Note that this implies that PyROS defaults to proving robust feasibility, regardless of the objective function focus.

Additionally, PyROS provides a user option for *p-robustness* [108]. The *p-robustness* constraints utilized in PyROS, shown in Equation 7.2, can be added via the option `p_robustness` and then supplying the value of the non-negative parameter, ρ .

$$f_1(x) + f_2(x, z^p, y^p; q^p) \leq (1 + \rho)[f_1(x) + f_2(x, z^0, y^0; q^0)] \quad \forall p \in \mathcal{P} \quad (7.2)$$

Here, the set \mathcal{P} is a finite set of realizations from the uncertainty set, \mathcal{Q} , and may be chosen from the set of violating parameter realizations,

\mathcal{K} . The constraints in 7.2 define a bound on the ratio of the objective that any scenario may exhibit relative to the nominal objective. In the case that the nominal objective is selected, p-robustness constraints can be added in order to bound the second-stage objective value contribution relative to other, non-nominal scenarios to potentially improve the robust objective value.

7.3.2.2 Decision Rules

The decision rules in PyROS are specified via the `decision_rule_order` option. This parameter can take a value from the set $\{0, 1, 2\}$, as follows:

<code>decision_rule_order = 0</code>	Constant decision rules, e.g. the static approximation. In this case, second-stage degrees of freedom are first-stage decisions not influenced by uncertain parameter realization information revealed in the second-stage.
<code>decision_rule_order = 1</code>	Affine decision rules. In this case, second-stage degrees of freedom depend on uncertain parameter realizations via an affine function of $ q $ uncertain parameters.
<code>decision_rule_order = 2</code>	Quadratic decision rules. In this case, second-stage degrees of freedom depend on uncertain parameter realizations via a complete homogeneous symmetric polynomial of degree 2 in $ q $ uncertain parameters, with an additional intercept (constant) term.

7.3.2.3 Separation Options

PyROS implements several efficiencies for the separation subproblem which may benefit the user. Firstly, PyROS allows for the specification of an option called `separation_priority_order`, which allows the user to specify a priority value for which inequality constraints to separate. This is specified via a dictionary which maps inequality constraint names in the deterministic model to positive integer priorities for separation. Constraints not referenced in the dictionary assume a priority of 0 (low-

est priority). By default, the in the case of `ObjectiveFocus.worst_case = True`, the worst-case objective is relegated to separation priority 0. Any inequality constraints that originated from bounds on second-stage variables are also given priority 0.

Within the algorithm, the priority-ranked inequality constraints are considered as objectives in separation. These ranked objectives are separated in order of decreasing priority, until a violation is identified. Once a violation is found, the separation subroutine returns.

An additional option for separation subproblems is the `bypass_local_separation` option, which will direct PyROS to utilize the specified `global_solver(s)` for all separation subproblems. By default, PyROS only utilizes the global NLP subsolver at the final iteration to confirm the robustness of the solution.

7.3.3 Calling PyROS

For the following section, we will consider the simple robust optimization model presented in [74] and shown in Equations 7.3a – 7.3b. This problem features a certain quadratic objective and an uncertain inequality constraint that is nonlinear in the uncertain parameter u . The uncertainty set in this example is simply an interval on the uncertain parameter u .

$$\min_{x \geq 0} (x_1 - 4)^2 + (x_2 - 1)^2 \quad (7.3a)$$

$$\text{s.t. } \sqrt{u}x_1 - ux_2 \leq 2 \quad \forall u \in \left[\frac{1}{4}, 2\right] \quad (7.3b)$$

```

1 # Import the Pyomo modeling environment and the PyROS module
2 from pyomo.environ import *
3 import pyomo.contrib.pyros as pyros
4
5 # Write the deterministic Pyomo model
6 m = ConcreteModel()
7 m.x1 = Var(initialize = 0, bounds = (0, None))
8 m.x2 = Var(initialize = 0, bounds = (0, None))
9 m.u = Param(initialize = 1.125, mutable = True)
10
11 m.con = Constraint(expr = sqrt(m.u) * m.x1 - m.u * m.x2 <= 2)
12 m.obj = Objective(expr = (m.x1 - 4)**2 + (m.x2 - 1)**2)
13
14 # Define the uncertainty set
15 interval = pyros.BoxSet(bounds = [(0.25, 2)])
16
17 # Instantiate the PyROS solver
18 pyros_solver = SolverFactory("pyros")
19
20 # Define subsolvers utilized in the algorithm
21 local_subsolver = SolverFactory("ipopt")
22 global_subsolver = SolverFactory("baron")
23
24 # Call the PyROS solver
25 results = pyros_solver.solve(model = m,
26                             first_stage_variables = [m.x1, m.x2],
27                             second_stage_variables = [],
28                             uncertain_params = [m.u],
29                             uncertainty_set = interval,
30                             local_solver = local_subsolver,
31                             global_solver = global_subsolver,
```



```

32         options = {
33             "objective_focus": pyros.ObjectiveType.worst_case,
34             "solve_master_globally": True
35         })

```

Code 7.1: Example of how to solve a robust optimization problem using the PyROS solver given a deterministic Pyomo model.

Given the mathematical formulation, we can write a Pyomo model representing the deterministic optimization problem and illustrate the usage of PyROS to solve the uncertain optimization problem, shown in Code 7.1.

Note that the deterministic Pyomo optimization model to be solved must be instantiated with any prospective uncertain parameters as Pyomo Param objects with the property `mutable=True`. Alternatively, the directive `Param.DefaultMutable = True` can be added after the Pyomo import statement and before defining the model object.

The PyROS solver is callable through the Pyomo SolverFactory in the same way as all other optimization solver. Because PyROS utilizes Pyomo optimization solver objects to solve master and separation subproblems, we provide a table mapping Pyomo solver statuses to PyROS GRCS algorithm actions in Table 7.9 of Appendix 7.6.3.

In general, PyROS accepts optimal or feasible solutions from the subordinate solvers for both the master and separation subproblems when appropriate for algorithmic correctness. If a master problem is determined to be infeasible by the subordinate solver, PyROS will return with a `pyrosTerminationCondition.robust_infeasible` status, as explained in Table 7.8 in Appendix 7.6.2. In the case that a subsolver returns any other status, backup solvers may be employed via the `backup_local_solvers` or `backup_global_solvers` options. If all solver resources are unable to identify an acceptable solution to a subproblem in PyROS, a `pyrosTerminationCondition.subsolver_error` status is returned. The user may then optionally retrieve the pathological subproblem for debugging purposes as a text file by setting the PyROS option `keepfiles=True` and specifying a writable directory via the `subproblem_file_directory` option.

The PyROS solver requires additional functional arguments to solve an uncertain optimization via the generalized robust cutting-set algorithm. These required arguments are explained here with respect to keyword designations in PyROS. First, the partition of degrees of freedom into first- and second-stage decision variables must be provided as positional arguments via `first_stage_variables` and `second_stage_variables`. The second-stage variables are meant to represent potential second-stage degrees of freedom that can be adjusted after the first-stage variables are determined. Therefore, true state variables abiding by the assumption outlined in Chapter 5, Section 5.2 should be excluded from the second-stage

variables list provided here. Next, the parameters (as Pyomo Param objects) which will be considered uncertain are provided via `uncertain_params`, along with the uncertainty set representing the set of possible values for said parameters via `uncertainty_set`. The final set of required arguments are the subsolvers to be utilized in solving sequential master and separation subproblems within the underlying GRCS algorithm. PyROS requires at least one local and one global NLP solver to be specified via `local_solver` and `global_solver`. These should be Pyomo Solver objects referencing the appropriate optimization solvers.

In the example shown in Code 7.1, there are only first-stage degrees of freedom, therefore both `m.x1` and `m.x2` are passed via the `first_stage_variables` argument, while an empty list is passed for the `second_stage_variables` argument. The uncertain parameter `m.u` is supplied via the `uncertain_params`. The uncertainty set object is defined as an interval on the sole uncertain parameter, `m.u`, via a one dimensional `BoxSet` and is supplied via the `uncertainty_set` object. For this example, we illustrate the creation of subsolvers for IPOPT[125] (local NLP solver) and BARON[112] (global NLP solver) and pass them with the arguments `local_solver` and `global_solver`.

In this example, we will select a worst-case objective and global master problem solutions via the options `objective_focus` and `solve_master_globally`. As outlined in Section 7.3.2.1, this will allow us to identify worst-case robust optimal solutions to the problem.

When we solve the above model, we retrieved the expected optimal solution shown in Leyffer, Menickelly, Munson, Vanaret, and Wild [74]: ($x_1 := 3.52, x_2 := 1.55$) with an optimal (worst-case) objective value of 0.53.

PyROS is able to solve this model in 3 iterations and 0.24 seconds. The first iteration is simply solving the deterministic problem, then identifying a violating parameter realization. Over the course of the solve execution, PyROS identified two violating parameter realizations to add back to the master problem before terminating with status `pyrosTerminationCondition.robust_optimal`.

In the case of adjustable second-stage degrees of freedom, PyROS implements polynomial order 0, 1, and 2 decision rules via the option `decision_rule_order`. As noted in Section 7.3.2.2, PyROS supports decision rules representing the complete homogeneous symmetric polynomials of degree `decision_rule_order` in the space of the uncertain parameters. This leads to decision rule functions that are the sum of all monomials of total degree from 0,...,`decision_rule_order` in the uncertain parameters. An example of how to specify this solve command via the options dictionary is shown in Code 7.2. In this case, we specify `m.x2` as a second-stage degree of freedom via `second_stage_variables`, and we select a decision rule order of 2. We note that if a decision rule order of 0 is selected, this

reduces to solving the static approximation via constant decision rules. Additionally, if a decision rule order of 1 or 2 is selected, but no second-stage degrees of freedom are specified, PyROS does not apply the decision rule approximation.

```

1 # Call the PyROS solver with second-stage degrees of freedom
2 results = pyros_solver.solve(model = m,
3                               first_stage_variables = [m.x1],
4                               second_stage_variables = [m.x2],
5                               uncertain_params = [m.u],
6                               uncertainty_set = interval,
7                               local_solver = local_subsolver,
8                               global_solver = global_subsolver,
9                               options = {
10                                   "objective_focus": pyros.ObjectiveType.worst_case,
11                                   "solve_master_globally": True,
12                                   "decision_rule_order": 2
13                               })

```

Code 7.2: Example of how to solve a robust optimization problem with uncertain equality constraints using the PyROS solver given a deterministic Pyomo model.

When we solve the above model, we retrieved the same optimal solution as before. For this small example, we see that there is no improvement in robust cost with second-stage recourse. PyROS is able to solve this model in 4 iterations and 1.17 seconds.

To illustrate the PyROS capability of handling general nonlinear problems featuring equality constraints, we augment the formulation above and instead solve the model shown in Equations 7.4a – 7.4c.

$$\min_{x \geq 0} (x_1 - 4)^2 + (x_2 - 1)^2 \quad (7.4a)$$

$$\text{s.t. } \sqrt{u}x_1 - ux_2 \leq 2 \quad \forall u \in \left[\frac{1}{4}, 2\right] \quad (7.4b)$$

$$u^2(x_2 - 1) + u(x_1^3 + 0.5) - 5ux_1x_2 + u(x_1 + 2) = 0 \quad \forall u \in \left[\frac{1}{4}, 2\right] \quad (7.4c)$$

In this case, we have an additional equality constraint shown in Equation 7.4c. This uncertain equality constraint is non-linear in the uncertain parameter u and the variable x_1 . In the notation of the robust counterpart, this equality constraint is of the form $h(x, q) = 0 \forall q \in \mathcal{Q}$ and represents a more general uncertain equality, as it does not represent a state variable definition. In PyROS, this equality is handled via coefficient matching, as outlined in Section 7.2.1.

A constraint of the form $h(x, q) = 0 \forall q \in \mathcal{Q}$ restricts the feasible set. In order to be satisfied for all uncertain parameter realizations, the coefficient terms on uncertain parameters must go to zero in order for the constraint to be trivially satisfied. In PyROS, coefficient matching is used to replace equations of the form $h(x, q) = 0 \forall q \in \mathcal{Q}$ by grouping

coefficients of polynomial powers of uncertain parameters q to create a new set of constraints $\tilde{h}(x) = 0$. In the master problem, the certain equalities $\tilde{h}(x) = 0 \in \mathcal{X}$ replace the uncertain equalities $h(x, q) = 0 \forall q \in \mathcal{Q}$. In the separation problem, this sort of equality constraint need not be carried through to the separation formulation, since it is not a state variable definition. In this section, we also note that PyROS can execute the coefficient matching procedure for constraints with second stage variables $h(x, z, q) = 0$, although it is not shown in this example.

In the example shown in Code 7.4, PyROS identifies the following $\tilde{h}(x) = 0$ constraints:

$$x_1^3 - 5x_1x_2 + x_1 + 2.5 = 0 \quad (7.5a)$$

$$x_2 - 1 = 0 \quad (7.5b)$$

Equation 7.5a represents the coefficients of u terms, while Equation 7.5b represents the coefficients of u^2 terms. It is immediately apparent from Equation 7.5b that $x_2 := 1$. Given this, the cubic equation in 7.5a has three real roots at -2.26, 0.72, and 1.54. Each of these solutions for x_1 are feasible in light of Constraint 7.4b. Therefore, the optimizer selects the value which minimizes the objective, leading to $x_1 := 1.54$. This leads to a final objective value of 6.03 and PyROS solves the problem at the first iteration in 0.22 seconds. The extra restrictions imposed by the uncertain equality constraint leads to a significant objective increase here. However, this solution satisfies Equation 7.4b for all realization of u , and therefore represents the robust optimal solution. If 7.4b was not robustly satisfiable, PyROS would instead return with a proven robust infeasible status. The benefit of this coefficient matching procedure is that PyROS may more quickly identify robust infeasible problems or robust optimal solutions due to the additional constraints.

```

1 # Import the PyROS module
2 import pyomo.contrib.pyros as pyros
3
4 # Specify (lb, ub) tuples
5 parameter_bounds = [(0.25, 2), (0.5, 1.5)]
6
7 # Define the uncertainty set
8 box_set = pyros.BoxSet(bounds = parameter_bounds)

```

Code 7.3: Example of how to solve a robust optimization problem with uncertain equality constraints using the PyROS solver given a deterministic Pyomo model.

```

1 # Import the Pyomo modeling environment and the PyROS module
2 from pyomo.environ import *
3 import pyomo.contrib.pyros as pyros
4
5 # Write the deterministic Pyomo model
6 m = ConcreteModel()
7 m.x1 = Var(initialize = 0, bounds = (0, None))
8 m.x2 = Var(initialize = 0, bounds = (0, None))
9 m.u = Param(initialize = 1.125, mutable = True)
10

```

```

11 m.con = Constraint(expr = sqrt(m.u) * m.x1 - m.u * m.x2 <= 2)
12 m.eq_con = Constraint(expr = m.u**2 * (m.x2 - 1) + m.u * (m.x1**3 - 0.5) + m.u * (m.x1 - 2)
13 == 0)
14
15 # Define the uncertainty set
16 interval = pyros.BoxSet(bounds = [(0.25, 2)])
17
18 # Instantiate the PyROS solver
19 pyros_solver = SolverFactory("pyros")
20
21 # Define subsolvers utilized in the algorithm
22 local_subsolver = SolverFactory("ipopt")
23 global_subsolver = SolverFactory("baron")
24
25 # Call the PyROS solver
26 results = pyros_solver.solve(model = m,
27                               first_stage_variables = [m.x1, m.x2],
28                               second_stage_variables = [],
29                               uncertain_params = [m.u],
30                               uncertainty_set = interval,
31                               local_solver = local_subsolver,
32                               global_solver = global_subsolver,
33                               options = {
34                                   "objective_focus": pyros.ObjectiveType.worst_case,
35                                   "solve_master_globally": True
36                               })

```

Code 7.4: Example of how to solve a robust optimization problem with uncertain equality constraints using the PyROS solver given a deterministic Pyomo model.

Additional PyROS user options can be found in the Pyomo documentation https://pyomo.readthedocs.io/en/latest/contributed_packages/pyros.html.

7.4 TRACTABILITY AND PERFORMANCE

A comprehensive set of tests are presented here to validate and qualify PyROS solver performance on representative problems.

A library of 6,264 test instances was derived from the base models specified in Table 7.3. The test instances possess an array of different uncertainty set types and sizes, different degree of freedom partitions between first- and second-stage decisions, and different decision rule relationships. The procedure for deriving these test instances is outlined in Appendix 7.6.1.

Model Name	No. Variables	No. State Variables	No. Parameter Data	No. Constraints	No. Inequality Constraints	No. Equality Constraints	Nonlinear Objective	Notes
<i>optctrl</i>	32	20	62	21	1	20	✓	Quadratic objective and (in)equalities
353	5	2	17	4	2	2		Linear objective, quadratic equality
<i>lewisopt</i>	7	3	6	9	6	3	✓	Quadratic objective, cubic terms in inequalities
<i>huverly</i>	12	7	4	9	2	7		Bilinear terms in equalities
<i>s381</i>	13	1	41	4	3	1		Linear objective and constraints
<i>s382</i>	13	1	71	4	3	1		Square root and quadratic terms in constraints

Table 7.3: Details regarding models used to derive robust optimization test problems for benchmarking PyROS.

The base models used to create the test instances were chosen to represent an array of nonlinear characteristics, as shown in the *Notes* column. The models also vary in number of degrees of freedom, number of potentially uncertain parameter data, and number of state variables.

Some example statistics regarding the test instances are shown in Table 7.4. This information is ultimately passed as arguments to construct `UncertaintySet` object (see Appendix 7.6.1.2) or as arguments to the `PyROS solve` command.

Model Name	Uncertainty Set Type	Uncertainty Set Description	First-stage DOF	Second-stage DOF	Uncertain Parameters	Decision rule order
$s381$	BoxSet	+ / - 15% nominal	x_1, x_2, x_3	x_4, \dots, x_{12}	p_0, p_{10}	0
$s381$	BoxSet	+ / - 15% nominal	x_1, x_2, x_3	x_4, \dots, x_{12}	p_0, p_{10}	1
$s381$	BoxSet	+ / - 15% nominal	x_1, x_2, x_3	x_4, \dots, x_{12}	p_0, p_{10}	2
			...			
$s381$	AxisAlignedEllipsoidalSet	+ / - 20% nominal		x_1, \dots, x_{12}	$p_0, p_{10}, p_{20}, p_{29}$	0
$s381$	AxisAlignedEllipsoidalSet	+ / - 20% nominal		x_1, \dots, x_{12}	$p_0, p_{10}, p_{20}, p_{29}$	1
$s381$	AxisAlignedEllipsoidalSet	+ / - 20% nominal		x_1, \dots, x_{12}	$p_0, p_{10}, p_{20}, p_{29}$	2

Table 7.4: Example instance information for derived benchmark problems from base problem $s381$.

We solved each of the model instances shown here using PyROS with BARON 21.1.3 with CPLEX 12.10 as the LP sub-solver. The PyROS runs were conducted with a time limit of 300 seconds and a robust feasibility tolerance of 1×10^{-3} . All computational experiments were conducted on an a single thread of an Intel(R) Xeon(R) CPU E5-2687Wv3 @ 3.10GHz with 64GB of RAM.

A summary of PyROS return statuses across all the instances is shown in Table 7.5. Given the total number of instances solved, PyROS can identify robust optimal solutions to 87% of the instances. From the results, we can see that the *haverly* instances featured more time out and sub-solver error statuses. For these small problems, we do not expect time out instances to arise due to increasing master problem size at each iteration. Instances which time out may be due to numerical sub-solver difficulties that arise due to bilinear terms and other complex, compound polynomials, or due to the selection of robust feasibility tolerance which leads to stalling algorithmic progress.

We note that at the time these results were collected, some of these time out and error statuses are known to arise due to existing Pyomo/BARON solver interface issues due to spurious constraints written to the model file. We also highlight that across the models tested, the average CPU time is low relative to the time limit. This is because the vast majority of PyROS instances terminate quickly and in very few iterations. The pathological instances that time out at the time limit are far fewer than those which solve almost instantaneously.

From the suite of results, we will highlight some interesting outcomes with respect to hierarchical uncertainty sets. For example, the box sets utilized for each test problem are designed such that $Q_X^c = \{q \in R^n : q^\ell \leq q \leq q^u : q^\ell := [q^0 - cq^0], q^u := [q^0 + cq^0] \forall c \in \{0, 0.15, 0.30\}\}$. Therefore, we naturally have $Q_X^0 \subset Q_X^{15} \subset Q_X^{30}$. In Table 7.6, we show how the value of the robust worst-case objective value ζ^* evolves for a particular instance of the $s353$ test problem.

Model Name	No. Robust Opt./Feas.	No. Time Outs	No. Sub-solver Error	Average CPU Time (s)	Average No. Iterations
<i>optcntrl</i>	1,060	16	4	42.4	2.0
<i>s353</i>	846	18	0	9.84	1.3
<i>lewispol</i>	1,072	5	3	21.1	1.1
<i>haverly</i>	918	83	79	58.6	3.3
<i>s381</i>	1073	5	2	22.9	2.2
<i>s382</i>	990	67	23	40.4	2.4
<i>hydro</i>	757	200	112	118.2	3.8
<i>optmass</i>	951	127	2	59.6	2.1
<i>hydrothermal</i>	601	384	94	175.4	1.9
<i>himmelp6</i>	772	9	83	51.0	2.1
Total	9,040	914	402		

Table 7.5: Overall performance statistics (statuses at termination and average time and iterations) for benchmark problems solved via PyROS.

Model Name	Uncertainty Set Type	Uncertainty Set Description	First-stage DOF	Second-stage DOF	Uncertain Parameters	Decision rule order	ζ^*
<i>s353</i>	BoxSet	+ / - 0% nominal	x_1	x_2, x_3	p_0, p_2, p_4, p_5, p_8	0	-39.9
<i>s353</i>	BoxSet	+ / - 15% nominal	x_1	x_2, x_3	p_0, p_2, p_4, p_5, p_8	0	-37.7
<i>s353</i>	BoxSet	+ / - 30% nominal	x_1	x_2, x_3	p_0, p_2, p_4, p_5, p_8	0	-35.5

Table 7.6: Robust worst-case objective values ζ^* for 5-D box uncertainty sets for a 353 problem instance.

In the deterministic base model for *s353*, the optimal objective value is -39.9 , as shown in the tabulated results. When we consider the *price-of-robustness* for this hierarchy of box uncertainty sets, we see that $\zeta^{*,0} \leq \zeta^{*,15} \leq \zeta^{*,30}$. This is expected, as increasing robustness by optimizing in light of a larger uncertainty set incurs an increase in the final robust objective value. This illustrates an additional utility in PyROS for simple price-of-robustness analyses via hierarchical uncertainty sets.

7.5 CONCLUSIONS

In this chapter we introduced PyROS, a Python-based robust optimization solver in Pyomo for solving nonlinear, two-stage robust optimization problems. We showcased the facile use of the robust optimization solver through the Pyomo solver interface and outlined the set of options available to the user. PyROS has been shown to be an effective solver for identifying robust solutions to general nonlinear robust optimization problems. The automatic robust optimization capabilities in PyROS will allow owners of nonlinear uncertain optimization models to study various topics of interest, such as the price of robustness and the effects of uncertainty set selection and recourse flexibility.

7.6 APPENDIX

7.6.1 Construction of Benchmark Problems

Because there is no test suite available for general two-stage nonlinear robust optimization problems, the instances used in the benchmarking study had to be generated from the initial problems listed in Table 7.3. First, an analysis was done to determine which variables in each model constitute a degree of freedom (DOF) and which are state variables. Given the set of free variables, we generate 5 instances with 5 partitions between first- and second-stage degrees of freedom. We split the degrees of freedom by selecting a fraction n of the DOF as first-stage decision variables, where $n = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$, and the remaining DOF become second-stage. Next, a set of 5 parameters is selected to be potentially uncertain. Given this information, a suite of uncertainty sets with different geometries and sizes were devised for each of the variable partitions. More details regarding this procedure are provided here.

7.6.1.1 Selecting Uncertain Parameters

From the set of numerical parameter data within the deterministic model, a subset of 5 parameters was chosen to be potentially uncertain. For consistency between each problem instance, the following set of rules for selecting these parameters was followed:

First, select a parameter from the objective. Second, select a parameter from a non-linear (if any exist, else linear) equality constraint wherein at least 1 state variable participates. Third, select a parameter from a non-linear (if any exist, else linear) inequality constraint. Fourth, select another parameter from the objective. Fifth, select another parameter from a linear (if any exist, else non-linear) equality constraint.

In the case where there are no equality constraints, the second parameter is chosen from a non-linear (if any exist, else linear) inequality. In the case that there are no parameters in the objective, the first and fourth parameters are chosen from either a non-linear (if any exist, else linear) inequality constraint.

Given the 5 potentially uncertain parameters, uncertainty sets ranging from 1- to 5-dimensions were devised for each PyROS UncertaintySet type. These are outlined in Table 7.7. In the case of FactorModelSet uncertainty sets, *all* parameter data in the model was considered uncertain. This way, the dimensionality of the space of the uncertain parameters is significantly larger than that of the individual factors.

We note that since the scale of the numerical parameter data within each problem varied, we normalize all parameter data to a value of 1 for relative scaling of the uncertainty sets.

7.6.1.2 Uncertainty Set Construction for Benchmark Instances

In this section we describe the uncertainty set construction for the benchmarking study in Section 7.4 wherein all uncertain parameters q are normalized to a nominal value of 1, i.e. $q^0 = [1]^n$. Each set is defined for uncertain parameters q with 2, 3, 4, and 5 dimensions. The relevant set information is given in Table 7.4.

Set Type	Set Parameter Definitions
BoxSet	$q^l := [q^0 - cq^0], q^u := [q^0 + cq^0], \forall c \in \{0.15, 0.30\}$
CardinalitySet	$\Gamma := \begin{cases} \lceil \frac{ q }{2} \rceil & q \geq 2 \\ \lceil \frac{ q }{3} \rceil & q \geq 3 \end{cases}, \hat{q} := [0.3q^0]$
BudgetSet	$\begin{cases} L := 1, b_\ell := [(1 + 0.1) \sum_{i=1}^n q_i^0], B_\ell := \{1, \dots, n\} & q \geq 2 \\ L := \lceil \frac{ q }{2} \rceil, b_\ell := 2[q^0 + 0.1q^0], B_\ell := B_2^{ii} & q \geq 3 \end{cases}$
FactorModelSet	$\beta \in \{0, 0.25, 0.5, 0.75, 1\}, F \in \{3, 4, 5\}$ $\Psi \sim \mathcal{N}_{n \times F}(\mu, \sigma^2) : 0.1 \leq \sum_{j=1}^F \Psi_{i,j} \leq 0.2 \forall i \in \{1, \dots, n\}$
PolyhedralSet	$Q_p := \{q \in \mathbb{R}^n : q_i \geq q_i^0 \forall i \in \{1, \dots, n\}, \sum_i^n (q_i - q_i^0) \leq (q^0 + 0.25q^0)\}$
AxisAlignedEllipsoidalSet	$\alpha := [0.2]^n$
EllipsoidalSet	See below.
DiscreteScenarioSet	$S \in \{10, 15, 20\}$ $Q_D^S = \{q^s \sim \mathcal{N}_n(\mu, \sigma^2) : q^0 - 0.3q^0 \leq q \leq q^0 + 0.3q^0, s = 0, \dots, S\} : Q_D^{10} \subset Q_D^{15} \subset Q_D^{20}$
IntersectionSet	$Q_I = \{q \in \mathbb{R}^n : q \in Q_X \cap Q_A\}$ $Q_X := \{q \in \mathbb{R}^n : q^0 - 0.15q^0 \leq q \leq q^0 + 0.15q^0\}, Q_A := \left\{q \in \mathbb{R}^n : \sum_{i=1}^n \left(\frac{q_i - q_i^0}{0.2}\right)^2 \leq 1\right\}$

Table 7.7: Uncertainty set information to generate the uncertainty sets used in the PyROS benchmarking study.

To construct the non-axis aligned ellipsoidal uncertainty sets in Table 7.7, the following protocol was followed to determine the shape matrices, P .

1. $|q| := 2, \alpha = [0.30, 0.10], 45^\circ$ rotation applied in $q_1 - q_2$ plane.
2. $|q| := 3, \alpha = [0.30, 0.20, 0.10], 45^\circ$ rotations applied in $q_1 - q_2, q_1 - q_3$ planes.
3. $|q| := 4, \alpha = [0.30, 0.20, 0.20, 0.10], 45^\circ$ rotations applied in $q_1 - q_2, q_1 - q_3, q_2 - q_3$ planes.
4. $|q| := 5, \alpha = [0.30, 0.25, 0.20, 0.15, 0.10], 45^\circ$ rotations applied in $q_1 - q_2, q_1 - q_3, q_2 - q_3, q_1 - q_4$ planes.

ii The set B_2 represents the set created from all combinations of 2 elements from the set $B_\ell := \{1, \dots, n\}$.

For each dimension noted in the list below, the vector a represents the half-lengths for the initial axis-aligned ellipsoid. These values are used to construct a diagonal covariance matrix. A sequence of 45° rotations was applied via multiplying the corresponding covariance matrix with an appropriate rotation matrix. This results in the shape matrix for the rotated ellipsoid.

7.6.2 PyROS Termination Conditions

Table 7.8 in this section outlines the PyROS solver return statuses and what conditions must be met for the status to be returned.

PyROS Return Status	Conditions When Status Is Returned
robust_optimal	objective_focus=worst_case AND solve_masters_globally=True AND no violations in global separation
robust_feasible	(objective_focus=nominal OR solve_masters_globally=False) AND no violations in global separation
robust_infeasible	Master problem subsolver returned infeasible
max_iter	Exceeded max_iter GRCS iterations
time_out	Exceeded pyros_time_limit seconds, including time spent by user-supplied solver(s)
subsolver_error	User-supplied solver(s) did not return acceptable status (see Table 7.9)

Table 7.8: PyROS Return Statuses.

7.6.3 Pyomo Subsolver Statuses in PyROS

Table 7.9 in this section explains how native Pyomo solver return codes are handled within the PyROS solver. All PyROS sub-solvers are Pyomo solver objects which return specific statuses that are interpreted by PyROS via the tabulated information below.

Pyomo Termination Condition of User-Supplied Solver	Pyomo Solver Status	PyROS Master Action	PyROS Separation Action
optimal	ok		Accept solution
globallyOptimal	ok		Accept solution
locallyOptimal	ok		Accept solution
feasible	ok	Try next backup solver*	If $g(.) > 0$, accept solution. Else, try next backup solver.*
maxTimeLimit	ok		Try next backup solver*
maxIterations	ok		Try next backup solver*
maxEvaluations	ok		Try next backup solver*
minStepLength	ok		Try next backup solver*
minFunctionValue	ok		Try next backup solver*
other	ok		Try next backup solver*
unbounded	warning		Try next backup solver*
infeasible	warning	Return robust_infeasible	Try next backup solver*
infeasibleOrUnbounded	warning		Try next backup solver*
invalidProblem	warning		Try next backup solver*
intermediateNonInteger	warning		Try next backup solver*
noSolution	warning		Try next backup solver*
solverFailure	error		Try next backup solver*
internalSolverError	error		Try next backup solver*
error	error		Try next backup solver*
unknown	unknown		Try next backup solver*
userInterrupt	aborted		Return subsolver_error
resourceInterrupt	aborted		Return subsolver_error
licensingProblems	aborted		Return subsolver_error

* If no backup solver available, return subsolver_error.

Table 7.9: Mapping Pyomo sub-solver termination conditions to PyROS master and separation problem actions.

NOTES

- ¹AIMMS. See <https://www.aimms.com>
- ²JuMPeR. See <https://github.com/IainNZ/JuMPeR.jl>
- ³ROC++. See <https://sites.google.com/usc.edu/robust-opt-cpp/>
- ⁴ROME. See <https://robustopt.com/resources.html>
- ⁵RSOME. See <https://www.rsomerso.com/>
- ⁶YALMIP. See <https://yalmip.github.io/>
- ⁷SIPAMPL. See <http://www.norg.uminho.pt/aivaz/sipampl.html>
- ⁸ROModel. See <https://github.com/cog-imperial/romodel>

CONCLUSIONS AND FUTURE WORK

In this thesis, we have proposed mathematical optimization approaches for the study of nanocluster morphology. Using this approach, we were able to identify highly stable, unintuitive nanocluster morphologies for small transition metal nanoclusters.

We also propose extensions to existing robust optimization methods in order to address general uncertain nonlinear optimization problems. We implement our proposed method in a general solver called PyROS. In doing so, we have lowered the barrier to researchers for applying robust optimization for general nonlinear programming problems. Here, we summarize the key contributions of the work presented in this thesis and then present several future research directions.

8.1 CONTRIBUTIONS

In Chapter 2, we proposed a mathematical optimization approach for identifying optimally cohesive transition metal nanoclusters.

- We devised a discrete, three-dimensional design space called a *canvas* to represent crystallographic arrangements of atoms in a nanocluster.
- A structure-function relationship between cohesive energy and local atom coordination was identified as mixed-integer linear representable.
- A mixed-integer linear programming model to identify arrangements of atoms in a canvas which maximizes the total cohesive energy of the resulting nanocluster was devised and solved to identify optimal nanocluster geometries for various numbers of atoms.

In Chapter 3, we identified metal-specific cohesive energy functions via regression with density-functional theory data, as well as new optimal nanocluster geometries at certain sizes.

- By comparing with density-functional theory evaluations of cohesive energy, we showed that the square root bond-cutting model of

cohesive energy typically overestimates energetic contributions for an array of transition metals.

- We utilized the density-functional theory predictions to regress corrected, metal-specific structure-function relationships between coordination number and cohesive energy.
- We embedded our corrected cohesive energy functions as objective functions in our nanocluster optimization model and identified new optimal geometries for certain metals at various sizes.

In Chapter 4, we proposed a preliminary mathematical optimization approach and formulation for discrete nanocluster optimization with off-lattice locations.

- We acknowledged the need to capture non-ideal inter-atomic distances between atoms of relaxed or non-crystallographic nanoclusters in our mixed-integer linear programming modeling framework.
- We proposed an off-lattice canvas for specifying a set of finite, non-ideal lattice locations for each atom in the nanocluster.
- We drafted a mixed-integer linear programming model for identifying maximally cohesive nanoclusters given an off-lattice canvas.

In Chapters 5 and 6, we identified a need for generally applicable methods for applying robust optimization methodology to process systems engineering-type problems. We then provided three process systems engineering case studies wherein we utilized the generalized robust cutting-set algorithm to identify robust solutions.

- We first proposed a generalized robust cutting-set algorithm to identify solutions for two-stage nonlinear robust optimization models.
- We proposed general nonlinear decision rules for improving recourse flexibility in the adjustable robust optimization framework.
- We outlined a general methodology for evaluating robust solution quality by calculating second-stage variable and objective expectations.
- We provide several case studies to show that the generalized robust cutting-set algorithm can identify robust solutions in light of postulated uncertainty sets for complex, nonlinear process models.

Finally, in Chapter 7, we have shown how the generalized robust cutting-set algorithm has been implemented in the algebraic modeling language, Pyomo, to provide generic robust optimization solver capability.

- We outlined the PyROS solver capabilities and other key elements, such as the provided classes for defining uncertainty sets.
- We illustrate how to use PyROS as a solver via the existing Pyomo solver interface.
- We show key performance evaluations of the solver via a suite of benchmark instances.

8.2 FUTURE DIRECTIONS

Here, I propose several areas of future research based on the work presented in this thesis.

In the context of the nanocluster design project, there is work to be done regarding how to incorporate repulsive energy term in the cohesive energy evaluation. In the case the atoms in the nanocluster are at non-equilibrium distances, repulsive effects arise. Currently, we propose this may be handled via effective coordination number and an appropriate scaling parameter, likely determined via density-functional theory evaluations. Alternatively, we may consider incorporating hard-core potentials directly into the objective function to account for the repulsive energy effects. Additional work would need to be done to consider how these can be treated in a mixed-integer linear context. Another potential research direction for the mixed-integer linear modeling of nanomaterials could be to work with larger fragments of matter as building blocks, instead of singular atoms. This can improve tractability at nanoscales of interest (more than 100 atoms). There is also an opportunity to consider the impact of symmetry breaking constraints on the MILP model tractability in the context of a branch-and-cut solver. Conducting a study to understand the impact of the existing canvas symmetry breaking constraints will motivate improving these constraints to further reduce the number of isomorphically equivalent solutions explored.

Furthermore, there is an opportunity to consider discrete nanocluster or nanomaterial optimization under epistemic uncertainty. Work by Yin et al. [141] has shown how square root bond-cutting models of cohesive energy can be used to identify optimally cohesive bimetallic nanoclusters. In this model of cohesive energy, data for the dimer bond dissociation energies (BDE) is used. However, these data have uncertainty associated with them that then factor into the calculation of bond weighting factors. To identify optimally cohesive bimetallic nanoclusters in light of this uncertainty, it is possible to pose the bimetallic nanocluster design problem as a robust optimization task wherein the discrete decisions to place atoms in the canvas are the primary decision made under BDE uncertainty. This framework may be posed as a single-stage decision making task, in which

no recourse is permitted, or a multi-stage decision making task in which uncertainty is revealed progressively and recourse decisions are made in response. For this research direction, it is important to understand the nature of the uncertainty present in the model and to pose meaningful uncertainty sets, as well as which decision making framework is most suitable.

Finally, it is worth noting that the MILP modeling framework can be extended to identify optimal nanocluster geometries against any structure-function relationships of interest. The key research task in this context is to identify an MILP representable structure function relationship. By employing a similar work-flow as described in this Thesis, we can use density-functional theory data to regress surrogate models with coordination number-based descriptors. Such a surrogate model can then be used to cast and solve an MILP formulation for optimal structures.

The generalized robust cutting-set algorithm also has many natural extensions to be considered in the future, as well as some efficiencies to consider. For example, we know that the master problems grow at each iteration, but there is special structure in the constraints because they are effectively duplicated at every uncertain parameter realization. This special structure could be exploited to expedite the identification of master solutions. Additional extensions may arise when considering a mixed-integer context. There are opportunities for supporting discrete variables in the first-stage, second-stage, or state variables of the deterministic problem. There is likely much more to consider regarding other algorithmic implications for mixed-integer support in the generalized robust cutting-set algorithm. However, we outline several general ideas here:

In the case that first-stage variables are mixed-integer, the GRCS master problems simply become mixed-integer optimization problems which require appropriate optimization solvers. The separation problems are unchanged, as first-stage variables are fixed in separation. The case that second-stage variables are binary is slightly different. Decision rule functions utilized must be binary-valued to enforce the logical nature of the second-stage variable domains in both the master and separation subproblems. The case that state variables are binary can be thought of conceptually as representing *model* uncertainty. The key detail in this case would be to ensure that all binary state variables map to 0-1 values for all uncertain parameter realizations in the postulated uncertainty set.

There is also an opportunity to improve the tractability of sequential master problems in the GRCS by considering the special structure of the master problems. This structure is illustrated in Figure 8.1. In the first box, we show the constraints for a master problem at iteration k , MP_k . In solving this master problem, we identify feasible first-stage variables x^* and d^* . The first-stage variable values are used in the following separation

problem, SP_k , to identify a violating parameter realization q^{k*} . Solving SP_k also leads to feasible values for state and second-stage variables, y^{k*} and z^{k*} , in light of the previous optimal design (x^*, d^*) and violation q^{k*} . When we add the k^* violation to the following master problem MP_{k+1}^{init} , all constraints are feasible in light of the MP_k and SP_k solutions, excluding the inequality constraints that led to violations in SP_k . These infeasible inequalities are shown in the red box in 8.1. Therefore, there is potential to utilize slack variables or an objective penalty to drive MP_{k+1}^{init} to a feasible initial point. This feasible initial point for the master problem can improve the numerical performance of the subsolvers used in the GRCS.

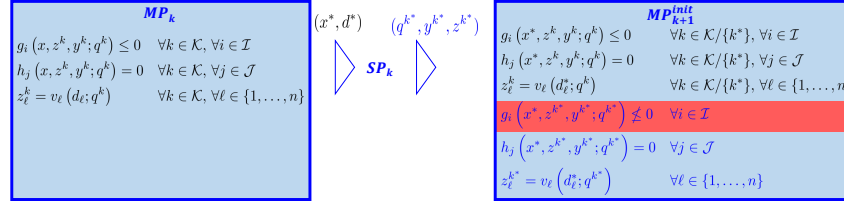


Figure 8.1: Illustration of how initialization of master problems (MP_{k+1}) in the GRCS can be improved based on information from previous master (MP_k) and separation problems (SP_k).

Additionally, there should be targeted efforts made to reduce the rate at which master problems grow in the constraints and variables. This will become especially important for large-scale deterministic models. One possible approach may be to update which q^k are part of the master problem at each iteration. It is possible that, after some iterations, a given realization q^k being explicitly accounted for in the master problem is redundant in light of another realization in the set \mathcal{K} . This means that all variables y^k, z^k and constraints h_j, g_i, v_ℓ for that given k may be removed. Performing some “clean-up” of the master problems as the algorithm progresses may abet the model growth, at the cost of some additional computations to determine redundant q^k .

We also propose a few efficiencies that may be added to PyROS. Currently, PyROS assumes that both a local and global subsolver are required to prove robust optimality. This is because we make no assumptions regarding the convexity of the subproblems solved in PyROS. However, with automatic model structure identification, as is available with the Pyomo package SUSPECT developed by Ceccon, Siirola, and Misener [19], PyROS could detect when global optimization solvers are required for solving both master and separation subproblems. This has the potential to improve numerical performance in PyROS. Additionally, the concept of general de-

cision rules could be explored in the context of PyROS. Currently, PyROS supports constant, affine, and quadratic decision rule relationships. There is an additional opportunity to devise a targeted study regarding the impact of decision rule flexibility on closing the adaptivity gap in two-stage robust optimization problems. This could be conducted in the context of increasing polynomial order decision rule functions, or with general nonlinear decision rule relationships. PyROS may be modified to support other nonlinear decision rule functions to improve the approximation of the fully adaptive case.

BIBLIOGRAPHY

- Acevedo, J. and E. N. Pistikopoulos (1998). "Stochastic optimization based algorithms for process synthesis under uncertainty." *Computers & Chemical Engineering* 22.4-5, pp. 647–671 (cit. on p. 4).
- Aiken III, J. D. and R. G. Finke (1999). "A review of modern transition-metal nanoclusters: their synthesis, characterization, and applications in catalysis." *Journal of Molecular Catalysis A: Chemical* 145.1-2, pp. 1–44 (cit. on p. 1).
- Atamtürk, A. and V. Narayanan (2007). "Cuts for conic mixed-integer programming." In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer, pp. 16–29 (cit. on p. 13).
- Austin, N., J. Johnson, and G. Mpourmpakis (2015). "Au₁₃: CO adsorbs, nanoparticle responds." *Journal of Physical Chemistry C* 119, pp. 18196–18202 (cit. on p. 27).
- Avraamidou, S. and E. N. Pistikopoulos (2020). "Adjustable robust optimization through multi-parametric programming." *Optimization Letters* 14.4, pp. 873–887 (cit. on p. 41).
- Baletto, F. (2018). "Structural properties of sub nanometer metallic clusters." *Journal of Physics: Condensed Matter* (cit. on p. 1).
- Baletto, F. and R. Ferrando (2005). "Structural properties of nanoclusters: Energetic, thermodynamic, and kinetic effects." *Reviews of modern physics* 77.1, p. 371 (cit. on pp. 1, 25, 26, 33).
- Barcaro, G., L. Sementa, and A. Fortunelli (2014). "A grouping approach to homotop global optimization in alloy nanoparticles." *Phys. Chem. Chem. Phys.* 16 (44), pp. 24256–24265 (cit. on p. 2).
- Bell, A. T. (2003). "The impact of nanoscience on heterogeneous catalysis." *Science* 299.5613, pp. 1688–1691 (cit. on p. 1).
- Ben-Tal, A., A. Goryashko, E. Guslitzer, and A. Nemirovski (2004). "Adjustable robust solutions of uncertain linear programs." *Mathematical Programming* 99.2, pp. 351–376 (cit. on pp. 5, 38, 41).
- Benson, H. Y. and Ü. Sağlam (2013). "Mixed-integer second-order cone programming: A survey." In: *Theory Driven by Influential Applications*. INFORMS, pp. 13–36 (cit. on p. 13).
- Bertsimas, D., D. A. Iancu, and P. A. Parrilo (2011). "A hierarchy of near-optimal policies for multistage adaptive optimization." *IEEE Transactions on Automatic Control* 56.12, pp. 2809–2824 (cit. on p. 47).

- Bertsimas, D. and Y. Ng (2019). "Robust and stochastic formulations for ambulance deployment and dispatch." *European Journal of Operational Research* 279.2, pp. 557–571 (cit. on p. 94).
- Bertsimas, D., O. Nohadani, and K. M. Teo (2010). "Nonconvex robust optimization for problems with constraints." *INFORMS Journal on Computing* 22.1, pp. 44–58 (cit. on p. 4).
- Bruni, M. E., L. D. P. Pugliese, P. Beraldi, and F. Guerriero (2017). "An adjustable robust optimization model for the resource-constrained project scheduling problem with uncertain activity durations." *Omega* 71, pp. 66–84 (cit. on p. 38).
- Buendía, F., J. A. Vargas, M. R. Beltrán, J. B. Davis, and R. L. Johnston (2016). "A comparative study of Au_mRh_n ($4 \leq m+n \leq 6$) clusters in the gas phase versus those deposited on (100) MgO." *Physical Chemistry Chemical Physics* 18.32, pp. 22122–22128 (cit. on p. 2).
- Bynum, M. L. et al. (2021). *Pyomo—Optimization Modeling in Python*. Vol. 67. Springer Nature (cit. on p. 98).
- Cai, Y., Y. Guo, and J. Liu (2020). "Geometric effect of Au nanoclusters on room temperature CO oxidation." *Chemical Communications* 56.6, pp. 876–879 (cit. on p. 7).
- Ceccon, F., J. D. Sirola, and R. Misener (2020). "SUSPECT: MINLP special structure detector for Pyomo." *Optimization Letters* 14.4, pp. 801–814 (cit. on p. 122).
- Cerrillo-Briones, I. M. and L. A. Ricardez-Sandoval (2019). "Robust optimization of a post-combustion CO₂ capture absorber column under process uncertainty." *Chemical Engineering Research and Design* 144, pp. 386–396 (cit. on p. 68).
- Che, G., B. B. Lakshmi, C. R. Martin, and E. R. Fisher (1999). "Metal-nanocluster-filled carbon nanotubes: catalytic properties and possible applications in electrochemical energy storage and production." *Langmuir* 15.3, pp. 750–758 (cit. on p. 23).
- Chen, Z., M. Sim, and P. Xiong (2020). "Robust stochastic optimization made easy with RSOME." *Management Science* 66.8, pp. 3329–3339 (cit. on p. 95).
- Chen, Z. and P. Xiong (2021). "RSOME in Python: An Open-Source Package for Robust Stochastic Optimization Made Easy" (cit. on p. 95).
- Chinen, A. S. et al. (2018). "Development of a Rigorous Modeling Framework for Solvent-Based CO₂ Capture. 1. Hydraulic and Mass Transfer Models and Their Uncertainty Quantification." *Industrial & Engineering Chemistry Research* 57.31, pp. 10448–10463 (cit. on pp. 67, 68).
- Cleri, F. and V. Rosato (1993). "Tight-binding potentials for transition metals and alloys." *Physical Review B* 48.1, p. 22 (cit. on p. 8).
- Corp., I. (2017). *IBM ILOG CPLEX Optimizer 12.8.0* (cit. on p. 16).

- Darby, S., T. V. Mortimer-Jones, R. L. Johnston, and C. Roberts (2002a). "Theoretical study of Cu Au nanoalloy clusters using a genetic algorithm." *The Journal of Chemical Physics* 116.4, pp. 1536–1550 (cit. on p. 33).
- Darby, S., T. V. Mortimer-Jones, R. L. Johnston, and C. Roberts (2002b). "Theoretical study of Cu–Au nanoalloy clusters using a genetic algorithm." *The Journal of Chemical Physics* 116.4, pp. 1536–1550 (cit. on p. 2).
- De, M., P. S. Ghosh, and V. M. Rotello (2008). "Applications of nanoparticles in biology." *Advanced Materials* 20.22, pp. 4225–4241 (cit. on p. 1).
- Doye, J. P. K. and D. J. Wales (1998). "Thermodynamics of Global Optimization." *Phys. Rev. Lett.* 80 (7), pp. 1357–1360 (cit. on p. 2).
- Doye, J. P. and L. Meyer (2005). "Mapping the magic numbers in binary Lennard-Jones clusters." *Physical review letters* 95.6, p. 063401 (cit. on p. 1).
- Doye, J. P. and D. J. Wales (1997). "Structural consequences of the range of the interatomic potential a menagerie of clusters." *Journal of the Chemical Society, Faraday Transactions* 93.24, pp. 4233–4243 (cit. on p. 1).
- Duff, I. S. and J. K. Reid (1982). *MA27-a set of Fortran subroutines for solving sparse symmetric sets of linear equations*. UKAEA Atomic Energy Research Establishment (cit. on p. 57).
- Ferrando, R, A Fortunelli, and G Rossi (2005). "Quantum effects on the structure of pure and binary metallic nanoclusters." *Physical Review B* 72.8, p. 085449 (cit. on p. 9).
- Fisher, K. S. et al. (2005). *Integrating MEA regeneration with CO₂ compression and peaking to reduce CO₂ capture costs*. Tech. rep. Trimeric Corporation (cit. on p. 69).
- Fournier, R. and S. Bulusu (2013). "Closed-Shell Metal Clusters." In: *Metal Clusters and Nanoalloys*. Springer, pp. 81–103 (cit. on pp. 16, 27).
- Futschek, T, M Marsman, and J Hafner (2005). "Structural and magnetic isomers of small Pd and Rh clusters: an ab initio density functional study." *Journal of Physics: Condensed Matter* 17.38, p. 5927 (cit. on pp. 26, 33).
- Garzón, I. L. et al. (1998). "Lowest Energy Structures of Gold Nanoclusters." *Phys. Rev. Lett.* 81 (8), pp. 1600–1603 (cit. on p. 2).
- Garzon, I. L. and A. Posada-Amarillas (1996). "Structural and vibrational analysis of amorphous Au 55 clusters." *Physical Review B* 54.16, p. 11796 (cit. on p. 33).
- Garzón, I. et al. (1998). "Lowest energy structures of gold nanoclusters." *Physical review letters* 81.8, p. 1600 (cit. on p. 1).
- Goedecker, S., M. Teter, and J. Hutter (1996). "Separable dual-space Gaussian pseudopotentials." *Phys. Rev. B* 54 (3), pp. 1703–1710 (cit. on p. 23).
- Goh, J. and M. Sim (2011). "Robust optimization made easy with ROME." *Operations Research* 59.4, pp. 973–985 (cit. on p. 95).

- Gong, J. and F. You (2018). "Resilient design and operations of process systems: Nonlinear adaptive robust optimization model and algorithm for resilience analysis and enhancement." *Computers & Chemical Engineering* 116, pp. 231–252 (cit. on p. 4).
- Grossmann, I. E., B. A. Calfa, and P. Garcia-Herreros (2014). "Evolution of concepts and models for quantifying resiliency and flexibility of chemical processes." *Computers & Chemical Engineering* 70, pp. 22–34 (cit. on p. 4).
- Grossmann, I. E. and C. A. Floudas (1987). "Active constraint strategy for flexibility analysis in chemical processes." *Computers & Chemical Engineering* 11.6, pp. 675–693 (cit. on p. 4).
- Grossmann, I. E. and R. W. H. Sargent (1978). "Optimum design of chemical plants with uncertain parameters." *AIChE Journal* 24.6, pp. 1021–1028 (cit. on pp. 4, 58).
- Häberlen, O. D., S.-C. Chung, M. Stener, and N. Rösch (1997). "From clusters to bulk: A relativistic density functional investigation on a series of gold clusters Au_n , $n=6, \dots, 147$." *The Journal of chemical physics* 106.12, pp. 5189–5201 (cit. on p. 26).
- Häkkinen, H., M. Moseler, and U. Landman (2002). "Bonding in Cu, Ag, and Au clusters: relativistic effects, trends, and surprises." *Physical review letters* 89.3, p. 033401 (cit. on p. 2).
- Häkkinen, H. et al. (2004). "Symmetry and electronic structure of noble-metal nanoparticles and the role of relativity." *Physical review letters* 93.9, p. 093401 (cit. on p. 33).
- Halemane, K. P. and I. E. Grossmann (1983). "Optimal process design under uncertainty." *AIChE Journal* 29.3, pp. 425–433 (cit. on p. 63).
- Hanselman, C. L. and C. E. Gounaris (2016). "A mathematical optimization framework for the design of nanopatterned surfaces." *AIChE Journal* 62.9, pp. 3250–3263 (cit. on p. 8).
- Hart, W. E., J.-P. Watson, and D. L. Woodruff (2011). "Pyomo: modeling and solving mathematical programs in Python." *Mathematical Programming Computation* 3.3, pp. 219–260 (cit. on pp. 57, 98).
- Hart, W. E. et al. (2017). *Pyomo—optimization modeling in python*. Second. Vol. 67. Springer Science & Business Media (cit. on p. 57).
- Hartke, B. (1993). "Global geometry optimization of clusters using genetic algorithms." *The Journal of Physical Chemistry* 97.39, pp. 9973–9976 (cit. on p. 2).
- Hessen, E. T., T. Haug-Warberg, and H. F. Svendsen (2010). "The refined e-NRTL model applied to CO_2 – H_2O –alkanolamine systems." *Chemical Engineering Science* 65.11, pp. 3638–3648 (cit. on p. 67).
- Hijazi, I. A. and Y. H. Park (2010). "Structure of pure metallic nanoclusters: Monte Carlo simulation and ab initio study." *The European Physical Journal D* 59.2, pp. 215–221 (cit. on pp. 26, 27).

- Huang, D., F. Liao, S. Moles, D. Redinger, and V. Subramanian (2003). "Plastic-compatible low resistance printable gold nanoparticle conductors for flexible electronics." *Journal of the electrochemical society* 150.7, G412–G417 (cit. on p. 1).
- Hutter, J., M. Iannuzzi, F. Schiffmann, and J. VandeVondele (2014). "cp2k: atomistic simulations of condensed matter systems." *Wiley Interdisciplinary Reviews: Computational Molecular Science* 4.1, pp. 15–25 (cit. on p. 23).
- Isenberg, N. M. et al. (2021). "A generalized cutting-set approach for nonlinear robust optimization in process systems engineering." *AIChE Journal* 67.5, e17175 (cit. on p. 5).
- Janthon, P. et al. (2014). "Bulk Properties of Transition Metals: A Challenge for the Design of Universal Density Functionals." *Journal of Chemical Theory and Computation* 10.9. PMID: 26588528, pp. 3832–3839 (cit. on p. 23).
- Jenkins, P. R., B. J. Lunday, and M. J. Robbins (2020). "Robust, multi-objective optimization for the military medical evacuation location-allocation problem." *Omega* 97, p. 102088 (cit. on p. 94).
- Kammammettu, S. and Z. Li (2019). "Two-Stage Robust Optimization of Water Treatment Network Design and Operations Under Uncertainty." *Industrial & Engineering Chemistry Research* (cit. on pp. 4, 41).
- Karim, A. M. et al. (2009). "Correlating particle size and shape of supported Ru/ γ -Al₂O₃ catalysts with NH₃ decomposition activity." *Journal of the American Chemical Society* 131.34, pp. 12230–12239 (cit. on p. 1).
- Kelley Jr, J. E. (1960). "The cutting-plane method for solving convex programs." *Journal of the society for Industrial and Applied Mathematics* 8.4, pp. 703–712 (cit. on pp. 38, 54).
- Kelley, M. T., R. Baldick, and M. Baldea (2020). "Demand response scheduling under uncertainty: chance-constrained framework and application to an air separation unit." *AIChE Journal* 66.9, e16273 (cit. on p. 4).
- Kılınç-Karzan, F. (2016). "On minimal valid inequalities for mixed integer conic programs." *Mathematics of Operations Research* 41.2, pp. 477–510 (cit. on p. 13).
- Kılınç-Karzan, F. and S. Yıldız (2015). "Two-term disjunctions on the second-order cone." *Mathematical Programming* 154.1, pp. 463–491 (cit. on p. 13).
- Kim, D., J. Resasco, Y. Yu, A. M. Asiri, and P. Yang (2014). "Synergistic geometric and electronic effects for electrochemical reduction of carbon dioxide using gold–copper bimetallic nanoparticles." *Nature communications* 5, p. 4948 (cit. on p. 23).
- Koga, K., T. Ikeshoji, and K.-i. Sugawara (2004). "Size-and temperature-dependent structural transitions in gold nanoparticles." *Physical review letters* 92.11, p. 115507 (cit. on p. 33).

- Kwon, S. K. et al. (2005). "Surface energy and stress release by layer relaxation." *Phys. Rev. B* 72 (23), p. 235423 (cit. on pp. 23, 27).
- Lappas, N. H. and C. E. Gounaris (2016). "Multi-stage adjustable robust optimization for process scheduling under uncertainty." *AIChE Journal* 62.5, pp. 1646–1667 (cit. on p. 38).
- Lappas, N. H. and C. E. Gounaris (2018). "Theoretical and computational comparison of continuous-time process scheduling models for adjustable robust optimization." *AIChE Journal* 64.8, pp. 3055–3070 (cit. on p. 38).
- Lappas, N. H., L. A. Ricardez-Sandoval, R. Fukasawa, and C. E. Gounaris (2019). "Adjustable Robust Optimization for multi-tasking scheduling with reprocessing due to imperfect tasks." *Optimization and Engineering* 20.4, pp. 1117–1159 (cit. on p. 38).
- Leyffer, S., M. Menickelly, T. Munson, C. Vanaret, and S. M. Wild (2020). "A survey of nonlinear robust optimization." *INFOR: Information Systems and Operational Research* 58.2, pp. 342–373 (cit. on pp. 105, 107).
- Li, C. and I. E. Grossmann (2018). "An improved L-shaped method for two-stage convex 0–1 mixed integer nonlinear stochastic programs." *Computers & Chemical Engineering* 112, pp. 165–179 (cit. on p. 4).
- Li, J., X. Li, H.-J. Zhai, and L.-S. Wang (2003). "Au₂₀: a tetrahedral cluster." *Science* 299.5608, pp. 864–867 (cit. on pp. 23, 27).
- Löfberg, J. (2004). "YALMIP : A Toolbox for Modeling and Optimization in MATLAB." In: *In Proceedings of the CACSD Conference*. Taipei, Taiwan (cit. on p. 95).
- Löfberg, J. (2012). "Automatic robust convex programming." *Optimization Methods and Software* 27.1, pp. 115–129 (cit. on p. 95).
- Lu, Y. and W. Chen (2015). "Application of Mass Spectrometry in the Synthesis and Characterization of Metal Nanoclusters." *Analytical Chemistry* 87.21, pp. 10659–10667 (cit. on p. 1).
- Matthews, L. R., C. E. Gounaris, and I. G. Kevrekidis (2019). "Designing networks with resiliency to edge failures using two-stage robust optimization." *European Journal of Operational Research* 279.3, pp. 704–720 (cit. on p. 38).
- Matthews, L. R., Y. A. Guzman, O. Onel, A. M. Niziolek, and C. A. Floudas (2018). "Natural Gas to Liquid Transportation Fuels under Uncertainty Using Robust Optimization." *Industrial & Engineering Chemistry Research* 57.32, pp. 11112–11129 (cit. on p. 4).
- Methfessel, M., D. Hennig, and M. Scheffler (1992a). "Trends of the surface relaxations, surface energies, and work functions of the 4d transition metals." *Phys. Rev. B* 46 (8), pp. 4816–4829 (cit. on pp. 23, 25).
- Methfessel, M., D. Hennig, and M. Scheffler (1992b). "Calculated surface energies of the 4d transition metals: A study of bond-cutting models." *Applied Physics A* 55.5, pp. 442–448 (cit. on p. 8).

- Michaelian, K, N Rendón, and I. Garzón (1999). "Structure and energetics of Ni, Ag, and Au nanoclusters." *Physical Review B* 60.3, p. 2000 (cit. on p. 1).
- Miller, D. C. et al. (2018). "Next generation multi-scale process systems engineering framework." In: *Computer Aided Chemical Engineering*. Vol. 44. Elsevier, pp. 2209–2214 (cit. on p. 57).
- Misener, R. and C. Floudas (2014). "ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations." *Journal of Global Optimization* 59.2, pp. 503–526 (cit. on p. 57).
- Mores, P., N. Rodríguez, N. Scenna, and S. Mussati (2012). "CO₂ capture in power plants: Minimization of the investment and operating cost of the post-combustion process using MEA aqueous solution." *International Journal of Greenhouse Gas Control* 10, pp. 148–163 (cit. on pp. 58, 67, 69).
- Morgan, J. C., D. Bhattacharyya, C. Tong, and D. C. Miller (2015). "Uncertainty quantification of property models: Methodology and its application to CO₂-loaded aqueous MEA solutions." *AIChE Journal* 61.6, pp. 1822–1839 (cit. on p. 68).
- Morgan, J. C. et al. (2017). "Thermodynamic modeling and uncertainty quantification of CO₂-loaded aqueous MEA solutions." *Chemical Engineering Science* 168, pp. 309–324 (cit. on p. 68).
- Mottet, C, J. Goniakowski, F Baletto, R Ferrando, and G Treglia (2004). "Modeling free and supported metallic nanoclusters: structure and dynamics." *Phase Transitions* 77.1-2, pp. 101–113 (cit. on p. 36).
- Mutapcic, A. and S. Boyd (2009). "Cutting-set methods for robust convex optimization with pessimizing oracles." *Optimization Methods & Software* 24.3, pp. 381–406 (cit. on pp. 5, 38, 43, 46, 54).
- Nicholson, B., J. D. Sirola, J.-P. Watson, V. M. Zavala, and L. T. Biegler (2018a). "pyomo. dae: A modeling and automatic discretization framework for optimization with differential and algebraic equations." *Mathematical Programming Computation* 10.2, pp. 187–223 (cit. on p. 98).
- Nicholson, B., J. D. Sirola, J.-P. Watson, V. M. Zavala, and L. T. Biegler (2018b). "Pyomo.DAE: a modeling and automatic discretization framework for optimization with differential and algebraic equations." *Mathematical Programming Computation* 10.2, pp. 187–223 (cit. on p. 57).
- Perdew, J. P., K. Burke, and M. Ernzerhof (1996). "Generalized Gradient Approximation Made Simple." *Phys. Rev. Lett.* 77 (18), pp. 3865–3868 (cit. on p. 23).
- Pistikopoulos, E. N. and M. G. Ierapetritou (1995). "Novel approach for optimal process design under uncertainty." *Computers & Chemical Engineering* 19.10, pp. 1089–1110 (cit. on p. 4).
- Pyykkö, P. (2004). "Theoretical chemistry of gold." *Angewandte Chemie International Edition* 43.34, pp. 4412–4456 (cit. on p. 27).

- Pyykkö, P. (2007). "Structural properties: Magic nanoclusters of gold." *Nature nanotechnology* 2.5, p. 273 (cit. on p. 23).
- Rahal, S., Z. Li, and D. J. Papageorgiou (2019). "Proactive and Reactive Scheduling of the Steelmaking and Continuous Casting Process through Adaptive Robust Optimization." *Computers & Chemical Engineering*, p. 106658 (cit. on p. 41).
- Roberts, C., R. L. Johnston, and N. T. Wilson (2000). "A genetic algorithm for the structural optimization of Morse clusters." *Theoretical Chemistry Accounts* 104.2, pp. 123–130 (cit. on p. 1).
- Rogan, J., A. Varas, J. A. Valdivia, and M. Kiwi (2013). "A strategy to find minimal energy nanocluster structures." *Journal of computational chemistry* 34.29, pp. 2548–2556 (cit. on p. 1).
- Rooney, W. C. and L. T. Biegler (2001). "Design for model parameter uncertainty using nonlinear confidence regions." *AIChE Journal* 47.8, pp. 1794–1804 (cit. on pp. 58, 59).
- Rooney, W. C. and L. T. Biegler (2003). "Optimal process design with model parameter uncertainty and process variability." *AIChE Journal* 49.2, pp. 438–449 (cit. on p. 63).
- Sanchez, A et al. (1999). "When gold is not noble: nanoscale gold catalysts." *The Journal of Physical Chemistry A* 103.48, pp. 9573–9578 (cit. on p. 7).
- Sebetci, A. and Z. B. Güvenç (2005). "Global minima of Al N , Au N and Pt N , $N \leq 80$, clusters described by the Voter–Chen version of embedded-atom potentials." *Modelling and Simulation in Materials Science and Engineering* 13.5, p. 683 (cit. on p. 26).
- Seider, W. D., J. D. Seader, and D. R. Lewin (2009). *Product & process design principles: synthesis, analysis and evaluation*. John Wiley & Sons (cit. on p. 67).
- Shandiz, M. A. (2008). "Effective coordination number model for the size dependency of physical properties of nanocrystals." *Journal of Physics: Condensed Matter* 20.32, p. 325237 (cit. on p. 34).
- Shi, H. and F. You (2016). "A computational framework and solution algorithms for two-stage adaptive robust scheduling of batch manufacturing processes under uncertainty." *AIChE Journal* 62.3, pp. 687–703 (cit. on p. 38).
- Snyder, L. V. and M. S. Daskin (2006). "Stochastic p-robust location problems." *Iie Transactions* 38.11, pp. 971–985 (cit. on pp. 49, 103).
- Subramanyam, A., F. Mufalli, J. M. Pinto, J. Lainez-Aguirre, and C. E. Gounaris. "Robust multi-period vehicle routing under customer order uncertainty." *Optimization Research* (cit. on p. 38).
- Sutton, A. P. (1993). *Electronic structure of materials*. Clarendon Press (cit. on p. 8).

- Swaney, R. E. and I. E. Grossmann (1985). "An index for operational flexibility in chemical process design. Part I: Formulation and theory." *AIChE Journal* 31.4, pp. 621–630 (cit. on p. 4).
- Tawarmalani, M. and N. V. Sahinidis (2005). "A polyhedral branch-and-cut approach to global optimization." *Mathematical Programming* 103 (2), pp. 225–249 (cit. on pp. 57, 107).
- Taylor, M. G., N. Austin, C. E. Gounaris, and G. Mpourmpakis (2015). "Catalyst design based on morphology-and environment-dependent adsorption on metal nanoparticles." *ACS Catalysis* 5.11, pp. 6296–6301 (cit. on p. 1).
- Tejeda-Iglesias, M., N. H. Lappas, C. E. Gounaris, and L. Ricardez-Sandoval (2019). "Explicit model predictive controller under uncertainty: An adjustable robust optimization approach." *Journal of Process Control* 84, pp. 115–132 (cit. on p. 38).
- Tománek, D., S. Mukherjee, and K. H. Bennemann (1983). "Simple theory for the electronic and atomic structure of small clusters." *Phys. Rev. B* 28 (2), pp. 665–673 (cit. on pp. 8, 22, 34, 36).
- Towler, G. and R. Sinnott (2012). *Chemical engineering design: principles, practice and economics of plant and process design*. Elsevier (cit. on p. 67).
- Underwood, A. V. (1970). *Simple formula to calculate mean temperature difference* (cit. on p. 63).
- VandeVondele, J. and J. Hutter (2007). "Gaussian basis sets for accurate calculations on molecular systems in gas and condensed phases." *The Journal of Chemical Physics* 127.11, p. 114105 (cit. on p. 23).
- Vanithakumari, S. and K. Nanda (2008). "A universal relation for the cohesive energy of nanoparticles." *Physics Letters A* 372.46, pp. 6930–6934 (cit. on p. 8).
- Varvarezos, D. K., L. T. Biegler, and I. E. Grossmann (1994). "Multi-period design optimization with SQP decomposition." *Computers & Chemical Engineering* 18.7, pp. 579–595 (cit. on p. 63).
- Vayanos, P., A. Georghiou, and H. Yu (2020). "Robust optimization with decision-dependent information discovery." *arXiv preprint arXiv:2004.08490* (cit. on p. 95).
- Vayanos, P., Q. Jin, and G. Elissaios (2020). "ROC++: Robust Optimization in C++." *arXiv preprint arXiv:2006.08741* (cit. on p. 95).
- Vaz, A. I. F., E. M. Fernandes, and M. P. S. Gomes (2004). "SIPAMPL: Semi-infinite programming with AMPL." *ACM Transactions on Mathematical Software (TOMS)* 30.1, pp. 47–61 (cit. on p. 96).
- Vila, F. D., S. T. Hayashi, J. M. Moore, and J. J. Rehr (2016). "Molecular dynamics simulations of supported pt nanoparticles with a hybrid sutton-chen potential." *The Journal of Physical Chemistry C* 120.27, pp. 14883–14891 (cit. on p. 2).

- Wächter, A. and L. T. Biegler (2006). "On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming." *Mathematical Programming* 106.1, pp. 25–57 (cit. on pp. 57, 107).
- Wales, D. J. and J. P. Doye (1997). "Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms." *The Journal of Physical Chemistry A* 101.28, pp. 5111–5116 (cit. on p. 1).
- Wang, J., G. Wang, and J. Zhao (2003). "Structures and electronic properties of Cu₂₀, Ag₂₀, and Au₂₀ clusters with density functional method." *Chemical physics letters* 380.5-6, pp. 716–720 (cit. on p. 2).
- Wang, Y., L. T. Biegler, M. Patel, and J. Wassick (2020). "Robust optimization of solid-liquid batch reactors under parameter uncertainty." *Chemical Engineering Science* 212, p. 115170 (cit. on p. 4).
- Watson, J.-P., D. L. Woodruff, and W. E. Hart (2012). "PySP: modeling and solving stochastic programs in Python." *Mathematical Programming Computation* 4.2, pp. 109–149 (cit. on p. 98).
- Wei, C., C. Cheng, and C.-M. Chang (2006). "Transition between icosahedral and cuboctahedral nanoclusters of lead." *The Journal of Physical Chemistry B* 110.48, pp. 24642–24645 (cit. on p. 33).
- Weiland, R., M. Rawal, and R. Rice (1982). "Stripping of carbon dioxide from monoethanolamine solutions in a packed column." *AIChE Journal* 28.6, pp. 963–973 (cit. on p. 67).
- Wendt, M., P. Li, and G. Wozny (2002). "Nonlinear chance-constrained process optimization under uncertainty." *Industrial & Engineering Chemistry Research* 41.15, pp. 3621–3629 (cit. on p. 4).
- Wiebe, J., I. Cecilio, and R. Misener (2019). "Robust optimization for the pooling problem." *Industrial & Engineering Chemistry Research* (cit. on p. 4).
- Wiebe, J. and R. Misener (2021). "ROmodel: Modeling robust optimization problems in Pyomo." *arXiv preprint arXiv:2105.08598* (cit. on p. 96).
- Wille, L. T. and J. Vennik (1985). "Computational complexity of the ground-state determination of atomic clusters." *Journal of Physics A: Mathematical and General* 18.8, p. L419 (cit. on p. 2).
- Xiao, L. and L. Wang (2004). "From planar to three-dimensional structural transition in gold clusters and the spin-orbit coupling effect." *Chemical physics letters* 392.4, pp. 452–455 (cit. on p. 27).
- Xing, X. et al. (2016). "Insights into the geometries, electronic and magnetic properties of neutral and charged palladium clusters." *Scientific reports* 6, p. 19656 (cit. on p. 26).
- Xing, X., B. Yoon, U. Landman, and J. H. Parks (2006). "Structural evolution of Au nanoclusters: From planar to cage to tubular motifs." *Physical Review B* 74.16, p. 165423 (cit. on p. 27).

- Xiong, P., P. Jirutitijaroen, and C. Singh (2016). "A distributionally robust optimization model for unit commitment considering uncertain wind power generation." *IEEE Transactions on Power Systems* 32.1, pp. 39–49 (cit. on p. 94).
- Yan, Z., M. G. Taylor, A. Mascareno, and G. Mpourmpakis (2018). "Size-, Shape-, and Composition-Dependent Model for Metal Nanoparticle Stability Prediction." *Nano letters* 18.4, pp. 2696–2704 (cit. on pp. 9, 23).
- Yin, X. et al. (2021). "Designing Stable Bimetallic Nanoclusters via an Iterative Two-Step Optimization Approach." *Molecular Systems Design & Engineering* (cit. on pp. 3, 120).
- Yoo, E. et al. (2009). "Enhanced electrocatalytic activity of Pt subnanoclusters on graphene nanosheet surface." *Nano letters* 9.6, pp. 2255–2259 (cit. on p. 23).
- Yuan, Y., Z. Li, and B. Huang (2018). "Nonlinear robust optimization for process design." *AIChE Journal* 64.2, pp. 481–494 (cit. on pp. 4, 58, 63).
- Zhang, H., D. Tian, and J. Zhao (2008). "Structural evolution of medium-sized Pd_n (n= 15–25) clusters from density functional theory." *The Journal of chemical physics* 129.11, p. 114302 (cit. on p. 26).
- Zhang, Q., I. E. Grossmann, and R. M. Lima (2016). "On the relation between flexibility analysis and robust optimization for linear systems." *AIChE Journal* 62.9, pp. 3109–3123 (cit. on p. 38).
- Zhang, Q., M. F. Morari, I. E. Grossmann, A. Sundaramoorthy, and J. M. Pinto (2016). "An adjustable robust optimization approach to scheduling of continuous industrial processes providing interruptible load." *Computers & Chemical Engineering* 86, pp. 106–119 (cit. on p. 38).
- Zhang, X., M. Kamgarpour, A. Georghiou, P. Goulart, and J. Lygeros (2017). "Robust optimal control with adjustable uncertainty sets." *Automatica* 75, pp. 249–259 (cit. on p. 38).
- Zhang, Y. (2007). "General robust-optimization formulation for nonlinear programming." *Journal of Optimization Theory and Applications* 132.1, pp. 111–124 (cit. on p. 4).
- Zhang, Y. et al. (2009). "Rate-based process modeling study of CO₂ capture with aqueous monoethanolamine solution." *Industrial & engineering chemistry research* 48.20, pp. 9233–9246 (cit. on p. 67).
- Zhao, L., C. Ning, and F. You (2019). "Operational optimization of industrial steam systems under uncertainty using data-Driven adaptive robust optimization." *AIChE Journal* 65.7, e16500 (cit. on p. 38).
- Zhao, M. et al. (2007). "Atomistic origin, temperature dependence, and responsibilities of surface energetics: An extended broken-bond rule." *Physical Review B* 75.8, p. 085427 (cit. on p. 8).