Discrete Approximations, Relaxations, and Applications in Quadratically Constrained Quadratic Programming

Benjamin J. Beach

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Robert Hildebrand, Chair
Kimberly Ellis
Douglas Bish
Manish Bansal

Mar. 22, 2022
Blacksburg, Virginia

Keywords: Quadratically Constrained Quadratic Programming, Mixed Integer Programming approximations, Binarization

Discrete Approximations, Relaxations, and Applications in Quadratically Constrained Quadratic Programming

Benjamin J. Beach

(ABSTRACT)

We present works on theory and applications for Mixed Integer Quadratically Constrained Quadratic Programs (MIQCQP). We introduce new mixed integer programming (MIP)-based relaxation and approximation schemes for general Quadratically Constrained Quadratic Programs (QCQP's), and also study practical applications of QCQP's and Mixed-integer QCQP's (MIQCQP).

We first address a challenging tank blending and scheduling problem regarding operations for a chemical plant. We model the problem as a discrete-time nonconvex MIQCP, then approximate this model as a MILP using a discretization-based approach. We combine a rolling horizon approach with the discretization of individual chemical property specifications to deal with long scheduling horizons, time-varying quality specifications, and multiple suppliers with discrete arrival times.

Next, we study optimization methods applied to minimizing forces for poses and movements of chained Stewart platforms (SPs). These SPs are parallel mechanisms that are stiffer, and more precise, on average, than their serial counterparts at the cost of a smaller range of motion. The robot will be used in concert with several other types robots to perform complex assembly missions in space. We develop algorithms and optimization models that can efficiently decide on favorable poses and movements that reduce force loads on the robot, hence reducing wear on this machine, and allowing for a larger workspace and a greater overall payload capacity.

In the third work, we present a technique for producing valid dual bounds for nonconvex quadratic

optimization problems. The approach leverages an elegant piecewise linear approximation for univariate quadratic functions and formulate this approximation using mixed-integer programming (MIP). Combining this with a diagonal perturbation technique to convert a nonseparable quadratic function into a separable one, we present a mixed-integer convex quadratic relaxation for nonconvex quadratic optimization problems. We study the strength (or *sharpness*) of our formulation and the tightness of its approximation. We computationally demonstrate that our model outperforms existing MIP relaxations, and on hard instances can compete with state-of-the-art solvers.

Finally, we study piecewise linear relaxations for solving quadratically constrained quadratic programs (QCQP's). We introduce new relaxation methods based on univariate reformulations of nonconvex variable products, leveraging the relaxation from the third work to model each univariate quadratic term. We also extend the NMDT approach (Castro, [1]) to leverage discretization for both variables in a bilinear term, squaring the resulting precision for the same number of binary variables. We then present various results related to the relative strength of the various formulations.

Discrete Approximations, Relaxations, and Applications in Quadratically Constrained Quadratic Programming

Benjamin J. Beach

(GENERAL AUDIENCE ABSTRACT)

Quadratically Constrained Quadratic Programs (QCQP's) are a class of nonlinear optimization problems that can contain products of multiple variables in both the objective function and constraints. Nonconvex QCQP's are a particularly challenging class of QCQP's, often requiring far more sophisticated techniques and far more time to solve. These problems are useful for a wide variety of real-world applications, including pooling problems in the crude oil and chemical industries, optimal power flow problems for power companies, and 3-D motion planning for robots, to name a few. In this document, we present two works introducing new mixed-integer programming (MIP)-based relaxation and approximation schemes for general QCQP's, and two works which focus on practical applications of QCQP's and Mixed-integer QCQP's (MIQCQP).

First, we study a challenging long-horizon supply acquisition problem for a chemical plant. For this problem, constraints with products of variables are required to track raw material composition from supply carriers to storage tanks to the production feed. We apply a mixed-integer nonlinear program (MIP) approximation of the problem combined with a rolling planning scheme to obtain good solutions for industry problems within a reasonable time frame.

Next, we study optimization methods applied to a robot designed as a stack of Stewart platforms (SPs), which will be used in concert with several other types robots to perform complex space missions. When chaining these SPs together, we obtain a robot that is generally stiffer more precise than a classic robot arm, enabling their potential use for a variety of purposes. Our methods

can efficiently decide on favorable poses and movements for the robot that reduce force loads on the robot, hence reducing wear on this machine, and allowing for a larger usable range of motion and a greater overall payload capacity.

Our final two works focus on MIP-based techniques for nonconvex QCQP's. In the first work, we break down the objective into an easy-to-handle term minus some squared terms. We then introduce an elegant new MIP-based approximation to handle these squared terms. We prove that this approximation has strong theoretical guarantees, then demonstrate that it is effective compared to other approximations. In the second, we directly model each variable product term using a MIP relaxation. We introduce two new formulations to do this that build on previous formulations, increasing the accuracy with the same number of integer variables. We then prove a variety of useful properties about the presented formulations, then compare them computationally on two families of problems.

Acknowledgments

This work was supported by AFOSR (grant FA9550-21-0107) and ONR (Grant N00014-20-1-2156). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research or the Air Force Office of Scientific Research. I would like to thank Eastman Chemical Company for the internship in 2018 which started this whole process, and fueled the first publication in this dissertation. I would like to thank NASA Langley (contract 80LARC20P0020) and the Northrop Grumman Undergraduate Research Experience in Industrial & Systems Engineering at Virginia Tech for their interest and support in the robotics project.

This dissertation was made possible by the help and support of many people. I thank my advisor, Dr. Robert Hildebrand, for his incredible help, guidance, patience, support, and encouragement throughout this process. His encouragement to be constantly thinking about the theoretical side of the problems and approaches studied throughout this work led to far better communication and quality in each of these works. I deeply appreciate all of our whiteboard sessions to collaborate on problems and proofs.

I thank our collaborators, Baptiste Lebreton, Kimberly Ellis, Joey Huchette, William Chapin, Erik Komendera, Lukas Hager, Robert Burlacu, and Andreas Bärmann for their many ideas and contributions to the works presented herein. I have thoroughly enjoyed working closely with these collaborators in solving these problems, and in communicating the solutions employed for each problem.

I thank Dr. Kimberly Ellis for her contributions, ideas, and support as a co-advisor for the first work.

Finally, I thank my wife, Tilly Beach, and my parents and siblings for their unyielding love and support.

Attribution

Several colleagues assisted with the writing and research for the works presented herein. We provide a brief summary of their contributions below.

Robert Hildebrand, PhD., my research advisor and committee chair, is currently an Assist Professor of Industrial and Systems Engineering at Virginia Tech. He has helped to write, re-frame, proofread, and provide direction for all manuscripts presented herein.

The first work, *An Approximate Method for the Optimization of Long-Horizon Tank Blending and Scheduling Operations*, began as an internship with Eastman Chemical Company. This manuscript has two additional collaborators: Drs. Baptiste Lebreton and Kimberly Ellis.

Baptiste Lebreton, PhD., is currently a manager in forecasting, planning and analytics systems at Eastman Chemical Company. He was my manager as an intern, and contributed by defining the original problem, providing a starting point for its solution, and directing my work throughout the internship.

Kimberly Ellis, PhD., is currently an Associate Professor of Industrial and Systems Engineering at Virginia Tech. She was my co-advisor for the first manuscript, and contributed to the framing of the research from a business perspective, and helped with guidance and editing for this work.

The second work, *Force-Controlled Pose Optimization and Trajectory Planning for Chained Stewart Platforms*, is a collaboration with the FASER Lab in the Mechanical Engineering Department at Virginia Tech. This manuscript has three additional collaborators: William Chapin, Samantha Glassner, and Dr. Erik Komendera.

William Chapin is currently a Master's student in the Computer Science Department at Virginia Tech. He is the primary co-contributor for this project, and created the core programming framework for computations and visualizations, built a physical version of our test robot, and extensively helped with writing and research direction.

Samantha Glassner is currently a PhD student in the Mechanical Engineering Department at Virginia Tech. She constructed CAD models and simulations to assist with hardware and figures.

Erik Komendera, PhD., is currently an Assistant Professor of Mechanical Engineering at Virginia Tech. As the director of FASER Lab, he has contributed resources, research direction, and editing for this work.

The third work, *Compact mixed-integer programming formulations in quadratic optimization*, is a collaboration with Dr. Joey Huchette.

Joey Huchette, PhD., is currently a software engineer on the Large Scale Optimization team at Google. He designed and performed the majority of the computational experiments in this manuscript, and generally assisted with writing and editing.

The fourth work, *A Comparison of Discretization Approaches for Mixed Integer Nonconvex Quadratic Programming*, is a collaboration with the Department of Data Science at Friedrich–Alexander University Erlangen–Nürnberg (FAU) in Germany. This manuscript has three additional collabora-

tors: Lukas Hager, Dr. Robert Burlacu, and Dr. Andreas Bärmann.

Lukas Hager is currently a PhD student in the Department of Data Science at FAU. He is the main collaborator on this work, contributing substantially to the writing, theory, and computations.

Robert Burlacu, PhD., is currently a group leader at Fraunhofer Institute for Integrated Circuits in Nürnberg. He contributed to editing and provided the ACOPF instances studied in this work.

Andreas Bärmann, PhD., is currently the Chair of Analytics & Mixed-Integer Optimization in the Department of Data Science at FAU. As the research advisor of Lucas Hager, he has contributed via research guidance and editing.

Contents

Chapter 1

Introduction

In this theses, we present an assortment of applied and theoretical works in quadratic optimization. All problems presented are nonconvex quadratically constrained quadratic programming (QCQP's) or mixed-integer QCQP (MIQCQP) problems, and most present mixed integer linear programming (MILP, or MIP)-based models for the nonlinear quadratic portions of the problems. All integer variables in this work are binary, so that the possible values are 0 or 1.

In Chapter 2, we present a manuscript titled *An Approximate Method for the Optimization of Long-Horizon Tank Blending and Scheduling Operations*. In this work, we tackle a challenging problem related to procurement optimization in a chemical plant, which requires procurement decisions to be made up to a year in advance. We model the underlying problem for this work as an MIQCQP, where binary variables are used to enforce scheduling and operational constraints and bilinear product terms are used to track chemical compositions of raw materials to be used in production. As the core model proved intractable for problem periods of longer than about a month and a half, we combine a MIP approximation approach specific to pooling-type problems with a rolling horizon planning approach to handle the long time horizons. We then computationally demonstrate the effectiveness of the approach using industry-representative data sets, and explore variations on the MIP model and on the rolling horizon planning approach.

In Chapter 3, we present a manuscript titled "Force-Controlled Pose Optimization and Trajectory Planning for Chained Stewart Platforms". In this work, we study an assembler robot comprised of four stacked Stewart Platforms. Each platform consists of two plates connected by six legs which are free to rotate about their connection joints, and is actuated by varying the length of the legs. In this work, we present a fast method for force-based pose optimization of the assembler robot. We then present a trust-region method for force-controlled trajectory planning of the robot. Finally, we computationally demonstrate that the proposed methods are effective and robust on a diverse set of target poses and motions for a test robot under a moderate load.

In Chapter 4, we present a manuscript titled "Compact mixed-integer programming formulations in quadratic optimization". In this work, we present a novel approach for the solution of general bounded nonconvex quadratic programs (QP's). Our approach begins by applying an established diagonalization-based reformulation to isolates all problem nonconvexities to univariate quadratic terms. We then apply a novel, compact MIP overapproximation for the univariate quadratic to obtain a convex mixed-integer quadratic program (MIQP). This model leverages a recursive piecewise linear approximation for the univariate quadratic by Yarotsky in 2016 based on the so-called sawtooth functions. We give an analysis of the strength of the new model for the univariate quadratic with respect to sharpness, and study relaxation volumes of a MIP relaxation derived form the overapproximation. Finally, we demonstrate that the presented approach is more effective than competing MIP approaches, and that the presented approach can compete with commercial solvers on a class of difficult instances.

Finally, in Chapter 5, we present a manuscript titled "A Comparison of Discretization Approaches for Mixed Integer Nonconvex Quadratic Programming". In this work, we study various MIP relaxations for individual bivariate terms to solve general nonconvex QCQP's. We begin by extending the sawtooth-based relaxation in Chapter 4 to form a compact lower-bounding relaxation for the univariate quadratic that does not require any integer variables. We then leverage this relaxation to strengthen the lower-bounding (epigraph) portion of the original sawtooth relaxation. Next, we apply this strengthened sawtooth relaxation to a univariate reformulation for bivariate

terms by our collaborators. We then leverage the new sawtooth epigraph relaxation to build a novel univariate reformulation that requires significantly fewer binary variables to reach the same level of precision. Next, we extend the normalized multiparametric disaggregation technique (NMDT) by Castro in 2015 to provide a double-discretized version, which requires half the binaries to reach the same level of precision if all possible bilinear terms must be modeled.

We explore the respective strengths of the various formulations presented in terms of sharpness and relaxation volumes. Finally, we computationally compare the presented formulations on two sets of instances: nonconvex boxQPs, and alternating current optimal power flow (ACOPF) problems.

1.1 An Approximate Method for the Optimization of Long-Horizon Tank Blending and Scheduling Operations

In this work, we address a challenging tank blending and scheduling problem regarding operations for a chemical plant, which is illustrated in Figure 1.1. The challenge for the problem stems from several primary concerns. First, integer variables are required to determine which tank each barge unloads its inventory to, and when, and which storage tanks are pulled from for each production run. Next, the composition of the mixtures must be tracked throughout the system, to ensure that the concentrations and ratios between useful components of the mixture fall within required production specifications. Finally, the problem requires daily decisions over a time horizon of a full year, since the relatively small size of the storage tanks in the instances studied may necessitate replenishment of a storage tank during a production run, which requires constant volume feeds from all tanks in use. This results in a fairly large number of binary variables and nonlinear terms. The primary optimization problem is a feasibility problem at its core: we need to determine a solution to unload all barges within their given unloading windows while meeting the production feed requirements throughout each production run. To ensure a feasible solution, however, we add slack variables to the volume-based feasibility constraints, then minimize the total value of lost

supply materials and missed production requirements.

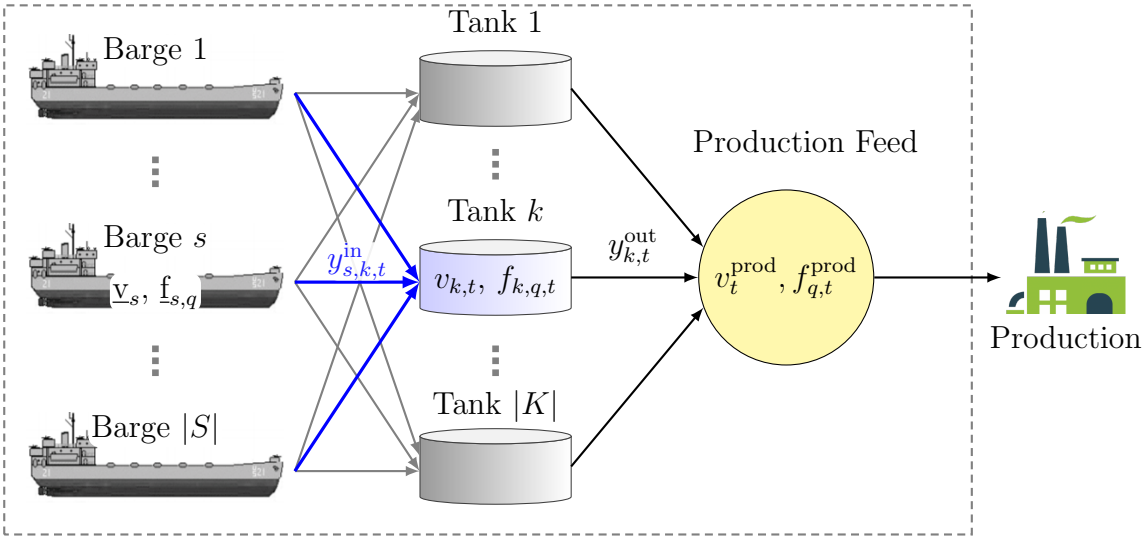


Figure 1.1: We schedule the the front-end portion of the problem, before the production stage (as indicated by the large box). This includes the scheduling of when and where to unload each barge and the determination of flow plans from tanks to production feed.

To model the problem, we make the simplifying assumption that, within a given day, all loading into a tank occurs before all production outflows. This model and with related variables definitions are illustrated in Figure 1.2. For a given time period t and chemical characteristic q, the key characteristic-related constraints are of the form

$$
\begin{aligned}
v^{\text{mid}}_{k,t} f_{k,q,t} &= \sum_{s \in S} y^{\text{in}}_{s,k,t} \underline{f}_{s,q} + v_{k,t-1} f_{k,q,t-1} && \forall k \\
v_{k,t} f_{k,q,t} &= v^{\text{mid}}_{k,t} f_{k,q,t} - y^{\text{out}}_{k,t} f_{k,q,t} && \forall k \\
f^{\text{prod}}_{q,t} v^{\text{prod}}_t &= \sum_{k \in K} \underline{f}_{k,q,t} y^{\text{out}}_{k,t} \\
f^{\text{prod}}_{q,t} &\in [\underline{f}^{\text{min}}_{q,t}, \underline{f}^{\text{max}}_{q,t}] \\
\frac{f^{\text{prod}}_{q_1,t}}{f^{\text{prod}}_{q_2,t}} &\in [\underline{r}^{\text{min}}_{q_1,q_2,t}, \underline{r}^{\text{max}}_{q_1,q_2,t}] && \forall q_1, q_2 \in Q_{\text{rat}},
\end{aligned}
$$

where Q_{rat} is the set of characteristic ratios to track. Note that the third of these constraints is a flow constraint, multiplied through by $f_{k,q,t}$.

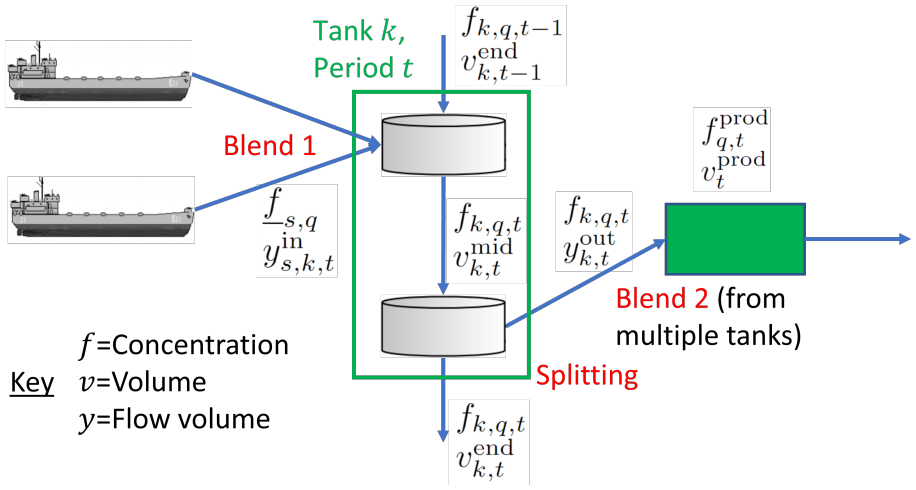


Figure 1.2: The flow model for the problem, including flow-related variables, for tank k in time period t.

The core thrust of this work is twofold. First, we reformulate these spec-related constraints as a MIP that is similar to the Normalized Multiparametric Disaggregation Technique [1]. With our version, though, we circumvent the $f_{k,q,t}$ variables entirely, and relax the problem in a different manner than NMDT. Furthermore, we tighten the spec requirements according to the approximation accuracy to help ensure a feasible solution. Second, we implement a rolling horizon planing approach to handle the long time-horizon.

We evaluate this combined approach using industry-representative data sets from the specialty chemical industry, demonstrating that we can find feasible solutions with good consistency within a reasonable timeframe. This work has been completed and published as [2].

1.2 Force-Controlled Pose Optimization and Trajectory Planning for Chained Stewart Platforms

Stewart platforms consist of two plates separated by six legs, and are actuated via the length of the legs. The legs are attached at an angle, with three legs in one direction and three in the opposite direction (at rest). In this way, the platform can achieve a wide range of motion only via changes in

the leg lengths. Our collaborators have constructed a serial-parallel kinematic chain by joining four of these Stewart platforms together. This structure yields no less flexibility than a pure serial arm, with far more stability. However, computing valid poses for the structure is a somewhat challenging computational problem, and force-optimization of such poses has not yet been attempted in the literature, to the best of our knowledge. In fact, relatively few works in the literature mention chaining these platforms together, and to the best of our knowledge, there is only only one prior work that attempts any sort of explicit pose optimization for these chained structures [3]. However, this work acts only on a simplified 2-D version of the structure, and required a large computational footprint.

We present a fast approach for the force-based pose optimization and trajectory planning of assemblers consisting of general n-stack Stewart platforms. After a fast, natural initialization, the force optimization problem is formulated as a QCQP, and the problem is then solved locally via ipopt with Pyomo to refine the solution. This approach has proven to frequently yield high-quality solutions extremely quickly, usually within about one or two seconds for a stack of 4 SP's. To solve the force-controlled trajectory planning problem, we apply a trust-region method employing local optimization with small steps to build a smooth set of motion waypoints. We then demonstrate the effectiveness and robustness of the proposed approaches, particularly compared to simpler alternatives without optimization.

1.3 Compact mixed-integer programming formulations in quadratic optimization

In this work, we introduce a novel formulation for general QPs. In this work, we consider problems of the form

$$\min_{x \in X} \quad h(x) := x'Qx + c \cdot x \tag{1.1}$$

where $Q \in \mathbb{R}^{n \times n}$ is a symmetric matrix that may is not positive semidefinite, so that the problem is nonconvex. We convert the problem to the form

$$\min_{x \in X} \quad h^D(x, y) \coloneqq x'(Q + D)x + c \cdot x - Dy \tag{1.2a}$$

$$\text{s.t.} \quad y_i = x_i^2 \qquad \forall i \in [\![n]\!], \tag{1.2b}$$

where D is a diagonal matrix of nonnegative numbers, and $Q + D$ is a symmetric matrix. We then model each constraint $y_i = x_i^2$ via a novel, compact, recursive, log-sized MIP approximation. This approximation leverages an elegant, compact piecewise linear approximation by Yarotsky in 2016 involving recursively-defined 'sawtooth' functions. We prove a variety of useful results about the MIP formulation related to formulation strength and connections to the reflective gray code. In particular, we show that the proposed sawtooth-based formulation is hereditarily sharp. Finally, we numerically compare the method to other existing approaches, and find that even a naïve application of the method can be computationally competitive with top solvers, and outshine other prominent MIP approximations for $y = x^2$.

1.4 A Comparison of Discretization Approaches for Mixed Integer Nonconvex Quadratic Programming

In this work, we study the effectiveness of several MIP formulations for bilinear terms in solving nonconvex QCQP's. To begin, we extend the sawtooth relaxation from Chapter 4 to construct a compact relaxation for $y \geq x^2$ that requires no binary variables. We then apply this epigraph relaxation to strengthen the original sawtooth relaxation. We then leverage these extensions to build on the Bin2 and Bin3 formulations presented in Barmann et. al. [4], presenting a MIP formulation for bilinear terms with far fewer binary variables than the formulations presented

therein. These formulations utilize the separable reformulations

$$xy = \tfrac{1}{2}((x+y)^2 - x^2 - y^2)$$
$$xy = \tfrac{1}{2}(x^2 + y^2 - (x+y)^2),$$

then apply other techniques, such as MIP approximations or adaptive piecewise linear approximations, to each univariate quadratic term to form a MIP approximation. In this work, we hybridize these two approaches to form a relaxation that does not require binary variables to model the $(x+y)^2$ and $(x-y)^2$ terms. To this end, we model

$$xy \geq \tfrac{1}{2}((x+y)^2 - x^2 - y^2)$$
$$xy \leq \tfrac{1}{2}(x^2 + y^2 - (x+y)^2),$$

so that we need only to model upper-bounds on the $(x+y)^2$ and $(x-y)^2$ terms. We then apply our sawtooth epigraph relaxation to model these terms with no binary variables and only logarithmically many continuous variables and linear constraints, so that binary variables are only necessary for the terms x^2 and y^2. Then, for a QCQP with n variables that includes all bilinear terms, we require only $O(n)$ binary variables instead of $O(n^2)$ to reach a specified level of precision for the relaxation.

We further present a double-discretized version of NMDT [1], which provides twice the number of bits of precision for the same number of binary variables as the original NMDT for QCQP's with dense Q-matrices, so that the bilinear connectivity graph is complete. In the base-2 version of NMDT for a constraint of the form $z = xy$ for $x, y \in [0,1]$ we first discretize x via a direct binarization with L binary variables, then multiply by y, to obtain

$$\begin{aligned} x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L, \quad z = \sum_{i=1}^{L} 2^{-i}\alpha_i \cdot y + \Delta_x^L \cdot y \\ \Delta_x^L &\in [0, 2^{-L}], \quad y \in [0,1], \quad \boldsymbol{\alpha} \in \{0,1\}^L. \end{aligned} \tag{1.3}$$

We then apply the well-known McCormick relaxation to the remaining bilinear terms to obtain

the NMDT relaxation. To construct a double-discretized variant, we further discretize y, then substitute into the $\Delta_x^L \cdot y$ term, to obtain

$$\begin{aligned} x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L, \quad y = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta_y^L \\ z &= \sum_{i=1}^{L} 2^{-i}[\beta_i((1-\lambda)\Delta_x^L + \lambda x) + \alpha_i(\lambda\Delta_y^L + (1-\lambda)y)] + \Delta_x^L\Delta_y^L \\ \Delta_x^L, \Delta_y^L &\in [0, 2^{-L}], \quad x, y \in [0,1], \quad \boldsymbol{\alpha}, \boldsymbol{\beta} \in \{0,1\}^L. \end{aligned} \tag{1.4}$$

We then apply McCormick relaxation to the remaining bilinear terms to obtain the final relaxation.

We prove various properties regarding the respective strengths of the presented formulations. In particular, we show that the strengthened sawtooth-based relaxation is hereditarily sharp. We then compute the relaxation volumes for each bivariate relaxation, and show that uniform breakpoints are optimal for both novel formulations. Finally, we computationally compare the various relaxations on two families of problem instances: Nonconvex boxQPs, and alternating current optimal power flow (ACOPF) problems.

Chapter 2

An Approximate Method for the Optimization of Long-Horizon Tank Blending and Scheduling Operations

Abstract *We address a challenging tank blending and scheduling problem regarding operations for a chemical plant. We model the problem as a nonconvex MIQCP, then approximate this model as a MILP using a discretization-based approach. We combine a rolling horizon approach with the discretization of individual chemical property specifications to deal with long scheduling horizons, time-varying quality specifications, and multiple suppliers with discrete arrival times. This approach has been evaluated using industry-representative data sets from the specialty chemical industry. We demonstrate that the proposed approach supports fast planning cycles.*

This work has been published as [2]: Benjamin Beach, Robert Hildebrand, Kimberly Ellis, and Baptiste Lebreton, "An approximate method for the opti- mization of long-horizon tank blending and scheduling operations," Computers & Chemical Engineering, vol. 141, p. 106 839, Copyright Elsevier, Oct. 2020. We include the full published work here.

2.1 Introduction

For speciality chemical manufacturing, companies purchase and blend supply streams from a variety of suppliers. Because the supply streams are derivatives of refinery operations, the chemical composition (specifications) of the streams may vary significantly by supplier as well as by season. Supply streams may have both desirable properties and less desirable properties that prevent their exclusive use during the production process. Furthermore, some streams might have limited available quantities or intermittent availability. Thus, supply streams of differing composition are often combined into intermediate storage tanks. The resulting compounds are then blended to meet demand for production feed (see Figure 2.1).

For our motivating application, supply streams typically arrive in barges from suppliers and may remain at the inbound docks for several days before unloading into one or multiple storage tanks. The time windows for the arrival and departure of barges are committed several months in advance. As the supply barges are unloaded into the tanks, the volume and chemical composition of the compounds in the storage tanks are changed. The compounds from one or multiple storage tanks are then blended to create the desired quantity and specification of production feed. The production feed is planned in product campaigns (runs), typically spanning 3-14 days per product. Approximately 3 to 12 months of product campaigns are planned in advance due to the wide variety of spec requirements over time for very different non-concurrent production runs, the non-periodic nature of the supply acquisition process, and the limited capacity of the storage tanks.

Given the volume and characteristics of the compounds on the inbound barges, the time windows for the availability of barges, and the desired production feed (both quantity and specifications), the company must determine an assignment of supply barges to tanks, an unloading plan from barges to tanks, and a blending plan from tanks to production feed. Although similar problems are often inherently multi-objective, the most important objective by far is to satisfy production requirements and to unload supply. Hence, the objective is to determine an operational schedule for the given barge arrivals and production plan. Rapid solutions are required for long scheduling

horizons to support effective planning decisions. Thus, we propose an efficient optimization-based approach to provide solutions for this assignment, mixing, and blending problem within ten minutes for a planning horizon of up to a year.

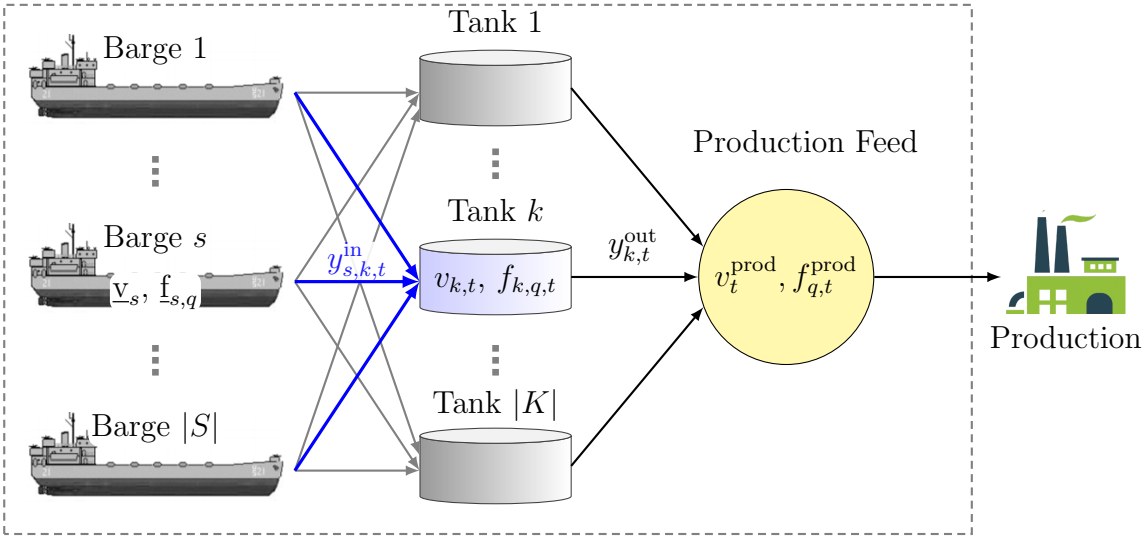


Figure 2.1: We schedule the the front-end portion of the problem, before the production stage (as indicated by the large box). This includes the scheduling of when and where to unload each barge and the determination of flow plans from tanks to production feed.

Within the context of supply chain management, this problem encompasses short-term decisions as highlighted in the Supply Chain Planning Matrix (Figure 2.2, adapted from [5]).

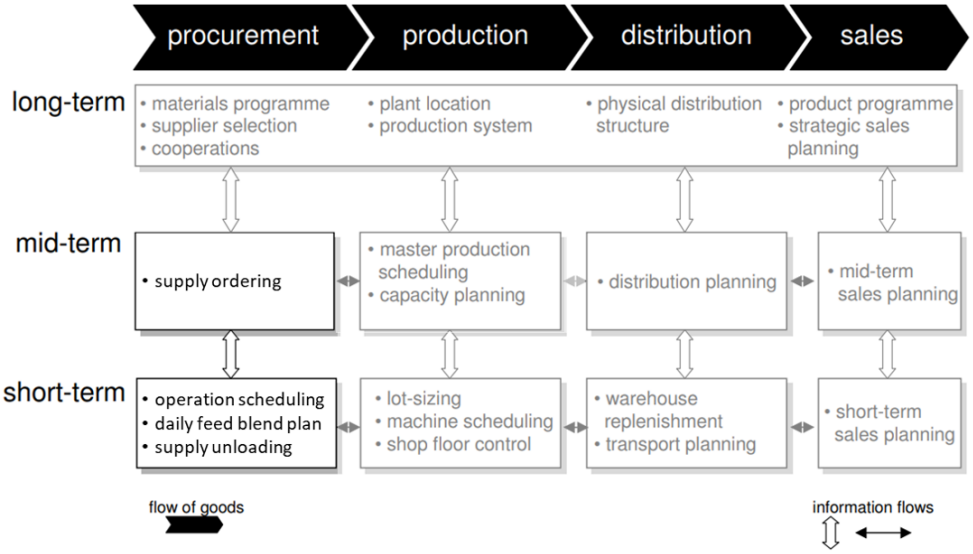


Figure 2.2: Context of this work in terms of supply chain planning (adapted from [5]).

In this work, we employ a discrete-time mixed-integer linear program (MILP) approximate formulation of the core nonconvex mixed-integer quadratically-constrained program (MIQCP), formulated via direct tracking of specs, using single-day scheduling periods with scheduling horizons of 90 to 368 days. The 'binary' version of this discretization combines aspects of multi-parametric disaggregation (MDT) [6] with normalized MDT (NMDT, [1]), while avoiding the explicit tracking of spec qualities. The demand spec and spec ratio requirements are tightened to help ensure that the approximate representation of the specs does not lead to violations of the demand composition requirements. To satisfy the long-horizon scheduling goals within a reasonable computation time, we use a rolling-horizon solution approach. After optimization of the model approximation, the scheduling solution is simulated to obtain the correct tank inventory and demand feed specs.

In the following sections, we provide a formal problem description, highlight the related research, formulate a mixed integer non-linear representation of the problem, propose approaches for solving the model, and compare the approaches with computational studies based on industry-representative data sets.

2.2 Problem Description

For a given a set S of supply nodes (barges), assume each supply node s contains a given volume $\underline{\mathrm{v}}_s$ of raw material, with a level for chemical property q of $\underline{\mathrm{f}}_{s,q}$, where $q \in Q$. The levels of these chemical properties are often specified as a percentage of the volume for a material component of interest. The percentages may not necessarily sum to 100% of the material, because some components may overlap and others may not be of interest. Each supply node s is available during a set T_s of consecutive time periods, from $t_s^{\min}$ to $t_s^{\max}$. With the availability of all supply nodes known, the set of supply nodes available each time period t is denoted as S_t. The number of barges that are unloaded each day is limited to $\underline{\mathrm{ul}}^{\max}$. If a barge is unloaded in a given day, the minimum percentage of inventory that can be unloaded is $\underline{\mathrm{ulp}}_s^{\min}$. If a supply barge has less than $\underline{\mathrm{ulp}}_s^{\min}$ remaining, then the inventory is not unloaded and the barge departs.

A set K of intermediate storage tanks is available to hold raw material or a blend of raw materials. Each tank k has an initial inventory volume $v_{k,0}$, minimum inventory level $v_k^{\min}$, and maximum inventory level $v_k^{\max}$. The initial quality level for chemical property q is denoted by $f_{k,q,0}$. Each tank can be supplied by the supply nodes in K_s. If the mixture in a tank k is used to fulfill production feed, the minimum percentage of feed that can originate from the tank is $\underline{\mathrm{tkp}}_k^{\min}$. The demand for production feed is known for each time period $t \in T$. This demand is characterized by a volume $\underline{\mathrm{d}}_t$, bounds $\underline{\mathrm{f}}_{t,q}^{\min}$ and $\underline{\mathrm{f}}_{t,q}^{\max}$ on the quality level of the chemical properties, and bounds $\underline{\mathrm{r}}_{t,q_1,q_2}^{\min}$ and $\underline{\mathrm{r}}_{t,q_1,q_2}^{\max}$ on certain ratios of chemical properties. Production feed is planned in campaigns or runs of consecutive days. During a run, $\underline{\mathrm{f}}_{t,q}^{\min}$ and $\underline{\mathrm{f}}_{t,q}^{\max}$ remain constant, and we require the daily flow plan for this feed to remain constant.

In order to meet production feed $\underline{\mathrm{d}}_t$, each time period, the volume of material transferred needs to be determined:

- volume to unload from supply node s to tank k during time t, $y_{s,k,t}^{\mathrm{in}}$ and
- volume to unload from tank k to meet production feed in time t, $y_{k,t}^{\mathrm{out}}$.

In addition, the following assignment variables on these flows are needed to address various operational constraints:

- if supply barge s is unloaded during time t, then $\gamma_{s,t} = 1$.
- If tank k provides mixture to meet production feed in time t, then $\sigma_{k,t} = 1$.

To model our objective of meeting demand and unloading requirements, we maximize a weighted sum of the amount of demand met and product unloaded.

Additional operational constraints include the following. The number of times a barge can be unloaded is limited to two, and all unload operations for the barge must be performed within a range of $\underline{\text{ult}}^{\max}$ time periods (e.g., seven days). Flow from tanks to meet production feeds must be continuous for each campaign (or demand run).

Conservation of flow must be maintained, such that the sum of the initial tank inventory level and the volume of supply that flows into the tank less the volume that flows out for production feed equals the final tank inventory level. The resulting levels of chemical properties from this flow must meet the specified composition requirements. Due the relatively small sizes of the storage tanks in this work, we allow storage tanks to both receive supply and provide material for production feed simultaneously. If this occurs in a given time period, for the sake of simplicity, we model that complete mixing of inflows occurs before any outflows within that time period. The problem and constraints are formally specified in Section 2.4.

2.3 Literature Review

The blending and scheduling problems, also known as multi-period pooling problems, studied in the literature vary widely in network size, scope, objective, and detail. Such problems, similar to the standing pooling problem, are typically characterized as bilinear nonconvex MIQCP's due to the balance of specs (or properties) in mixing and splitting of product. This is referred to as

'linear blending', and is often an approximation of what occurs in practice (as e.g. the mixing process may not finish completely before material is split into multiple streams). However, similar problems can also involve linear [7] representation for spec balance, or a more highly nonlinear [8]–[11] representations for balance of some specs ('nonlinear blending', as in the extended pooling problem), depending on the nature of the operational network or the specifications to track.

The underlying resource networks can be small and separated into discrete layers, and can represent front-end acquisition, as in e.g. [12]–[16] and the current work; back-end operations and sales, as in e.g. [11]; or the full process from acquisition to sales, as in [12], [17]. The networks can also span larger, more complex supply chains, as in [18], [19]. The scope of the problem can include scheduling only, as in [16], [20]–[22] and the current work, but can also extend to include more high-level acquisition [13], production planning decisions [23]–[25], or more detailed equipment control decisions [26]. Some works, such as [27], [28], integrate all three levels of planning, scheduling, and control in a single model.

Common examples of objective functions used in similar problems include maximization of profits [15], [17], [19], [29]–[34] or gross margins [12], [13], [16], [35], [36], or minimization of operating costs [7], [12], [16], [35], [37], [38]. The objective function used in [39] in particular is similar to that in this work, but includes additional penalties for performing mixing operations, missing the demand composition requirements, and for not respecting the maintenance requirements. [11] maximizes the yields of final products. Many problems have only a single type of supply source, which can be characterized either by discrete arrivals [12]–[14], [16], [35] or continuous supply streams e.g. [7], [11], [33], [34], [36], [38]. Some works require multiple types of supply sources. For example, in [15], [17], [29]–[31], there are both small barges with a single type of inventory and larger barges, referred to as very large crude carriers (VLCCs), that carry multiple packages with varied compositions. Some works explicitly include vessel scheduling decisions to be made at unloading points, such as docks [29], [31]. Most works assume known supply and demand quantities or bounds, but some, such as [11], [23], [26], [40]–[43], explicitly take into account stochastic supply and demand.

Muti-period pooling problems classically come in the forms of P, Q, or PQ formulations. The Q formulation is source-based, the P formulation is spec-based, and the PQ is a hybrid of the two. That is, the Q formulation tracks compositions based on origin fractions or volumes, while the P formulation tracks compositions explicitly based on the concentrations of the physical qualities to track. See [41] for a discussion of these models. Some examples of spec-based formulations can be found in [11], [14], [33], while examples of source-based formulations can be found in [12], [15], [29], [36]. Comparing to the traditional pooling problem, a spec-based formulation mirrors the traditional P-formulation while a source-based formulation mirrors the Q or PQ-formulations. This choice of representation affects both the required number of bilinear terms and the MILP tightness of the formulation. Overall, the number of bilinear terms for a source-based formulation scales with the number of mixing or splitting nodes in the network and the number of distinct source compositions, while the number of bilinear terms for a spec-based formulation scales instead with the number of specs that require tracking. As such, this choice of representation can have a significant effect on both the tightness of the MIP and the number of required bilinear terms. While source-based formulations can be tighter [34], they are only viable in situations where the number of distinct sources is relatively small. In this work, as many as 35 distinct source compositions can be represented within the scheduling horizon, rendering source-based formulations computationally expensive.

2.3.1 Comparison to Similar Research

We highlight and compare three research streams that are similar to this current work. Note that many differences between this work and others stem from the specialty chemical nature of our problem, combined with the properties of the production site for the application of interest.

Castro [16] studies a similar problem motivated by crude oil operations in refineries. This problem is also addressed in [12], [35], [44], [45], among others. The problem involves barge-based supply where the barges carry a single type of crude, arrive within given time intervals, and unload according to

the arrival sequence. The barges unload to storage tanks. The material in the storage tanks are mixed into charging tanks that supply crude oil into charging tanks that supply crude oil distillation units (CDUs).

The similarities of the core problem include a similar network structure, a barge-based supply stream, and the capacity of each storage tank (∼1-2 barges worth). Both works solve front-end operational scheduling problems and determine feed plans for downstream operations. In [16], such feeds are routed through crude oil distillation units (CDUs). In this current work, the chemicals feed a single production feed.

However, the differences between the problems affect the modeling and solution approaches. In [16], the barges arrive at regular intervals and are unloaded in the order of arrival, with only one barge unloaded at a time. In the current work, multiple barges (up to a limit) may be unloaded simultaneously and the order of unloading needs to be determined. The shorter scheduling horizon in [16] ($\leq$15 days, $\leq$3 barges) allows for solution over the entire scheduling horizon at once, while the long scheduling horizon (up to 365 days, ∼30 barges) in the current work necessitates a heuristic such as a rolling horizon scheme. At the same time, the larger network in [16] reduces the length of the scheduling horizon over which the problem can be solved within a reasonable time. A significant observation of [16] is that, for the problem considered, the choice of objective function significantly impacted the relative efficacy of the discrete-time and continuous-time formulations. Another key difference is that the production schedule for each CDU is not fixed in [16], but is chosen based on gross margins per crude or on operating costs. On the other hand, a CDU can be sourced from only a single charging tank at a time.

In Castro's work, the source-based choice is convenient because there are few composition types to track. In the current work, the source-based formulation is prohibitive due to the large amount of different possible composition types, and a spec-based formulation is used instead. As mentioned in [16], when viable, source-based formulations yield tighter MILP relaxations [34], and can yield better computational performance [33].

Castro [33] addresses a similar multi-period pooling problem with composition constraints on the flows to demand stream (tested on both per-flow and a mixture bases). Similar to the solution approach used in [16], Castro employs an iterated MILP-NLP approach, where the MILP stage is represented using base-10 NMDT. However, the utilized objective functions and formulations differ. In [33], a discrete-time formulation is used, and both spec-based and source-based formulations are presented, and compared with commercial global solvers. The objective function is profit maximization as opposed to gross margin maximization or cost minimization. The problem instance tested has two source streams, up to four blending/storage tanks for which tank-to-tank transfer operations are allowed, and two demand streams, and are solved with a time horizon of up to four periods.

Oddsdottir [13] addresses a similar network, with the same structure (barges to storage tanks to demand (CDUs)). The main differences in this respect involve barge availability (less restrictive time windows), the number of storage tanks (6), and the number of demand streams (2). Another common feature is the requirement of a much longer scheduling horizon (3 months) and in the flexibility in barge arrivals. However, the goal in [13] is related to procurement planning instead of operational scheduling. As such, the choice of which barges to order and when is a problem decision, and a courser time-grid is used (three days per period), resulting in a total of 30 time periods to model. Additionally, in both works, multiple barges might be unloaded within a single time period (up to a limit), possibly contributing to the choice of a discrete-time formulation. Further, a given barge may unload to multiple storage tanks in a given time period. As with [16], however, the limited number of crude types in the system in [13] allow for a reasonably compact source-based formulation, which would be unwieldy in the current work.

2.3.2 Related Solution Methodologies

Rolling planning aside, the solution approach selected in [13] is comparable to the approach in [16], but with some key differences. As in [16], a source-based formulation is used and composition

constraints are formulated with respect to crude volumes instead of crude volume fractions. However, instead of discretizing, [13] applies a MILP approximation to the problem in which the constraints enforcing correct proportions for the outflow composition are reformulated in a manner that allows composition discrepancy in the outflows. A nonconvex MIQCP stage is then applied to resolve the composition discrepancy, yielding near-optimal feasible solutions in a much shorter time frame than the full nonconvex MIQCP: Running in GAMS with BARON, over 20 datasets, the average solution gap for this PRONODIS method (compared to the full nonconvex MIQCP) was 1.1%, with an average computation time of 20s (compared to over 13000s for the full nonconvex MIQCP).

A wide variety of formulations and approaches have been developed to tackle these difficult scheduling problems, with varying time-scheduling formulations, spec representations, and algorithms to deal with the problem nonlinearity, which usually incorporate some form of MILP approximation to the basic nonconvex MIQCP.

A number of continuous-time and discrete-time scheduling formulations have been presented. The choice of time-formulation can have a significant impact on problem performance, as showcased in works such as [30] and [12], [45]. The preferred scheduling formulation can depend on the type of objective function used, as observed in [16], as well as on the types of operational constraints to be considered. Compared to continuous-time formulations, discrete-time formulations frequently yield simpler representations, with a trade-off of more integer variables. Even so, problems with certain types of operational constraints, such as variable flow rates and constraints on the number of simultaneous operations (such as barge unloads), can be significantly easier to model and solve in discrete-time (as seen in e.g. [16]). A discrete-time formulation is also used in our work.

A number of solution methodologies have emerged to deal with the handling of the nonlinear nature of these scheduling problems, particularly over long scheduling horizons. Such methods include single-step [12], [13], [16], [33], [35], [46] and iterated MILP-NLP [15], [17], [39], [47], [48] approaches, which have emerged as some of the most efficient methods to ensure small optimality gaps for relatively short scheduling horizons. Other methods include MILP-(smaller MINLP) iterations [13],

[34] and iterative MILP-refinement methods [21], [22], [29]–[31].

For longer scheduling horizons, rolling planning methods are often used to limit the number of integer variables. Such models typically yield good feasible solutions far faster than the full model, with far better scaling of computation times with the scheduling horizon. In general, a rolling horizon model operates on a discrete time grid by representing in detail only the 'present' segment $\{t^h, \ldots, t^h + H^{\text{roll}} - 1\}$ of the scheduling horizon $\{1, \ldots, H\}$ at each iteration, then tanking a step forwards in time of size $t^{\text{step}} \leq t^h$ for the next iteration, freezing the decisions made in the interval $\{t^h, \ldots, t^h + t^{\text{step}} - 1\}$ and proceeding until the full horizon is reached.

In some rolling planning models, such as those in [11], [13], [24], [49], [50] and heuristic H2 in [25], all aspects of the 'past' decisions in $\{1, \ldots, t^h - 1\}$ are frozen; in others, such as [15] and heuristics H1, H3, and H4 in [25], only certain variables are frozen (most commonly the binary/integer decisions). The treatment of the 'future' portion of the scheduling horizon $\{t + H^{\text{roll}}, \ldots, H\}$ varies considerably between models: Some models exclude the 'future' from the current model entirely, as in [11], [15], [49]–[51], whereas some relax complicating constraints [23], [25] and/or treat integer variables as continuous [24], [25]. Some works, such as [11], [13], [52], [53], deal with uncertain events by updating uncertain data or events at each (or certain) rolling planning steps(s). For bi-level integrated planning and scheduling models, it is common to solve the scheduling subproblem only for the 'present' segment, while including (a subset of) the future segments in the planning subproblem, as is done in [37], [49], [53], [54]. Some works use an alternate framework based on supply arrivals to determine the time steps for each rolling iteration, as in [15], [29]–[31]; [31] in particular allows for some backtracking to correct spec quality discrepancies in the current plan.

Finally, many ideas for MILP approximations to deal with bilinear terms have emerged in recent years, especially for the pooling problem. Some methods are either binary-free, or add a number of binary variables that scales linearly with the desired level of precision. These include various linearization techniques [20], [55], piecewise-linear approximation [36], [56], outer-approximation methods [17], and McCormick or piecewise McCormick envelopes [33], [39], [57]–[60]. [61] in

particular gives a numerical comparison for fifteen different piecewise McCormick-related schemes, split into three classes of schema. For other methods, often utilizing digit-based representations for a single variable, the number of binaries scales logarithmically with the desired precision. Such methods include multi-parametric disaggregation [1], [6], [16], [33], [35], [62], [63] and other techniques [22], [64]–[67]. Approximation via generalized disjunctive programming (GDP) is another, far more expressive, approach that has been used to tackle pooling-type problems. Castro and Grossmann et. al. have derived a MILP-based formulation for variants of multi-parametric disaggregation as a disjunctive program [1], [33], [35], [62], [68], and have also applied it separately [16], [34], [69]. In this work, we choose to directly discretize the inventory specs of each tank using a formulation based on normalized multi-parametric disaggregation (NMDT), as set forth in [1], yielding a relatively simple and highly efficient model in conjunction with a rolling planning approach.

We were unable to find any other work in the literature which combines a version of the multi-parametric disaggregation technique with a version of rolling planning; this combination is a key novelty of this work. The primary results of this work include the speed of the method achieved while yielding near-optimal solutions without compromising the qualities of the specs in the demand feeds, combined with the incorporation of constraints on the ratios of specs in the demand feeds.

2.4 Mathematical Model

We describe a nonconvex MIQCP model in this section that introduces variables to handle the constraints. Data in the problem is written with an underscore, continuous variables are written with lower-case letters, binary variables are written with Greek letters, and sets are written with capital letters.

2.4.1 Terminology

Run	Production run, consisting of a number of consecutive days for which the production feed must be constant in terms of volume.
Feed	Flow from a storage tank to the downstream production processes.
Mixing	The mixing of material from a number of different sources to a single destination.
Splitting	The splitting of material from a single source to a number of different destinations.
Spec	Concentration of a certain type of chemical property.

2.4.2 Sets and Parameters

Sets

S : Set of supply nodes (barges with raw material), indexed by s.

S_k : Set of supply barges allowed to unload to tank $k \in K$.

S_t : Set of supply barges available during time period $t \in T$.

K : Set of tanks, indexed by k.

K_s : Set of tanks to which supply barge $s \in S$ is allowed to unload.

Q : Set of chemical properties, indexed by q.

Q_{rat} : Pairs of specification for which ratio bounds are to be enforced.

R : Set of non-overlapping demand runs (consecutive days with constant production feed).

T : Set of time periods (days), such that $T := \{1, 2, \ldots, H\}$.

T_r : Time periods corresponding to run r where runs do not overlap, such that $T_r = \{t_r^i, t_r^i + 1, \ldots, t_r^f\} \subseteq T$.

T_s : Time periods during which supply $s \in S$ can be unloaded, such that $T_s = \{t_s^{\min}, t_s^{\min} + 1, \ldots, t_s^{\max}\}$.

T_d : Time periods where there is some demand, such that $T_d = \bigsqcup_{r \in R} T_r$.

Parameters

$\underline{c}_s^{\text{inb}}$: Penalty cost per volume of not unloading supply from supply node s.

$\underline{c}_t^{\text{dmd}}$: Penalty cost per volume of not meeting production feed volume for time period t.

$\underline{d}_t$: Demand to meet in time step $t \in T$. Must be constant throughout each run.
Thus, $\underline{d}_t = \underline{d}_{t'}$ for all $t, t' \in T_r$ for $r \in R$. Also, $\underline{d}_t = 0$ for $t \notin T_d$.

$\underline{f}_{s,q}$: Value of chemical characteristic $q \in Q$ in supply source $s \in S$.

$\underline{f}_{k,q,0}$: Initial value of chemical characteristic $q \in Q$ in tank $k \in K$.

$\underline{f}_{q,t}^{\min}, \underline{f}_{q,t}^{\max}$: Bounds on the value of chemical characteristic $q \in Q$ for the demand to be filled in time step $t \in T$.

H : Scheduling time horizon (implicit in the definition of T).

$\underline{r}_{q_1,q_2,t}^{\min}, \underline{r}_{q_1,q_2,t}^{\max}$: Bounds on the ratio of chemical characteristics f_{q_1} and f_{q_2},$(q_1, q_2) \in Q_{\text{rat}}$ for the demand to be filled in time step $t \in T$.

$\underline{\text{tkp}}_k^{\min}$: Minimum percentage of feed each day that can originate from tank k (if used).

$\underline{\text{ul}}^{\max}$: Maximum number of supply barges that can be unloaded each day.

$\underline{\text{ulp}}_s^{\min}$: Minimum percentage of inventory from supply barge s. that can be unloaded each day that unloading occurs for s.

$\underline{\text{ult}}^{\max}$: Maximum permissible gap between the first and last unloads for each supply barge.

$\underline{v}_s$: Volume of supply available for unloading from supply node $s \in S$.

$\underline{v}_{k,0}$: Initial inventory volume of tank $k \in K$.

$[\underline{v}_k^{\min}, \underline{v}_k^{\max}]$: Bounds on permissible inventory for each tank $k \in K$.

Variables (All variables are non-negative)

Assignment: $\gamma_{s,t}$ $:= \begin{cases} 1: & \text{supply barge } s \text{ is unloaded in time step } t \\ 0: & \text{otherwise} \end{cases}$ $: s \in S, t \in T$

$$\sigma_{k,t} \quad := \begin{cases} 1: & \text{tank } k \text{ supplies production feed in time step } t \\ 0: & \text{otherwise} \end{cases} \quad : k \in K, t \in T$$

Transfer: $y^{\text{in}}_{s,k,t}$: Volume of raw material to transfer from supply $s \in S$ to tank $k \in K$ in time step $t \in T$

$y^{\text{out}}_{k,t}$: Volume of material to feed to demand from node $k \in K$ in time step $t \in T$

Resource: $v^{\text{end}}_{k,t}$: Volume of inventory in tank $k \in K$ at the end of time step $t \in T$

$v^{\text{mid}}_{k,t}$: Volume of inventory in tank $k \in K$ after material is supplied from supply barges, but before feed operations, in time step $t \in T$

Demand: v^{prod}_{t} : Total volume fed to production in time step $t \in T$

Tank specs: $f_{k,q,t}$: Value of chemical characteristic $q \in Q$ in tank $k \in K$ at the end of time step $t \in T$

Final specs: $f^{\text{prod}}_{q,t}$: Value of chemical characteristic $q \in Q$ at production at the end of time step $t \in T$

Unload time: t^{first}_{s} : Earliest unload date for supply barge s

t^{last}_{s} : Latest unload date for supply barge s

Slack: v^{unused}_{s} : Volume of material that is not unloaded from supply barge $s \in S$

d^{unmet}_{t} : Volume of production demand missed in time step $t \in T$

Objective: Our objective is to minimize the amount of material that is not unloaded from barges (v^{unused}_{s}) and the demand that is unmet (d^{unmet}_{t}). To accomplish this, we maximize the sum of the value of the raw material unloaded from the barges and the value of the production feed demand that is met. The amount of material unloaded for a barge s is exactly $\underline{\text{v}}_s - v^{\text{unused}}_{s}$, and the amount of demand met at time t is $\underline{\text{d}}_t - d^{\text{unmet}}_{t}$. Hence, we have

$$\max \quad \sum_{s \in S} \underline{\text{c}}^{\text{inb}}_{s} \cdot (\underline{\text{v}}_s - v^{\text{unused}}_{s}) + \sum_{t \in T} \underline{\text{c}}^{\text{dmd}}_{t} \cdot (\underline{\text{d}}_t - d^{\text{unmet}}_{t}) \tag{2.1}$$

2.4.3 Constraints

The following constraints limit the solution space of the assignment, blending, and mixing problem. In the following discussion, note that variables $y^{\text{in}}, y^{\text{out}}$ correspond to flows, while the variables v correspond to stationary volumes at certain times including volumes on barges v_s, volumes in tanks $v_{k,t}$, and volumes required for production feed v_t.

Initial Conditions: The volume of material in the tanks and the chemical characteristics of the material are initialized. Specifically, (2.2a) enforces that the initial volumes in each storage tank are correctly represented, and (2.2b) enforces that the initial chemical characteristics in each storage tank are specified.

$$v_{k,0}^{\text{end}} = \underline{\text{v}}_{k,0} \qquad k \in K \tag{2.2a}$$

$$f_{k,q,0} = \underline{\text{f}}_{k,q,0} \qquad k \in K, q \in Q \tag{2.2b}$$

Flow: For these flow constraints, we assume that inbound supply material is unloaded and blended in the tanks before any material is drawn from the demand tanks to meet production feed on any given day. The model assumes that in the first half of a time period, barges unload to tanks (2.3a), adding to the volume previously in the tank and creating an intermediate volume $v_{k,t}^{\text{mid}}$. In the second half of the time period, material is unloaded from the tanks (2.3b) to support production

(2.3c), while within required volume bounds at production (2.3d) and (2.3e).

$$\sum_{s \in S_k} y^{\text{in}}_{s,k,t} + v^{\text{end}}_{k,t-1} = v^{\text{mid}}_{k,t} \qquad k \in K, t \in T \tag{2.3a}$$

$$v^{\text{mid}}_{k,t} - y^{\text{out}}_{k,t} = v^{\text{end}}_{k,t} \qquad k \in K, t \in T \tag{2.3b}$$

$$\sum_{k \in K} y^{\text{out}}_{k,t} = v^{\text{prod}}_{t} \qquad t \in T \tag{2.3c}$$

$$v^{\text{end}}_{k,t} \geq \underline{\text{v}}^{\text{min}}_{k} \qquad k \in K, t \in T \tag{2.3d}$$

$$v^{\text{mid}}_{k,t} \leq \underline{\text{v}}^{\text{max}}_{k} \qquad k \in K, t \in T \tag{2.3e}$$

Chemical Characteristics: Bi-linear The following constraints capture the blending and mixing of chemical characteristics. Specifically, (2.4a) ensures the characteristics of the material in a tank at the end of the previous time period are combined with the material from supply barges to create a mix. Expression (2.4b) determines the characteristics of the material provided from tanks to meet production feed.

$$f_{k,q,t} \cdot v^{\text{mid}}_{k,t} = \sum_{s \in S_k} \underline{\text{f}}_{s,q} \cdot y^{\text{in}}_{s,k,t} + f_{k,q,t-1} \cdot v^{\text{end}}_{k,t-1} \qquad k \in K, q \in Q, t \in T \tag{2.4a}$$

$$f^{\text{prod}}_{q,t} \cdot v^{\text{prod}}_{t} = \sum_{k \in K} f_{k,q,t} \cdot y^{\text{out}}_{k,t} \qquad q \in Q, t \in T \tag{2.4b}$$

Thus, while supply unloads and production feeds can occur in the same time period (day), they are not considered to occur simultaneously.

Demand: The following constraint set captures the production feed volume that is not met each time period, d^{unmet}_{t}.

$$v^{\text{prod}}_{t} = \underline{\text{d}}_{t} - d^{\text{unmet}}_{t} \qquad t \in T \tag{2.5}$$

Feed Characteristics: Both the chemical characteristics (2.6a) and ratios of characteristics (2.6b)

need to be within the required bounds. Note that in practice for (2.6b) the denominator would be multiplied through, as numerical issues can arise when the denominator is small.

$$\underline{\mathrm{f}}^{\min}_{q,t} \leq f^{\mathrm{prod}}_{q,t} \leq \underline{\mathrm{f}}^{\max}_{q,t} \qquad t \in T,\ q \in Q \tag{2.6a}$$

$$\underline{\mathrm{r}}^{\min}_{q_1,q_2,t} \leq \frac{f^{\mathrm{prod}}_{q_1,t}}{f^{\mathrm{prod}}_{q_2,t}} \leq \underline{\mathrm{r}}^{\max}_{q_1,q_2,t} \qquad t \in T,\ (q_1,q_2) \in Q_{\mathrm{rat}} \tag{2.6b}$$

Total Inflow: Ideally, the volume of supply on each barge is fully unloaded. Expression (2.7a) captures the volume of supply that is not unloaded (v^{unused}_s). In addition, (2.7b) ensures no material is unloaded from a supply barge when the barge is not available.

$$\sum_{k\in K_s}\sum_{t\in T_s} y^{\mathrm{in}}_{s,k,t} = \underline{\mathrm{v}}_s - v^{\mathrm{unused}}_s \qquad s \in S \tag{2.7a}$$

$$y^{\mathrm{in}}_{s,k,t} = 0 \qquad s \in S, k \in K_s, t \in T\backslash T_s \tag{2.7b}$$

Continuous Feed During Runs: The volume of supply provided by each tank to meet production feed needs to remain constant during a production run.

$$y^{\mathrm{out}}_{k,t} = y^{\mathrm{out}}_{k,t-1} \qquad r \in R, k \in K,\ t \in T_r\backslash\{t^i_r\} \tag{2.8}$$

Tank Feed Share Constraints: If production feed is supplied by tank k during time period t (such that $\sigma_{k,t} = 1$), then the volume of production feed ($y^{\mathrm{out}}_{k,t}$) must be at least $\underline{\mathrm{tkp}}^{\min}_k$ of the demand $\underline{\mathrm{d}}_t$ at time t, as enforced by (2.9a). If $\sigma_{k,t} = 0$, then (2.9b) ensures no feed is provided by tank k for time t.

$$y^{\mathrm{out}}_{k,t} \geq (\underline{\mathrm{tkp}}^{\min}_k \cdot \underline{\mathrm{d}}_t) \cdot \sigma_{k,t} \qquad k \in K, t \in T \tag{2.9a}$$

$$y^{\mathrm{out}}_{k,t} \leq \underline{\mathrm{d}}_t \cdot \sigma_{k,t} \qquad k \in K, t \in T \tag{2.9b}$$

Supply Limitations: Although it is preferred that barges are completely unloaded in one day, due to

tank storage capacity, this may need to be done over several days. Hence, we constrain the number of unloads to be bounded by $\underline{\text{bul}}^{\max}$, which are 2 in all of our instances (2.10a). The number of times any barge is unloaded within a time period is also limited (2.10c). Expression (2.10b) serves as a tightening constraint to ensure that barges are not unloaded when they are not available. If a supply barge is not unloaded during a time period (such that $\gamma_{s,t}$ is zero), then no volume is unloaded from the barge (2.10d).

$$\sum_{t \in T_s} \gamma_{s,t} \leq \underline{\text{bul}}^{\max} \qquad s \in S \tag{2.10a}$$

$$\gamma_{s,t} = 0 \qquad s \in S, t \in T \backslash T_s \tag{2.10b}$$

$$\sum_{s \in S} \gamma_{s,t} \leq \underline{\text{ul}}^{\max} \qquad t \in T \tag{2.10c}$$

$$y^{\text{in}}_{s,k,t} \leq \underline{\text{v}}_s \cdot \gamma_{s,t} \qquad s \in S, t \in T,\ k \in K_s \tag{2.10d}$$

Unload Inventory Percentage Constraints: (2.11) enforces that the percent of inventory unloaded per supply barge remains above required minimum $\underline{\text{ulp}}^{\min}_s$ each day, if unloading is to occur that day.

$$\sum_{k \in S_k} y^{\text{in}}_{s,k,t} \geq (\underline{\text{v}}_s \cdot \underline{\text{ulp}}^{\min}_s) \cdot \gamma_{s,t} \qquad s \in S, t \in T \tag{2.11}$$

Unload Time Gap Constraints: In an effort to minimize the cost associated with barges remaining in the area for an extended time, these constraints limit the time duration for a supply barge to remain in the unloading area. To accomplish this, constraints are included to limit the interval $[t^{\text{first}}_s, t^{\text{last}}_s]$. Expression (2.12a) ensures that t^{first}_s is no later than the first period that any inventory is unloaded for barge s. Expression (2.12b) enforces that t^{last}_s is no earlier than the last period that any inventory is unloaded for barge s. Finally, (2.12c) enforces that the difference between the first and last unloading times for each barge is no larger than the allotted time gap. Although t^{first}_s may assume a value earlier than the actual earliest unload time and t^{last}_s may assume a value after the

last unload time, if the interval is *shorter* than the required length, then the actual interval will also be shorter. Thus, this set of constraints is sufficient.

$$t_s^{\text{first}} \leq t \cdot \gamma_{s,t} + H(1 - \gamma_{s,t}) \qquad s \in S, t \in T \tag{2.12a}$$

$$t_s^{\text{last}} \geq t \cdot \gamma_{s,t} - H(1 - \gamma_{s,t}) \qquad s \in S, t \in T \tag{2.12b}$$

$$t_s^{\text{last}} - t_s^{\text{first}} \leq \underline{\text{ult}}^{\max} \qquad s \in S \tag{2.12c}$$

All of the constraints described above are linear except for the chemical characteristic constraints described by (2.4a) and (2.4b), which contain bilinear terms. MILP approximations for these constraints are formulated in Section 2.5.2.

2.4.4 Reformulation without spec variables $f_{q,t}^{\text{prod}}$

For the implementation of this model, we adjust the model to eliminate terms consisting of only a single f^{prod} variable from the model, restricting the model to product terms $f \cdot v$ or $f^{\text{prod}} \cdot v$, referred to as spec-volume terms. In this way, after discretization, we no longer have need to explicitly keep track of these specs at all, relying solely on these spec-volume product terms. In addition, we eliminate the $f_{q,t}^{\text{prod}}$ and v_t^{prod} variables.

The initial constraint (2.2b) is replaced with

$$f_{k,q,0} \cdot v_{k,0}^{\text{end}} = \underline{\text{f}}_{k,q,0} \cdot \underline{\text{v}}_{k,0} \qquad k \in K,\ q \in Q \tag{2.2b-prod}$$

By multiplying (2.3b) by $f_{k,q,t}$ we obtain

$$f_{k,q,t} \cdot v_{k,t}^{\text{mid}} - f_{k,q,t} \cdot y_{k,t}^{\text{out}} = f_{k,q,t} \cdot v_{k,t}^{\text{end}} \qquad k \in K,\ t \in T \tag{2.3b-prod}$$

Note that the $v_{k,t}^{\text{mid}}$ variables could also be eliminated via the constraint (2.3b). However, as this elimination choice can impact the quality of the MILP approximation, we defer this discussion to a

later section.

We reformulate the constraints (2.6a) and (2.6b) by multiplying through by volume (and any denominators) to obtain the following:

$$\underline{\mathrm{f}}^{\min}_{q,t} \cdot v^{\text{prod}}_t \leq f^{\text{prod}}_{q,t} \cdot v^{\text{prod}}_t \leq \underline{\mathrm{f}}^{\max}_{q,t} \cdot v^{\text{prod}}_t \qquad t \in T,\ q \in Q \qquad \text{(2.6a-prod)}$$

$$\underline{\mathrm{r}}^{\min}_{q_1,q_2,t} \cdot f^{\text{prod}}_{q_2,t} \cdot v^{\text{prod}}_t \leq f^{\text{prod}}_{q_1,t} \cdot v^{\text{prod}}_t \leq \underline{\mathrm{r}}^{\max}_{q_1,q_2,t} \cdot f^{\text{prod}}_{q_2,t} \cdot v^{\text{prod}}_t \qquad t \in T,\ (q_1,q_2) \in Q_{\text{rat}} \qquad \text{(2.6b-prod)}$$

Finally, we substitute the definitions of v_t and $f_{k,q,t} \cdot v_t$ in (2.3c) and (2.4b) into constraints (2.5), (2.6a-prod) and (2.6b-prod) to obtain:

$$\sum_{k \in K} y^{\text{out}}_{k,t} = \underline{\mathrm{d}}_t - d^{\text{unmet}}_t \qquad t \in T \qquad \text{(2.5-sub)}$$

$$\underline{\mathrm{f}}^{\min}_{q,t} \cdot \sum_{k \in K} y^{\text{out}}_{k,t} \leq \sum_{k \in K} f_{k,q,t} \cdot y^{\text{out}}_{k,t} \leq \underline{\mathrm{f}}^{\max}_{q,t} \cdot \sum_{k \in K} y^{\text{out}}_{k,t} \qquad t \in T,\ q \in Q \quad \text{(2.6a-prod-sub)}$$

$$\underline{\mathrm{r}}^{\min}_{q_1,q_2,t} \cdot \sum_{k \in K} f_{k,q_2,t} \cdot y^{\text{out}}_{k,t} \leq \sum_{k \in K} f_{k,q_1,t} \cdot y^{\text{out}}_{k,t} \leq \underline{\mathrm{r}}^{\max}_{q_1,q_2,t} \cdot \sum_{k \in K} f_{k,q_2,t} \cdot y^{\text{out}}_{k,t} \qquad t \in T,\ (q_1,q_2) \in Q_{\text{rat}}$$

(2.6b-prod-sub)

To summarize, the MINLP-Mixing reformulation is the initial model, but with (2.2b), (2.3b), (2.3c), (2.4b), (2.5), (2.6a), and (2.6b) replaced by (2.2b-prod), (2.3b-prod), (2.5-sub), (2.6a-prod-sub) and (2.6b-prod-sub).

2.4.5 Model MINLP-SPLIT: Volume-based reformulation

The model presented above can be reformulated to keep track of volumes of chemical characteristics instead of percentages $f_{k,q,t}$, removing $f_{k,q,t}$ from the model. This reformulation results in a linear representation for the mixing constraints in (2.4a), (2.6a-prod-sub) and (2.6b-prod-sub) and modified flow balance constraint in (2.3b-prod). However, there is now a nonlinear representation for the splitting process, in which some volume is split into multiple parts (e.g. future inventory

and demand feed). In the original model, the definition and usage of $f_{k,q,t}$ ensures that the outflow characteristics are correct. However, after reformulating without $f_{k,q,t}$, this is not longer the case, so we must ensure that the outflow characteristics are correct.

To enact the reformulation, we make the following substitutions:

$$f_{k,q,t} \cdot v^{\text{end}}_{k,t} = \mathsf{v}^{\text{f,end}}_{k,q,t} \qquad k \in K, q \in Q, t \in T \tag{2.15a}$$

$$f_{k,q,t} \cdot v^{\text{mid}}_{k,t} = \mathsf{v}^{\text{f,mid}}_{k,q,t} \qquad k \in K, q \in Q, t \in T \tag{2.15b}$$

$$f_{k,q,t} \cdot y^{\text{out}}_{k,t} = \mathsf{y}^{\text{f,out}}_{k,q,t} \qquad k \in K, q \in Q, t \in T \tag{2.15c}$$

Thus, (2.3b-prod), (2.4a), (2.6a-prod-sub) and (2.6b-prod-sub) are written as

$$\mathsf{v}^{\text{f,end}}_{k,q,t} = \mathsf{v}^{\text{f,mid}}_{k,q,t} - \mathsf{y}^{\text{f,out}}_{k,q,t} \qquad k \in K, t \in T \tag{2.3b-prod-vol}$$

$$\mathsf{v}^{\text{f,mid}}_{k,q,t} = \sum_{s \in S_k} \underline{\mathrm{f}}_{s,q} \cdot y^{\text{in}}_{s,k,t} + \mathsf{v}^{\text{f,end}}_{k,q,t-1} \qquad k \in K, q \in Q, t \in T \tag{2.4a-vol}$$

$$\underline{\mathrm{f}}^{\text{min}}_{q,t} \cdot \sum_{k \in K} y^{\text{out}}_{k,t} \leq \sum_{k \in K} \mathsf{y}^{\text{f,out}}_{k,q,t} \leq \underline{\mathrm{f}}^{\text{max}}_{q,t} \cdot \sum_{k \in K} y^{\text{out}}_{k,t} \qquad t \in T,\ q \in Q \tag{2.6a-prod-sub-vol}$$

$$\underline{\mathrm{r}}^{\text{min}}_{q_1,q_2,t} \cdot \sum_{k \in K} \mathsf{y}^{\text{f,out}}_{k,q_2,t} \leq \sum_{k \in K} \mathsf{y}^{\text{f,out}}_{k,q_1,t} \leq \underline{\mathrm{r}}^{\text{max}}_{q_1,q_2,t} \cdot \sum_{k \in K} \mathsf{y}^{\text{f,out}}_{k,q_2,t} \qquad t \in T,\ (q_1,q_2) \in Q_{\text{rat}} \tag{2.6b-prod-sub-vol}$$

In addition, the initial condition (2.2b-prod) is replaced with

$$\mathsf{v}^{\text{f,end}}_{k,q,0} = \underline{\mathrm{f}}_{k,q,0} \cdot \underline{\mathrm{v}}_{k,0} \qquad k \in K,\ q \in Q \tag{2.2b-prod-vol}$$

Now, to enforce the requirement that the chemical characteristics sent to production feed match match the chemical characteristics in the tank; that is:

$$f_{k,q,t} = \frac{\mathsf{v}^{\text{f,mid}}_{k,q,t}}{v^{\text{mid}}_{k,t}} = \frac{\mathsf{y}^{\text{f,out}}_{k,q,t}}{y^{\text{out}}_{k,t}} \tag{2.17a}$$

we add the following nonlinear constraint

$$\mathsf{v}^{\mathrm{f,mid}}_{k,q,t} \cdot y^{\mathrm{out}}_{k,t} = \mathsf{y}^{\mathrm{f,out}}_{k,q,t} \cdot v^{\mathrm{mid}}_{k,t} \qquad k \in K,\ q \in Q,\ t \in T \tag{2.18a}$$

To summarize, the MINLP-Splitting model is the initial model, but with (2.2b), (2.3b), (2.3c), (2.4a), (2.4b), (2.5), (2.6a) and (2.6b) replaced by (2.2b-prod-vol), (2.3b-prod-vol), (2.5-sub), (2.6a-prod-sub-vol), (2.6b-prod-sub-vol), and (2.18a).

2.5 MILP Approximation Methods

We consider approaches of changing the nonlinear portions of the model in Section 2.4.4 to an MILP to more efficiently solve the problem and more easily interact with a rolling planning approach. The core change is that we will replace all products $y^{\mathrm{in}} \cdot f$ and $y^{\mathrm{out}} \cdot f$ with approximations. The new models, given sufficiently accurate approximations, will produce feasible flows y^{in} and y^{out} that completely determine the actions of the system.

For a spec variable $f \in [l, u]$ we describe it by a discete part $\dot{f} \in \{l, l+\epsilon, l+2\epsilon, \ldots, l+\zeta\epsilon\}$ and a continuous part $\Delta f \in [0, \epsilon]$, where we choose ϵ such that $u - \epsilon < 1 + \zeta\epsilon \leq u$. The discrete part $\dot{f}$ will be modeled with binary variables in a way known as Normalized Multi-Parametric Disaggregation Techniques (NMDT). For the products with Δf, we will consider two options - restricting the value to a midpoint or using the McCormick relaxation. That is, the two main ideas considered are the models:

$$x \cdot \tilde{f} = \underbrace{x \cdot \dot{f}}_{\text{NMDT}} + \underbrace{x \cdot \Delta f}_{\Delta f \leftarrow \frac{\epsilon}{2}} \tag{Center}$$

$$x \cdot \tilde{f} = \underbrace{x \cdot \dot{f}}_{\text{NMDT}} + \underbrace{x \cdot \Delta f}_{\text{McCormick}} \tag{McCormick}$$

When using (Center), we relax (2.4a) to maintain flexibility on inflow volumes. We do this by

replacing the left-hand side with its upper and lower bounds with respect to Δf. These choices, along with various relaxations of redundant constraints are described in detail in this section.

We will first describe common linearizations that we will consider, then describe our two competing models, and then have a discussion of rolling planning approaches.

2.5.1 Linearization

We define here two types of linearizations that we will consider. First, the McCormick envelope is a 3-dimensional polytope $\mathcal{M}(x, \beta)$ that is the smallest convex set containing the equation $\mathsf{x}^\beta = x \cdot \beta$ with upper and lower bounds on β and x. We will assume $0 \leq \beta \leq 1$ and $\underline{x}^{\min} \leq x \leq \underline{x}^{\max}$. The McCormick envelope is

$$\mathcal{M}(x,\beta) = \left\{ \begin{array}{c} \underline{x}^{\min} \cdot \beta \leq \mathsf{x}^\beta \leq \underline{x}^{\max} \cdot \beta \\ (x, \beta, \mathsf{x}^\beta) \ : \ x - \underline{x}^{\max} \cdot (1-\beta) \leq \mathsf{x}^\beta \leq x - \underline{x}^{\min} \cdot (1-\beta) \\ (x, \beta, \mathsf{x}^\beta) \in [\underline{x}^{\min}, \underline{x}^{\max}] \times [0,1] \times \mathbb{R} \end{array} \right\}. \tag{2.19}$$

More generally, suppose we have $\beta_0, \ldots, \beta_m \in \{0,1\}$ such that $\sum_{j=0}^m \beta_j = 1$ hence $\sum_{j=0}^m x \cdot \beta_j = x$. Defining $\mathsf{x}^{\beta_j} = x \cdot \beta_j$, we can then model the convex hull of all such solutions as

$$\mathcal{M}^m(x,\beta) = \left\{ \begin{array}{c} \sum_{j=0}^m \beta_j = 1, \quad \sum_{j=0}^m \mathsf{x}^{\beta_j} = x \\ (x, \beta, \mathsf{x}^\beta) \ : \quad \underline{x}^{\min} \cdot \beta_j \leq \mathsf{x}^{\beta_j} \leq \underline{x}^{\max} \cdot \beta_j \\ x - \underline{x}^{\max} \cdot (1-\beta_j) \leq \mathsf{x}^{\beta_i} \leq x - \underline{x}^{\min} \cdot (1-\beta_i) \\ (x, \beta, \mathsf{x}^\beta) \in [\underline{x}^{\min}, \underline{x}^{\max}] \times [0,1]^{m+1} \times \mathbb{R}^{m+1} \end{array} \right\}. \tag{2.20}$$

In the special case where $m = 1$, the set $\mathcal{M}^1(x, \beta)$ is a bijective lifting of $\mathcal{M}(x, \beta)$.

2.5.2 Discretization

We discuss two methods of linearizing or relaxing bilinear products: Normalized Multiparametric Descretization Technique (NMDT) [1] and McCormick envelopes. The discrete variable $\dot{f}$, contained in the interval $[l, u]$, is converted into binary variables as

$$\dot{f} = \lambda_0 + \varepsilon \sum_{i=1}^{n} \sum_{j=0}^{m} (j\, \lambda_i) \cdot \alpha_{ij}, \qquad \sum_{j=0}^{m} \alpha_{ij} = 1 \text{ for all } i = 1, \ldots, n, \tag{2.21}$$

where $\alpha_{ij} \in \{0, 1\}$ for all i, j and λ_i are appropriately chosen scalars. Given a desired level of precision $0 < \hat{\varepsilon} \leq u - l$, the choices of λ_i (and resulting choices for ε, m, and n) will affect the number of binary variables α_{ij} that are needed to reach the desired precision.

1. Multiparametric Disaggregation Technique with base b provided $l \geq 0$:

$$\lambda_0 = 0, \lambda_i = b^{i-1}, m = b - 1; \rightarrow \ \varepsilon = b^{\lfloor \log_b(\hat{\varepsilon}) \rfloor}, n = \left\lceil \log_b(\tfrac{u}{\varepsilon}) \right\rceil. \tag{MDT}$$

2. Normalized Multiparametric Disaggregation Technique, with base b:

$$\lambda_0 = l, \lambda_i = b^{i-1}, m = b - 1; n = \left\lceil \log_b\left(\tfrac{u-l}{\hat{\varepsilon}}\right) \right\rceil, \varepsilon = (u - l)b^{-n}. \tag{NMDT}$$

3. Monolithic Disaggregation:

$$\lambda_0 = l, n = 1, \lambda_1 = 1; m = \left\lceil \tfrac{u-l}{\hat{\varepsilon}} \right\rceil, \varepsilon = (u - l)/m. \tag{Mono}$$

(MDT) discretizes $\dot{f}$ independent of the lower bound l, while (NMDT) and (Mono) use the lower bound and can be interpreted as discretizing $\dot{f}$ after normalizing the bounds $[l, u]$ to $[0, 1]$. Incorporating the lower bound is crucial for reducing the number of binary variables needed (and hence the complexity of the model) to obtain a certain precision ε. Furthermore, the exponential approach of (NMDT) is vastly preferred over the unary approach of (Mono), allowing only

($\mathcal{O}(\log(1/\varepsilon))$ as opposed to $\mathcal{O}(1/\varepsilon)$).

In the originating work for (NMDT) [1], the same number of digits n (and hence, precision ε) is applied to all discretized variables (to maintain the same precision in the *normalized* space), while in the version presented here, n is computed based on the desired precision $\hat{\varepsilon}$ for each individual discretized variable (to maintain similar precision in the *original* space).

$$\mathcal{D}(x,f,\alpha) = \left\{ (x,f,\dot{f},\Delta f,\alpha,\mathsf{x}^{\mathsf{f}},\mathsf{x}^{\alpha}) : \begin{array}{ll} f = \dot{f} + \Delta f & \\ \dot{f} = \lambda_0 + \varepsilon \sum_{i=1}^{n}\sum_{j=1}^{m} (j\,\lambda_i)\cdot\alpha_{ij} & \\ \sum_{j=0}^{m} \alpha_{ij} = 1 & i = 1,\dots,n \\ \mathsf{x}^{\mathsf{f}} = x\cdot f = \lambda_0 x + x\cdot\Delta f + \varepsilon \sum_{i=1}^{n}\sum_{j=0}^{m} (j\,\lambda_i)\cdot \mathsf{x}^{\alpha_{ij}} & \\ \mathsf{x}^{\alpha_{ij}} = x\cdot\alpha_{ij} & i = 1,\dots,n,\ j = 0,\dots,m \\ \underline{\mathrm{x}}^{\min} \le x \le \underline{\mathrm{x}}^{\max} & \\ \alpha_{ij} \in \{0,1\} & i = 1,\dots,n,\ j = 0,\dots,m \end{array} \right\}. \tag{2.22}$$

Most of the nonlinear products are now linearized and we drop the variables $\dot{f}$ and f. In this way, f is now only recorded implicitly (approximately) through product terms $\mathsf{x}^{\alpha_{ij}}$, x^{f}, and $\mathsf{x}^{\Delta\mathsf{f}}$. This produces the set

$$\tilde{\mathcal{D}}(x,f,\alpha) = \left\{ (x,\alpha,\mathsf{x}^{\mathsf{f}},\mathsf{x}^{\Delta\mathsf{f}},\mathsf{x}^{\alpha}) \ : \begin{array}{ll} \mathsf{x}^{\mathsf{f}} = \lambda_0\cdot x + \mathsf{x}^{\Delta\mathsf{f}} + \varepsilon \sum_{i=1}^{n}\sum_{j=0}^{m} (j\,\lambda_i)\cdot \mathsf{x}^{\alpha_{ij}} & \\ (x,\alpha_i,\mathsf{x}^{\alpha_i}) \in \mathcal{M}^m(x,\alpha_i) & \\ \alpha_{ij} \in \{0,1\} & i = 1,\dots,n,\ j = 0,\dots,m \end{array} \right\}, \tag{2.23}$$

where λ_0, n, and ε are computed from a specified $\hat{\varepsilon}$, $\underline{\mathrm{x}}^{\min}$ and $\underline{\mathrm{x}}^{\max}$.

The remaining nonlinear product $x\cdot\Delta f$, approximated by $\mathsf{x}^{\Delta\mathsf{f}}$, will be dealt with in two different ways.

The binarization in [67] is similar to (NMDT) with $b = 2$ after projecting out the α_{i0} variables using the equation $\sum_{j=0}^{1} \alpha_{ij} = 1$, i.e., $\alpha_{i0} + \alpha_{i1} = 1$.

Choosing base $b = 2$ reduces variables.

We suggest the best model uses $b = 2$ since it uses asymptotically fewer binary variables than any other choice $b > 2$ (as $\hat{\varepsilon} \to 0^+$).

Proposition 2.1. *Let* $z_{\hat{\varepsilon}}(b) := (b-1) \cdot n = (b-1) \left\lceil \log_b \left(\frac{u-l}{\hat{\varepsilon}} \right) \right\rceil$ *be the number of binary variables used to represent* $\dot{f}$*, and let* b_1, b_2 *be two different bases. Then* $z'(b_1, b_2) := \lim_{\hat{\varepsilon} \to 0^+} \frac{z_{\hat{\varepsilon}}(b_1)}{z_{\hat{\varepsilon}}(b_2)} = \frac{(b_1 - 1)\ln(b_2)}{(b_2 - 1)\ln(b_1)}$.

Proof. Asymptotically, as $\hat{\varepsilon} \to 0^+$, we can remove the ceiling function in the definition of $z_{\hat{\varepsilon}}$. Hence

$$z'(b_1, b_2) = \lim_{\hat{\varepsilon} \to 0^+} \frac{(b_1 - 1)\log_{b_1}\left(\frac{u-l}{\hat{\varepsilon}}\right)}{(b_2 - 1)\log_{b_2}\left(\frac{u-l}{\hat{\varepsilon}}\right)} = \frac{(b_1 - 1)}{(b_2 - 1)} \log_{b_1}(b_2) = \frac{(b_1 - 1)\ln(b_2)}{(b_2 - 1)\ln(b_1)}.$$

□

Because $z_{\hat{\varepsilon}}(b)$ is an increasing function of b, $z'(b, 2) > 1$, for all $b > 2$. Using the proposition, we compute $z'(b, 2)$ for various values of b in Table 2.3. In [1], Castro uses $b = 10$. As the table shows, this binarization will use nearly three times as many binary variables.

b	3	4	5	6	7	8	9	10
$z'(b, 2)$	1.26186	1.5	1.72271	1.93426	2.13724	2.33333	2.52372	2.70927

Table 2.3: Asymptotic number of binary variables required for base b, as a multiple of the number required for base 2.

In summary, we suggest to use NMDT with a base 2 because it will require the fewest number of binary variables to achieve a desired accuracy.

Implied and redundant equations

Constraint (2.3b) (and subsequently (2.3b-prod)) couples the variables $v^{\text{mid}}_{k,t}, y^{\text{out}}_{k,t}$, and $v^{\text{end}}_{k,t}$. For convenience, let us write $x_1 = v^{\text{mid}}_{k,t}$, $x_2 = y^{\text{out}}_{k,t}$, $x_3 = v^{\text{end}}_{k,t}$, and $f = f_{k,q,t}$. That is,

$$x_1 = x_2 + x_3. \tag{2.24}$$

As a consequence of (2.24), by multiplying through by $f, \Delta f$, and α_i, we obtain three sets of valid equations

$$\mathsf{x}^{\mathsf{f}}_1 = \mathsf{x}^{\mathsf{f}}_2 + \mathsf{x}^{\mathsf{f}}_3, \tag{2.25a}$$

$$\mathsf{x}^{\Delta\mathsf{f}}_1 = \mathsf{x}^{\Delta\mathsf{f}}_2 + \mathsf{x}^{\Delta\mathsf{f}}_3, \tag{2.25b}$$

$$\mathsf{x}^{\alpha_i}_1 = \mathsf{x}^{\alpha_i}_2 + \mathsf{x}^{\alpha_i}_3 \qquad i = 1, \dots, n. \tag{2.25c}$$

Consider the approximation of f and the resulting product form of the equations

$$f = l + \Delta f + \varepsilon \sum_{i=1}^{n} 2^{i-1} \cdot \alpha_i \tag{2.26a}$$

$$\mathsf{x}^{\mathsf{f}}_\kappa = l \cdot x_\kappa + \mathsf{x}^{\Delta\mathsf{f}}_\kappa + \varepsilon \sum_{i=1}^{n} 2^{i-1} \cdot \mathsf{x}^{\alpha_i}_\kappa \qquad \kappa = 1, 2, 3 \tag{2.26b}$$

$$\mathsf{x}^{\alpha_i}_\kappa = x_\kappa \cdot \alpha_i \qquad i = 1, \dots, n,\ \kappa = 1, 2, 3 \tag{2.26c}$$

Of key importance here for model reduction and a stronger formulation are the observations: (1) Bilinear terms can be eliminated using the above equations (2) Applying McCormick relaxations to these bilinear terms before variable elimination produces a tighter formulation.

First, although other model choices are possible, we project out the variables associated with $\kappa = 1$. Conveniently, equations (2.26b) and (2.26c), for $\kappa = 1$, are both implied by all other equations. In fact, variables $\mathsf{x}^{\mathsf{f}}_1$, $\mathsf{x}^{\Delta\mathsf{f}}_1$ and $\mathsf{x}^{\alpha_i}_1$ by substitution, can be projected out of the model using equations (2.24), (2.25b), and (2.25c), then the equations (2.26b) and (2.26c) become tautologies. This

observation provides convenient reduction choices in the model in terms of both variables and constraints.

Second, applying the projection/linear relaxation in the right order can improve the model. In particular, let $\mathcal{F}$ represent the feasible set given by the above equations, let $\mathcal{M}^{123}(x, \alpha_\kappa)$ be the McCormick relaxation in the space of all variables (similarly, $M^{23}(x, \alpha_\kappa)$ in the space of variables associated with $\kappa = 2, 3$), proj_{23} be the projecting into the space of variables with $\kappa = 2, 3$, and denote the removal of bilinear equations (2.26c).

Then we have the strict inclusion

$$\text{relax}\left(\text{proj}_{23}\left(\mathcal{F} \cap \bigcap_{\kappa=1,2,3} \mathcal{M}^{123}(x, \alpha_\kappa)\right)\right) \subsetneq \text{relax}\left(\text{proj}_{23}(\mathcal{F}) \cap \bigcap_{\kappa=2,3} \mathcal{M}^{23}(x, \alpha_\kappa)\right). \tag{2.27}$$

That is, using the inequalities from the McCormick relaxation from the $\kappa = 1$ variables improves the formulation.

2.5.3 MILP Models

We will now outline the two models we propose to consider and then discuss how to integrate them with rolling planning.

Center Approximation

We create a mixed integer linear approximation to the main problem that will only approximately track constraints on specifications. To check feasibility of the resulting flows y^{in} and y^{out}, these solutions must be plugged into the main model to recover spec values and ensure the constraints are met. The approximation error will be guided by how fine the discrete approximation of the spec variables is. For the model, a prevision $\hat{\varepsilon}_q$ is chosen such that, given an accurate discretization, we should have $|f_{k,q,t} - \dot{f}_{k,q,t}| \leq \hat{\varepsilon}$. The choice of $\hat{\varepsilon}_q$, as discussed earlier, defines the choices of ϵ_q, n_q

and m_q which are implied parameters in the sets $\mathcal{D}(x, f_{k,q,t}, \alpha)$. Thus, for each spec q, there are (potentially) unique ϵ, n and m that are used for the set $\mathcal{D}(x, f_{k,q,t}, \alpha)$. We write all products in terms of single variables and we relax (2.4a) to increase the number of feasible solutions to the approximate model.

We thus model (2.4a) and (2.3b-prod) to (2.6b-prod-sub) as

$$\mathsf{v}^{\mathsf{f,mid}}_{k,q,t} - \mathsf{y}^{\mathsf{f,out}}_{k,q,t} = \mathsf{v}^{\mathsf{f,end}}_{k,q,t} \qquad k \in K, t \in T, \qquad \text{(2.3b-prod-appx)}$$

$$\mathsf{v}^{\mathsf{f,mid}}_{k,q,t} - \frac{\varepsilon_q}{2} \cdot v^{\text{mid}}_{k,q,t} \leq \sum_{s \in S_k} \underline{\mathrm{f}}_{s,q} \cdot y^{\text{in}}_{s,k,t} + \mathsf{v}^{\mathsf{f,end}}_{k,q,t-1} \qquad k \in K, q \in Q, t \in T, \qquad \text{(2.4a-relax-ub)}$$

$$\mathsf{v}^{\mathsf{f,mid}}_{k,q,t} + \frac{\varepsilon_q}{2} \cdot v^{\text{mid}}_{k,q,t} \geq \sum_{s \in S_k} \underline{\mathrm{f}}_{s,q} \cdot y^{\text{in}}_{s,k,t} + \mathsf{v}^{\mathsf{f,end}}_{k,q,t-1} \qquad k \in K, q \in Q, t \in T, \qquad \text{(2.4a-relax-lb)}$$

$$\underline{\mathrm{f}}^{\text{min}}_{q,t} \cdot \sum_{k \in K} y^{\text{out}}_{k,t} \leq \sum_{k \in K} \mathsf{y}^{\mathsf{f,out}}_{k,q,t} \leq \underline{\mathrm{f}}^{\text{max}}_{q,t} \cdot \sum_{k \in K} y^{\text{out}}_{k,t} \qquad t \in T,\ q \in Q, \qquad \text{(2.6a-prod-sub-appx)}$$

$$\underline{\mathrm{r}}^{\text{min}}_{q_1,q_2,t} \cdot \sum_{k \in K} \mathsf{y}^{\mathsf{f,out}}_{k,q_2,t} \leq \sum_{k \in K} \mathsf{y}^{\mathsf{f,out}}_{k,q_1,t} \leq \underline{\mathrm{r}}^{\text{max}}_{q_1,q_2,t} \cdot \sum_{k \in K} \mathsf{y}^{\mathsf{f,out}}_{k,q_2,t} \qquad t \in T,\ (q_1, q_2) \in Q_{\text{rat}}, \qquad \text{(2.6b-prod-sub-appx)}$$

The remaining constraints are generated using $\tilde{\mathcal{D}}$ with $\lambda_0 = \underline{\mathrm{f}}^{\text{min}}_{k,q}$ and $\Delta f_{k,q,t} = \frac{\varepsilon_q}{2}$ and $b = 2$,

$$(v^{\text{mid}}_{k,q,t}, \alpha_{k,q,t}, \mathsf{v}^{\mathsf{f,mid}}_{k,q,t}, \mathsf{v}^{\Delta\mathsf{f,mid}}_{k,q,t}, \mathsf{v}^{\alpha,\text{mid}}_{k,q,t}) \in \tilde{\mathcal{D}}(v^{\text{mid}}_{k,q,t}, f_{k,q,t}, \alpha_{k,q,t}) \qquad k \in K, q \in Q, t \in T, \qquad \text{(2.29a)}$$

$$(v^{\text{end}}_{k,q,t}, \alpha_{k,q,t}, \mathsf{v}^{\mathsf{f,end}}_{k,q,t}, \mathsf{v}^{\Delta\mathsf{f,end}}_{k,q,t}, \mathsf{v}^{\alpha,\text{end}}_{k,q,t}) \in \tilde{\mathcal{D}}(v^{\text{end}}_{k,q,t}, f_{k,q,t}, \alpha_{k,q,t}) \qquad k \in K, q \in Q, t \in T, \qquad \text{(2.29b)}$$

$$(y^{\text{out}}_{k,q,t}, \alpha_{k,q,t}, \mathsf{y}^{\mathsf{f,out}}_{k,q,t}, \mathsf{y}^{\Delta\mathsf{f,out}}_{k,q,t}, \mathsf{y}^{\alpha,\text{out}}_{k,q,t}) \in \tilde{\mathcal{D}}(y^{\text{out}}_{k,q,t}, f_{k,q,t}, \alpha_{k,q,t}) \qquad k \in K, q \in Q, t \in T. \qquad \text{(2.29c)}$$

Note that each $\alpha_{k,q,t}$ is a vector of $n_{k,q,t}$ binary variables, which may vary in number depending on the accuracy used to approximate $f_{k,q,t}$.

Redundant equations

Defining both sets of constraints in (2.29a), (2.29b) creates some redundant inequalities. In particular, after omitting the k, q, t indices for convenience, the inequalities

$$\mathsf{v}_{i,j}^{\alpha,\text{mid}} \geq \underline{\mathrm{v}}_k^{\min} \cdot \alpha_{i,j} \qquad i \in I_{k,q,t}, j \in \{0,1\} \tag{2.30a}$$

$$\mathsf{v}_{i,j}^{\alpha,\text{end}} \leq \underline{\mathrm{v}}_k^{\max} \cdot \alpha_{i,j} \qquad i \in I_{k,q,t}, j \in \{0,1\} \tag{2.30b}$$

are redundant due to (2.3b) and the non-negativity of all volume variables.

Tightening feasible region to produce feasible solutions

Because the proposed model is an approximation of the original problem, the model might produce a schedule with flows that result in infeasible specs - violating either the spec bounds or spec ratio bounds. However, in most cases, the magnitude of these violations we observed to be within $\hat{\varepsilon}/2$ for the proposed discretization methods.

Thus, to mitigate this error, we add a buffer of $\hat{\varepsilon}/2$ to each spec requirement, so that the interval $[\underline{\mathrm{f}}_{t,q}^{\min}, \underline{\mathrm{f}}_{t,q}^{\max}]$ is replaced with $[\underline{\mathrm{f}}_{t,q}^{\min} + \hat{\varepsilon}_q/2, \underline{\mathrm{f}}_{t,q}^{\max} - \hat{\varepsilon}_q/2]$.

As for the spec ratios, we used differentials of the ratio to obtain an upper approximation for the magnitude of the buffer to use: If the spec ratio to control is $\dfrac{f_{q_1}}{f_{q_2}}$, where f_{q_1} and f_{q_2} are spec qualities to control, the buffer to use is given via the differential:

$$\Delta\left(\frac{f_{q_1}}{f_{q_2}}\right) = \frac{1}{f_2^{\min}}\Delta f_{q_1} + \frac{f_1^{\max}}{\left(f_2^{\min}\right)^2}\Delta f_{q_1}, \tag{2.31}$$

where Δf_{q_1} and Δf_{q_2} are the computed buffers $\hat{\varepsilon}/2$ for f_{q_1} and f_{q_2}, respectively. Thus, the interval $[\underline{\mathrm{r}}_{q_1,q_2,t}^{\min}, \underline{\mathrm{r}}_{q_1,q_2,t}^{\max}]$ enforced for $\left(\frac{f_{q_1}}{f_{q_2}}\right)$ is replaced by $[\underline{\mathrm{r}}_{q_1,q_2,t}^{\min} + \Delta\left(\frac{f_{q_1}}{f_{q_2}}\right), \underline{\mathrm{r}}_{q_1,q_2,t}^{\max} - \Delta\left(\frac{f_{q_1}}{f_{q_2}}\right)]$. These buffers proved effective in largely resolving the issue of incorrect feed specs and spec ratios for the problems tested. The buffers could be reduced if they would result in possible ranges for specs or spec ratios

that are smaller than the range provided by a single discretization bin (based on initial tank and supply inventories and the parameter $\hat{\varepsilon}_q$ specified by the user). However, for the purposes of this work, we will assume that $\hat{\varepsilon}_q$ is small enough that the resulting ranges for the demand specs are sufficiently large (with width $\geq \hat{\varepsilon}$ after tightening) that feasibility issues do not become problematic for the proposed discretization scheme.

These buffers result in a replacement of the parameters $\underline{\mathrm{f}}^{\max}_{t,q}$, $\underline{\mathrm{f}}^{\min}_{t,q}$, $\underline{\mathrm{r}}^{\max}_{t,q}$, and $\underline{\mathrm{r}}^{\min}_{t,q}$ in Section 2.4 with their 'buffered' counterparts, tightened further by the possible feed specs based on initial tank inventory and inbound supply specs.

McCormick Based Approximation

In a similar way, the final model for the original NMDT method (with McCormick envelopes around the continuous product terms) is given as follows:

$$\mathsf{v}^{\mathrm{f,mid}}_{k,q,t} - \mathsf{y}^{\mathrm{f,out}}_{k,q,t} = \mathsf{v}^{\mathrm{f,end}}_{k,q,t} \qquad k \in K, t \in T \tag{2.32a}$$

$$\mathsf{v}^{\mathrm{f,mid}}_{k,q,t} = \sum_{s \in S_k} \underline{\mathrm{f}}_{s,q} \cdot y^{\mathrm{in}}_{s,k,t} + \mathsf{v}^{\mathrm{f,end}}_{k,q,t-1} \qquad k \in K, q \in Q, t \in T \qquad \text{(2.4a-relax)}$$

$$\underline{\mathrm{f}}^{\min}_{q,t} \cdot \sum_{k \in K} y^{\mathrm{out}}_{k,t} \leq \sum_{k \in K} \mathsf{y}^{\mathrm{f,out}}_{k,q,t} \leq \underline{\mathrm{f}}^{\max}_{q,t} \cdot \sum_{k \in K} y^{\mathrm{out}}_{k,t} \qquad q \in Q, t \in T \tag{2.32b}$$

$$\sum_{k \in K} \mathsf{y}^{\mathrm{f,out}}_{k,q_2,t} \cdot \underline{\mathrm{r}}^{\min}_{q_1,q_2,t} \leq \sum_{k \in K} \mathsf{y}^{\mathrm{f,out}}_{k,q_1,t} \leq \sum_{k \in K} \mathsf{y}^{\mathrm{f,out}}_{k,q_2,t} \cdot \underline{\mathrm{r}}^{\max}_{q_1,q_2,t} \qquad t \in T,\ (q_1,q_2) \in Q_{\mathrm{rat}} \tag{2.32c}$$

Where, for all $k \in K, q \in Q, t \in T$,

$$(v^{\text{mid}}_{k,q,t}, \Delta f_{k,q,t}, \alpha_{k,q,t}, \mathsf{v}^{\mathsf{f},\text{mid}}_{k,q,t}, \mathsf{v}^{\alpha,\text{mid}}_{k,q,t}) \in \tilde{\mathcal{D}}(v^{\text{mid}}_{k,q,t}, f_{k,q,t}, \alpha_{k,q,t}) \tag{2.33a}$$

$$(v^{\text{end}}_{k,q,t}, \Delta f_{k,q,t}, \alpha_{k,q,t}, \mathsf{v}^{\mathsf{f},\text{end}}_{k,q,t}, \mathsf{v}^{\alpha,\text{end}}_{k,q,t}) \in \tilde{\mathcal{D}}(v^{\text{end}}_{k,q,t}, f_{k,q,t}, \alpha_{k,q,t}) \tag{2.33b}$$

$$(y^{\text{out}}_{k,q,t}, \Delta f_{k,q,t}, \alpha_{k,q,t}, \mathsf{y}^{\mathsf{f},\text{out}}_{k,q,t}, \mathsf{y}^{\alpha,\text{out}}_{k,q,t}) \in \tilde{\mathcal{D}}(y^{\text{out}}_{k,q,t}, f_{k,q,t}, \alpha_{k,q,t}) \tag{2.33c}$$

$$\Delta\mathsf{v}^{\mathsf{f},\text{mid}}_{k,q,t} \in [0, \varepsilon \cdot \underline{\mathrm{v}}^{\max}_{k,q,t}] \tag{2.33d}$$

$$\Delta\mathsf{v}^{\mathsf{f},\text{end}}_{k,q,t} \in [0, \varepsilon \cdot \underline{\mathrm{v}}^{\max}_{k,q,t}] \tag{2.33e}$$

$$\Delta\mathsf{y}^{\mathsf{f},\text{out}}_{k,q,t} \in [0, \varepsilon \cdot \underline{\mathrm{d}}_{k,q,t}] \tag{2.33f}$$

Finally, the continuous product terms $\Delta\mathsf{v}^{\mathsf{f},\text{mid}}_{k,q,t}$, $\Delta\mathsf{v}^{\mathsf{f},\text{end}}_{k,q,t}$, and $\Delta\mathsf{y}^{\mathsf{f},\text{out}}_{k,q,t}$ are modeled via their McCormick envelopes, for all $k \in K, q \in Q, t \in T$,

$$\Delta f_{k,q,t} \in [0, \varepsilon] \tag{2.34a}$$

$$(\Delta f_{k,q,t}/\varepsilon, v^{\text{mid}}_{k,q,t}, \Delta\mathsf{v}^{\mathsf{f},\text{mid}}_{k,q,t}) \in \mathcal{M}(\Delta f_{k,q,t}, v^{\text{mid}}_{k,q,t}) \tag{2.34b}$$

$$(\Delta f_{k,q,t}/\varepsilon, v^{\text{end}}_{k,q,t}, \Delta\mathsf{v}^{\mathsf{f},\text{end}}_{k,q,t}) \in \mathcal{M}(\Delta f_{k,q,t}, v^{\text{end}}_{k,q,t}) \tag{2.34c}$$

$$(\Delta f_{k,q,t}/\varepsilon, y^{\text{out}}_{k,q,t}, \Delta\mathsf{y}^{\mathsf{f},\text{out}}_{k,q,t}) \in \mathcal{M}(\Delta f_{k,q,t}, y^{\text{out}}_{k,q,t}) \tag{2.34d}$$

Error propagation in time horizon

From a perspective of numerical error propagation, the McCormick-based approach results in an exactly-represented tank-inflow blending process (constraint (2.4a)) for our problem: since only two 'spec · volume' terms contain variable specs, the volumes of the specs are coupled correctly. On the other hand, the splitting processes in (2.6a) are inexact, since the McCormick envelopes around the Δf product terms effectively allow the specs leaving a tank to differ slightly from the specs propagated to the next time period. However, due to the tightness of the McCormick envelopes, once a post-blending value $\tilde{f}$ is chosen for a tank, the represented specs will remain in the interval $[\tilde{f}, \tilde{f} + \varepsilon]$ until the next blending operation in the tank, allowing a maximum possible spec error in

the interval $[\varepsilon/2, \varepsilon]$ to accrue between blending operations.

On the other hand, the Center approximation presented here incurs an error of up to $\varepsilon/2$ immediately after blending into the tank, but then accrues no additional numerical error within that tank until the next blending operation. This allows the model to remain closer to feasibility for cases in which each tank does not receive supply from barges very frequently.

2.5.4 Rolling Horizon Approach

As mentioned in Section 2.1, we need a blending and scheduling optimizer that can efficiently optimize over a scheduling horizon of at least 3 months and up to 12 months. For the instances of interest, the target time for this optimization is typically preferred to be no more than five minutes (with a ten minute maximum). However, even the linear discrete approximation above is extremely slow even for a scheduling horizon of 120 days, requiring several hours of computation time in Gurobi 8.1, even for extremely course spec discretization (with a total of roughly *six* binary variables per tank per day for only three tanks). As such, we employ a rolling planning approach.

We present three major alternatives for rolling planning. These alternatives are characterized treatment of the 'past', 'present', and 'future', coupled with the choice of time periods ('present') to use for each rolling planning step. The 'past' is the portion of the planning horizon before the current rolling horizon window, the 'present' is the current horizon window, and the 'future' is the portion of the planning horizon after the current rolling horizon window. For the considered schemes, the 'future' is split into two partitions: the 'near future' and the 'far future'. We consider two alternatives for the treatment of the 'past' and 'future', and two major schemes for the choice of time periods.

We will compare two schedules for rolling planning - *fixed-length* periods and *run-based* periods. In both schemes, periods are chosen to be disjoint since previous performed test with overlapping periods performed slower.

Our fixed-length periods are simply equal length periods of some length Δt, except the last period may be truncated to not exceed the horizon length H. See Figure 2.3.

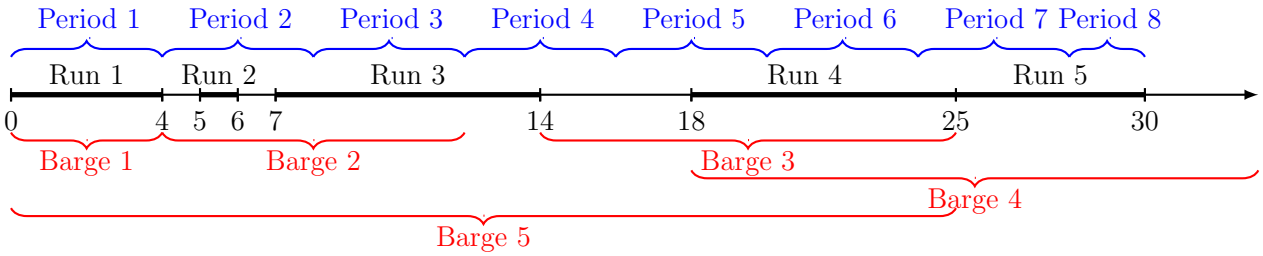


Figure 2.3: This figure shows specified demand runs, the barge time windows, and the periods that we use in the rolling horizon calculation for the fixed-length periods scheme (and considering a horizon $H = 30$ days).

Although this period schedule is consistent, it subdivides runs. This makes modeling consistent volume feed throughout a run (2.8) complicated in the rolling planning process and can lead to poor choices early in the rolling planning. To alleviate this issue, we propose a 'run-based' schedule.

The run-based periods either

- Continue in a run to the end of the run or,
- Encompass all periods and gaps (days between runs) that are completely contained within a time change of Δt from the beginning of period.

This choice makes it so that periods and gaps are never subdivided. This is depicted in Figure 2.4, where a sample partition resulting from this procedure, using $\Delta t = 7$, is illustrated. Formally, our choice of periods is described by Algorithm 1.

Algorithm 1: Rolling Planning with Run Based Sub-periods.

Input: Desired approximate length Δt for a single rolling planning sub-period
Output: Sub-periods P_i for rolling planning.
$t_0 \leftarrow 0$, $i \leftarrow 0$ {Initialize beginning of current rolling planning step and index.} **while** $t_i < H$ **do**
end while
$t^{\text{end}} \leftarrow$ first ending time period of a run that is $\geq t_i$
$t^{\text{change}} \leftarrow$ latest ending time period of a run or gap between runs that is $\leq t_i + \Delta t$
$t_{i+1} \leftarrow \max(t^{\text{end}}, t^{\text{change}}) + 1$
$P_i \leftarrow [t_i, t_{i+1})$
$i \leftarrow i + 1$

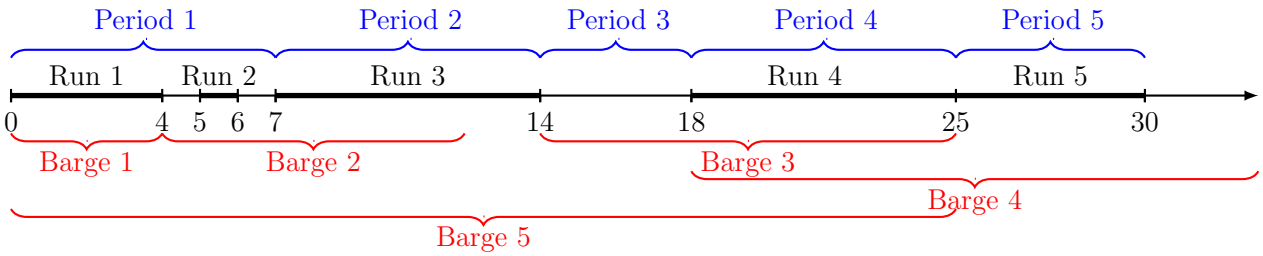


Figure 2.4: This figure shows specified demand runs, the barge time windows, and the periods that we use in the rolling horizon calculation for the 'run-based periods' scheme (and considering $H = 30$ days).

Next, we will discuss the major schemes for the treatment of the past, present, near future, and future segments of the scheduling horizon. These schemes include a full-horizon scheme and a partial-horizon scheme.

Full-Horizon Scheme Given the current rolling horizon period $P_i = [t_i, t_i + 1)$ and a near future horizon H^{NF}, let $t_i^{\text{NF}} = \min(H, t_i + H^{\text{NF}} - 1)$. In our examples, we use $H^{\text{NF}} = 90$.

1. In the past, we freeze the binary decisions made, but allow all continuous variables to continue to vary. This allows the model some additional ability to compensate for poor early decisions, which (as we will see) helps to improve the optimality gap substantially.

2. In the present, the full model is used.

3. In the near future, the binary variables related to demand decisions and discretized tank

compositions are relaxed to the interval $[0, 1]$, while the binary variables related to supply decisions are enforced to be binary. All other constraints are included.

4. In the far future, all binary variables are relaxed to the interval $[0, 1]$. All other constraints are included.

The decision not to relax supply decision binaries in the 'near future' is due to the inclusion of the 'limited unloading window' constraints for barges. These constraints rendered the inclusion of small objective function terms based on supply unloading decisions computationally prohibitive, due to poor branching performance of the solver. Before these were included, small objective function terms had been added to slightly favor earlier unloading times (all else equal), encouraging barges to be unloaded more quickly (and allowing for a model that more consistently avoided small unnecessary supply misses). After the inclusion of these constraints, the solver performance with these small objective terms became extremely poor, requiring at least an order of magnitude more computational time to solve to the same optimality gap in preliminary tests via Gurobi 8.1. When these terms were removed, the model began to exhibit very large extraneous supply misses, due in part to the poor linear relaxation of the 'limited unloading window' constraints, and in part to relaxations of the constraints limiting the number of unloads per barge. These poor relaxations caused the rolling planning model to favor unloading during the relaxed-binary periods, a decision which the model could not recover from. However, by including the supply-based binaries in the 'near future' (defined to end 90 days after the start of the 'present' period), preliminary testing showed that these unnecessary misses became small enough that their cost was considered to be acceptable, while the inclusion of these extra binary variables had only a moderate impact on computational cost (close to a 50% increase).

For all rolling horizon methods, we use 0.5% as the MIP gap. For the Full-Horizon scheme, though experimental results we learned that increasing this, on average, does not significantly reduce computation time, but may have a large effect on resulting objective values. In particular, we want to optimize as much as we can in each step since, due to the way we relax variables, the objective

value in each subsequent step can only decrease. In particular, since we are targeting the best objective value possible, we attempt to make optimal decisions at every step.

Partial-Horizon Scheme

1. In the past, we fix all decisions made, and use them to re-compute accurate values for spec compositions to use for the next rolling planning period. Possible $[l, u]$ for the specs in each tank are re-computed, and the number of remaining unloads (and remaining volumes) for barges that have been unloaded in the past are reduced accordingly.

2. In the present, the full model is enforced.

3. In the near future, the binary variables related to demand decisions and discretized tank compositions are relaxed to the interval $[0, 1]$, while the binary variables related to supply decisions are enforced to be binary. All other constraints are included.

4. In the far future, all variables and constraints are omitted.

Full Horizon Scheme

Vars	Past $[0, t_i)$	Present P_i	Near $\left[t_{i+1}, t_i^{\mathrm{NF}}\right]$	Far $\left(t_i^{\mathrm{NF}}, H\right]$
y, v	active	active	active	active
y, v	active	active	active	active
d	active	active	active	active
t_s	active	active	active	active
γ	fixed	active	active	relaxed
σ	fixed	active	relaxed	relaxed
α	fixed	active	relaxed	relaxed

Partial Horizon Scheme

Vars	Past $[0, t_i)$	Present P_i	Near $\left[t_{i+1}, t_i^{\mathrm{NF}}\right]$	Far $\left(t_i^{\mathrm{NF}}, H\right]$
y, v	fixed	active	active	omitted
y, v	fixed	active	active	omitted
d	fixed	active	active	omitted
t_s	fixed	active	active	omitted
γ	fixed	active	active	omitted
σ	fixed	active	relaxed	omitted
α	fixed	active	relaxed	omitted

Table 2.4: Description of how variables are either fixed, fully active, relaxed to be continuous, or omitted in the two versions of the rolling planning horizon schemes.

In the partial-horizon scheme, we note that it is possible that only a small part the arrival window, even just a single period, for a single barge overlaps with the 'present' and 'near future'. To compensate for this, we pro-rate the inventory to be unloaded from such barges: If only 10% of the volume on a barge overlaps with the currently-considered period, then we enforce that only 10% of

	Tank 1	Tank 2	Tank 3
Total Volume	1179 metric tons	2948 metric tons	1225 metric tons
Minimum Volume	158 metric tons	272 metric tons	136 metric tons

Table 2.5: The capacities of the three tanks we consider. Minimums are about 10% of total volumes. Note that barges hold between 1200 - 1400 metric tons. Hence a barge is comparable to our small tanks, and our large tank can hold slightly more than 2 barges.

the volume from the barge is to be unloaded in the current rolling planning step. In this way, as the model rolls forwards, more of the barge's unloading window enters the visible horizon window, allowing for improved decisions regarding the volume on the barge.

2.5.5 Post-processing and Error Mitigation

After solving the approximation model, we check feasibility of the scheduled flows in terms of required spec quality bounds. To do so, using the flow variables $y^{\text{in}}, y^{\text{out}}$, we plug their computed values into the original model from Section 2.4 to uniquely determine the spec values $f_{k,q,t}$.

After the optimization of the discretized model with rolling planning, we compute specs $f_{k,q,t}$ from simulate the scheduling solution to reveal the true specs throughout the scheduling horizon, and report the true tank and feed volumes and compositions throughout the horizon in addition to the basic scheduling plan.

During preliminary computational testing, after simulating the scheduling solution to reveal the true specs throughout the scheduling horizon, we found that, without tightening the spec requirements, the MILP discretization techniques for the model often resulted in solutions for which the actual spec and spec ratios for the demand feeds were not within the specified ranges. However, this was largely remedied after tightening the feasible region as described in Section 2.5.3

Barge Type	Type 1	Type 2	Type 3	Type 4
Volume	1240	1360	1182	1360
$\underline{c}^{\text{inb}}$	1000	800	800	1000

Table 2.6: Example data for barge types, their capacities, and related objective values.

2.6 Numerical Results

To evaluate the performance of the proposed methods, we constructed a realistic problem with a 368-day scheduling horizon from real operational data with a 119-day scheduling horizon, extended to the desired horizon in a periodic fashion. The resulting problem had a total of 33 supply arrivals over the scheduling horizon, three available storage tanks, two specs (S_1 and S_2) to track, and one spec ratio to control. Note that the range of possible spec qualities was relatively narrow for some tanks, resulting in relatively few binary variables used to approximate the specs for each tank. See Table 2.5 for example data on the storage tanks. Other relevant parameters include $\underline{\text{ul}}^{\max} = 2$, $\underline{\text{ult}}^{\max} = 7$, $\underline{\text{tkp}}_k^{\min} = 10\%$ for each tank, and $\underline{\text{ulp}}^{\min} = 10\%$ for each supply barge. For the objective coefficients, we globally use $\underline{c}_t^{\text{dmd}} = 3000$. $\underline{c}_t^{\text{inb}}$ depends on the barge type, and is either \$800 or \$1000 for each type, as specified in Table 2.6. Figure Figure 2.6 showcases the unloading and arrival windows for a sample instance with $H = 150$, while Figures Figures 2.5 and 2.7 showcase sample solution schedules with $H = 40$. Note that the high variability in bounds for the $S1/S2$ ratios is due to the substantially different requirements for product produced in different runs.

Most computational experiments are performed on a Dell desktop with an Intel Xeon 3.2GHz CPU with 8 cores and 16 threads, and 32 GB of RAM, using Gurobi 8.1 in Python 3 on a Windows 10 operations system. The computational experiments in Section 2.6.2 are performed on a Dell desktop with an Intel Core i7-7820X 3.6GHz CPU with 8 cores and 16 threads, and 32 GB of RAM, using Gurobi 9.0.1 in Python 3 on a Windows 10 operations system.

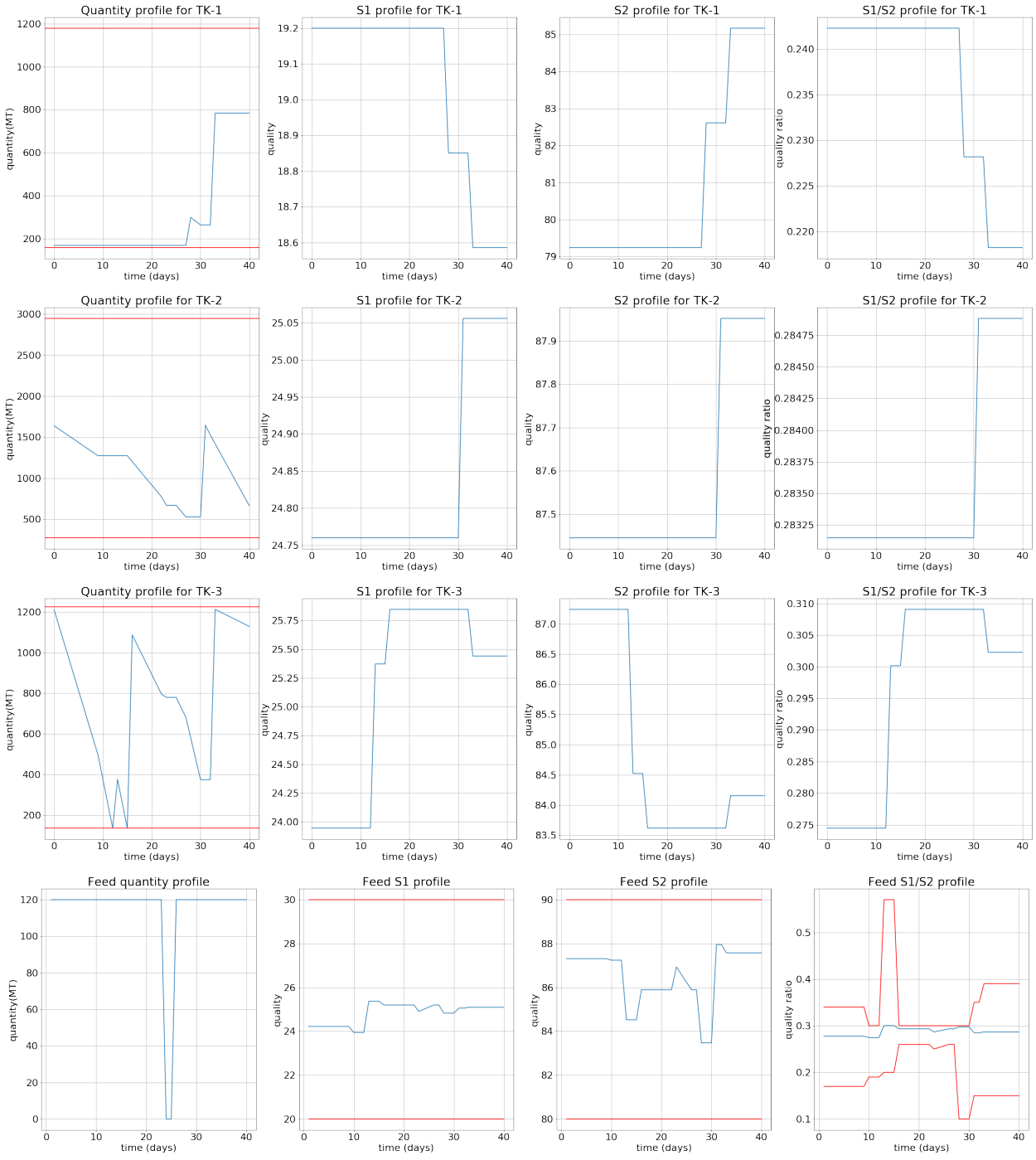


Figure 2.5: Volume profiles with some bounds describing allowable ranges of spec and spec ratios.

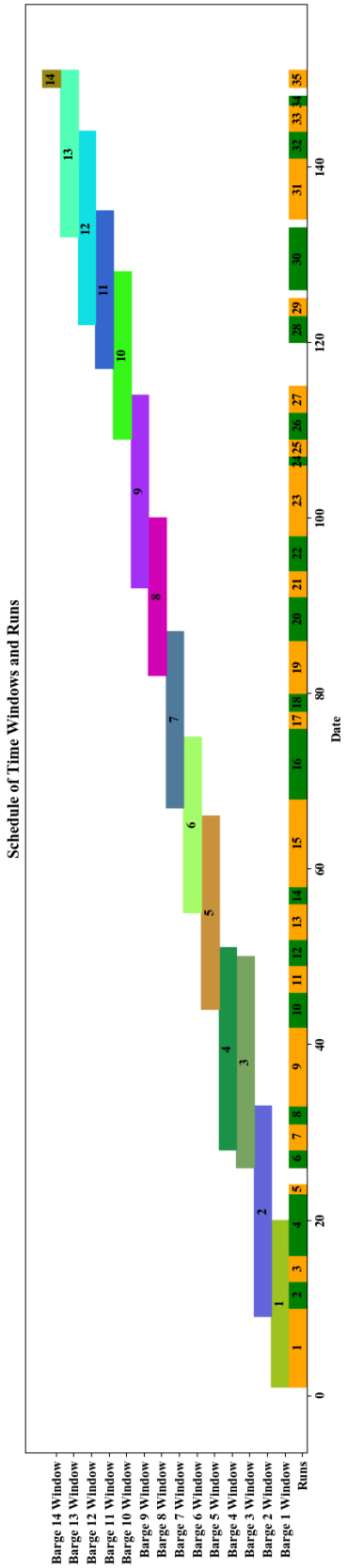


Figure 2.6: Example of time windows for barge arrivals and dates of runs on a 150 day time horizon.

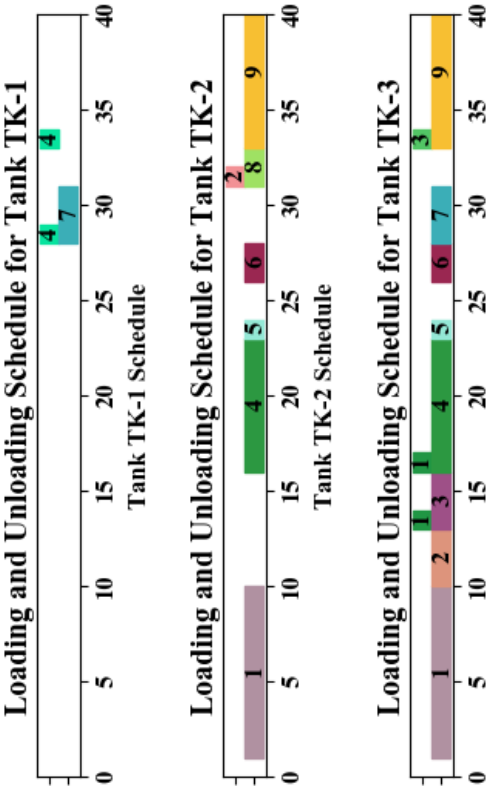


Figure 2.7: Example of solution for schedules of loading to tanks and unloading (feed) from tanks to production for a 40 day time horizon.

2.6.1 Comparison with McCormick

To perform our computational tests, we obtained three years of demand data for the same system, and generated randomized supply data for those three years. We generated ten different such randomizations, and performed each computational test with five different starting dates (separated by half a year), obtaining a total of fifty test problems for each scheduling horizon.

In this section, we compare the effectiveness of two different methods for handling the product terms $v \cdot \Delta f$: McCormick and the proposed Center method, each using a base-2 binarization of the specs. The tests were repeated using several different values of H, and with several values for $\hat{\varepsilon}$ (chosen to be the same for all specs). For $\hat{\varepsilon} = 1$, we used $H \in \{10, 20, 30, 45, 70, 90\}$, and for $\hat{\varepsilon} = 0.25$, we used $H \in \{10, 20, 30, 45, 60\}$ (since McCormick became computationally prohibitive over longer scheduling horizons). A total time limit of 600 seconds was enforced for all computations in this section. The average-case results are shown in Figure 2.8. In this section, we investigate performance in terms of computation time and percentage of value lost (%loss) in the objective, with respect to solution values v_s^{unused} and d_t^{unmet}, computed via

$$val^{\text{target}} = \sum_{s \in S} \underline{\mathrm{c}}_s^{\text{inb}} \cdot \underline{\mathrm{v}}_s + \sum_{t \in T} \underline{\mathrm{c}}_t^{\text{dmd}} \cdot \underline{\mathrm{d}}_t \tag{2.35a}$$

$$val^{\text{missed}} = \sum_{s \in S} \underline{\mathrm{c}}_s^{\text{inb}} \cdot v_s^{\text{unused}} + \sum_{t \in T} \underline{\mathrm{c}}_t^{\text{dmd}} \cdot d_t^{\text{unmet}} \tag{2.35b}$$

$$\%\text{loss} = \frac{val^{\text{missed}}}{val^{\text{target}}} \cdot 100\% \tag{2.35c}$$

In all cases, we see that the proposed Center method performs much faster than McCormick, with a difference of more than an order of magnitude for longer scheduling horizons. However, this performance improvement came at a cost: as shown in 2.8 (b), the average optimality gap for $\hat{\varepsilon} = 1$ yielded by the Center method is consistently quite a bit worse that that yielded by McCormick. As showcased in Figure 2.9(c), the majority of this difference seems to be caused by significantly worse solutions in roughly a quarter of the test cases. However, this disadvantage disappears at higher fidelity; with $\hat{\varepsilon} = 0.25$, the optimality results are very close (well within Gurobi's default optimality

gap of 10^{-4}), with the Center method taking a slight edge over McCormick. At the same time, the Center method proved to yield similar feasibility results as McCormick, with each becoming infeasible for no more than one out of the 50 trials in all test cases.

Figure 2.9 gives another perspective on the performance advantage of the proposed Center method, based on the number of instances solved by a given time for the longest scheduling horizons tested: for $\hat{\varepsilon} = 1$ and $H = 90$, the advantage is overwhelming: almost all of the 50 test instances have converged for the Center method before any have converged for McCormick. The results for $\varepsilon = 0.25$, $H = 60$ also show a strong advantage for the proposed method, though this advantage is somewhat less pronounced due to the shorter scheduling horizon.

These results suggest that, for longer scheduling horizons, the new Center idea fares far better computationally than does the McCormick-style discretization in NMDT for the same level of precision.

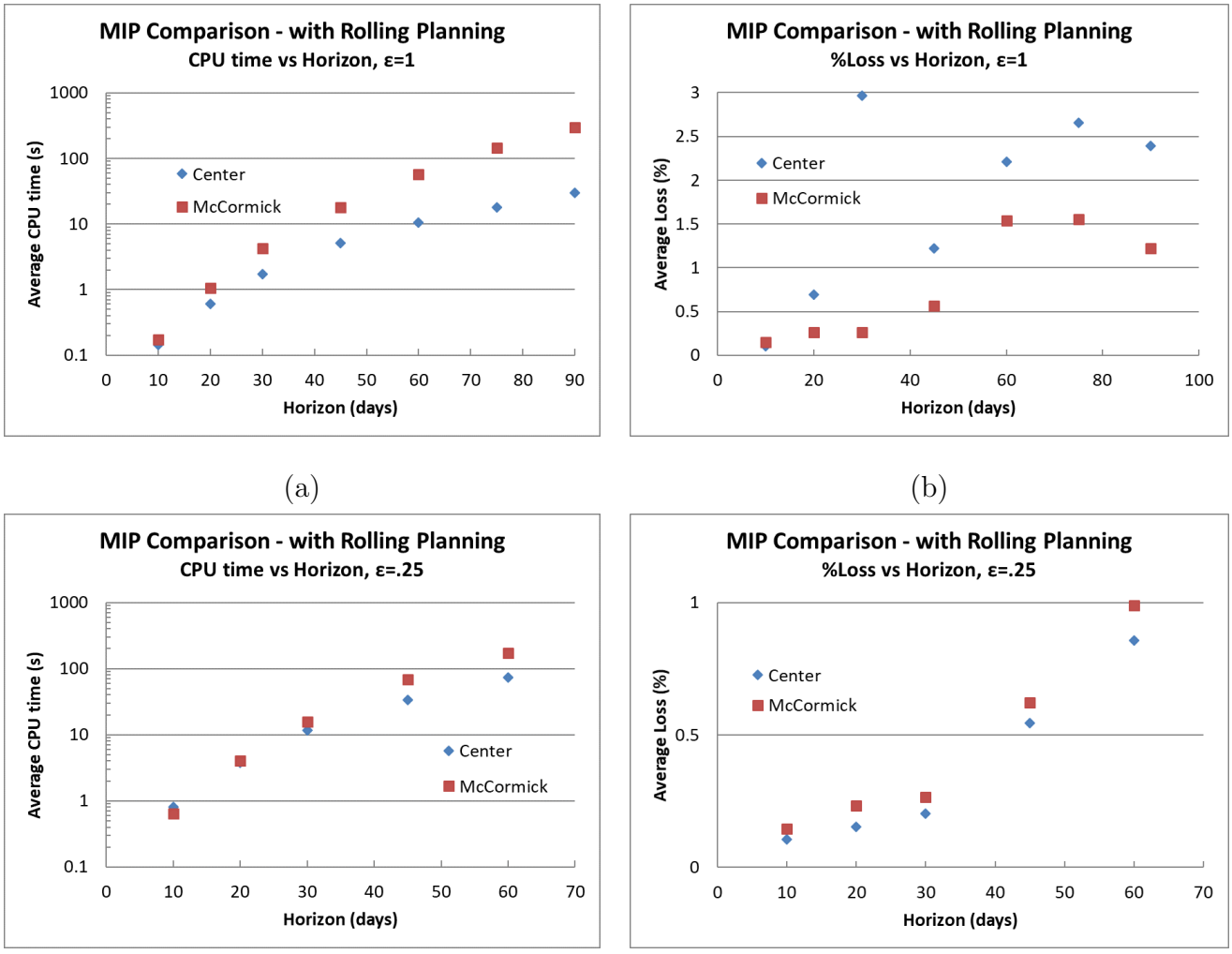
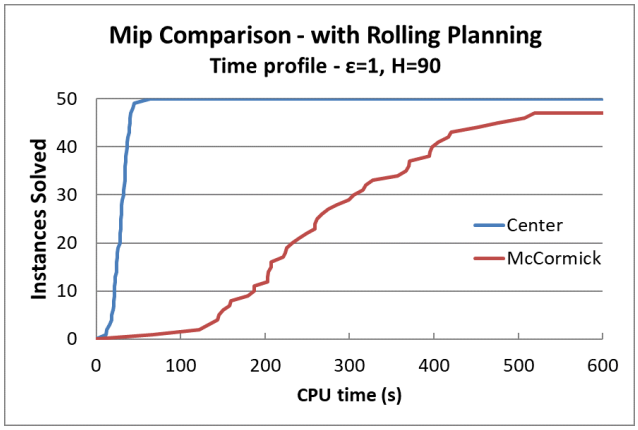
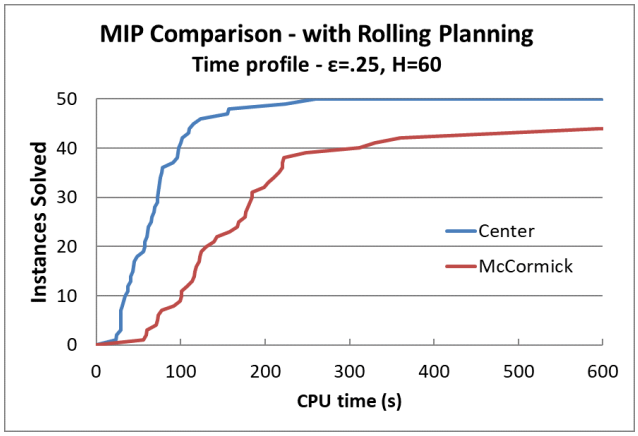


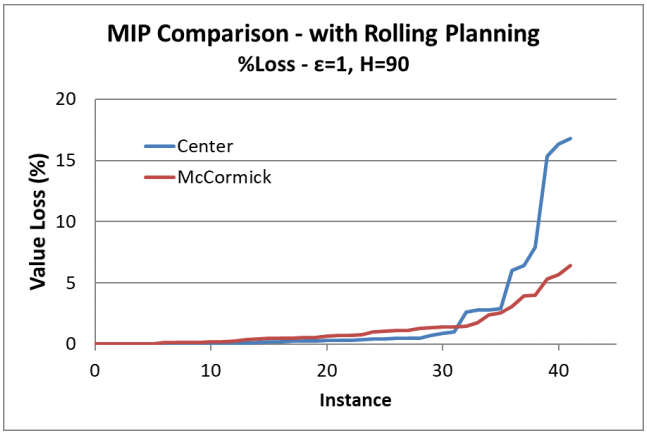
Figure 2.8: Comparison of computational performance of the Center approximation vs. the McCormick relaxation. Averaged over 50 similar instances with different randomized supply parameters. (a) and (c): Computational time using step sizes of $\hat{\varepsilon} = 1$ and 0.25. (b) and (d): % Loss computed as fraction of upper bound on objective value not obtained (see (2.35c)), where model accuracy is set to $\hat{\varepsilon} = 1$ and 0.25.



(a)



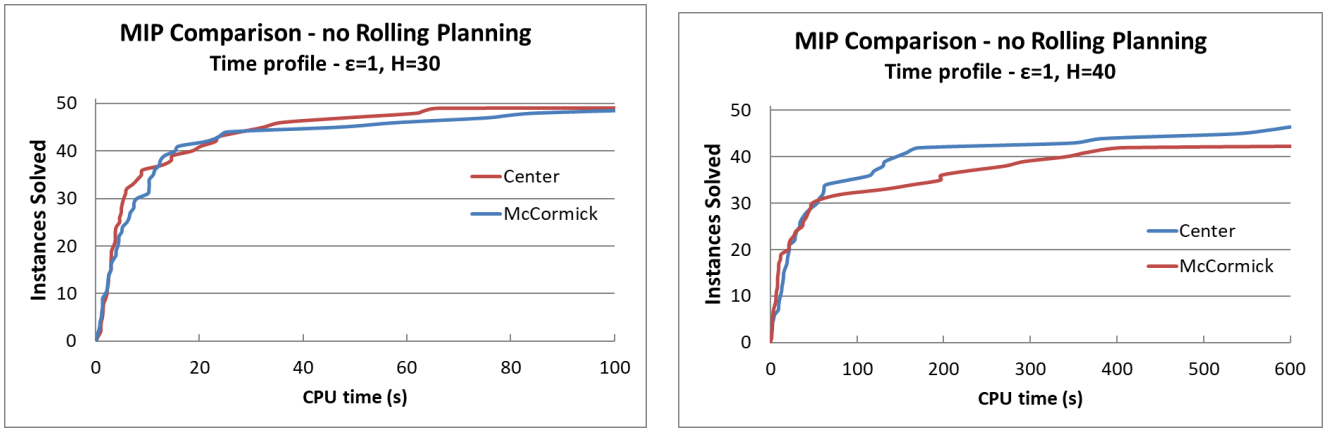
(b)



(c)

Figure 2.9: Time-completion (a,b) and % value missed (c) profiles for the comparison of computational performance for McCormick vs. the proposed approximate method, using the largest values of H tested for (a,c) $\hat{\varepsilon} = 1$, and (b) $\hat{\varepsilon} = 0.25$.

Without rolling planning, the difference is far less pronounced: using 30-day and 40-day scheduling horizons with fidelity of $\hat{\varepsilon} = 1$, we find little difference between the performance of the two methods, as showcased in figure Figure 2.10. As a significant performance difference between the methods had already manifested by such small scheduling horizons in the cases with rolling planning, we find that, for these problem instances, the primary advantage with the Approx method lies in its interaction with rolling planning.



(a) (b)

Figure 2.10: Time-completion profiles for the comparison of computational performance for NMDT vs. the proposed approximate method, using $\varepsilon = 1$ tested for (a) $H = 30$ and (b) $H = 40$.

2.6.2 Comparison to Nonconvex MIQCP in Gurobi 9.0

In this section, we compare the computational performance of the methods presented, using $\varepsilon = 1$ and without rolling planning, to solving the models MINLP-Mix and MINLP-Split defined in Sections 2.4.4 and 2.4.5 with Gurobi 9.0, using scheduling horizons of $H = 20$, $H = 30$, and $H = 40$. The MINLP-Mix model timed out (5-minute limit) for $H = 30$ on all instances but one, so we omit it from our comparisons beyond the first plot with $H = 20$. The results are shown in Figure 2.11. Note that the MINLP-Mix results are omitted from (b), as many instances also didn't finish for $H = 20$.

The performance of MINLP-Mix is far inferior to that of MINLP-Split: For $H = 20$, ∼44 instances finished for MINLP-Split within 50s, while only 10 had finished by that point for MINLP-Mix, with only 19 finishing within 5 minutes. For $H = 30$, no more than a single instance finish within 5 minutes for MINLP-Mix.

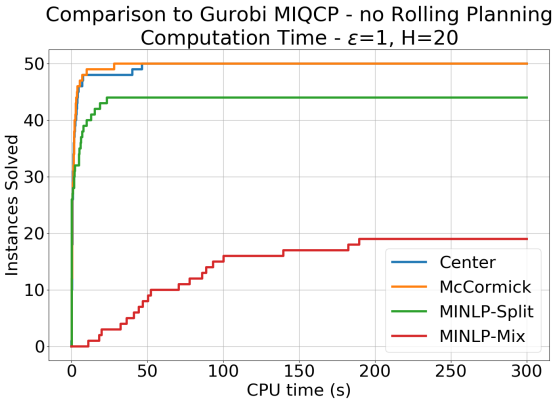
It is interesting to note that, in quite a few instances, Gurobi is able to solve the MINLP-Split model very quickly (∼26 for $H = 20$, ∼21 for $H = 30$, and ∼15 for $H = 40$. The results show that more instances get stuck with long solution times than with the NMDT-based options, with

the slowest 40-60% of solutions slower than the NMDT-based methods. However, the solutions it returns are precise in terms of spec feasibility, while the level of solution precision $\varepsilon = 1$ for the discrete options is quite coarse. It seems likely that, using to $\varepsilon = 0.5$ or $\varepsilon = 0.25$, the Gurobi-based solver may overtake the NMDT-based variants in both solution time and quality for these methods. As a result, one might imagine that a rolling planning scheme based on this MINLP-split model, e.g. relaxing the bilinear constraint (2.18a) for the 'future' portion of the horizon, could be quite fruitful. We leave this exploration as future work.

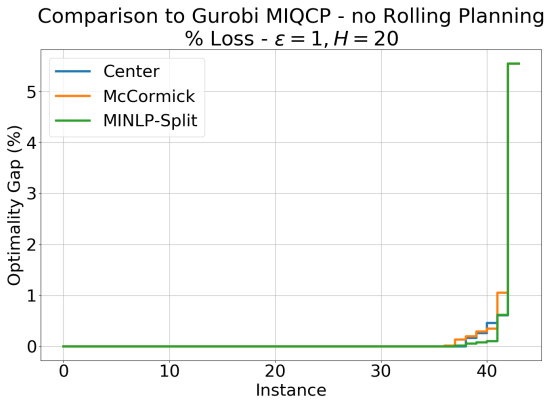
The %loss results show that, even at such course fidelity, the approximation scheme without rolling planning consistently yields near-optimal results. Indeed, the relative difference between the solutions is consistently within the MIP-gap, 0.5%, used for the optimization.

Computation Time

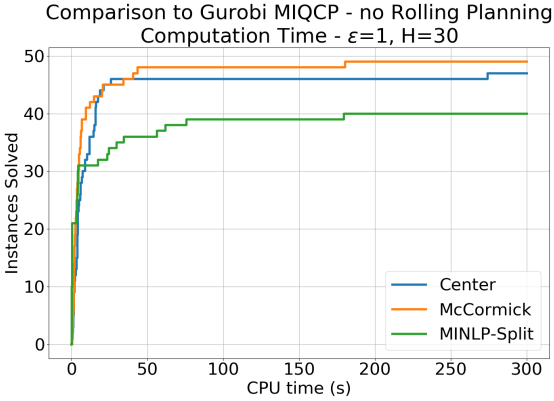
% Loss



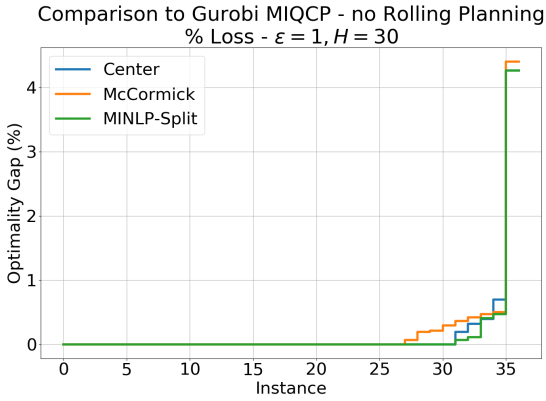
(a)



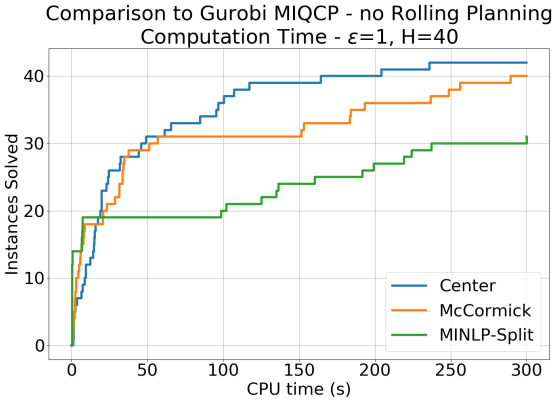
(b)



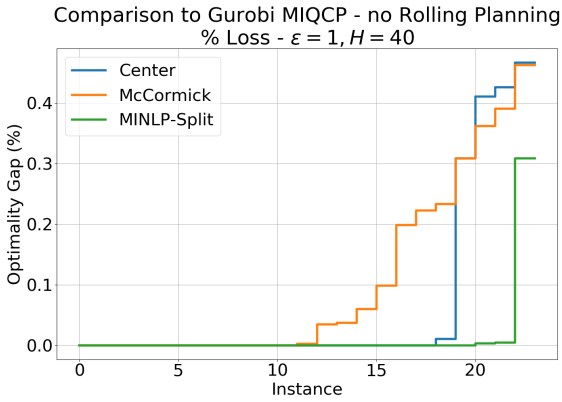
(c)



(d)



(e)



(f)

Figure 2.11: Time-completion (a,c,e) and percent-loss (b,d,f) profiles for the comparison of computational performance for Gurobi 9.0.1 vs. the proposed NMDT-based methods, using $\varepsilon = 1$ tested for (a,b) $H = 20$, (c,d) $H = 30$, and (e,f) $H = 40$.

2.6.3 Performance Enhancement

We explore how to enhance the performance of the method via two simple changes to the Center model that preserve the mixed-integer feasible solutions. The first of these is to add the coupling constraints described in Section 2.5.2

$$\mathsf{v}_{i,j}^{\alpha,\text{mid}} = \mathsf{v}_{i,j}^{\alpha,\text{end}} + \mathsf{y}_{i,j}^{\alpha,\text{out}} \qquad i \in I_{k,q,t}, j \in \{0,1\} \tag{2.36a}$$

where $I_{k,q,t}$ is defined as in Section 2.5.3. We then remove constraints (2.30a) and (2.30b), since they are now made redundant in the associated LP by (2.36a) combined with the nonnegativity of $\mathsf{v}_{p,i,1}^{\alpha,\text{mid}}$, $\mathsf{v}_{p,i,1}^{\alpha,\text{end}}$, and $\mathsf{y}_{p,i,1}^{\alpha,\text{out}}$.

The second of these is to relax the MILP approximation of the $\mathsf{v}_{p,i,j}^{\alpha,\text{mid}}$ terms. This corresponds then removing constraints (2.30a), (2.30b), and the constraints

$$\mathsf{v}_{i,j}^{\alpha,\text{end}} \geq \underline{\mathrm{v}}_k^{\min} \cdot \alpha_{i,j} \qquad i \in I_{k,q,t}, j \in \{0,1\}, \tag{2.37a}$$

$$\mathsf{y}_{i,j}^{\alpha,\text{out}} \leq \underline{\mathrm{d}}_t \cdot \alpha_{i,j} \qquad i \in I_{k,q,t}, j \in \{0,1\}, \tag{2.37b}$$

which are part of the representation of (2.29a) and (2.29c). The viability of the removal of (2.37b) is shown in Section 2.5.2: since any one of the three associated variables $\mathsf{v}_{p,i,1}^{\alpha,\text{mid}}$, $\mathsf{v}_{p,i,1}^{\alpha,\text{end}}$, and $\mathsf{y}_{p,i,1}^{\alpha,\text{out}}$ can be eliminated completely without compromising the MILP, a valid model is still obtained when relaxing any number of constraints related to the definition of $\mathsf{y}_{p,i,1}^{\alpha,\text{out}}$. Constraint (2.37a) is always safe to remove, as it is implied by the fact that only one $\mathsf{v}_{i,j}^{\alpha,\text{mid}}$ variable can nonzero for each i, combined with the fact that $\mathsf{v}_{i,j}^{\alpha,\text{mid}}$ sum to $v_{k,t}^{\text{mid}}$, which is bounded below by $\underline{\mathrm{v}}_k^{\min}$. Note that either (2.30b) or (2.37b) must be included if the coupling constraints (2.36a) are not included. In this case, we choose to remove (2.37b).

The results, displayed in Figure 2.12, show that while both modeling changes make a large difference in computational cost, the simple relaxation of the MILP yields a larger improvement

in computational time. This improvement is increased further when the coupling constraints are added, yielding a method in which the majority of instances finish within the desired 10 minutes when using a scheduling horizon of a full year with $\hat{\varepsilon} = 1$.

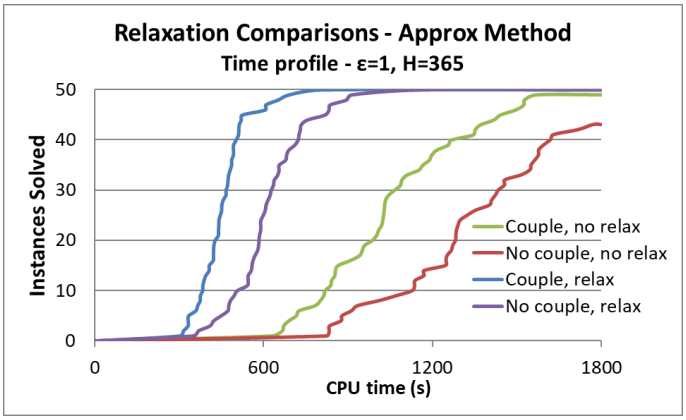
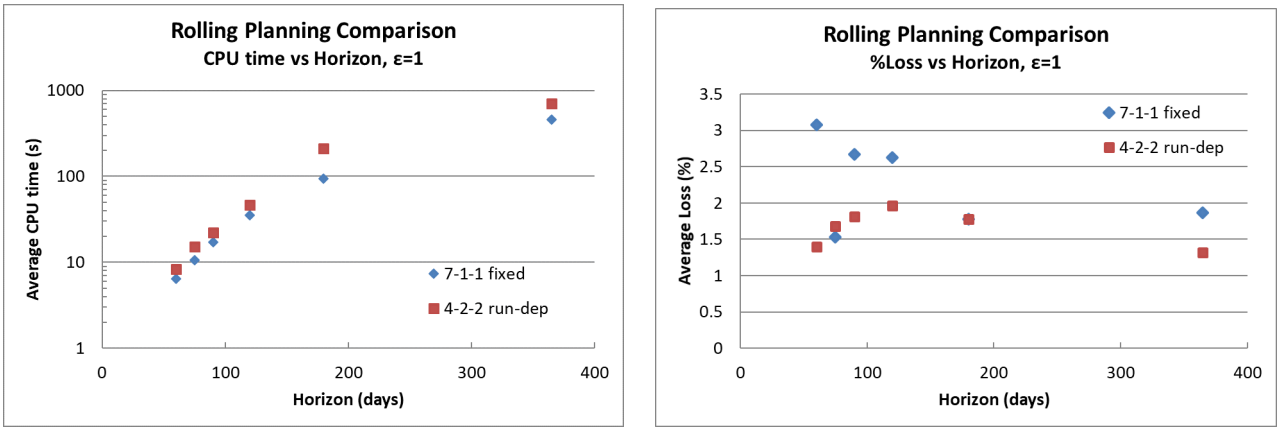


Figure 2.12: Time-completion profiles using different combinations of relaxing (constraints) and tightening (coupling constraint), with $H = 365$ and a 7-1-1 fixed-step-size rolling planning scheme.

2.6.4 Comparison of Rolling Planning Ideas

In this section, we compare the computational performance of different rolling planning regimes. We restrict ourselves here to the full-horizon scheme, and compare the performance obtained using two different ideas. For the first, we use fixed-length periods with $H^{\text{int}} = 7$. For the second, we use run-based periods using $\tilde{H}^{\text{int}} = 4$ to generate the periods, with $n^H = \Delta n = 2$ as our stepping parameters. We compare the methods using a time limit of $1800s$ using $\hat{\varepsilon} = 1$ and scheduling horizons of $H \in \{60, 75, 90, 129, 180, 365\}$ (in days).

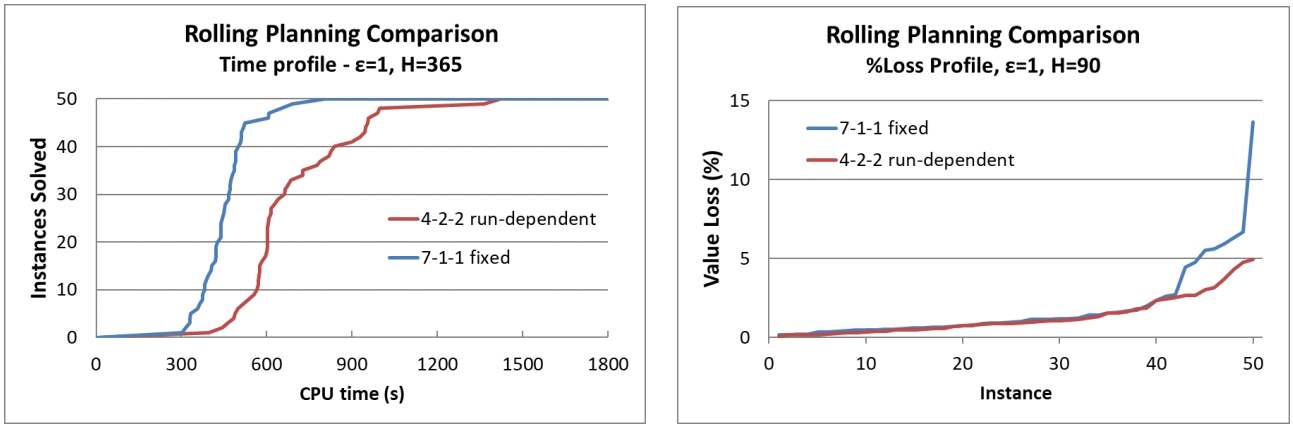
The results, as shown in Figure 2.13, indicate that the regime with a fixed step size yields a more consistent, and often lower, optimality gap on average while incurring a higher computational cost. This outcome is emphasized in Figure 2.14: when $H = 365$, the 7-1-1 fixed-step scheme results in nearly all test cases finishing within about a 5-10 minute time window, while incurring high optimality gaps in a handful of instances. On the other hand, for the 4-2-2 run-dependent-step scheme, most instances finish within about a 10-20 minute time window, but optimality gap for the



(a) (b)

Figure 2.13: Comparison of fixed-step-length and run-dependent-step rolling planning methods. Averaged over 50 similar instances with different randomized supply parameters, using spec step size $\hat{\varepsilon} = 1$. (a): Computational cost, (b) Optimality gap.

worst instances is far more controlled.



(a) (b)

Figure 2.14: Time-completion profiles of fixed-step-length and run-dependent-step rolling planning methods, using $H = 365$ days and $\hat{\varepsilon} = 1$. Run using both the relaxation and coupling things. (a) Computational cost, (b) Optimality gap.

2.7 Conclusion

In this work, we have developed an approximation method for the tank mixing and scheduling problem that combines rolling horizon planning with two different discretization schemes, yielding reasonably high-quality solutions very quickly over long scheduling horizons, while with high probability ensuring that demand specs remain within the required ranges. Due to this performance, the model could be especially useful for obtaining rough, fast plan feasibility results during the planning of supply acquisitions and product production runs in an industry setting (counting solutions with no (or only small) supply and demand inventory misses as 'feasible').

Comparing the in-house "Center" discretization scheme to the original "McCormick"-based NMDT scheme [1], we find that the in-house method yields much faster performance when combined with a rolling horizon scheme, while sometimes yielding worse solutions. However, without a rolling horizon scheme, the methodologies yield comparable performance, suggesting that the "Center" scheme may yield relaxations more compatible with a rolling horizon scheme. Further, when comparing rolling horizon schemes, we find that a scheme using run-based rolling horizon periods yields better solutions than a fixed-duration scheme, at some cost to performance. To extend this work, we plan

to explicitly incorporate acquisition planning of barges, and possibly planning of production runs.

Chapter 3

Force-Controlled Pose Optimization and Trajectory Planning for Chained Stewart Platforms

Abstract *We study optimization methods applied to minimizing forces for poses and movements of chained Stewart platforms (SPs) that we call an "Assembler" Robot. These chained SPs are parallel mechanisms that are stronger, stiffer, and more precise, on average, than their serial counterparts at the cost of a smaller range of motion. Linking these units in a series overcomes their individual limitations and yet maintains their trusslike rigidity, enabling their potential use for a variety of purposes. The assembler robot will be used in concert with several other types of robots to perform complex space missions. We develop algorithms and optimization models that can efficiently decide on favorable positions and movements that reduce forces loads on the robot and hence reducing wear on this machine. The objective of this research is to maneuver the interior plates of the Assembler such that the load forces distributed through the legs of each constituent SP are more even, allowing for a larger workspace and a greater overall payload capacity. The Assembler is designed to function concert with several other robots for a variety of space missions. Our computations focus*

on assemblers with four chained SPs, but our methods apply to an arbitrary number of SPs, and can be extended to general over-actuated truss-like robot architectures.

3.1 Introduction

Robotic space missions are a complex, expensive endeavor, often resulting in multiple mission extensions or scope expansions in order to maximize the robotic system's potential over its useful lifespan. Ensuring that a system is precise and robust enough to accomplish its mission in addition to unknown future mission requirements drives up development and deployment cost significantly, especially due to one-off hardware design. The overall cost of system deployment can be driven down by mass production of smaller, modular robots that can join together to enhance their capabilities. Specifically, we seek to use parallel mechanisms called Stewart Platforms (SPs), which are stronger, stiffer, and more precise, on average, than their serial counterparts at the cost of a smaller range of motion.Linking these units in a series overcomes their individual limitations and yet maintains their trusslike rigidity, enabling their potential use for a variety of purposes. We designate such a configuration as an "Assembler" robot. While the number of SPs in a stack is arbitrary, the experimentation discussed in this article concerns stacks of four. A stack of Stewart platforms has the property that it is over-actuated, which implies that there is a continuum of internal plate poses for any pair of end plate poses. This allows internal plate poses be chosen to satisfy constraints and optimize a variety of metrics. The objective of this research is to maneuver the interior plates of the Assembler such that the load forces distributed through the legs of each constituent SP are balanced, allowing for a larger workspace and a greater overall payload capacity.

We propose to use these chained SPs as an alternative to robot serial arms, for eventual use in fully automated robot assembly on the moon or in space. Each SP consists of a pair of plates with six legs in between. The structure is actuated simply by extending or contracting the linear actuator legs. An example of the structure is shown in Figure 3.1.

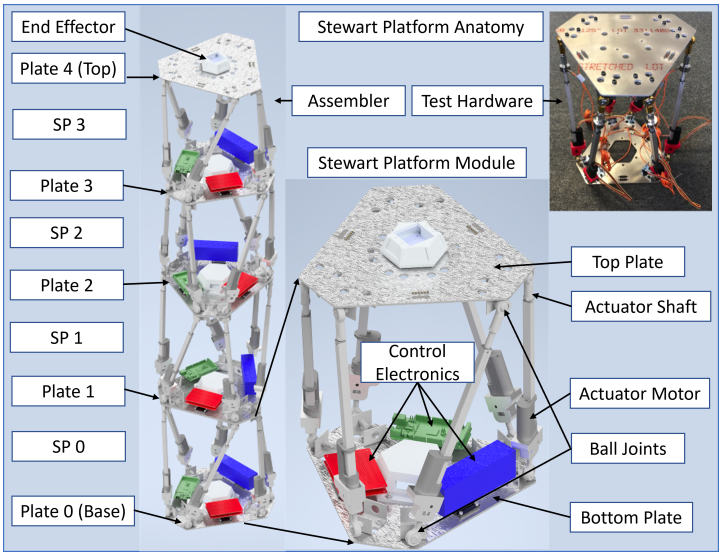


Figure 3.1: Anatomy of an Assembler Robot. The robot is comprised of four chained steward platforms, giving it a larger range of motion and many more degrees of freedom. This assembler has four chained Stewart platforms (SPs), six legs per SP, and 5 plates that serves as the upper and lower boundaries of each SP. The legs are connected via ball joints that are mounted below and above each plate. Thus, the leg connections do not lie in the same plane as the plates.

Why SP-stacks: truss-like robots can be made very lightweight vs. serial arms. Can be used as reconfigurable long-term support structures in addition to manipulation. Higher precision and stiffness for tasks requiring those things. High stiffness means higher fundamental frequency and lower oscillation amplitude (i.e. cantilevering less of an issue). Holds pose when powered down. May self-locomote. Methods here may be extendable to related truss-like robots. Overactuated means optimizable. Downsides: not good for speed, not good for situations requiring compliance. Due to limited angular range of motion even for stacks, it may be good to add a spherical wrist to the end-effector if such angular range needed.

We propose an optimization approach to reduce structural forces in poses and throughout motion. The complexity of the problem stems from the serial combination of parallel kinematic structures (each SP), which can render difficult the problem of even finding feasible poses for a given end effector position and load. From an optimization perspective, the primary source of complexity stems from nonlinearity and nonconvexity of the feasible space, resulting from the trigonometric functions required to model the physical kinematic transformations throughout the structure, along

with the many other bilinear and quadratic terms. These include coordinate distance computations to determine leg lengths, wrench and force computations, and coordinate transformations after computation of the transformation matrices.

There have been many works in the literature exploring design optimization of individual SPs, including [70]–[76]. These works typically optimize the design of a single Stewart platform with respect to parameters such as stiffness, manipulability, and accuracy. In particular, [77] designs and implements a variant of SP with passive control of leg forces. However, we found the literature on stacked Stewart platforms to be incredibly sparse, and were unable to find any prior works optimizing poses for stacked SP chains based on leg forces. There is a related field studying more general actuated truss structures, such as in [78]–[81]. Among other structures, these works study variable geometry truss (VGT) structures, which are similar to SP's, except that each plate is replaced with a triangle of three actuated legs, and the legs between 'plates' are not actuated. [78], [79] also study a version of an SP where the plates are replaced with triangles of stiff legs. However, none of these works study the problem of controlling forces while maneuvering these devices.

For trajectory planning, state-of-the-art approaches include the random tree search-based method RRT [82] and variants RRT-Connect [83], RRT* [84], RRT*-SMART [85], among others. Other state-of-the-art approaches include CHOMP [86], TrajOpt [87], and ROMP [88]. A number of works have explored trajectory planning for individual SP's, including [89]–[94]. Several such works, such as [91], [94], rely on variants of RRT. In one example, [91] even applies this method to an obstacle-avoiding trajectory planning problem for a 4-stack of Stewart platforms. [3] performs stiffness optimization on a 2-D version of the stacked platform structure. Aside from RRT variants, some obstacle-avoiding trajectory planning approaches applied primarily to simple structures such as serial arms are introduced in [86], [87], [95]. Prior works involving trust-region based approaches for trajectory planning, as used in this work, include [88], [96], [97] for serial arms, and [98]–[100] for more difficult robots involving closed kinematic chains.

We were unable to find any works in the literature performing any sort of force-controlled trajectory

planning of stacked SPs in three dimensions, with very few performing any sort of trajectory planning for stacked SPs. Such optimization can significantly improve maximum leg forces throughout the structure, thereby increasing the workspace of an SP under large loads.

Contributions. We present an optimization model and approach to solve these complex kinematic optimization and trajectory planning problems. We are not aware of prior models for chained Stewart platforms, though for a single SP, [101] gives a dynamic model. We first introduce a near-instantaneous spline-based heuristic to find a (near-)feasible initial pose for the structure given a target end effector position and load. We then construct an optimized solution by modeling the structure as nonlinear program (NLP), then solving it locally using IPOPT and modeling it with Pyomo. To obtain a good initial solution for the optimizer, we construct a simple closed-form initialization approach for which each SP has the same pose. The optimization process typically takes just over 1 second on average for a single pose, for an Assembler consisting of four chained SPs. We computationally demonstrate the effectiveness of the approach, and demonstrate the effectiveness of the method in improving the range of motion of the assembler under a payload compared to the spline-based heuristic.

In Section 3.2 we describe the structure of the assembler and its abilities in terms of movement and positions. We then introduce the forward and inverse kinematics problems for single SP's and the full assembler, and mathematically define a kinematically valid SP. In Section 3.3.3, we propose a spline-based construction heuristic (SIK) to produce valid poses, and a simpler same-SP heuristic to produce starting solutions for the optimizer. In Section 3.3.4, we propose an optimization model and simple initial-solution scheme (OPT) to directly generate locally force-optimal poses. In Section 3.4.2, we propose a trust region approach for trajectory planning based on the optimization model. In Section 3.5.2, we compare the effectiveness of the SIK and OPT schemes to quickly generate workable poses for the assembler. Finally, in Section 3.5.3, we compare the trust region trajectory planning scheme with a naive approach based on linearly interpolating leg lengths between the initial and target poses.

3.2 Assembler Robot, Notation, and Transformations

A *Stewart platform* consists of a pair of plates linked by six linear actuators, constituting a parallel mechanism with six degrees of freedom (DOF). The actuators for each platform are connected via ball joints on either end. We refer to the linear actuators as legs. The motor of each actuator is situated on the leg bottom (LB), while the shaft of the actuator is on the leg top (LT). We define the *pose* of a plate as its position and orientation in a given reference frame, and we characterize the pose of an assembler via the poses of its plates in the assembler's reference frame, also referred to as the *global reference frame*. That is, a pose is a list of plate positions $\mathbf{p}_i^{\mathcal{G},P}$ and rotations $\mathbf{r}_i^{\mathcal{G},P}$ for each plate i with $i = 0, \ldots, N_P$. See Figure 3.2(b). The pose of each SP can be expressed in terms of the pose of its top plate in the reference frame of its bottom plate. We primarily use this local 'SP-frame' to mathematically define a kinematically valid SP. See Figure 3.2(c) for examples of local reference frame variables.

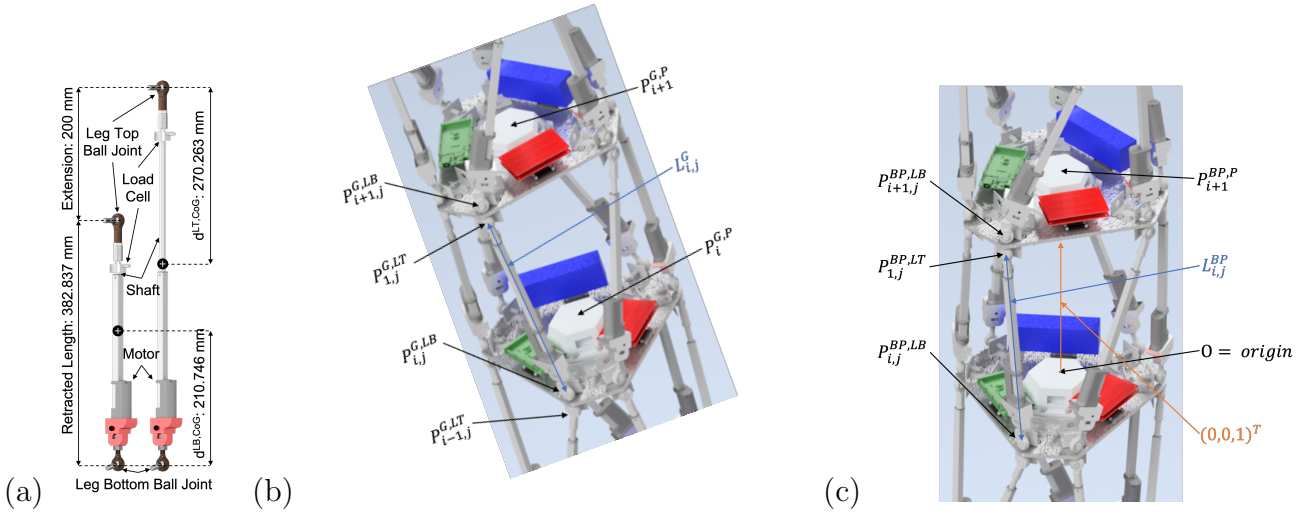
Note that, crucially, a top plate pose is not uniquely defined by only the leg lengths [102], [103].

An important pose is the *resting pose*. We define this as the pose where all the legs are set to be 50% of their total possible length, as a heuristic estimate of the center of a Stewart platform's workspace. We define parameters $\boldsymbol{L}^{\text{rest}}$ for each leg as the vector that leg follows when in resting pose. Note that, in the resting pose, due to the symmetry of the SP design used here, each plate is translated vertically with no rotation, so that $\mathbf{r}_i = \mathbb{0}$ and $\mathbf{p}_{i,[1:2]} = \mathbb{0}$.

We use $\mathbb{0}$ to denote a vector or matrix of zeros, and we leave size of this vector/matrix to be deduced by context.

Given a goal position for the end effector of the assembler, the goal of this work is to quickly find kinematically valid poses and motions for a given assembler while controlling the worst-case forces on the legs. We consider only mass-related forces and external forces applied to the end-effector of the assembler, and neglect any external forces applied to other parts of the assembler by e.g. wind.

To resolve torques resulting from the masses of each leg, we must compute the center of gravity



(a) (b) (c)

Figure 3.2: (a) Image of a linear actuator in two positions. The legs of the assembler are made from these linear actuators. Linear actuator has two parts: the motor (on the leg bottom) and the shaft (on the leg top). The actuator works by extending and contracting the shaft. For force computations, we record as data the distance from the bottom of the leg to the center of gravity of the motor, and from the top of the leg to the center of gravity of the shaft. (b) A section of the assembler with several variable locations labeled in the global reference frame. Note that the center of gravity of plate i is $\mathbf{p}_i^{\mathcal{G},P}$. Variables in the global reference frame are denoted that a superscript $\mathcal{G}$. (c) The same pose, but rotated to look from the reference frame $\mathcal{BP}$ of SP i with respect to the bottom plate. We also occasionally use $\mathcal{TP}$ to denote the reference frame with respect to the top plate of the SP.

(CoG) for each part the leg. Crucially, the CoG for a leg is not in the center of each leg; rather, each leg is a linear actuator consisting of two connected parts, a motor and a shaft. See Figure 3.2(a). The CoG of each part is a fixed distance from its attachment location. This fixed-distance property gives the model for the CoG a measure of mathematical complexity, as one cannot simply multiply the vector defining the leg's position by a fixed number to obtain the position of the center of gravity for each part. Rather, the unit vector defining the direction of the leg is multiplied by the fixed distance from the attachment location of a part to its CoG. In this work, the motor of each leg is connected to the bottom plate of its SP, while the shaft is connected to the top plate.

In the remainder of this section, we first define our notation and mathematics for coordinate transformations. We then define the Forward Kinematics and Inverse Kinematics problems for an assembler and for a single Stewart platform, define a kinematically valid Stewart Platform, and give some reasoning for the use of four Stewart platforms for the assembler in this work.

3.2.1 Notation

In this section, we introduce some useful notation and abbreviations to be used in the remainder of this work.

Shorthand: Here, we summarize our shorthand notation. TAA, SP, and KVC are used in text descriptions, while the rest are used in variable superscripts as object specifiers.

TAA : Translation-Axis-Angle

SP : Stewart Platform

KVC : Kinematic Validity Constraint

CoG : Center of Gravity

BP : Bottom Plate of SP

TP : Top Plate of SP

P : Plate

LB : Leg Bottom

LT : Leg Top

For parameter and variable definitions, we use the following notation,

$$VarType^{\text{RefFrame(Optional)},\text{DesiredObject},\text{specifier}(Optional)}_{VarIndex,[ValIndex]}, \tag{3.1}$$

where RefFrame is the reference frame, $VarIndex$ gives indices for the related object, and $ValIndex$ gives vector and matrix indices. We implicitly define the reference frame of a plate by its leg attachment locations. The reference frame for each plate of an SP has its origin at the center of its exterior surface, the surface opposite its leg joints. In the assembler stack, the top plate of each SP joins the bottom plate of the next at the plate centers, so that so that the two plates share the same origin, with the top SP rotated by 30 degrees. To simplify calculations, we treat the top bottom plate of a SP and the top plate of the preceding SP as a single plate, providing only a single reference frame. This is accomplished by rotating the leg attachment locations for the bottom SP by 30 degrees about the z-axis to obtain the leg attachment locations for the top SP.

Reference Frames These three reference frames are used throughout the paper.

$\mathcal{BP}$: 'SP' frame, the reference frame of the Bottom Plate of an SP

$\mathcal{TP}$: Reference frame of the Top Plate of an SP

$\mathcal{G}$: 'Global' assembler frame, the reference frame of the bottom plate of the bottom SP

Miscellaneous notation We aggregate some useful miscellaneous notation here for reference.

$\mathbb{0}$	A vector or matrix of zeros.
$\mathbb{I}^3$	The 3×3 identity matrix.
$\diamond$	The operator to apply a coordinate transformation T to a point $\mathbf{p}$.
$\hat{\boldsymbol{e}}_k$	The standard unit vectors for $k = 1, 2, 3$, such that $\hat{\boldsymbol{e}}_{k,[k]} = 1$ and all other $\hat{\boldsymbol{e}}_{k,[j]} = 0$.

3.2.2 Coordinate transformations

We use $\|\cdot\|$ to denote the 2-norm (Euclidean norm). For a vector $\boldsymbol{v}$, we use $\hat{\boldsymbol{v}}$ to denote $\boldsymbol{v}/\|\boldsymbol{v}\|$, that is, the unit direction of $\boldsymbol{v}$. We define $\hat{\boldsymbol{e}}_k \in \mathbb{R}^3$ for $k = 1, 2, 3$ as the standard unit vectors with indices starting at 1. That is, $\hat{\boldsymbol{e}}_1 = [1, 0, 0]^\top$, $\hat{\boldsymbol{e}}_2 = [0, 1, 0]^\top$, and $\hat{\boldsymbol{e}}_3 = [0, 0, 1]^\top$.

We use $\boldsymbol{\Upsilon}$ to denote coordinate transformations via a vector containing the Translation-Axis-Angle representation (TAA)

$$\boldsymbol{\Upsilon} = \begin{pmatrix} \mathbf{p} \\ \mathbf{r} \end{pmatrix} = \begin{pmatrix} \mathbf{p} \\ \|\mathbf{r}\|\hat{\mathbf{r}} \end{pmatrix} = \begin{pmatrix} \mathbf{p} \\ \theta\hat{\mathbf{r}} \end{pmatrix}. \tag{3.2}$$

The TAA format is used to represent position $\mathbf{p}$ and orientation data $\mathbf{r}$ in place of transformation matrices because total translational and rotational errors can be found by taking the norm of the top 3 and the bottom 3 components, respectively.

The related rotation matrix can be defined via Rodrigues' formula as

$$\mathbf{R} := e^{[\mathbf{r}]} = I + \sin(\|\mathbf{r}\|)\frac{[\mathbf{r}]}{\|\mathbf{r}\|} + (1 - \cos(\|\mathbf{r}\|))\frac{([\mathbf{r}])^2}{\|\mathbf{r}\|^2}, \tag{3.3}$$

where $[\mathbf{r}]$ is the vector cross-product operator for $\mathbf{r}$

$$[\mathbf{r}] := \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}$$

so that, for any vector $\mathbf{v}$, we have $[\mathbf{r}]\mathbf{v} = \mathbf{r} \times \mathbf{v}$. We use the TAA scheme to represent reference

frames in space due to its compactness, with the theoretically minimal six degrees of freedom, combined with the mathematical ease of converting the translational and rotational components to matrix transformation form. The matrix $\mathbf{R}$ is used to define transformation matrices via

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{p} \\ \mathbb{0} & 1 \end{bmatrix}. \tag{3.4}$$

To apply the coordinate transformation defined by transformation matrix $\mathbf{T}$ to a point $\tilde{\mathbf{p}}$, we use

$$\mathbf{T} \diamond \tilde{\mathbf{p}} := \left(\mathbf{T} \begin{pmatrix} \tilde{\mathbf{p}} \\ 1 \end{pmatrix} \right)_{[1:3]} = \mathbf{R}\tilde{\mathbf{p}} + \mathbf{p}. \tag{3.5}$$

Where, for a vector $\boldsymbol{v}$, we use a bracketed MATLAB style subscript to denote components, for example, we define $\boldsymbol{v}_{[1:3]} = [v_1, v_2, v_3]^\top$.

3.2.3 Forward and Inverse Kinematics

There are two primary problems of interest to solve for a general robot with an end-effector: the Forward Kinematics (FK) and Inverse Kinematics (IK) problems. The FK problem is to compute the end-effector pose given the actuator lengths. Conversely, the IK problem is to compute the actuator values given the end-effector pose.

For a single SP, the FK problem corresponds to computing the pose of the top plate w.r.t. the bottom plate given a vector of leg lengths. However, in general, the FK problem is difficult: it has multiple reachable solutions [102], [103], most of which cannot be reversed, such as 'pretzeled' poses for which the SP has twisted excessively, and spirals downwards until the legs collide. On the other hand, the IK problem corresponds to computing the leg lengths of the SP given the pose of the top plate w.r.t. the bottom plate. This computation is straightforward, with a simple closed-form solution, as described below. In this section, we describe all computations in the reference frame of the SP.

Define the pose of the top plate of an SP as $\mathbf{T}^{\mathrm{TP}}$, and let J be the set of legs for the SP. Then, for each leg of an SP, there are corresponding rest positions for the leg's joint attachment locations to the top and bottom plates. For a leg $j \in J$, these are $\mathbf{p}_j^{\mathcal{TP},\mathrm{LT},\mathrm{rest}}$ for the top plate and $\mathbf{p}_j^{\mathrm{LB}}$ for the bottom plate. These locations are defined in the reference frame of the corresponding plate, and are required for leg-vector and leg-length computations. We compute the coordinates of the top-plate leg joint locations via

$$\mathbf{p}_j^{\mathrm{LT}} = \mathbf{T}^{\mathrm{TP}} \diamond \mathbf{p}_j^{\mathcal{TP},\mathrm{LT},\mathrm{rest}} \qquad j \in J. \tag{3.6}$$

Note that the bottom-plate joint attachment locations $\mathbf{p}^{\mathrm{LB}}$ are known constants. Once joint locations have been determined, it is straightforward to calculate the resulting leg lengths by taking the magnitude of the positional displacement between corresponding leg joint pairs, as

$$\boldsymbol{L}_j^{\mathrm{len}} = \|\mathbf{p}_j^{\mathrm{LT}} - \mathbf{p}_j^{\mathrm{LB}}\| \quad j \in J. \tag{3.7}$$

The vector of leg lengths $\left(\boldsymbol{L}_j^{\mathrm{len}}\right)_{j\in J}$ is then the solution to the single-SP IK problem.

Considering that IK on an SP is a single step mathematical computation [104], it lends itself to rapid successive iteration. Solutions to the FK problem for SPs are based in this principle, with most algorithms performing calls to IK in order to numerically approximate the true position. However, as FK is not an integral component to the assembler IK methodologies described in this paper, it is not described here in full. For further information, consult [105].

For an assembler consisting of a serial chain of stacked SP's, which are parallel kinematic structures, neither the FK or IK problems are straightforward to solve. For the IK problem, there is generally a continuous space (or set of disjoint continuous spaces) of feasible plate positions for a given end-effector position. Moreover, this space is difficult to characterize, and can contain many bad solutions, such as those with extreme SP poses or very high leg forces. As such, before computing the leg lengths, one must first choose a 'good' solution for the plate positions from the space

of feasible solutions. On the other hand, to solve the FK problem for an assembler, one must individually solve the difficult FK problem for each SP.

In this work, we focus on solving the IK problem for an assembler. We then extend our approach via a trust region method to solve point-to-point trajectory planning.

3.2.4 Kinematic validity constraints

In this section, we define what it means for an SP to have a valid pose. In this section, we will define constraints in the reference frame of the base plate of SP, so that the origin is the center of the bottom of the bottom plate of the SP, with no rotation. In effect, we omit the reference frame superscript $\mathcal{BP}$, as defined in the following section.

Leg Length bounds: The legs have minimum and maximum lengths that can be attained. For the robots we consider here, the legs all have uniform length bounds $L^{\min}$ and $L^{\max}$, but this is easily changed in our model if more general situations are of interest. Formally, for a particular leg, let $\mathbf{p}^{\mathrm{LT}}$ and $\mathbf{p}^{\mathrm{LB}}$ be the coordinates of the top and bottom connections of the leg. Then the vector $\boldsymbol{L} = \mathbf{p}^{\mathrm{LT}} - \mathbf{p}^{\mathrm{LB}}$ describes the direction and magnitude of the leg. This vector is then bounded in magnitude as

$$L^{\min} \leq \|\boldsymbol{L}\| \leq L^{\max}. \tag{3.8}$$

Note that, in the reference frame of the SP, the bottom-plate attachment locations $\mathbf{p}^{\mathrm{LB}}$ are known constants.

Leg Angular Deviation: Each leg must not exceed a certain angular deviation from its normal resting pose, as this could break the ball joints in physical hardware. We must constrain the leg lengths in both the top and bottom pose.

Formally, for a particular leg, let $\mathbf{p}^{\mathrm{LT,rest}}$ and $\mathbf{p}^{\mathrm{LB,rest}}$ be the rest coordinates of the top and bottom connections of the leg. Then the vector $\boldsymbol{L}^{\mathrm{rest}} = \mathbf{p}^{\mathrm{LT,rest}} - \mathbf{p}^{\mathrm{LB,rest}}$ describes the direction and magnitude of the leg in the resting pose.

The bottom-plate constraint for leg angle deviation is

$$\cos(\theta^{\max}) \leq \frac{\boldsymbol{L}}{\|\boldsymbol{L}\|} \cdot \frac{\boldsymbol{L}^{\text{rest}}}{\|\boldsymbol{L}^{\text{rest}}\|}. \tag{3.9}$$

Similarly, we require the same constraints for the top plate angles. Let $\mathbf{R}$ be the rotation matrix defining the orientation of the top plate w.r.t. the bottom plate. Then the top-plate angles are defined as in (3.9), except that the rest coordinates are pre-multiplied by $\mathbf{R}$ to move them to the top plate, yielding

$$\cos(\theta^{\max}) \leq \frac{\boldsymbol{L}}{\|\boldsymbol{L}\|} \cdot \frac{\mathbf{R}\boldsymbol{L}^{\text{rest}}}{\|\mathbf{R}\boldsymbol{L}^{\text{rest}}\|}.$$

Since rotation operations are distance-invariant, we have $\|\mathbf{R}\boldsymbol{L}^{\text{rest}}\| = \|\boldsymbol{L}^{\text{rest}}\|$, yielding

$$\cos(\theta^{\max}) \leq \frac{\boldsymbol{L}}{\|\boldsymbol{L}\|} \cdot \frac{\mathbf{R}\boldsymbol{L}^{\text{rest}}}{\|\boldsymbol{L}^{\text{rest}}\|}. \tag{3.10}$$

Legs Point Up: To prevent legs from colliding with the base plate of an SP, we require that each leg is pointing 'up', so that the z-component $\boldsymbol{L}_{[3]}$ of $\boldsymbol{L}$ is nonnegative:

$$\boldsymbol{L}_{[3]} \geq 0. \tag{3.11}$$

Extreme Pose Prevention: We wish to prevent extreme and difficult-to-reach poses for each SP, particularly 'pretzeling,' a phenomenon where the top plate of an SP over-rotates in the z-direction, causing it to collapse, spinning down until the legs of the SP collide. See e.g. [77], [103] for more information on singularities, such as pretzeling, that can be encountered with some Stewart Platform designs. To help prevent such extreme poses, we set a limit of $\theta^{\mathbf{R},\max}$ for the action of the plate rotation matrix $\mathbf{R}$ on any principle unit vector $\hat{\boldsymbol{e}}_k$, where $k \in \{1, 2, 3\}$. This corresponds to enforcing that each deviation angle $\theta_k^{\mathbf{R}} \leq \theta^{\mathbf{R},\max}$, which is equivalent to

$$\|\hat{\boldsymbol{e}}_k\|\|\mathbf{R}\hat{\boldsymbol{e}}_k\| \cos(\theta_k^{\mathbf{R}}) = \hat{\boldsymbol{e}}_k^{\top}\mathbf{R}\hat{\boldsymbol{e}}_k \geq \|\hat{\boldsymbol{e}}_k\|\|\mathbf{R}\hat{\boldsymbol{e}}_k\| \cos(\theta^{\mathbf{R},\max}).$$

Now, since $\|\hat{\boldsymbol{e}}_k\| = \|\mathbf{R}\hat{\boldsymbol{e}}_k\| = 1$, and $\hat{\boldsymbol{e}}_k^\top \mathbf{R}\hat{\boldsymbol{e}}_k = \mathbf{R}_{[k,k]}$, the kth diagonal element of $\mathbf{R}$, this simplifies to

$$\mathbf{R}_{[k,k]} \geq \cos(\theta^{\mathbf{R},\max}). \tag{3.12}$$

In this work, we use a maximum rotation of 60 degrees, or in radians, $\theta^{\mathbf{R},\max} = \frac{\pi}{3}$.

3.2.5 Problem Difficulty

For most cases, if forces are ignored, a feasible IK solution for the assembler can be found very quickly. For example, with N=4, and using the SP parameters and computers specified in the numerical results section, IPOPT will typically converge (or report a locally infeasible solution) within 0.1s, so that a feasible solution to reach goal poses in the workspace can usually be obtained via local optimization from several different initial solutions.

However, finding an globally optimal solution to the IK problem, in terms of e.g. minimizing the maximal leg forces, is far more difficult in general. Due to the nonconvexity of both the objective and constraints, there are often multiple locally optimal solutions. Moreover, these locally optimal solutions can differ greatly in solution quality, as seen in Figure 3.3. When attempting a direct global solve of our QCQP model in Section 3.3.4 with Gurobi 9.1.1, applied to the assembler with parameters defined in Section 3.5, even directly optimizing an assembler with only 2 SP's requires an inordinate amount of computational time (over an hour), despite the fact that the original problem has only 6 degrees of freedom, the pose of the middle plate, as the poses of the top and bottom plates are fixed.

Moreover, to demonstrate the potentially high number of locally optimal solutions, we optimized the 2-SP assembler with a goal pose of $\mathbf{p} = [0, 0, 0.65]^\top$ and $\mathbf{r} = [0, 0, 0]^\top$, so that a vertical pose is impossible due to the leg length lower-bounds. Using 100 randomized starting poses for the middle plate drawn from a normal distribution, and minimizing the maximal leg forces via IPOPT, we obtained 22 different locally optimal solutions, with maximum leg forces ranging from 457N-519N.

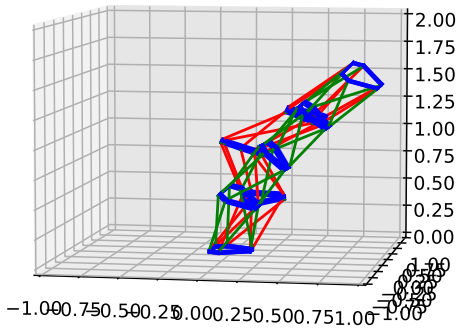


Figure 3.3: An example of a goal pose with two very different locally optimal solutions in IPOPT, for the assembler defined in Section 3.5 with a 5kg weight on the top plate. The red pose has a maximum leg force of 703N, while the green pose has a maximum leg force of only 282N.

3.2.6 Justification of Four SPs in a Stack

We choose the number of chained SP's for an assembler so that it can reach a 'bent-over' pose, with the end effector is in-plane with the bottom plate, facing downwards. This enables the assembler roughly a hemisphere of motion, allowing a reasonably large workspace while keeping internal forces under control. While adding additional SP's would increase the kinematic flexibility of motion in a zero-gravity environment, on Earth, it would also increase the loads on the legs particularly for the bottom SP, thereby shrinking the workspace of the assembler due to excessive forces.

For the specifications of the Stewart platform used in this work, due to leg length and joint motion limitations, a minimum of four chained SP's are required to reach this bent-over pose. Thus, we use four chained SP's for the computational tests in this work.

3.3 Assembler IK

3.3.1 Definitions

In this section, we formally define the notation used for variables and parameters needed for this work.

Sets

$i \in I \quad := 0, 1, \ldots, N_P$: The plates.

$i \in I^{\mathrm{P}} \quad := 0, 1, \ldots, N_P - 1$: The SPs. Platform i connects plates i and $i+1$.

$j \in J \quad := 1, 2, \ldots, 6$: The legs for each SP.

Parameters

To reduce notation definitions, we implicitly define some coordinate variables $\mathbf{p}$, rotational variables $\mathbf{r}$, and matrix variables $\mathbf{R}$ and $\mathbf{T}$ from $\boldsymbol{\Upsilon}$ via (3.2).

Parameter	Size	Description
$\boldsymbol{\Upsilon}^{\mathcal{G},\mathrm{goal}}$	$\mathbb{R}^6$	: Target global-frame TAA pose for the end effector
$\boldsymbol{\Upsilon}_i^{\mathcal{BP},\mathrm{TP,rest}}$	$\mathbb{R}^6$	: SP-frame rest pose of the top plate for SP $i \in I^P$
$\mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LB}}$	$\mathbb{R}^3$	: SP-frame position of bottom joint $j \in J$ for SP $i \in I^P$
$\mathbf{p}_{i,j}^{\mathcal{TP},\mathrm{LT,rest}}$	$\mathbb{R}^3$	: Rest position of top joint $j \in J$ w.r.t. top plate for SP $i \in I^P$
$\boldsymbol{L}_{i,j}^{\mathcal{BP},\mathrm{rest}}$		$:= \mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LT,rest}} - \mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LB}}$: SP-frame rest-position leg vector for leg $j \in J$ for SP $i \in I^P$
$\theta^{\max}$	$\mathbb{R}$	: Maximal angle deviation from rest position for any leg
$\theta^{\mathbf{R},\max}$	$\mathbb{R}$	: Maximal angle between $\mathbf{R}_i^{\mathrm{BP,TP}}\hat{\boldsymbol{e}}_k$ and $\hat{\boldsymbol{e}}_k$ for plate $i \in I^P$ and $k \in [1:3]$
$(L^{\min}, L^{\max})$	$\mathbb{R}$	: Bounds on the length of any leg
$f^{\max}$	$\mathbb{R}$	: Upper bound on the compressive and tensile force on a leg
m_i^{P}	$\mathbb{R}$	: Mass of plate $i \in I$
m^{LT}	$\mathbb{R}$	: Mass of a leg motor
m^{LB}	$\mathbb{R}$	: Mass of a leg shaft

$d^{\text{LT,CoG}}$	$\mathbb{R}$	: Distance from the top joint to the CoG for a leg shaft
$d^{\text{LB,CoG}}$	$\mathbb{R}$	: Distance from the bottom joint to the CoG for a leg motor
g	$\mathbb{R}$	: Gravitational constant. For this work, we use Earth gravity, $g \approx 9.81\frac{m}{s^2}$.

Note that, as the assembler studied in this work consists of N_P identical stacked Stewart platforms, there are really 2 plates between consecutive sets of legs, so that $m_i^P = 2m_0^P = 2m_{N_P}^P$ for $i = 1, 2, \ldots, N_P - 1$.

Variables We color variables in blue to help readability of formulas.

Variable	Size	Description
$\boldsymbol{\Upsilon}_i^{\mathcal{BP},\text{TP}}$	$\mathbb{R}^6$	: SP-frame pose of top plate for SP $i \in I^P$
$\boldsymbol{\Upsilon}_i^{\mathcal{G},\text{P}}$	$\mathbb{R}^6$	: Global-frame pose of plate $i \in I$, where $\boldsymbol{\Upsilon}_{N_P}^{\mathcal{G},\text{P}} = \boldsymbol{\Upsilon}^{\mathcal{G},\text{goal}}$
$\mathbf{p}_{i,j}^{\mathcal{BP},\text{LT}}$	$\mathbb{R}^3$	: SP-frame position of top joint $j \in J$ for SP $i \in I^P$
$\mathbf{p}_{i,j}^{\mathcal{G},\text{LB}}$	$\mathbb{R}^3$	: Global-frame position of bottom joint $j \in J$ for SP $i \in I^P$
$\mathbf{p}_{i,j}^{\mathcal{G},\text{LT}}$	$\mathbb{R}^3$	: Global-frame position of top joint $j \in J$ for SP $i \in I^P$
$\mathbf{p}_{i,j}^{\mathcal{G},\text{LB,CoG}}$	$\mathbb{R}^3$	: Global-frame CoG position of motor for leg $j \in J$ for SP $i \in I^P$
$\mathbf{p}_{i,j}^{\mathcal{G},\text{LT,CoG}}$	$\mathbb{R}^3$	: Global-frame CoG position of shaft for leg $j \in J$ for SP $i \in I^P$
$\boldsymbol{L}_{i,j}^{\mathcal{G}}$		$:= \mathbf{p}_{i,j}^{\mathcal{G},\text{LT}} - \mathbf{p}_{i,j}^{\mathcal{G},\text{LB}}$; Global-frame leg vector for leg $j \in J$, SP $i \in I^P$
$\boldsymbol{L}_{i,j}^{\mathcal{BP}}$		$:= \mathbf{p}_{i,j}^{\mathcal{BP},\text{LT}} - \mathbf{p}_{i,j}^{\mathcal{BP},\text{LB}}$; SP-frame leg vector for leg $j \in J$, SP $i \in I^P$
$\boldsymbol{L}_{i,j}^{\text{len}}$		$:= \|\boldsymbol{L}^{\mathcal{BP}}\|$; Leg length for leg $j \in J$, SP $i \in I^P$
$\tau_{i,j}$	$\mathbb{R}$	: Gravitational force on leg $j \in J$ for SP $i \in I^P$
$\tau^{\max}$	$\mathbb{R}$	: Maximum force on any leg of the assembler
W_i	$\mathbb{R}^6$	: Global-frame wrench forces on the top plate of SP $i \in I^P$
		$W_i = [W_i^R, W_i^P]^\top$, where W_i^R relates to the torque and W_i^P is the force.
${J^{\text{s}}}_i$	$\mathbb{R}^{6\times6}$	: Spatial Jacobian related to leg-force computations for SP $i \in I^P$
${J^{\text{t}}}_i$		$:= ({J^{\text{s}}}_i)^{-\top}$

We assume that the assembler is oriented so that gravity pulls directly downward in the reference frame of the assembler, i.e. $\boldsymbol{g} = [0, 0, -g]^\top$. To handle different orientations of the assembler, one needs only to redefine the gravity vector $\boldsymbol{g}$. Note that each $\boldsymbol{\Upsilon}$ transformation term implicitly defines corresponding TAA-form terms $\mathbf{p}$ and $\mathbf{r}$, and matrix-form terms $\mathbf{R}$ and $\mathbf{T}$. Finally, as the bottom plate is positioned at the origin of the global reference frame, we enforce that $\boldsymbol{\Upsilon}_0^{\mathcal{G},\mathrm{P}} = \boldsymbol{\Upsilon}_0^{\mathcal{G},\mathrm{P,rest}} = \mathbb{0}$.

In order to define a kinematically valid assembler pose, we enforce the SP-frame kinematic validity constraints constraints in Section 3.2.4 for each SP, combined with the definition $\boldsymbol{\Upsilon}_{N_P}^{\mathcal{G},\mathrm{P}} = \boldsymbol{\Upsilon}^{\mathcal{G},\mathrm{goal}}$, which ensures that the end-effector is where it should be.

This constraint is enforced with a small implicit tolerance within the nonlinear solver.

3.3.2 Force Calculation

Force analysis follows the procedures set out in [104] with a few additional considerations.

For SP $i \in I^P$, we can define the Jacobian with the relationship

$$J_i^{\mathrm{s}-1} = \begin{bmatrix} \mathbf{p}_{i,1}^{\mathcal{G},\mathrm{LB}} \times \hat{\boldsymbol{L}}_{i,1}^{\mathcal{G}} & \dots & \mathbf{p}_{i,6}^{\mathcal{G},\mathrm{LB}} \times \hat{\boldsymbol{L}}_{i,6}^{\mathcal{G}} \\ \hat{\boldsymbol{L}}_{i,1}^{\mathcal{G}} & \dots & \hat{\boldsymbol{L}}_{i,6}^{\mathcal{G}} \end{bmatrix}^\top \tag{3.13}$$

where $\hat{\boldsymbol{L}}_{i,j}^{\mathcal{G}}$ is the unit vector for $\boldsymbol{L}_{i,j}^{\mathcal{G}}$.

Given a Wrench W defined in the global frame and acting on the SP end effector, the resultant forces on each of the SP's legs can be determined by the relation:

$$\boldsymbol{\tau}_{i,[1:6]} = (J^{\mathrm{s}}{}_i)^\top (\mathrm{Adj}(\mathbf{T}_i^{\mathcal{G},\mathrm{TP}}))^\top W_i \tag{3.14}$$

where, as in [104], we define the matrix adjoint for a transformation matrix $\mathbf{T}$ (see (3.4)) as

$$\mathrm{Adj}(\mathbf{T}) := \begin{bmatrix} \mathbf{R} & \mathbb{0} \\ [\mathbf{p}]\mathbf{R} & \mathbf{R} \end{bmatrix} \tag{3.15}$$

For wrench computations, we first define the center of gravity for the motor and shaft of each leg $j \in J$ for SP $i \in I^P$ as

$$\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB,CoG}} = d^{\mathrm{LB,CoG}} \frac{\boldsymbol{L}_{i,j}^{\mathcal{G},\mathrm{LT}}}{\|\boldsymbol{L}_{i,j}^{\mathcal{G},\mathrm{LT}}\|} + \mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB}} \quad \text{and} \quad \mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LT,CoG}} = d^{\mathrm{LT,CoG}} \frac{-\boldsymbol{L}_{i,j}^{\mathcal{G},\mathrm{LT}}}{\|\boldsymbol{L}_{i,j}^{\mathcal{G},\mathrm{LT}}\|} + \mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LT}}. \tag{3.16}$$

Then, to compute the global-frame wrenches W_i for each Stewart platform, we sum all the wrenches from forces applied on or above the platform by leg masses, platform masses, and the end effector load. For $i = 0, \ldots, N_P - 2$, this is computed as

$$W_i = W_{i+1} + m_{i+1}^P \begin{bmatrix} [\mathbf{p}_{i+1}^{\mathcal{G},\mathrm{P}}]\mathbf{g} \\ \mathbf{g} \end{bmatrix} + \sum_j \left(m^{\mathrm{LT}} \begin{bmatrix} [\mathbf{p}_{i+1,j}^{\mathcal{G},\mathrm{LT,CoG}}]\mathbf{g} \\ \mathbf{g} \end{bmatrix} + m^{\mathrm{LB}} \begin{bmatrix} [\mathbf{p}_{i+1,j}^{\mathcal{G},\mathrm{LB,CoG}}]\mathbf{g} \\ \mathbf{g} \end{bmatrix} \right). \tag{3.17}$$

For the last platform $i = N_P - 1$, this is computed as

$$W_{N_P-1} = W^{\mathrm{EE}} + m_{N_P}^P \begin{bmatrix} [\mathbf{p}_{N_P}^{\mathcal{G},T}]\mathbf{g} \\ \mathbf{g} \end{bmatrix}. \tag{3.18}$$

Note that, as $\mathbf{g} = (0, 0, -g)^\top$, for any vector $\mathbf{p} \in \mathbb{R}^3$, we have

$$[\mathbf{p}]\mathbf{g} = g \begin{bmatrix} -\mathbf{p}_{[1]} \\ \mathbf{p}_{[2]} \\ 0 \end{bmatrix}.$$

Consequently, as W^{EE} is a parameter, and since the 3rd component of $[\mathbf{p}]\boldsymbol{g}$ is zero, the 3rd-6th components of the wrench are known constants, while the first two depend linearly on the plate

and leg locations.

3.3.3 IK Heuristics

In this section, we introduce two heuristics for the initialization of Stewart platform poses.

Spline-Based IK

In this section, we introduce a splined kinematic approach, denoting a cubic spline originating at the platform base plate and ending at the end effector position. Define the following positions as helper points $\mathbf{p}^{\mathcal{G},1}$ and $\mathbf{p}^{\mathcal{G},2}$ from the formulas:

$$\mathbf{T}^{\mathcal{G},1} := \mathbf{T}^{\text{goal}} \left(-\tfrac{2}{3} \mathbf{T}_0^{\mathcal{TP},\text{BP,rest}} \right), \quad \mathbf{T}^{\mathcal{G},2} := \mathbf{T}^{\text{Base}} \left(\tfrac{2}{3} \mathbf{T}_{N_P}^{\mathcal{TP},\text{BP,rest}} \right). \tag{3.19}$$

Define the B-spline $p\colon [0,1] \to \mathbb{R}^3$ as

$$p(x) := \text{Spline}\left(\mathbf{p}^{\mathcal{G},\text{Base}}, \mathbf{p}^{\mathcal{G},1}, \mathbf{p}^{\mathcal{G},2}, \mathbf{p}^{\mathcal{G},\text{goal}} \right). \tag{3.20}$$

We use SciPy's B-spline interpolation [106], which requires a minimum of four points to produce the spline curve, these two helper points serve a dual purpose: ensure that spline function has enough input to produce the expected curve, and also to ensure that interior plates are placed behind the plane of the goal end effector, and above the plane of the base plate.

The resulting spline function provides the positions of the interior $N_P - 1$ plates via interpolation at $\mathbf{p}_i^{\mathcal{G},\text{P}} = p\left(\frac{i}{N_P}\right)$, for $i = 0, 1, \ldots, N_P$. We then recursively determine the rotation of the middle-most plate, and when there are an even number of plates left in the queue, we consider the two middle-most plates. The rotation of the middle-most plate(s) is then determined by an average of the rotation between the beginning and end plates, computed in TAA form in the reference frame of the beginning plate. Once the middle plate(s) rotation is determined, we recurse and determine

rotation of the next middle plate(s).

Within this heuristic, we consider a pose to be valid if all kinematic validity constraints, including the end-effector position, are satisfied within some small tolerance. If a pose fails kinematically, we (try the recourse stuff depending on what failed).

In the event of failure (such as a calculated leg being too long), all legs are re-scaled such that the legs are within bounds, maintaining orientation, and the end effector position is recalculated accordingly.

Validation of the Stewart platform proceeds in steps, described in the following algorithm.

Same-SP Initialization

We derive a simple initial guess for the assembler pose by assuming that each SP has the same SP-frame pose $\boldsymbol{\Upsilon}_i^{\mathcal{BP},\mathrm{TP}} = \boldsymbol{\Upsilon}$. It is then trivial to compute the SP-frame rotation matrices, as since all share the same axis of rotation, we have for two rotations $\mathbf{r}_i^{BP,TP}$ and $\mathbf{r}_j^{BP,TP}$ that

$$\mathbf{R}_i^{\mathrm{BP,TP}}\mathbf{R}_j^{\mathrm{BP,TP}} = e^{[\mathbf{r}_i^{BP,TP}]}e^{[\mathbf{r}_j^{BP,TP}]} = e^{[\mathbf{r}_i^{BP,TP}+\mathbf{r}_j^{BP,TP}]}.$$

Thus, if $\mathbf{R}_i^{\mathrm{BP,TP}} = \mathbf{R}$ for all $i \in I^P$, we obtain

$$\mathbf{R}_{NP}^{\mathcal{G},\mathrm{P}} = e^{[\mathbf{r}_{N_P}^{\mathcal{G},P}]} = e^{[N_P\mathbf{r}]} = \mathbf{R}^{NP}.$$

Thus, we simply compute $\mathbf{r}$ as

$$\mathbf{r} = \tfrac{1}{N_P}\mathbf{r}_{N_P}^{\mathcal{G},P}.$$

Next, to compute the shared translation vector $\mathbf{p}$, we first note that

$$\mathbf{R}_i^{\mathcal{G},\mathrm{P}} = \mathbf{R}^i$$

and

$$\begin{aligned}
\mathbf{p}_1^{\mathcal{G},\mathrm{P}} &= \mathbf{p} \\
\mathbf{p}_2^{\mathcal{G},\mathrm{P}} &= \mathbf{p}_1^{\mathcal{G},\mathrm{P}} + \mathbf{R}_1^{\mathcal{G},\mathrm{P}}\mathbf{p} = (\mathbb{I}^3 + \mathbf{R})\mathbf{p} \\
\mathbf{p}_3^{\mathcal{G},\mathrm{P}} &= \mathbf{p}_2^{\mathcal{G},\mathrm{P}} + \mathbf{R}_2^{\mathcal{G},\mathrm{P}}\mathbf{p} = (\mathbb{I}^3 + \mathbf{R} + \mathbf{R}^2)\mathbf{p} \\
\mathbf{p}_i^{\mathcal{G},\mathrm{P}} &= \left(\sum_{l=0}^{i-1} \mathbf{R}^l\right)\mathbf{p} \qquad\qquad i \in I,
\end{aligned}$$

where $\mathbb{I}^3$ is the 3×3 identity matrix, and notable that $\mathbf{R}^0 = \mathbb{I}^3$ for any invertible 3×3 matrix $\mathbf{R}$. We then obtain $\mathbf{p}$ via a single cheap linear solve. To summarize, we compute $\mathbf{r}$ and $\mathbf{p}$, and the resulting $\mathbf{R}_i^{\mathcal{G},\mathrm{P}}$ and $\mathbf{p}_i^{\mathcal{G},\mathrm{P}}$'s, via

$$\begin{aligned}
\mathbf{r} &= \tfrac{1}{N_P}\mathbf{r}_{N_P}^{\mathcal{G},P} \\
\mathbf{R} &= e^{[\mathbf{r}]} \\
\mathbf{p}^{\mathcal{G},\mathrm{P},\mathrm{goal}} &= \left(\sum_{i=0}^{N_P-1} \mathbf{R}^i\right)\mathbf{p} \\
\mathbf{R}_i^{\mathcal{G},\mathrm{P}} &= \mathbf{R}^i \qquad\qquad i \in I \\
\mathbf{p}_i^{\mathcal{G},\mathrm{P}} &= \left(\sum_{l=0}^{i-1} \mathbf{R}^l\right)\mathbf{p} \qquad i \in I
\end{aligned} \tag{3.21}$$

where, again, $\mathbf{p}$ is computed via a linear solve in the third equation. Note that this initialization can correspond to multiple solutions, as e.g. a 180 degree z-rotation can be achieved via four 45-degree or four -45-degree z-rotations. Some examples of this phenomenon are shown in Figure 3.4. As such, we try up to two solutions in TAA form. Given $\boldsymbol{\Upsilon}_0^{\mathcal{G},\mathrm{goal}}$, we first normalize $\|\mathbf{r}_0^{\mathcal{G},\mathrm{goal}}\|$ to ensure $\|\mathbf{r}_0^{\mathcal{G},\mathrm{goal}}\| < 2\pi$. To this end, if $\|\mathbf{r}_0^{\mathcal{G},\mathrm{goal}}\| \in [2k\pi, 2(k+1)\pi)$ for some integer $k \geq 1$, we apply

$$\mathbf{r}_0^{\mathcal{G},\mathrm{goal}} \leftarrow \mathbf{r}_0^{\mathcal{G},\mathrm{goal}} \frac{\|\mathbf{r}_0^{\mathcal{G},\mathrm{goal}}\| - 2k\pi}{\|\mathbf{r}_0^{\mathcal{G},\mathrm{goal}}\|}.$$

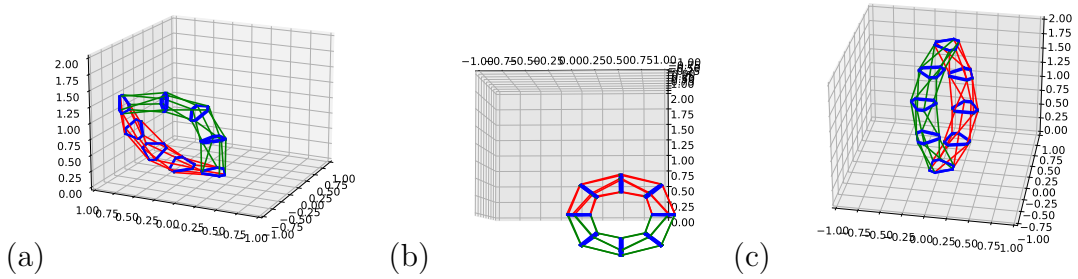
Then, if we find that the resulting SP-frame translation vector $\mathbf{p}^{\mathcal{BP},\mathrm{TP}}$ is too far (more than 60 degrees) from vertical, which is true if and only if

$$\mathbf{p}_{[2]} \leq \|\mathbf{p}\| \cos(\tfrac{\pi}{3}),$$

we reject the solution, then try a second one. We obtain this second solution by reflecting the rotation $\|\mathbf{r}_0^{\mathcal{G},\text{goal}}\|$ about π to achieve the same rotation from the opposite direction, via

$$\mathbf{r}_0^{\mathcal{G},\text{goal}} \leftarrow \mathbf{r}_0^{\mathcal{G},\text{goal}} \frac{2\pi - \|\mathbf{r}_0^{\mathcal{G},\text{goal}}\|}{\|\mathbf{r}_0^{\mathcal{G},\text{goal}}\|}.$$

We have found computationally that this procedure consistently yields useful initial guesses for the optimizer, though the guesses are often kinematically invalid.



(a) (b) (c)

Figure 3.4: Some examples of pose initialization results. The red poses are the originals, while the green poses are reflected. (a) First try yields extreme angles and flattened SP's; (b) reflected try yields inverted pose with legs clipping through the plates; (c) initializations yield mirrored poses.

3.3.4 Force-Based IK Optimization

We formulate the optimization problem for a stacked SP assembler as a quadratically constrained quadratic program (QCQP). In this model, $\mathbf{R}_i^{\mathcal{G},\text{P}}$ and $\mathbf{p}_i^{\mathcal{G},\text{P}}$ are the driving decision variables, in that they uniquely specify a pose. The rest of the variables are derived directly from these. Note that, due to our choice of reference frame, we have $\mathbf{R}_0^{\mathcal{G},\text{P}} = e^{[\mathbb{0}]} = \mathbb{I}^3$, where $\mathbb{I}^3$ is the 3×3 identity matrix.

Constraints related to kinematic validity are denoted via **(KVC)**; all other constraints establish definitions of intermediate variables.

Equations in this section are either labeled as $(M\#)$ to denote that they are used explicitly in the model, or as $(\#)$, to denote that this is just a calculation useful to deriving the model equations.

Additional Definitions

For the optimization model, we use all variables and parameters in Section 3.3.1 except J_i^{s}, including only $\mathbf{R}$ and $\mathbf{p}$ for each $\boldsymbol{\Upsilon}$ variable or parameter, and add the following additional variables. For clearer distinction between variables and parameters within this model, we will represent all variables using blue text.

Constraints

Vector norms To model the two-norm $\|\boldsymbol{L}_{i,j}^{\mathcal{BP}}\|$ of the leg length vector (which is equivalent to $\|\boldsymbol{L}_{i,j}^{\mathcal{G}}\|$), we introduce intermediate variable $\boldsymbol{L}_{i,j}^{\mathrm{len}}$, then add the constraint

$$\boldsymbol{L}_{i,j}^{\mathrm{len}^2} = \sum_{k=1}^{3} \boldsymbol{L}_{i,j,[k]}^2 \quad \forall i \in I^P, j \in J. \tag{M1}$$

Moreover, any expressions of the form $\|\boldsymbol{v}\|^2$, $\boldsymbol{v} \in \mathbb{R}^m$, or $\|\mathbf{R}\|_F^2$, $\mathbf{R} \in \mathbb{R}^{m\times m}$, as in (M4), (T4), and (T1), are substituted explicitly with the defining expressions

$$\|\boldsymbol{v}\|^2 = \sum_{k=1}^{m} \boldsymbol{v}_{[k]}^2, \tag{3.22a}$$

$$\|\mathbf{R}\|_F^2 = \sum_{k=1}^{m}\sum_{l=1}^{m} \mathbf{R}_{[k,l]}^2. \tag{3.22b}$$

Reference Frame Computations: The constraints in this section are needed to define $\mathbf{R}_i^{\mathrm{BP,TP}}$'s and $\mathbf{R}_i^{\mathcal{G},\mathrm{P}}$'s. First, we express the global rotation matrices via their columns as

$$\mathbf{R}_i^{\mathcal{G},\mathrm{P}} = \left[\begin{array}{c|c|c} \mathbf{R}_{i,[:,1]}^{\mathcal{G},\mathrm{P}} & \mathbf{R}_{i,[:,2]}^{\mathcal{G},\mathrm{P}} & \mathbf{R}_{i,[:,3]}^{\mathcal{G},\mathrm{P}} \end{array}\right] \quad i \in I. \tag{M2}$$

Note that, as rotation matrices define an orthonormal right-handed coordinate system, it is sufficient

to consider $\mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,1]}$ and $\mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,2]}$ as the driving variables, then compute $\mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,3]}$ as

$$\mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,3]} = \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,1]} \times \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,2]}. \tag{M3}$$

We then ensure the orthonormality of each $\mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,1]}$ and $\mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,2]}$ via

$$\begin{aligned} \left\| \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,1]} \right\|^2 &= 1, \\ \left\| \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,2]} \right\|^2 &= 1, \\ \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,1]} \cdot \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i,[:,2]} &= 0. \end{aligned} \tag{M4}$$

Next, to obtain the translation portion of each plate transformation in the SP frame, we inverse transform the global-frame point $\mathbf{p}^{\mathcal{G}}_i$ by $\mathbf{T}^{\mathcal{G}}_i$ by solving the forward transformation $\mathbf{p}^{\mathcal{G}}_i = \mathbf{T}^{\mathcal{G}}_i \diamond \mathbf{p}^{\mathcal{BP},\mathrm{TP}}_i$, as defined in (3.5), for the pre-transformed point $\mathbf{p}^{\mathcal{BP},\mathrm{TP}}_i$, yielding

$$\mathbf{p}^{\mathcal{BP},\mathrm{TP}}_i = (\mathbf{R}^{\mathcal{G},\mathrm{P}}_i)^\top (\mathbf{p}^{\mathcal{G},\mathrm{P}}_{i+1} - \mathbf{p}^{\mathcal{G},\mathrm{P}}_i) \quad i \in I^P. \tag{M5}$$

We constrain that the bottom plate is at the origin in the global assembler frame via

$$\begin{aligned} \mathbf{p}^{\mathcal{G},\mathrm{P}}_0 &= \mathbb{0} \\ \mathbf{R}^{\mathcal{G},\mathrm{P}}_0 &= \mathbb{I}^3. \end{aligned} \tag{M6}$$

Finally, we define the SP-frame rotation matrices $\mathbf{R}^{\mathrm{BP},\mathrm{TP}}_i$ via $\mathbf{R}^{\mathcal{G},\mathrm{P}}_i \mathbf{R}^{\mathrm{BP},\mathrm{TP}}_i = \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i+1}$, yielding

$$\mathbf{R}^{\mathrm{BP},\mathrm{TP}}_i = \left(\mathbf{R}^{\mathcal{G},\mathrm{P}}_i \right)^\top \mathbf{R}^{\mathcal{G},\mathrm{P}}_{i+1} \quad i \in I^P. \tag{M7}$$

Leg Attachment Locations: In (M8), we define the SP- and global-frame locations of the leg attachments on the top/bottom plates for each SP according to (3.6). Note that the position of the

bottom leg attachments are known constants in local space. See also Figure 3.2.

$$\begin{aligned}
\mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LT}} &= \mathbf{R}_i^{\mathrm{BP,TP}}\mathbf{p}_{i,j}^{\mathcal{TP},\mathrm{LT,rest}} + \mathbf{p}_i^{\mathcal{BP},\mathrm{TP}} && \forall i \in I^P, j \in J \\
\mathbf{p}_{0,j}^{\mathcal{G},\mathrm{LB}} &= \mathbf{p}_{0,j}^{\mathcal{G},B,\mathrm{rest}} && \forall j \in J \\
\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB}} &= \mathbf{R}_i^{\mathcal{G},\mathrm{P}}\mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LB}} + \mathbf{p}_i^{\mathcal{G}} && \forall i \in I^P, i \geq 1, j \in J \\
\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LT}} &= \mathbf{R}_{i+1}^{\mathcal{G},\mathrm{P}}\mathbf{p}_{i,j}^{\mathcal{TP},\mathrm{LT,rest}} + \mathbf{p}_{i+1}^{\mathcal{G}} && \forall i \in I^P, j \in J.
\end{aligned} \tag{M8}$$

Leg Centers of Gravity: To define the center of gravity for each leg $j \in J$ for SP $i \in I^P$, we start with (3.16), multiply through by denominators, and rearrange, yielding

$$\begin{aligned}
\boldsymbol{L}_{i,j}^{\mathrm{len}}(\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB,CoG}} - \mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB}}) &= d^{\mathrm{LB,CoG}}\boldsymbol{L}_{i,j}^{\mathcal{G}}, \\
\boldsymbol{L}_{i,j}^{\mathrm{len}}(\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LT}} - \mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LT,CoG}}) &= d^{\mathrm{LT,CoG}}\boldsymbol{L}_{i,j}^{\mathcal{G}}.
\end{aligned} \tag{M9}$$

Leg Length Bounds: **(KVC)** We define the leg length bounding constraints via (3.8), as

$$L^{\min} \leq \boldsymbol{L}_{i,j}^{\mathrm{len}} \leq L^{\max}. \tag{M10}$$

Note that the upper-bounding leg length constraints are second order cone constraints in terms of $\mathbf{p}_i^{\mathcal{BP},\mathrm{TP}}$ and $\mathbf{R}_i^{\mathcal{BP},\mathrm{TP}}$ (as the interiors of spheres), while the lower-bounding constraints are nonconvex as sphere exteriors.

Leg Angle Deviation: **(KVC)** The bottom-plate leg angle deviation constraints are defined via (3.9), after multplying through by denominators, as The bottom-plate leg angle deviation constraints are defined via (3.9), as

$$\boldsymbol{L}_{i,j}^{\mathrm{len}}\|\boldsymbol{L}_{i,j}^{\mathcal{BP},\mathrm{rest}}\|\cos(\theta^{\max}) \leq \boldsymbol{L}_{i,j}^{\mathcal{BP}} \cdot \boldsymbol{L}_{i,j}^{\mathcal{BP},\mathrm{rest}} \quad \forall i \in I^P, j \in J. \tag{M11}$$

while the top-plate constraints are defined via (3.10), after multiplying through by denominators, as

$$\boldsymbol{L}_{i,j}^{\mathrm{len}}\|\boldsymbol{L}_{i,j}^{\mathcal{BP},\mathrm{rest}}\|\cos(\theta^{\max}) \leq \boldsymbol{L}_{i,j}^{\mathcal{BP}} \cdot (\mathbf{R}_i^{\mathrm{BP,TP}}\boldsymbol{L}_{i,j}^{\mathcal{BP},\mathrm{rest}}) \quad \forall i \in I^P, j \in J. \tag{M12}$$

Note that all leg angle deviation constraints are second-order cone constraints given fixed rotation matrices.

Continuous Translation Enforcement: **(KVC)** We enforce that legs do not break the surface of the bottom plate of any SP via (3.11), as

$$\mathbf{p}^{\mathcal{BP},\mathrm{LT}}_{i,j,[3]} \geq \mathbf{p}^{\mathcal{BP},\mathrm{LB}}_{i,j,[3]} \tag{3.23}$$

Extreme Pose Prevention: **(KVC)** To help prevent extreme and difficult-to-reach poses for each SP, we enforce (3.12), as

$$\mathbf{R}^{\mathrm{BP,TP}}_{i,[k,k]} \geq \cos(\theta^{\mathbf{R},\max}), \quad k = 1,2,3 \tag{3.24}$$

End Effector: **(KVC)** We enforce that the end effector pose is exactly as desired via

$$\begin{aligned} \mathbf{p}^{\mathcal{G},\mathrm{P}}_{N_P-1} &= \mathbf{p}^{\mathcal{G},\mathrm{Pgoal}} \\ \mathbf{R}^{\mathcal{G},\mathrm{P}}_{N_P-1,[:,1]} &= \mathbf{R}^{\mathcal{G},\mathrm{P,goal}}_{[:,1]} \\ \mathbf{R}^{\mathcal{G},\mathrm{P}}_{N_P-1,[:,2]} &= \mathbf{R}^{\mathcal{G},\mathrm{P,goal}}_{[:,2]} \end{aligned} \tag{M13}$$

where $\mathbf{p}^{\mathcal{G},\mathrm{P,goal}}$ and $\mathbf{R}^{\mathcal{G},\mathrm{P,goal}}$ are computed from $\boldsymbol{\Upsilon}^{\mathcal{G},\mathrm{goal}}$.

Objective: The objective is to minimize maximal leg force

$$\min \tau^{\max}. \tag{M14}$$

where the constraints defining the maximum force $\tau^{\max}$ are introduced in Section 3.3.4.

Initialization:

As the model is solved only to local optimality via IPOPT due to the intractability of a global solve even with $N_P = 2$, an initial solution for the pose is required. We choose to initialize via the same-SP initialization scheme in Section 3.3.3, as it performed much better in our numerical

testing when compared to the spline-based scheme in Section 3.3.3, despite yielding valid poses less often before optimization.

Forces

Referring to (3.15), the *transposed adjoint* of a transformation matrix $\mathbf{T}$ is

$$\operatorname{Adj}(\mathbf{T})^\top = \begin{bmatrix} \mathbf{R}^\top & \mathbf{R}^\top[\mathbf{p}]^\top \\ \mathbb{0} & \mathbf{R}^\top \end{bmatrix}. \tag{3.25}$$

For shorthand, we let $\mathbf{A}_i := \operatorname{Adj}\left(\mathbf{T}_i^{\mathcal{G}}\right)^\top$ for each $i \in I$.

Each column of the transposed inverse Jacobian J^{t} as in (3.13) via (M15), can be computed as (see [104] for discussion on single SPs)

$$\boldsymbol{L}_{i,j}^{\mathrm{len}} J^{\mathrm{t}}{}_{i,[:,j]} = \begin{bmatrix} [\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB}}]\boldsymbol{L}_{i,j}^{\mathcal{G}} \\ \boldsymbol{L}_{i,j}^{\mathcal{G}} \end{bmatrix} \quad \forall i \in I^P, j \in J. \tag{M15}$$

Note that $\boldsymbol{L}_{i,j}^{\mathrm{len}} = \|\mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LT}} - \mathbf{p}_{i,j}^{\mathcal{G},\mathrm{LB}}\| = \|\mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LT}} - \mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LB}}\|$ is the length of the corresponding leg, as modelled in (M1).

To compute the global-frame wrenches for each Stewart platform, we use (3.17) for $i = 0, \ldots, N_P - 2$ and (3.18) for $i = N_P - 1$.

Finally, to compute the leg forces, and defining $\boldsymbol{\tau}_i = [\tau_{i,1}, \ldots, \tau_{i,6}]^\top$ for each i, we use the vector constraints

$$\begin{aligned} J^{\mathrm{t}}{}_i \boldsymbol{\tau}_i &= \mathbf{A}_i W_i \quad && \forall i \in I^P \\ \tau^{\max} &\geq \tau_{i,j} && \forall i \in I^P, j \in J \\ \tau^{\max} &\geq -\tau_{i,j} && \forall i \in I^P, j \in J \end{aligned} \tag{M16}$$

where the jth component of the resulting solution $\boldsymbol{\tau}_i$ is the force on the jth leg of the ith SP. We then minimize over τ.

Notice that $\mathbf{A}_i W_i$ can be written as:

$$\mathbf{A}_i W_i = \begin{bmatrix} (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top & (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top [\mathbf{p}_i^{\mathcal{G},\mathrm{P}}]^\top \\ \mathbb{0} & (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top \end{bmatrix} \begin{bmatrix} W_i^R \\ W_i^P \end{bmatrix} = \begin{bmatrix} (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top (W_i^R + [\mathbf{p}_i^{\mathcal{G},\mathrm{P}}]^\top W_i^P) \\ (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top W_i^P \end{bmatrix}.$$

Thus, the first constraint of (M16) is equivalent to, and implemented as,

$$J^{\mathrm{t}}{}_i \boldsymbol{\tau}_i = \begin{bmatrix} (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top (W_i^R + [\mathbf{p}_i^{\mathcal{G},\mathrm{P}}]^\top W_i^P) \\ (\mathbf{R}_i^{\mathcal{G},\mathrm{P}})^\top W_i^P \end{bmatrix} \quad i \in I^P. \tag{M17}$$

Note that this constraint is bilinear in nature, since W_i^P is a constant.

If $w_2 \neq 0$, we also enforce that the force on each leg does not exceed the maximum allowable force, via

$$\tau^{\max} \leq f^{\max} \tag{M18}$$

Improving Robustness

Due to the nature of iterative local nonlinear optimization via e.g. IPOPT, the equality constraints defining various intermediate variables such as leg lengths, rotation matrices, the inverse Jacobian etc, and even primary constraints such as end-effector position, can become violated as the solver attempts to resolve violations of the physical constraints defining a valid pose. We have observed that this can sometimes lead to instability within the optimizer, particularly if the end effector is moved from the goal position during optimization.

To address this instability while controlling for computational time, we implement an iterative-refinement scheme around the basic nonlinear optimizer. Here, we leverage the fact that the variables $\boldsymbol{\Upsilon}_i^{\mathcal{G},\mathrm{P}}$ uniquely define the assembler. Thus, if an optimization seems to be converging slowly or returns with a 'locally infeasible' status, we re-initialize the model every so often, and reset the end effector to the correct location.

To this end, we set the maximum total internal iteration count as 2500. Every k iterations, if the solver has not yet converged, we stop the solve, re-initialize using the plate locations and corrected the end effector location, then continue. We start with $k = 500$, but for each internal solver error we divide k in half and try again, with up to five such retries.

Occasionally, locally infeasible solutions can occur at valid poses, due to numerical difficulties in resolving force-related equality constraints. This results in sub-optimal, but kinematically valid, poses. When using IPOPT as the nonlinear solver, we have observed that the solver can often recover from such poses after re-initialization. Thus, to handle this contingency, on the first consecutive 'locally infeasible' result, we deduct k iterations from the remaining total (as if the solver had run k iterations) and re-initialize as usual. On the consecutive second locally infeasible result for the same pose, we report an optimization failure due to local infeasibility within the solver.

Extensions

Here, we describe some possible extensions to the model that could be useful to improve the stability of the optimized pose solutions.

Objective: In the objective, we could take into consideration both maximal leg force and the angular deviation λ_i of the consecutive plates from the resting poses. In this case, the objective becomes

$$\min w_1 \tau^{\max} + w_2 \|\boldsymbol{\lambda}\|^2, \tag{3.26}$$

where λ_i for $i = 1, \ldots, N_p - 1$ can be formally defined as

$$\|\mathbf{p}_i^{\mathcal{BP},\mathrm{TP}}\| \|\mathbf{p}_i^{\mathcal{BP},\mathrm{TP,rest}}\| \cos(\lambda_i) = \mathbf{p}_i^{\mathcal{BP},\mathrm{TP}} \cdot \mathbf{p}_i^{\mathcal{BP},\mathrm{TP,rest}}. \tag{3.27}$$

Now, as the model for λ_i^2 is highly nonlinear, we instead model 1-cos(λ_i), as it is nonnegative, zero at $\lambda_i = 0$, and approximately quadratic for λ_i near zero. Indeed, for small λ, we have

$\|\lambda\|^2 \approx 2\|1-\cos(\lambda)\|_1$, which is supported element-wise by the fact that $\lim_{x\to 0} \frac{1-\cos(x)}{x^2} = \frac{1}{2}$. Thus, for the objective, we use

$$\min w_1\tau + w_2 \sum_{i\in I^P} x_i, \tag{3.28}$$

where x_i is defined from $\cos(\lambda_i)$ via the second-order cone constraint.

$$x_i \geq (2(1-\cos(\lambda_i)))^2 \quad \forall i \in I^P. \tag{3.29}$$

Note that one only needs to model the lower bound on x_i in (3.27), as the optimization will seek to minimize x_i in order to maximize $\cos(\lambda_i)$. Furthermore, we replace $\cos(\lambda_i)$ with an intermediate variable $c\lambda_i$, defined via (3.27) as

$$\|\mathbf{p}_i^{\mathcal{BP},\mathrm{TP}}\|\|\mathbf{p}_i^{\mathcal{BP},\mathrm{TP,rest}}\|c\lambda_i = \mathbf{p}_i^{\mathcal{BP},\mathrm{TP}} \cdot \mathbf{p}_i^{\mathcal{BP},\mathrm{TP,rest}}. \tag{3.30}$$

Note that we require an intermediate variable and constraint to model $\|\mathbf{p}_i^{\mathcal{BP},\mathrm{TP}}\|$.

In practice, setting $w_2 > 0$ can help improve the stability and manipulability of the assembler in the resulting poses.

Sensitivity to end-effector forces: To improve the robustness of the pose to internal forces, we can add phantom forces to each leg corresponding with the sensitivity of the leg to the force applied to the end effector.

To compute the sensitivity of the force on leg i to the force $\boldsymbol{F}^{\mathrm{EE}}$ applied on the end effector, we first note

$$\begin{aligned} W^{R,\mathrm{EE}} &= [\mathbf{R}_{N_P}^{\mathcal{G},\mathrm{P}} \boldsymbol{v}^{\mathcal{TP},m,\mathrm{EE}} + \mathbf{p}_{N_P}^{\mathcal{G}}]\boldsymbol{F}^{\mathrm{EE}} \\ W^{P,\mathrm{EE}} &= \boldsymbol{F}^{\mathrm{EE}}, \end{aligned} \tag{3.31}$$

where $\boldsymbol{v}^{\mathcal{TP},m,\mathrm{EE}}$ is the position of the end effector extension from the top plate of the assembler, in the reference frame of the top plate. Thus, since the only component of the wrench depending on

$\boldsymbol{F}^{\text{EE}}$ is W^{EE}, we thus obtain

$$
\begin{aligned}
{J^{\text{t}}}_i(\nabla_{\boldsymbol{F}^{\text{EE}}}\boldsymbol{\tau}_i)^\top &= \begin{bmatrix} (\mathbf{R}_i^{\mathcal{G},\text{P}})^\top([\mathbf{R}_{N_P}^{\mathcal{G},\text{P}}\boldsymbol{v}^{\mathcal{TP},m,\text{EE}} + \mathbf{p}_{N_P}^{\mathcal{G}}] + [\mathbf{p}_i^{\mathcal{G},\text{P}}]^\top) \\ (\mathbf{R}_i^{\mathcal{G},\text{P}})^\top \end{bmatrix} \\
&= \begin{bmatrix} (\mathbf{R}_i^{\mathcal{G},\text{P}})^\top([\mathbf{R}_{N_P}^{\mathcal{G},\text{P}}\boldsymbol{v}^{\mathcal{TP},m,\text{EE}}] + [\mathbf{p}_{N_P}^{\mathcal{G}} - \mathbf{p}_i^{\mathcal{G},\text{P}}]) \\ (\mathbf{R}_i^{\mathcal{G},\text{P}})^\top \end{bmatrix} \quad i \in I^P,
\end{aligned}
\tag{3.32}
$$

where the jth row of $(\nabla_{\boldsymbol{F}^{\text{EE}}}\boldsymbol{\tau}_i)^\top$ gives the gradient of the force on leg j of SP i w.r.t. the force on the end effector.

Equations in this section are either labeled as $(N\#)$ to denote that they are used explicitly in the model, or as $(\#)$, to denote that this is just a calculation useful to deriving the model equations.

3.4 Trajectory Planning

The assembler is designed to be a movable platform that can help with complicated operations. As such, it is important that it can move from one position to another. We will adapt the tools in the prior section to develop trajectory optimization techniques. We present three approaches: a naïve direct transformation, a force optimization using trust regions, and a RRT* version that hinges on sub-paths computed from the force optimization approach.

Equations in this section are either labeled as $(N\#)$, $(T\#)$, or $(\#)$ to denote equations for naïve method, trust method, or calculations, respectively.

3.4.1 Naïve Direct Transformation

We establish the simplest approach to move from one pose to another that we call the Naïve method. This approach is ignorant to force calculations, and thus is very prone to generating infeasible

trajectories that violate the maximum force bounds. The approach simply uses a linear interpolation of the vector of leg lengths throughout the motion. For each target leg-length vector $\boldsymbol{L}_{i,j}^{\text{goal}}$, $i \in I^P$, $j \in J$, the forward kinematics (FK) problem is solved to obtain the full trajectory planning solution. To solve this problem, we first attempt the widely-used Newton-Raphson approach, as in e.g. [89], [105]. If this does not succeed, we then leverage a force-free version of the nonlinear optimizer to solve the FK problem, in which the end effector is allowed to move, and we optimize the squared leg-length-distance from the goal. More formally, to construct a QCQP model for the FK problem, we begin with the model in Section 3.3.4, then remove all force-related constraints and the end effector constraint (M13). We then optimize

$$\min \sum_{i \in I^P} \sum_{j \in J} (\boldsymbol{L}_{i,j} - \boldsymbol{L}_{i,j}^{\text{goal}})^2. \tag{N1}$$

Note that the FK problem for the assembler can decompose into separate FK problems for each SP.

3.4.2 Trust Region Optimization

We introduce a trust region method for trajectory planning of the Stewart platform. This algorithm first optimizes the starting and goal poses, then iteratively tries to make small steps towards the goal pose until it is sufficiently close, while trying to maintain forces that do not exceed those in the starting or ending poses.

Additional Definitions

To define the optimization problem for a single step, we first introduce the following additional parameters for the full optimization.

Parameter	Size	Description
$\boldsymbol{\Upsilon}^{\mathcal{G},EE,\text{start}}$		: Starting end effector position
$\boldsymbol{\Upsilon}^{\mathcal{G},EE,\text{goal}}$		: Goal end effector position

$\lambda^{\text{force}} \in [0, 1]$	: Coefficient for force-related objective terms
$\lambda^{\text{pose}} \in [0, 1]$	: Coefficient for objective terms related to distance from the final goal pose
$\lambda^{\text{avg}} \in [0, 1]$	: Additional multiplier for average-force-related objective term
ε^{pos}	: Positional plate motion limit for each Stewart platform per iteration
ε^{rot}	: Rotational plate motion limit for each Stewart platform per iteration
$\boldsymbol{F}^{\text{EE}}$	: The force vector applied to the end effector
$\boldsymbol{v}^{\mathcal{TP},m,\text{EE}}$	: The application point for $\boldsymbol{F}^{\text{EE}}$ w.r.t. the top plate of the assembler

Note that we normalize the non-negative objective weights with $\lambda^{\text{force}} + \lambda^{\text{pose}} = 1$. We then compute the full optimized starting and ending poses by solving the QCQP model in Section 3.3.4 with IPOPT, then compute the maximum force observed in either pose.

Parameter	Description
$\boldsymbol{\Upsilon}_i^{\mathcal{G},\text{P},\text{start}}$	: Global-frame starting pose for plate $i \in I$
$\boldsymbol{\Upsilon}_i^{\mathcal{BP},\text{TP},\text{start}}$	: SP-frame starting pose for plate $i \in I$
$\boldsymbol{\Upsilon}_i^{\mathcal{G},\text{P},\text{goal}}$	: Global-frame goal pose for plate $i \in I$
$\boldsymbol{\Upsilon}_i^{\mathcal{BP},\text{TP},\text{goal}}$	: SP-frame goal pose for SP $i \in I^P$
$f^{\text{max,ends}}$	: Maximum force in either path endpoint, the starting and goal poses

Note that, for practical use, the starting pose would be specified directly as the current pose of the assembler.

For each iteration, we define the pose from the previous iteration as

Parameter	Description
$\boldsymbol{\Upsilon}_i^{\mathcal{G},P,\text{init}}$	: Initial global-frame pose for plate $i \in I$ for the current iteration
$\boldsymbol{\Upsilon}_i^{\mathcal{BP},\text{TP},\text{init}}$	: Initial SP-frame pose for plate $i \in I^P$ for the current iteration

Finally, we introduce the following additional variables for each iteration.

Variable	Size	Description
τ^{viol}		: Maximum force above the maximum allowable force in any leg
τ^{ends}		: Maximum force above $f^{\max,\text{ends}}$ in any leg
$\tau^{\text{abs}}_{i,j}$		: Absolute value of $\tau_{i,j}$ for leg $i \in I^P$, $j \in J$
W^{EE}		: End-effector wrench at the current pose

Constraints

We begin with the model in Section 3.3.4, omitting the end effector constraints (M13) and the explicit force constraint (M18). We also redefine the end effector wrench definition W^{EE} to account for the fact that the end effector is no longer at a fixed position.

We then add constraints to ensure that no plate moves a distance further than ε^{pos} from $\mathbf{p}_i^{\mathcal{G},P,\text{init}}$, measured in two-norm, and rotates no further than ε^{rot} from $\mathbf{R}_i^{\mathcal{G},\text{P},\text{init}}$, measured in matrix Frobenius norm. The Frobenius norm was chosen for its superior performance and robustness compared to more direct angle-based measures in preliminary testing, combined with its simplicity and convexity. Finally, we add constraints to define τ^{viol} and τ^{ends}, then define the objective function.

Motion Limit: We ensure that positional and rotational motions are sufficiently controlled via

$$\begin{aligned} \|\mathbf{R}_i^{\mathcal{G},\text{P}} - \mathbf{R}_i^{\mathcal{G},\text{P},\text{init}}\|_F^2 &\leq (\varepsilon^{\text{rot}})^2 \quad i \in I, i \geq 1 \\ \|\mathbf{p}_i^{\mathcal{G},\text{P}} - \mathbf{p}^{\mathcal{G},\text{P},\text{init}}\|^2 &\leq (\varepsilon^{\text{pos}})^2 \quad i \in I, i \geq 1. \end{aligned} \tag{T1}$$

Force definitions:

To define wrenches, we first define the now-variable rotational component $W^{R,\text{EE}}$ of the end effector wrench W^{EE} via

$$\begin{aligned} \mathbf{p}^{\mathcal{G},m,\text{EE}} &= \mathbf{R}_{N_P}^{\mathcal{G},\text{P}} \boldsymbol{v}^{\mathcal{TP},m,\text{EE}} + \mathbf{p}_{N_P}^{\mathcal{G},\text{P}} \\ W^{R,\text{EE}} &= [\mathbf{p}^{\mathcal{G},m,\text{EE}}] \boldsymbol{F}^{\text{EE}} \end{aligned} \tag{T2}$$

where the translational component $W^{P,\text{EE}} = \boldsymbol{F}^{\text{EE}}$ of the wrench is constant. We then use (3.18)

and (3.17) to define plate wrenches as before. Next, we define the additional required force-related variables with

$$\begin{aligned}
\tau^{\text{viol}} &\geq \tau^{\max} - f^{\max} && \forall i \in I^P, j \in J \\
\tau^{\text{ends}} &\geq \tau^{\max} - f^{\max,\text{ends}} && \forall i \in I^P, j \in J \\
\tau^{\text{abs}}_{i,j} &\geq \tau_{i,j} && \forall i \in I^P, j \in J \\
\tau^{\text{abs}}_{i,j} &\geq -\tau_{i,j} && \forall i \in I^P, j \in J \\
\tau^{\text{viol}} &\geq 0 \\
\tau^{\text{ends}} &\geq 0
\end{aligned} \tag{T3}$$

Note that, through τ^{viol}, the max-force constraints on the legs are moved to the objective with a high coefficient. For the purposes of balancing objective terms related to forces, distances, and rotation angles, we assume forces and distances are measured in SI units (i.e. Newtons and meters). Note that the rotation terms are unit-independent, as they are measured via Frobenius norms of unitary rotation matrices.

Objective

To assist in defining the objective functions, we define the following expressions for the sake of readability. These definitions are the 'distance-to-goal' metrics for the objective function

$$\begin{aligned}
\tau^{\text{1-norm}} &= \sum_{i \in I^P} \sum_{j \in J} \tau^{\text{abs}}_{i,j}, \\
\mathcal{E}^{\text{pos}} &= \sum_{i \in I^P} \|\mathbf{p}_i^{\mathcal{BP},\text{TP}} - \mathbf{p}_i^{\mathcal{BP},\text{TP,goal}}\|^2, \\
\mathcal{E}^{\text{rot}} &= \sum_{i \in I^P} \|\mathbf{R}_i^{\mathcal{BP}} - \mathbf{R}_i^{\mathcal{BP},\text{goal}}\|_F^2.
\end{aligned} \tag{T4}$$

With these definitions, we define the objective function as

$$\min \lambda^{\text{force}}(\tau^{\text{ends}} + \tfrac{\lambda^{\text{avg}}}{6N_P}\tau^{\text{1-norm}} + 1000\tau^{\text{viol}}) + \lambda^{\text{pose}}(\mathcal{E}^{\text{pos}} + \tfrac{1}{4}\mathcal{E}^{\text{rot}}). \tag{T5}$$

Note that the decision to divide the $\mathcal{E}^{\text{rot}}$ by 4 serves to balance the position and rotation distance metrics, and performed well in preliminary testing compared to other coefficients. For assemblers consisting of SP's of different sizes, this divisor should be re-scaled according to the size of the SP, while the choice of λ^{force} should be re-scaled according to the weight of the SP.

Iteration

The trust region method proceeds by solving the problem described above until convergence. However, the process often stagnates, reaching positions where the objective to reduce forces overrides the objective to move towards the goal pose, or even moves in the wrong direction to improve average forces. When stagnation occurs due with high maximum forces, we divide λ^{force} by 2, set $\lambda^{\text{pose}} = 1 - \lambda^{\text{force}}$, and try again with the same initial positions. Similarly, if either stagnation or motion in the wrong direction occurs due to average forces, we divide λ^{avg} by 4 and try again with the same initial positions. The method converges when the distance between the current pose and the goal pose is small enough that the goal pose is a valid solution for the next iteration.

More formally, for the current iteration, define

$$\begin{aligned} \mathcal{E}^{\text{max,pos}} &= \max_{i \in I, i \geq 1} \|\mathbf{p}_i^{\mathcal{G},\text{P,init}} - \mathbf{p}^{\mathcal{G},\text{P,goal}}\|, \\ \mathcal{E}^{\text{max,rot}} &= \max_{i \in I, i \geq 1} \|\mathbf{R}_i^{\mathcal{G},\text{P,init}} - \mathbf{R}_i^{\mathcal{G},\text{P,goal}}\|_F. \end{aligned} \tag{3.33}$$

Then *convergence* occurs when both $\mathcal{E}^{\text{max,pos}} \leq \varepsilon^{\text{pos}}$ and $\mathcal{E}^{\text{max,rot}} \leq \varepsilon^{\text{rot}}$. Early termination occurs if the process has stagnated or moved in the wrong direction too many times, or if too many iterations have been reached.

Due to the multi-objective nature of the trust region method, it's possible that insufficient progress towards the goal can occur during the solve. We call this a *stagnation* and formally define this to

occur when one of the following criteria are met after optimizing:

$$\begin{aligned} \max_{i\in I, i\geq 1} \|\mathbf{p}_i^{\mathcal{G},\mathrm{P}} - \mathbf{p}^{\mathcal{G},\mathrm{P},\mathrm{init}}\| &\leq \tfrac{\varepsilon^{\mathrm{pos}}}{100}, \\ \max_{i\in I, i\geq 1} \|\mathbf{R}_i^{\mathcal{G},\mathrm{P}} - \mathbf{R}_i^{\mathcal{G},\mathrm{P},\mathrm{init}}\|_F &\leq \tfrac{\varepsilon^{\mathrm{rot}}}{100}. \end{aligned} \tag{3.34}$$

Even if $\lambda^{\mathrm{avg}} = 0$, stagnation can occur in the optimizer if $\tau^{\max} = f^{\max,\mathrm{ends}}$ and further progress requires forces above $f^{\max,\mathrm{ends}}$. Thus, if stagnation occurs, we conclude that the average-force term is the culprit only if $\tau^{\max} \leq f^{\max,\mathrm{ends}} - 0.001$, where the 1mN subtraction is added to be conservative, helping to prevent the algorithm from stagnating with repeated, futile reductions of λ^{avg}.

We define a step to be in the *wrong direction* if both the positional and rotational components of the motion have moved somewhat away from the goal pose, i.e. if for an iteration k we have $\mathcal{E}_k^{\max,\mathrm{pos}} \geq \mathcal{E}_{k-1}^{\max,\mathrm{pos}} + 10^{-4}$ and $\mathcal{E}_k^{\max,\mathrm{rot}} \geq \mathcal{E}_{k-1}^{\max,\mathrm{rot}} + 10^{-4}$. We allow motion in the wrong direction in order to reduce maximum leg forces that seemed excessive, i.e. if $\tau^{\max} > f^{\max,\mathrm{ends}}$. If we reject the motion step, then the position-related terms in the objective function have worsened while the max-force-related terms have not improved, and so the average force term must be the culprit.

Define the maximum allowed number of stagnated iterations as $n^{\max,\mathrm{stag}}$, and define the maximum number of iterations as $k^{\max}$. The iteration then proceeds as in Algorithm 2. For shorthand, we define the solution path as a list of poses from the starting pose to the goal pose. As defined in Section 3.2, a *pose* is the corresponding list of $\mathbf{p}_i^{\mathcal{G},\mathrm{P}}$'s and $\mathbf{R}_i^{\mathcal{G},\mathrm{P}}$'s.

To improve consistency of this algorithm, we run it within a backtracking scheme: if convergence fails, then we assume the iteration got sidetracked by forces, and restart after dividing λ^{force} by 4, for up to 5 total restarts. Then, if convergence succeeds, but maximal mid-path forces exceed maximum path-endpoint forces, we run again with $pose^{\mathrm{start}}$ and $pose^{\mathrm{ends}}$ switched to try to find a better path. We only keep this reversed solution if it converges and yields better worst-case mid-path forces.

Algorithm 2: General procedure for trust-region trajectory planning method

Input : A starting position $pose^{\text{start}}$ and ending position $pose^{\text{ends}}$.
Output : An integer K and a sequence of poses $pose_0, \ldots, \text{pose}_k$

$n^{\text{stag}} \leftarrow 0, \quad k \leftarrow 1, \; pose_0 \leftarrow pose^{\text{start}}$
while *not converged and* $n^{stag} < n^{\max,stag}$ *and* $k \leq k^{\max}$ **do**
  $pose^{\text{init}} \leftarrow pose_{k-1}$
  Solve the trust region problem from position $pose^{\text{init}}$ to compute solution $pose$
  if *stagnation detected or (wrong direction detected and* $\tau_{k-1}^{ends} = 0$*)* **then**
    if *wrong direction* **then**
      $\lambda^{\text{avg}} \leftarrow \frac{1}{4}\lambda^{\text{avg}}, \quad n^{\text{stag}} \leftarrow n^{\text{stag}} + 1$
    else
      $\lambda^{\text{force}} \leftarrow \frac{1}{2}\lambda^{\text{force}}, \quad \lambda^{\text{pose}} \leftarrow 1 - \lambda^{\text{force}}, \quad n^{\text{stag}} \leftarrow n^{\text{stag}} + 1$
    end if
  else
    $pose_k \leftarrow pose, \quad k \leftarrow k + 1$
  end if
end while
if *converged* **then**
  $pose_k \leftarrow pose^{\text{goal}}$
end if

3.5 Experimental Results

The results in Sections 3.5.2 and 3.5.3 were coded in Python 3.7, while the optimization steps were performed in IPOPT 3.11.1 [107] via Pyomo 5.7 [108], [109]. They were run on a laptop running Windows 10 with 32GB of RAM, using an Intel Core i7-9750H CPU processor (2.6GHz, 6 Cores, 12 threads). Additional computations were run on a desktop computer running Windows 10 with 64GB of RAM, using an AMD Ryzen 9 3900X 12-Core processsor running at 3.79Ghz.

For these computational studies, we use the following parameters. All parameters are given in SI units, i.e. kilograms, meters, seconds, and Newtons for masses, distances, time, and forces, respectively.

Parameter	Value
N_P	$= 4$
$\boldsymbol{\Upsilon}_i^{\mathcal{BP},\text{TP},\text{rest}}$	$= [0, 0, 0.5069351\ m, 0, 0, 0]^\top$

$$
\begin{aligned}
\mathbf{p}_{0,j}^{\mathcal{BP},\mathrm{LB}} &= \begin{bmatrix} 0.150037 & 0.150037 & -0.040202 & -0.109834 & -0.109834 & -0.040202 \\ -0.040202 & 0.040202 & 0.150037 & 0.109834 & -0.109834 & -0.150037 \\ 0.016637 & 0.016637 & 0.016637 & 0.016637 & 0.016637 & 0.016637 \end{bmatrix}^\top \\
\mathbf{p}_{0,j}^{\mathcal{TP},\mathrm{LT,rest}} &= \begin{bmatrix} 0.109834 & 0.109834 & 0.040202 & -0.150037 & -0.150037 & 0.040202 \\ -0.109834 & 0.109834 & 0.150037 & 0.040202 & -0.040202 & -0.150037 \\ -0.016637 & -0.016637 & -0.016637 & -0.016637 & -0.016637 & -0.016637 \end{bmatrix}^\top \\
\theta^{\max} &= 55^\circ \\
\theta^{\mathbf{R},\max} &= 60^\circ \\
(L^{\min}, L^{\max}) &= (0.38044, 0.580434) \\
f^{\max} &= 889.644 \\
m^{\mathrm{P}}_{(i=0:N_P)} &= [7.235, 14.47, 14.47, 14.47, 7.235]^\top \\
m^{\mathrm{LT}} &= 0.15 \\
m^{\mathrm{LB}} &= 0.2 \\
d^{\mathrm{LT,CoG}} &= 0.05 \\
d^{\mathrm{LB,CoG}} &= 0.089 \\
g &= 9.81
\end{aligned}
$$

For even i, The values of $\mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LB}}$ and $\mathbf{p}_{i,j}^{\mathcal{TP},\mathrm{LT,rest}}$ are equal to $\mathbf{p}_{0,j}^{\mathcal{BP},\mathrm{LB}}$ and $\mathbf{p}_{0,j}^{\mathcal{TP},\mathrm{LT,rest}}$, respectively. For odd i, we have

$$
\begin{aligned}
\mathbf{p}_{i,j}^{\mathcal{BP},\mathrm{LB}} &= R^{\mathrm{odd}} \mathbf{p}_{0,j}^{\mathcal{BP},\mathrm{LB}} \\
\mathbf{p}_{i,j}^{\mathcal{TP},\mathrm{LT,rest}} &= R^{\mathrm{odd}} \mathbf{p}_{0,j}^{\mathcal{TP},\mathrm{LT,rest}}
\end{aligned}
\tag{3.35}
$$

Where R^{odd} is a 30-degree rotation about the z-axis,

$$
R^{\mathrm{odd}} = \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} & 0 \\ \frac{1}{2} & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.36}
$$

3.5.1 Pose Generation

We describe the procedure for generating the dataset of poses that we use to generate end-effector goal positions for the computational studies in this work. This generation begins with the generation of individual random poses for the intended SP configuration. Given that the target 4-SP assembler consists of four separate robots, poses can be generated by applying kinematic operations on each constituent SP separately, then stacking them, resulting in a full pose for the assembler.

Pose generation for the individual SPs came in two varieties: Uniform and Extreme. Uniform poses were generated by applying the FK algorithm to a SP with a set of leg lengths uniformly generated from their minimum and maximum extensions. Configurations that resulted in an error or were at the "home" position (via error correction) were discounted, and iteration continued until a set number of poses (for the purpose of this paper, 10,000) were generated. Extreme poses followed the same procedure, with the added caveat that poses which did not meet a minimum measure of rotation magnitude were also rejected, leaving only poses with the requisite tilt factor. Tilt was selected as being more important than translation because a tilt in one platform has a much greater change potential in a stack end effector than does translation.

Once individual SP poses were completed, the 4-SP stacked poses could be constructed. There were three categories of poses which we trialed: Uniform, Extreme, and Repeated. Uniform poses drew from the aforementioned uniform individual poses, while Extreme drew from the extreme poses. For both, four poses are chosen at random from their constituent files and applied to create a stacked pose. The end effector position (topmost plate of the topmost SP) is recorded, along with the plate positions and leg lengths of all constituent platforms. The Repeated poses differ slightly in generation. They too draw from the extreme pose file, but only one individual SP pose is chosen, and is applied to each platform in the stack, such that the ultimate pose is severely biased towards tilt in the direction of the constituent SP pose. For the purposes of this analysis, each type of pose generator produced three files consisting of 100, 1,000, and 10,000 poses respectively. The 100 pose file is intended purely for testing, whereas the two larger datasets for each type were earmarked for

analysis.

3.5.2 Pose Optimization

In this section, we compare the SIK and OPT methods for solving the IK problem on the instances generated in Section 3.5.1. We also include the force-related information for the initially generated poses (GEN) as a reference.

Table 3.1 summarizes the performance of the SIK and OPT methods on various instances, while Table 3.2 summarizes some additional performance characteristics determined during meta-analysis. A pose is said to be *valid* the end effector and base plates are in the correct pose, and all constraints related to kinematic validity are satisfied. A valid pose is said to be *force-valid* if the maximum-force constraints are also satisfied. The Avg % and Avg Max % Reduced metrics measure the percent reduction of forces out of the instances for which both methods yield kinematically valid poses. The % Improved metric measures the percentage of poses for which OPT yielded a better pose then SIK, out of the poses for which at least one of the methods yielded a valid pose. Finally, the Avg OPT Time metric measures the average time required for the OPT method to terminate, regardless of termination status.

Dataset	Solver	Valid poses	Force-valid poses
Uniform	GEN	10,000	627
	SIK	9,048	4,416
	OPT	10,000	9,895
Extreme	GEN	10,000	323
	SIK	8,905	3,459
	OPT	10,000	9,903
Repeated	GEN	10,000	1479
	SIK	2,191	187
	OPT	10,000	8,317

Table 3.1: Performance metrics for SIK and OPT, out of 10,000 random poses.

Note that the optimizer is always able to find kinematically valid poses when initialized via the

Dataset	Avg % Reduced	Avg Max % Reduced	% Improved	Avg OPT Time
Uniform	34.03%	57.4%	100%	1.15s
Extreme	33.85%	58.51%	100%	1.18s
Repeated	37.81%	64.67%	100%	1.19s

Table 3.2: Improvement of OPT over SIK, along with average computational time for 10,000 random poses.

same-SP approach, even when the initial guesses are not kinematically valid. This effectiveness is especially noticeable for poses from the difficult Repeated datasets, for the success rate from SIK was only about 22%.

Figure 3.5 showcases the force-performance for the SIK and OPT approaches. From the figure, note that, particularly for the repeated dataset, the problems yielding kinematically valid poses for SIK are significantly less likely to yield force-invalid poses in OPT.

Figure 3.6 showcases the computational performance of the optimizer over the Extreme and Repeated instances. The Uniform performance plot is omitted due to its strong similarity to the Extreme plot, but with one outlier requiring more than 16s. Note that the time-performance for the Repeated poses is more polarized than for other datasets: $\sim$55% (vs. $\sim$40%) of poses solve in less than 1s, but $\sim$10% (vs. $\sim$3%) require more than 2s to solve.

From Table 3.1 and Figure 3.5, we see a very strong degree of improvement in the resulting forces compared to the SIK heuristic. The forces are at least halved between 70-80% of the time, with improvement factors over 8 in some cases. This improvement, combined with the fast computation times observed in Figure 3.6, renders the OPT approach a viable method for the generation of force-optimized poses for Stewart Platforms. Note that, in terms of forces, even the SIK approach was typically able to find much better poses than were initially generated, when it succeeded in generating a kinematically valid pose.

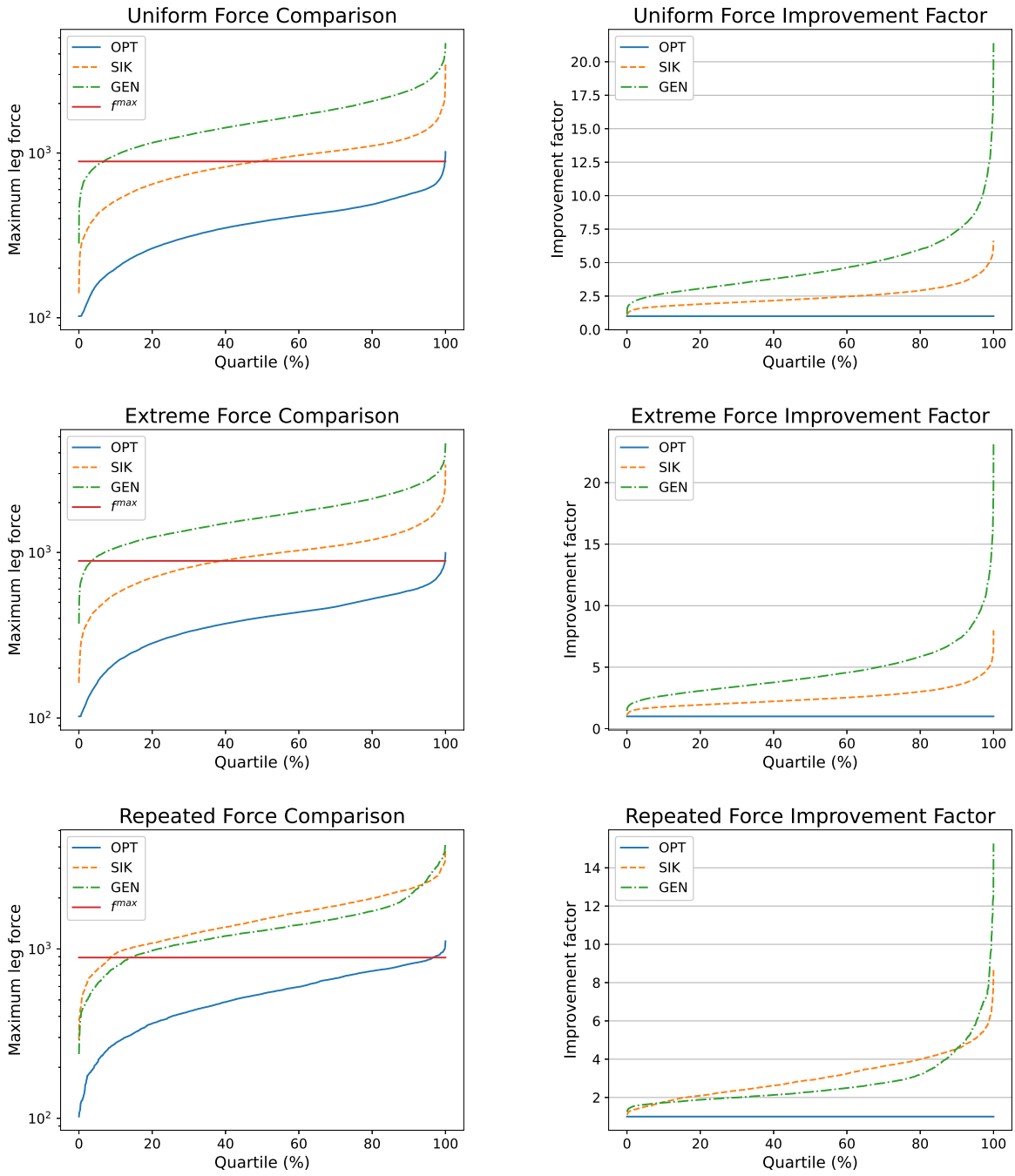


Figure 3.5: Pose comparison of OPT, GENerated, and SIK forces, considering the poses for which both approaches yielded kinematically valid poses. (a) Comparison of forces, (b) Factor of improvement of OPT over each method.

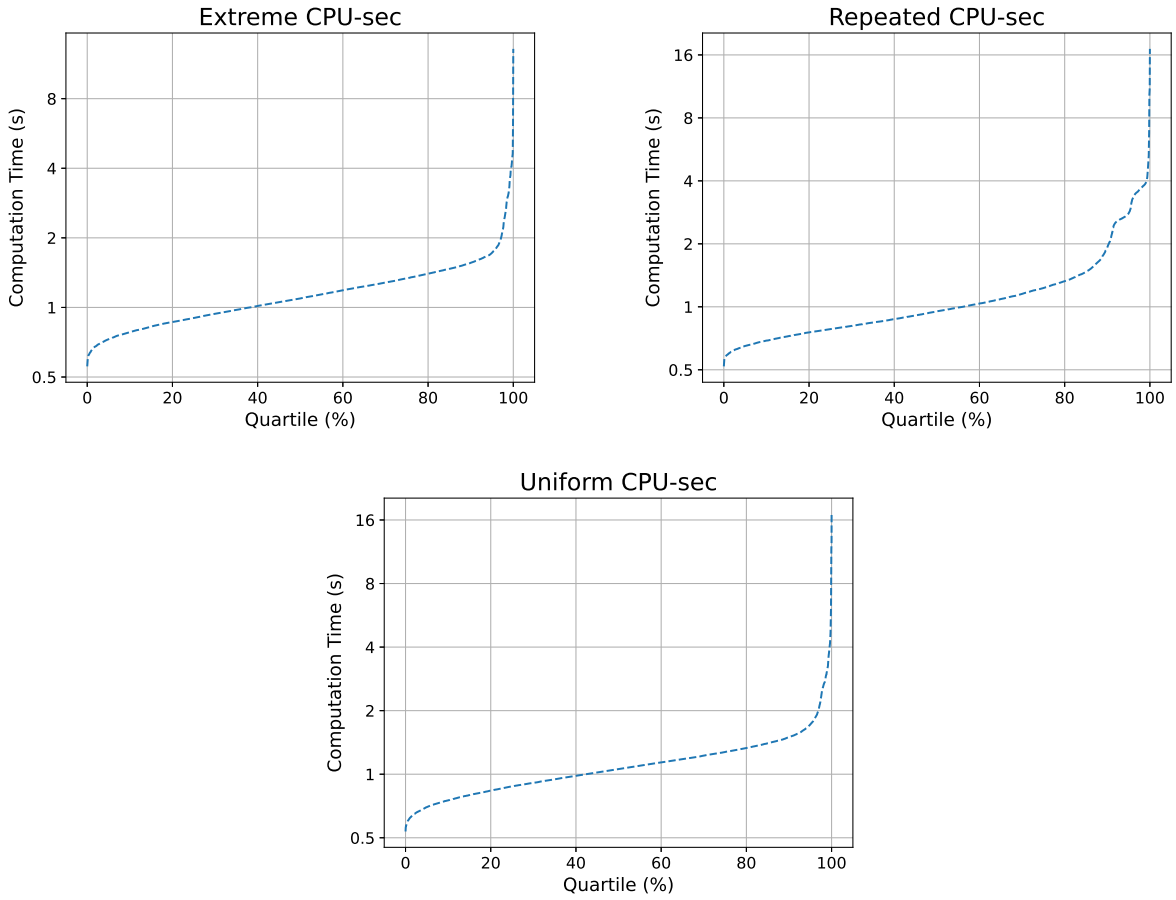


Figure 3.6: Computation times for OPT over 10,000 poses for the Extreme and Repeated instance families.

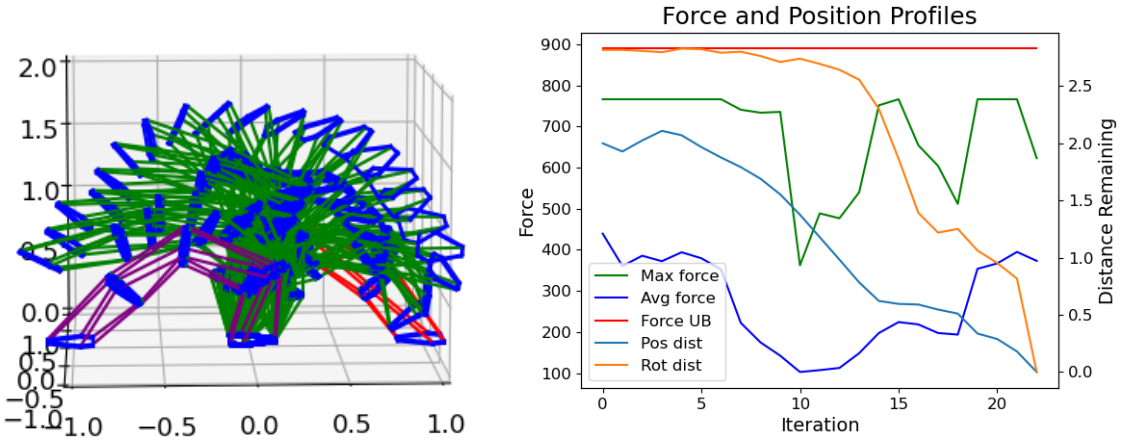


Figure 3.7: Force-controlled transition between bent-over poses using trust region method.

3.5.3 Trajectory Planning

In this section, we demonstrate the effectiveness of the trust region method, then showcase how this method can be used in conjunction with RRT* to obtain good obstacle-avoiding paths very quickly after pre-processing.

To demonstrate the effectiveness of the trust region method, we showcase a difficult motion: a transition between opposite bent-over poses. We showcase using the thin-plated SP with $\varepsilon^{\text{pos}} = 0.2$, $\varepsilon^{\text{rot}} = \frac{\pi}{3}$, $\lambda^{\text{force}} = 0.04$, $\lambda^{\text{pose}} = 0.96$, and $\lambda^{\text{avg}} = 0.05$. The sample motion, along with a plot showing the progression of the motion in terms of the forces and the maximum plate global-frame distances from any plate to the ending pose, is shown in Figure 3.7.

To demonstrate the consistency and performance of the trust region method, we compare the trust region method with the direct transformation and the RRT* approach.

A comparison trajectory planning results for all datasets are shown in Table 3.3. The settings used for the trust-region method were $\varepsilon^{\text{pos}} = 0.1$, $\varepsilon^{\text{rot}} = \frac{\pi}{6}$, $\lambda^{\text{force}} = 0.04$, $\lambda^{\text{pose}} = 0.96$, and $\lambda^{\text{avg}} = 0.05$. Some primary metrics of solution are given as

1. Behaved: A path is considered to be behaved if the maximum leg forces mid-motion do not exceed the maximum forces at the starting or ending pose.

2. Force-Valid: A path is considered to be force-valid if either all leg forces are valid throughout the motion, or if the path is behaved with excessive leg forces at either the starting or ending pose.

3. Avg OPT Time: The average optimization time across all tests.

Dataset	Method	% Behaved	% Force-Valid	Avg OPT Time
Uniform	Trust	100%	100%	15.06s
	Naïve	1.0%	78.5%	7.79s
Extreme	Trust	99.9%	100%	15.35s
	Naïve	1.3%	75.1%	8.34s
Repeated	Trust	99.8%	100%	22.20s
	Naïve	0.3%	19.7%	9.08s

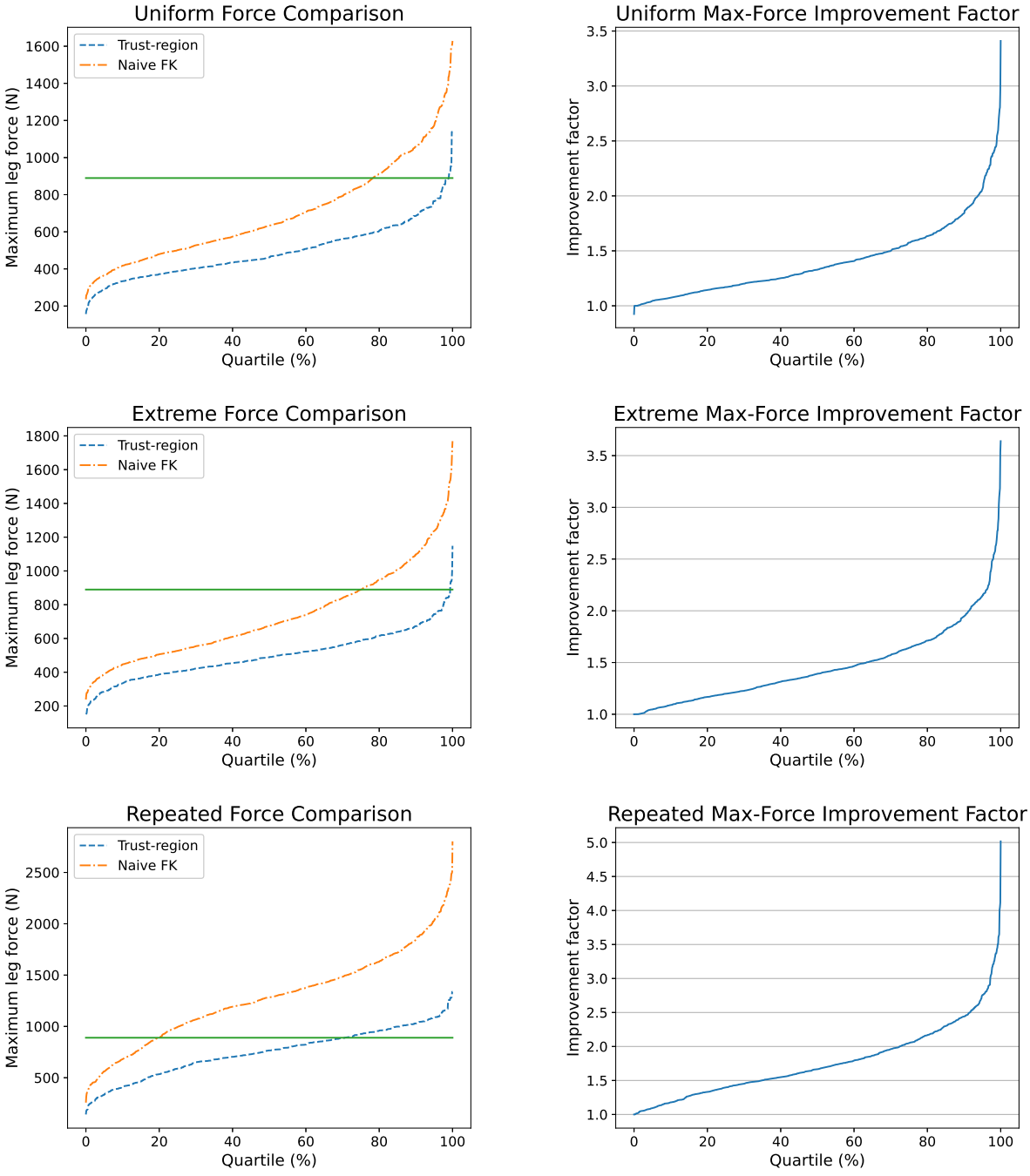
Table 3.3: Comparison of trajectory planning results for trust-region vs naïve FK, out of 1000 random poses.

We show a more complete picture of the maximum leg forces resulting from the two approaches in Figure 3.8, and compare the motion energies in Figure 3.9. To estimate the motion energies, we assume that extending a leg under tensile forces and contracting a leg under compression forces are free operations, and count only the portions of the motions that require energy input.

The results reveal that the trust region method is far more consistent, with maximum forces only rarely exceeding those at the end effectors (in 0.1% of cases), and with maximum-force improvements as high as a factor of 5. However, the gains in maximum forces and consistency come at a price: the motion energies tend to be far higher, exceeding the more direct motions by as much as a factor of 7. Furthermore, the naïve approach solves roughly twice as quickly as the trust region approach. Thus, in practice, we recommend first computing the naïve motion, then applying the trust region method if the motion is either near-infeasible or very poorly behaved, with mid-motion max-forces far higher then endpoint max-forces.

Note that, in practice, any leg motion requires some baseline motor energy usage even under zero

net forces, which would further strengthen the energy advantage gained from the shorter naïve motions.



(a) (b)

Figure 3.8: Comparison of maximum forces between trajectory planning methods. (a) Comparison of maximum forces, (b) Factor of improvement of trust-region method over naïve interpolated FK.

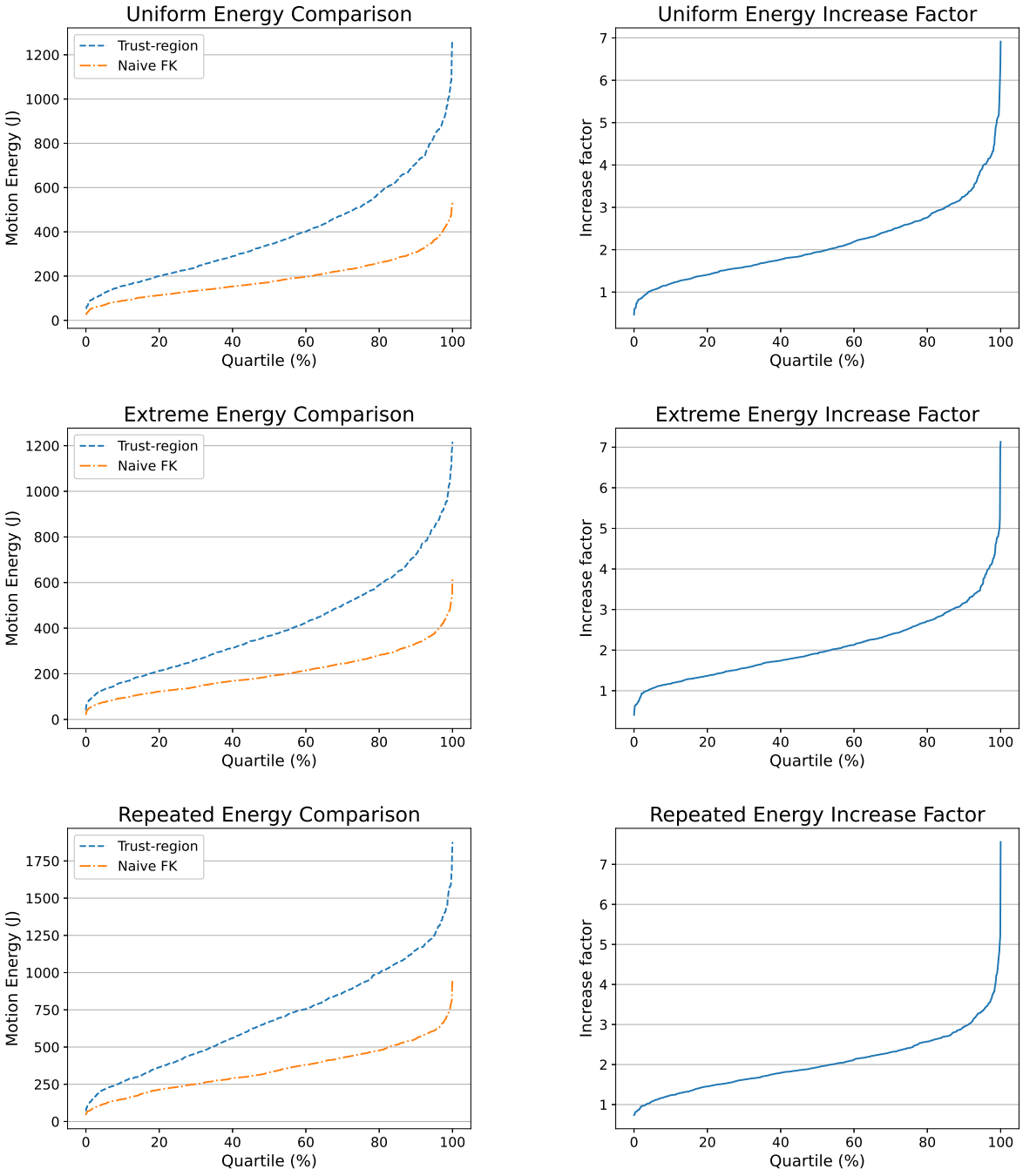


Figure 3.9: Comparison of motion energies between trajectory planning methods. (a) Comparison of motion energies, (b) Extra-energy factor of trust-region method over naïve interpolated FK.

3.6 Conclusion

We have presented a fast approach for the force-optimization of stacked SPs that can significantly improve their range of motion under heavy loads. In particular, the optimization step can result in reductions of the worst-case leg forces by as much as an order of magnitude in some cases. The model of forces used in this work has been verified via hardware testing, showing that the modeled and real forces are very similar under rotation. Further, we have presented a fast, consistent trust-region trajectory planning approaches based on this pose optimization scheme, and demonstrated the effectiveness of the approach in comparison to a simple length-based optimization approach.

In the future, the ultimate goal is to use these stacked SP structures to perform automated assembly operations. To improve on the pose optimization approach, we suggest exploring the use of more robust heuristics in conjunction with the local optimizer proposed in this work. For example, to improve pose optimization given fixed base and end-effector positions, one could quickly compute rough bounds on the possible poses for the middle plates, then apply heuristics such as a genetic algorithm or a bidirectional search, with frequent or intermittent local optimization steps, to quickly hone in on a solution. Alternatively, one could construct some fast heuristic for generating randomized feasible or near-feasible poses for a fixed end-effector position, then locally optimize a moderate number of random poses in parallel to seek a stronger solution with limited additional computational cost.

Funding

Chapter 4

Compact mixed-integer programming formulations in quadratic optimization

Abstract *We present a technique for producing valid dual bounds for nonconvex quadratic optimization problems. The approach leverages an elegant piecewise linear approximation for univariate quadratic functions due to Yarotsky [110], formulating this (simple) approximation using mixed-integer programming (MIP). Notably, the number of constraints, binary variables, and auxiliary continuous variables used in this formulation grows logarithmically in the approximation error. Combining this with a diagonal perturbation technique to convert a nonseparable quadratic function into a separable one, we present a mixed-integer convex quadratic relaxation for nonconvex quadratic optimization problems. We study the strength (or* sharpness*) of our formulation and the tightness of its approximation. Further, we show that our formulation represents feasible points via a Gray code. We close with computational results on problems with quadratic objectives and/or constraints, showing that our proposed method i) across the board outperforms existing MIP relaxations from the literature, and ii) on hard instances produces better bounds than exact solvers within a fixed time budget.*

4.1 Introduction

We are interested in methods to solve optimization problems with quadratic objectives and/or constraints. Consider the following generic problem with a quadratic objective:

$$\min_{x \in X} \quad h(x) := x'Qx + c \cdot x, \tag{4.1}$$

where $X \subseteq \mathbf{R}^n$ is some nonempty feasible region described by side constraints. When the quadratic objective matrix Q is not positive semidefinite, this is a difficult nonconvex optimization problem. We will focus on techniques to (approximately) reformulate nonconvex quadratic functions like the objective of (4.1).

Quadratic optimization problems naturally arise in a number of important applications across science and engineering (see [111], [112] and references therein). In the presence of nonconvexity, such problems are in general very difficult to solve from both a practical and theoretical perspective [113]. As a result, there has been a steady stream of research developing new algorithmic techniques to solve quadratic optimization problems, and variants thereof (see [114] for a survey).

Our approach to approximately solving problems of the form (4.1) will be to reformulate the objective of (4.1) using mixed-integer programming (MIP). Given some diagonal matrix D, we can equivalently write (4.1) as

$$\min_{x \in X} \quad h^D(x, y) := x'(Q + D)x + c \cdot x - Dy \tag{4.2a}$$

$$\text{s.t.} \quad y_i = x_i^2 \qquad i \in [\![n]\!], \tag{4.2b}$$

where $[\![n]\!] := \{1, \ldots, n\}$. If D is chosen such that $Q + D$ is positive semidefinite, the quadratic objective will be convex, meaning that all the nonconvexity of this problem has been isolated in the univariate quadratic equations $y_i = x_i^2$. This technique is sometimes called "diagonal perturbation" [115].

In this work, we present a compact, tight MIP formulation for the graph of a univariate quadratic term: $\{ (x, y) \mid l \leq x \leq u,\ y = x^2 \}$. We derive our formulation by adapting an elegant result of Yarotsky [110], who shows that there exists a simple neural network function that approximates $y = x^2$ exponentially well (in terms of the size of the network) over the unit interval. The resulting neural network can be interpreted as a function $F_L : \mathbb{R} \to \mathbb{R}$ that is built compositionally from a number of simple piecewise linear functions. There is a long and rich strain of research on MIP formulations for piecewise linear functions that serve as approximations for more complex nonlinear functions [116]–[122], with recent work focusing particularly on modeling neural networks [123]–[128].

We show that this approximation for univariate quadratic terms leads to a *relaxation* for optimization problems with quadratic objectives and/or constraints, meaning that it provides valid dual bounds for the true quadratic problem. We will show that our proposed formulation is *sharp*, meaning that its LP relaxation projects to the convex hull of all feasible points. Further, we show that the formulation is in fact *hereditarily sharp*, meaning that this sharpness property holds throughout the branch and bound tree. The key to reaching this result is connecting the binary reformulation to the *reflected Gray code*, a well-studied binary sequence in electrical engineering.

4.1.1 Literature review

Our approach hews most closely to that of Dong and Luo [129] and Saxena et al. [130]. The diagonal perturbation approach we follow have been applied throughout the years in a number of settings; for example, nonconvex quadratic optimization (with or without integer variables) [131]–[135], more general nonlinear [136], [137] optimization with binary variables, and general nonlinear optimization [138]–[140].

A string of recent work on optimization methods for nonconvex quadratic problems has focused on methods for relaxing bilinear terms using piecewise McCormick envelopes [1], [141]–[145]. These piecewise envelopes can be formulated using mixed-integer programming in multiple ways, typically

resulting in either a linear- or logarithmic-sized MIP formulation. Moreover, this piecewise relaxation can be refined dynamically to produce a tighter relaxation in a region of interest without resulting in an unduly large MIP formulation [141], [144]. In a similar vein, a paper of Galli and Letchford [146] presents a binarization heuristic for "box QP" problems, leveraging a structural result of Hansen et al. [147], and compares classical convexification techniques [148]–[150] within the heuristic.

An interesting recent paper of Xia et al. [151] reformulates optimization problems with quadratic objectives and linear constraints into MIPs via the KKT conditions. The approach outperforms commercial solvers on certain classes of instances; however, it does not seem to perform favorably on boxQP problems, and in general requires the careful computation of "big-M" coefficients which may lead to loose LP relaxations.

4.1.2 Outline

In Section 4.2 we describe our MIP approximation for $y = x^2$. In Section 4.3 we prove some properties of Gray codes that will be useful for proving the results in Section 4.4. In Section 4.4, we show that our formulation is strong (i.e. sharp), and establish the connection between our MIP approximation and the reflected Gray code. In Section 4.5, we show how to derive some facets of the full convex hull of our MIP approximation, with connections to the parity polytope. In Section 4.6, we present a relaxation version of our MIP approximation, derive the total area of the relaxation, and compare against the relaxation of Dong and Luo [129]. Finally, in Section 4.7, we numerically compare our relaxation with other competing methods, including other relaxations such as CDA [129] and NMDT [1], as well as state-of-the-art solvers with quadratic support like Gurobi, CPLEX, and BARON.

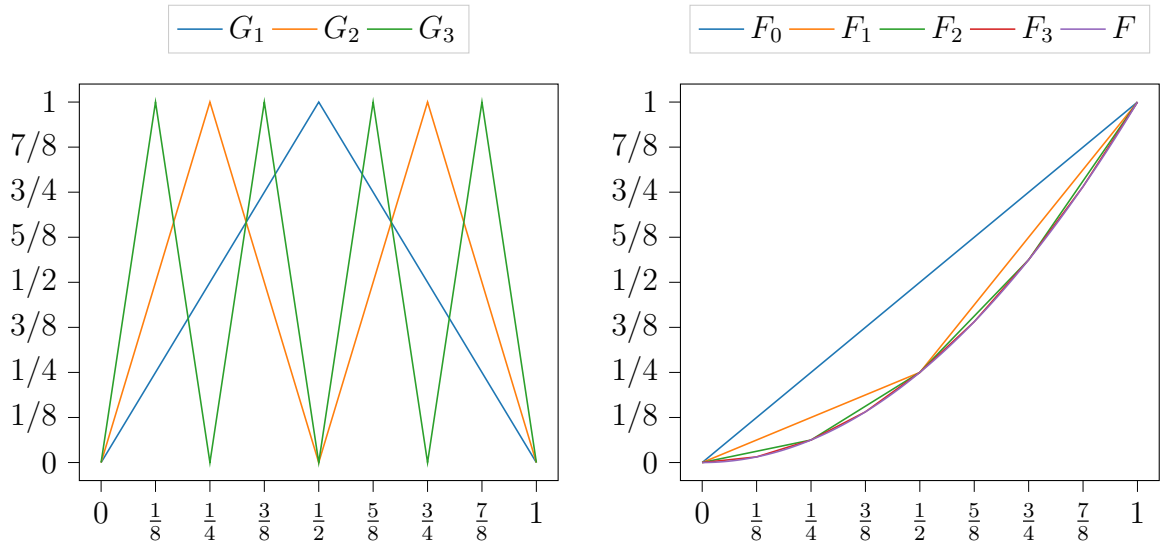


Figure 4.1: *Left:* The intermediary sawtooth functions $G_i = 2^{2i}(F_{i-1} - F_i)$. *Right:* The approximation for $F(x) = x^2$ of Yarotsky by functions F_i [110, Figure 2].

4.2 A piecewise-linear approximation for univariate quadratic terms

In this section, we present our mixed-integer programming relaxation for (4.1). We start by describing the construction of Yarotsky, which is a piecewise linear neural network approximation for the univariate quadratic function $F(x) = x^2$. We then formulate the graph of this piecewise-linear function using mixed-integer programming, and use it to build a tight under-approximation for the quadratic optimization problem (4.1).

For ease of notation, for any integers $i \leq j$, we define $[\![i, j]\!] := \{i, i + 1, \ldots, j\}$, and for integers $i \geq 1$ we define $[\![i]\!] := \{1, 2, \ldots, i\}$.

4.2.1 The construction of Yarotsky

For fixed $L \in \mathbb{N}$, we wish to model the function $F_L(x)$, defined as the piecewise linear interpolant to $y = x^2$ on the interval $[0, 1]$ at $2^L + 1$ uniformly spaced breakpoints:

$$F_L(x) = \tfrac{2i-1}{N}(x - \tfrac{i}{N}) + \tfrac{i^2}{N^2} \quad \text{if } x \in [\tfrac{i-1}{N}, \tfrac{i}{N}] \text{ for some } i \in [\![2^L]\!]. \tag{4.3}$$

Define the *sawtooth functions* $G_i \colon [0, 1] \to [0, 1]$ as $G_i = 2^i(F_{i-1} - F_i)$. Yarotsky [110] shows that G_i can be defined recursively as

$$G_0(x) = x, \tag{4.4a}$$

$$G_i(x) = \begin{cases} 2G_{i-1}(x) & G_{i-1}(x) < 1/2 \\ 2(1 - G_{i-1}(x)) & G_{i-1}(x) \geq 1/2 \end{cases} \quad i \in [\![L]\!], \tag{4.4b}$$

and, furthermore, that

$$F_L(x) = x - \sum_{i=1}^{L} 2^{-2i} G_i(x). \tag{4.5}$$

Yarotsky further shows that $F(x)$ approximates x^2 to a pointwise error of $|x^2 - F_L(x)| \leq 2^{-2L-2}$ [110, Proposition 2].[1] We include an illustration of G_L and F_L for different values of L in Figure 4.1(b). Crucially, we will later make use of the fact that $F_L(x) \geq F(x)$ for each $0 \leq x \leq 1$, i.e. F_L is an overestimator for F.

4.2.2 A MIP formulation for F_L

We now turn our attention to constructing a mixed-integer programming formulation for F_L. As (4.5) tells us that F_L depends linearly on the sawtooth functions G_i, we turn our attention to

[1] Furthermore, Yarotsky [110] observes that it is straightforward to represent each of the sawtooth functions as a composition of the standard ReLU activation function $\sigma(x) = \max\{0, x\}$. For example, $G_1(x) = 2\sigma(x) - 4\sigma(x - \frac{1}{2}) + 2\sigma(x - 1)$. In this way, F_L can be written as a neural network with a very particular choice of architecture and weight values.

formulating the piecewise-linear equations (4.4) using MIP.

For the remainder of the section we will use g_i as decision variables in our optimization formulation corresponding to the output of the i-th sawtooth function G_i. Therefore, $g_0 = x$, and for each of the other sawtooth functions G_i for $i \in [\![L]\!]$, we introduce a binary decision variable α_i. Given some input x, these binary variables serve to indicate which piece of the sawtooth the input lies on:

$$\alpha_i = 0 \implies (g_i = 2g_{i-1}) \wedge (0 \leq g_{i-1} \leq 1/2) \tag{4.6a}$$

$$\alpha_i = 1 \implies (g_i = 2(1 - g_{i-1})) \wedge (1/2 \leq g_{i-1} \leq 1) \tag{4.6b}$$

Define the set $S_i := \{ (g_{i-1}, g_i, \alpha_i) \in [0,1] \times [0,1] \times \{0,1\} \mid (4.6) \}$ for each $i \in [\![L]\!]$. It is not difficult to see that a convex hull formulation for S_i is given by

$$2(\alpha_i - g_{i-1}) \leq g_i \leq 2(1 - g_{i-1}), \tag{4.7a}$$

$$2(g_{i-1} - \alpha_i) \leq g_i \leq 2g_{i-1}. \tag{4.7b}$$

$$(g_{i-1}, g_i, \alpha_i) \in [0,1] \times [0,1] \times \{0,1\}. \tag{4.7c}$$

Chaining these formulations together for each i, we construct a MIP formulation for $\mathcal{G}_L := \{ (x,y) \in [0,1] \times [0,1] \mid y = F_L(x) \}$, the graph of the neural network approximation function F_L.

Proposition 4.1. *Fix some $L \in \mathbb{N}$. A MIP formulation for $(x,y) \in \mathcal{G}_L$ is*

$$\begin{aligned} & g_0 = x \\ & (g_{i-1}, g_i, \alpha_i) \in S_i \qquad i \in [\![L]\!] \\ & y = x - \sum_{i=1}^{L} 2^{-2i} g_i. \end{aligned} \tag{4.8}$$

We emphasize that this formulation is extremely compact: it requires only $\mathcal{O}(L)$ binary variables, auxiliary continuous variables, and constraints. As noted in Section 4.2.1, F_L approximates F to within $\mathcal{O}(2^{-L})$ pointwise, which implies that the size of our formulation scales logarithmically in

the desired accuracy.

It is a straightforward extension of Proposition 4.1 to consider more general interval domains $x \in [l, u]$ on the inputs. In particular, introducing two auxiliary variables $\tilde{x}, \tilde{y} \in [0, 1]$, we formulate $\tilde{y} = F(\tilde{x}) = \tilde{x}^2$ using (4.8), and then map them to the (x, y) variables via the linear transformation

$$x = l + (u - l)\tilde{x}, \quad y = l^2 + 2l(u - l)\tilde{x} + (u - l)^2\tilde{y}.$$

4.2.3 Tying it all together

We are now prepared to construct our mixed-integer programming approximation for (4.1). For the objective of (4.1), compute a nonnegative diagonal matrix D such that $Q + D$ is positive semidefinite.[2] Then, for a given L, the approximation for (4.1) is:

$$\min_{x \in X, y} \quad h^D(x, y) \equiv x'(Q + D)x + c \cdot x - Dy \tag{4.9a}$$

$$\text{s.t.} \quad (x_i, y_i) \in \mathcal{G}_L \qquad\qquad i \in [\![n]\!]. \tag{4.9b}$$

Using the formulation (4.8) for the constraint (4.9b), this yields a mixed-integer convex quadratic reformulation of the problem (ignoring the potential structure of X). This formulation requires at most nL binary variables and $\mathcal{O}(nL)$ auxiliary continuous variables and linear constraints. Furthermore, recall that we may set $L = \mathcal{O}(\log(1/\varepsilon))$ to attain an approximation of accuracy ε for the equations (4.2b).

Consider any $\hat{x} \in X$, along with any $\hat{y}$ such that $(\hat{x}, \hat{y})$ satisfies (4.9b). Since F_L overestimates F, for each $i \in [\![n]\!]$ we have $\hat{x}_i^2 \leq \hat{y}_i$. Therefore, $h^D(\hat{x}, \hat{y}) \leq h(\hat{x})$. Since there always will exist such a $\hat{y}$ for any $\hat{x} \in X$, (4.9) offers a valid dual bound on the optimal cost of (4.1).

Note that this approach can readily be adapted to handle quadratic constraints. In particular,

[2] This can be accomplished in a number of ways: for example, by computing the minimum eigenvalue of D, or by solving a semidefinite programming problem [129].

this transformation will offer a *relaxation* of the quadratically constrained problem. Note that the error bound derived above is with respect to the quadratic constraint that is being relaxed. It may not translate into an error bound on the objective value of the optimization problem, a known phenomena in the global optimization literature [152].

4.3 Gray Codes and Binary Representation

In this section, we introduce the reflected Gray code, and prove some of its useful properties.

For the remainder of this work, we will work with two notions of expressing integers as vectors in $\{0,1\}^*$. First, we consider the standard binarization with L bits. That is, for an integer $i \in [\![0, 2^L - 1]\!]$ we define $\boldsymbol{\beta}^i \in \{0,1\}^L$ such that

$$i = \sum_{j=1}^{L} 2^{L-j} \beta_j^i. \tag{4.10}$$

Next, we define the *reflected Gray code* sequence, which is a sequence of binary representations of integers that is extremely well-studied in electrical engineering and engineering [153]. Notably, each adjacent pair in the sequence differs in exactly one bit. As presented in [154] and references therein, the L-bit reflected Gray code $\boldsymbol{\alpha}^i \in \{0,1\}^L$ representing the integer i can be described by the recursion

$$\alpha_1^i = \beta_1^i \tag{4.11}$$

$$\alpha_j^i := \beta_j^i \oplus \beta_{j-1}^i \qquad \text{for all } j = 2, \ldots, L, \tag{4.12}$$

where we use $\oplus$ to denote addition modulo 2. By inverting the relation, we obtain the formula

$$\beta_j^i = \alpha_1^i \oplus \alpha_2^i \oplus \cdots \oplus \alpha_j^i \qquad \text{for } j = 1, \ldots, L. \tag{4.13}$$

$L = 1$

Code	Number
0	0
1	1

$L = 2$

Code		Number
0	0	0
0	1	1
1	1	2
1	0	3

$L = 3$

Code			Number
0	0	0	0
0	0	1	1
0	1	1	2
0	1	0	3
1	1	0	4
1	1	1	5
1	0	1	6
1	0	0	7

Figure 4.2: Building the reflected Gray code. The reflected Gray code with $L+1$ bits is build from the reflected Gray code on L bits by appending 0s in front of it, then reflecting the Gray code sequence, and then appending 1s in front of it.

In this way, flipping any α_j bit implies that we flip all less significant bits β_k for $k \geq j$. See Figure 4.2 for an illustration of how to build the reflected Gray code, which we will henceforth refer to as 'the Gray code'.

One key property of any Gray code is that successive integer representations differ by only 1 bit, i.e.,

$$\|\boldsymbol{\alpha}^i - \boldsymbol{\alpha}^{i+1}\|_1 = 1. \tag{4.14}$$

That is, only one bit changes between adjacent binary vectors in the sequence. We show a similar property holds if we restrict the set of integers we work with by fixing some of the bits in the Gray code vector. To help prove this property, we note the following well-known property of the reflected Gray code in this work.

Lemma 4.2. *For each $i \in [\![0, 2^L - 1]\!]$, let $\tilde{\boldsymbol{\alpha}}^i$ be the L-bit Gray code for i, and let $\boldsymbol{\alpha}^j$ be the $L+1$-bit Gray code for some $i \in [\![0, 2^{L+1} - 1]\!]$.*

1. *If $j \in [\![0, 2^L - 1]\!]$, then $\boldsymbol{\alpha}^j = [0, \tilde{\boldsymbol{\alpha}}^j]$.*
2. *If $j \in [\![2^L, 2^{L+1} - 1]\!]$, then $\boldsymbol{\alpha}^j = [1, \tilde{\boldsymbol{\alpha}}^i]$, where $i = 2^{L+1} - j - 1$.*

Proof. First, for each $i \in [\![0, 2^L - 1]\!]$, let $\tilde{\boldsymbol{\beta}}^i$ be the corresponding L-bit binarization. Similarly, for

each $j \in [\![0, 2^{L+1} - 1]\!]$, and let $\boldsymbol{\beta}^j$ be the corresponding $L+1$-bit binarization. Then, by (4.11) have that $\alpha_1^j = \beta_1^j = 0$ and $\boldsymbol{\beta}^i = [0, \tilde{\boldsymbol{\beta}}^i]$. Applying (4.11) recursively, this yields $\boldsymbol{\alpha}^i = [0, \tilde{\boldsymbol{\alpha}}^i]$, as desired.

Now, let $i \in [\![2^L, 2^{L+1} - 1]\!]$, and let $\tilde{i} = 2^{L+1} - i - 1$. Since $\tilde{i} \in [\![0, 2^L - 1]\!]$ we have as before that $\boldsymbol{\alpha}^{\tilde{i}} = [0, \tilde{\boldsymbol{\alpha}}^{\tilde{i}}]$. We wish to show that $\boldsymbol{\alpha}^i = [1, \tilde{\boldsymbol{\alpha}}^{\tilde{i}}]$. Now note that

$$\begin{aligned}\tilde{i} &= 2^{L+1} - i - 1 = 2^{L+1} - \sum_{j=1}^{L+1} 2^{L+1-j}\beta_j - 1 \\ &= 2^{L+1} - 1 - \sum_{j=1}^{L+1} 2^{L+1-j} + \sum_{j=1}^{L+1} 2^{L+1-j}(1 - \beta_j) \\ &= \sum_{j=1}^{L+1} 2^{L+1-j}(1 - \beta_j)\end{aligned}$$

That is, in the binarization for $\tilde{i}$, we have $\boldsymbol{\beta}^{\tilde{i}} = \boldsymbol{\beta}^i \oplus [1, \ldots, 1]$, so that every bit has been flipped. Observing (4.13), we see that this can be induced by enforcing $\alpha_1^{\tilde{i}} = 1 - \alpha_1^i$, with all other $\alpha_j^{\tilde{i}} = \alpha_j^i$: flipping the first α-bit induces a flip in all β-bits. Thus, we obtain that $\boldsymbol{\alpha}^i = [1, \tilde{\boldsymbol{\alpha}}^{\tilde{i}}]$, as desired. □

Lemma 4.3. *Let $J \subseteq [\![L]\!]$ and $\overline{\boldsymbol{\alpha}} \in \{0,1\}^J$. Let $X = \{x \in [\![0, 2^L - 1]\!] : \boldsymbol{\alpha}_J^x = \bar{\boldsymbol{\alpha}}\}$. We will write X as $X = \{x_1, \ldots, x_t\}$, ordered such that $x_j < x_{j+1}$. Let $I = [\![L]\!] \setminus J$. Then $\boldsymbol{\alpha}_I^{x_j}$ is a reflected Gray code for the indices j over X. That is, for any $j \in 1, \ldots, t$, we have*

$$\|\boldsymbol{\alpha}_I{}^{x_j} - \boldsymbol{\alpha}_I{}^{x_{j+1}}\|_1 = 1. \tag{4.15}$$

Furthermore, if $|I| \geq 1$, there exists a $\boldsymbol{\gamma} \in \{0,1\}^{|I|}$ such that, for all $j \in [\![0, t]\!]$, we have

$$\boldsymbol{\alpha}_I^{x_j} \oplus \boldsymbol{\gamma} = \boldsymbol{\alpha}^j_{[\![L-|I|+1, L]\!]}. \tag{4.16}$$

That is, the modified Gray code after fixing some bits is the original reflected Gray code on $|I|$ bits, with some bits flipped.

Proof. We will prove this by induction on L. To enable the use of Lemma 4.2, we will also prove

that, if $|I| \geq 1$, then for all $j \in [\![0, t]\!]$, $i = t - j$, we have

$$x^j = 2^L - x^i - 1. \tag{4.17}$$

Base case: $L = 1$

For $L = 1$, the possibilities are trivial, as there is only one bit. If we do not fix the bit, then we have $\alpha^j = [j]$ for each $j \in \{0, 1\}$; this sequence of two vectors is trivially a Gray code sequence. This yields the original reflected Gray code for $L = 1$, and so $\boldsymbol{\gamma} = [0]$. Finally, we have $t = 1$, and $x_j = j$, yielding, for $i = 1 - j$, $x_j = 1 - x_i = 2^1 - x^i - 1$, as required.

On the other hand, if we do fix the bit, then there are no pairs of consecutive bits, and (4.15) holds by default. In this case, we have $|I| = 0$, so that the other results do not apply.

Inductive step:

Let $L = k$, and suppose the desired properties hold for $L = k - 1$. Let J, $\overline{\boldsymbol{\alpha}}$ be a choice of fixed bits for $L = k$. First, note that if $|J| = L$, then all bits are fixed and the (4.15) holds by default, while the others do not apply, as $|I| = 0$.

Next, suppose $I = \{1\}$, so that the newly added bit is the first unfixed bit. Then we have $t = 1$, and $\boldsymbol{\alpha}^{x_j} = [j, \bar{\boldsymbol{\alpha}}]$, with $\boldsymbol{\alpha}_I^{x_j} = [j]$. Thus, defining $\boldsymbol{\gamma} = [0]$, we have that (4.15) and (4.16) hold trivially. Finally, by Lemma 4.2 and the uniqueness of the L-bit Gray code for j, we have that $x_0 = 2^L - x_1 - 1$ and $x_1 = 2^L - x_0 - 1$, as required.

Otherwise, suppose $|J| \leq L - 1$, with $I \neq \{1\}$. Then there are three cases: $1 \notin J$, or $1 \in J$ and either $\bar{\alpha}_1 = 1$ or $\bar{\alpha}_1 = 0$. Regardless of this choice, the corresponding choices $\tilde{J}$ and $\tilde{\boldsymbol{\alpha}}$ for $L = k - 1$ can be attained by defining $\tilde{J} = J \setminus \{1\}$, and defining $\tilde{I}$ and $\tilde{\boldsymbol{\alpha}}$ accordingly. Consider the corresponding sequence $\tilde{X}$ for $L = k - 1$. Then, by the inductive hypothesis, the desired results hold for $\tilde{X}$, and as $|\tilde{I}| \geq 1$, we can define a corresponding vector $\tilde{\boldsymbol{\gamma}} \in \{0, 1\}^{|\tilde{I}|}$ so that (4.16) holds.

Case 1: $1 \in J$ with $\bar{\alpha}_1 = 0$. In this case, define $\tilde{J} = J \setminus \{1\}$ and $\tilde{\alpha} = \bar{\alpha}_{[\![2,L]\!]}$, and define $\tilde{X}$ and $\tilde{\boldsymbol{\gamma}}$

accordingly, noting that $|X| = |\tilde{X}| = t = 2^{k-|J|-1} - 1$. Let $j \in [\![t]\!]$. Then we have by Lemma 4.2 that $\boldsymbol{\alpha}^{x_j} = [0, \tilde{\boldsymbol{\alpha}}^{\tilde{x}_j}]$, so that $\boldsymbol{\alpha}_I{}^{x_j} = \tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_j}$. Thus, (4.15) to (4.17) hold by the induction hypothesis, with $\boldsymbol{\gamma} = \tilde{\boldsymbol{\gamma}}$.

Case 2: $1 \in J$ with $\overline{\alpha}_1 = 1$. In this case, define $\tilde{J} = J \backslash \{1\}$ and $\tilde{\alpha} = \bar{\alpha}_{[\![2,L]\!]}$, and define $\tilde{X}$ accordingly, noting that $|X| = |\tilde{X}| = t$. Let $j \in [\![t]\!]$. Then, since $t - j = 2^{k-|J|-1} - j - 1$, we have by Lemma 4.2 that $\boldsymbol{\alpha}^{x_j} = [1, \tilde{\boldsymbol{\alpha}}^{\tilde{x}_{t-j}}]$, so that $\boldsymbol{\alpha}_I{}^{x_j} = \tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_{t-j}}$. That is, the sequence of $\boldsymbol{\alpha}_I^{x_j}$'s the sequence of $\tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_j}$'s, but in reversed order. Thus, (4.15) and (4.17) hold by the induction hypothesis, as reversing the order of a sequence has no impact on results for consecutive or centrally reflected terms. Furthermore, by Lemma 4.2 and the induction hypothesis, we have that reversing the order of the sequence corresponds with flipping only the first bit $\alpha_1^{x_j}$, so that we can define $\boldsymbol{\gamma} = \tilde{\boldsymbol{\gamma}} \oplus [1, 0, \ldots, 0]$ to obtain (4.16).

Case 3: $1 \notin J$, so that the first bit is unfixed. In this case, define $\tilde{J} = J$ and $\tilde{\boldsymbol{\alpha}} = \bar{\boldsymbol{\alpha}}$, and define $\tilde{X}$ accordingly. Then $\tilde{t} = |\tilde{X}| = 2^{k-|J|-1}$. Let $j \in [\![0, t]\!]$ and let $i = t - j = 2^{k-|J|} - j - 1$, so that $j = 2^{k-|J|} - i - 1$ Then, by Lemma 4.2, we can construct X as follows:

1. If $j \in [\![0, 2^{k-|J|-1} - 1]\!]$, then $\boldsymbol{\alpha}^{x_j} = [0, \tilde{\boldsymbol{\alpha}}^{\tilde{x}_j}]$
2. If $j \in [\![2^{k-|J|-1}, 2^{k-|J|} - 1]\!]$, then $\boldsymbol{\alpha}^{x_j} = [1, \tilde{\boldsymbol{\alpha}}^{\tilde{x}_i}]$.

This yields (4.17) immediately. Further, define $\boldsymbol{\gamma} = [0, \tilde{\boldsymbol{\gamma}}]$. Then for $j \in [\![0, 2^{k-|J|-1} - 1]\!]$, we have

$$\boldsymbol{\alpha}_I^{x_j} \oplus \boldsymbol{\gamma} = [0, \tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_j}] \oplus [0, \tilde{\boldsymbol{\gamma}}] = [0, \tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_j} \oplus \tilde{\boldsymbol{\gamma}}] = [0, \boldsymbol{\alpha}^j_{[\![L-|I|+2,L]\!]}] = \boldsymbol{\alpha}^j_{[\![L-(|I|)+1,L]\!]},$$

yielding (4.16). For $j \in [\![2^{k-|J|-1}, 2^{k-|J|-1} - 1]\!]$, defining $i = t - j = 2^{k-|J|} - j - 1 \in [\![\tilde{t} + 1, t]\!]$, we have by Lemma 4.2 that

$$\boldsymbol{\alpha}_I^{x_j} \oplus \boldsymbol{\gamma} = [1, \tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_i}] \oplus [0, \tilde{\boldsymbol{\gamma}}] = [1, \tilde{\boldsymbol{\alpha}}_I^{\tilde{x}_i} \oplus \tilde{\boldsymbol{\gamma}}] = [1, \boldsymbol{\alpha}^i_{[\![L-|I|+2,L]\!]}] = \boldsymbol{\alpha}^j_{[\![L-(|I|)+1,L]\!]},$$

again yielding (4.16).

Now, (4.15) trivially holds for all $j \neq 2^{k-|J|-1} - 1$ as in cases 1 and 2, since the first bits of $\boldsymbol{\alpha}^{x_j}$ and $\boldsymbol{\alpha}^{x_{j+1}}$ match, and exactly one other bit differs by the induction hypothesis. Otherwise, if $j = 2^{k-|J|-1} - 1$, then $i = 2^{k-|J|} - (2^{k-|J|-1} - 1) - 1 = 2^{k-|j|} = j + 1$, and thus x_j and x_{j+1} differ by exactly the first bit α_1 as indicated above. Thus, (4.15) holds. From these cases, (4.15) to (4.17) hold by induction. □

Next we show one more property of the code that occurs when extending the Gray code by 1 bit.

Proposition 4.4. *For an integer $i \in [\![0, 1, \ldots, 2^L - 1]\!]$, let $\boldsymbol{\alpha}^i$ and $\boldsymbol{\beta}^i$ be the L-bit Gray code and binary representations of i. Let $\tilde{\boldsymbol{\alpha}}^{2i}$ and $\tilde{\boldsymbol{\alpha}}^{2i+1}$ be the $(L+1)$-bit Gray codes of the integers $2i$ and $2i+1$.*

Then

$$\tilde{\boldsymbol{\alpha}}^{2i} = \left[\alpha_1^i, \ldots, \alpha_L^i, \beta_L^i\right] \quad \textit{and} \quad \tilde{\boldsymbol{\alpha}}^{2i+1} = \left[\alpha_1^i, \ldots, \alpha_L^i, 1 - \beta_L^i\right].$$

Proof. First, note that

$$\tilde{\boldsymbol{\beta}}^{2i} = [\beta_1, \ldots, \beta_L, 0] \ \textit{and} \ \tilde{\boldsymbol{\beta}}^{2i+1} = [\beta_1, \ldots, \beta_L, 1].$$

Then

$$\tilde{\alpha}_{L+1}^{2i} = \tilde{\beta}_{L+1}^{2i} \oplus \tilde{\beta}_L^{2i} = 0 \oplus \beta_L^i = \beta_L^i \quad \text{and} \quad \tilde{\alpha}_{L+1}^{2i+1} = \tilde{\beta}_{L+1}^{2i} \oplus \tilde{\beta}_L^{2i} = 1 \oplus \beta_L^i = 1 - \beta_L^i. \tag{4.18}$$

Furthermore, $\tilde{\beta}_j^{2i} = \beta_j^{2i+1} = \beta^i$ for all $j = 1, \ldots, L$, by definition, we have that $\tilde{\alpha}_j^{2i} = \tilde{\alpha}_j^{2i+1} = \alpha_j^i$ for all $j = 1, \ldots, L$. □

4.4 Formulation Strength

The *strength* of a MIP formulation is a commonly used metric to assess its potential computational performance. We will work with the three following notions of strength.

Definition 4.5. Consider a set $U \subseteq \mathbf{R}^n$. For a formulation $P^{\text{IP}} = \{(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{z}) \in P^{\text{LP}} : \boldsymbol{z} \in \mathbb{Z}^L\}$, where $P^{\text{LP}} \subseteq \mathbf{R}^{n+d+L}$ and $\text{proj}_{\boldsymbol{u}}(P^{\text{IP}}) = U$, we say the the formulation P^{LP} is

- *sharp* if $\text{proj}_{\boldsymbol{u}}(P^{\text{LP}}) = \text{conv}(U)$.
- *hereditarily sharp* if, for all $I \subseteq [\![L]\!]$ and $\bar{\boldsymbol{z}}_I \in \mathbb{Z}^{|I|}$, we have

$$\text{proj}_{\boldsymbol{u}}(P^{\text{LP}}|_{\bar{\boldsymbol{z}}_I = \boldsymbol{z}_I}) = \text{conv}(\{\boldsymbol{u} \in U : (\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{z}) \in P^{\text{LP}}\}|_{\bar{\boldsymbol{z}}_I = \boldsymbol{z}_I}).$$

- *ideal* if $\text{proj}_{\boldsymbol{z}}(\text{ext}(P^{\text{LP}})) \subseteq \mathbb{Z}^L$.

These definitions closely follow those in [155], except we define hereditary sharpness explicitly in terms of the current branch.

In this section, we explore the strength of our MIP formulation (4.8). We also draw an interesting connection between how our formulation represents feasible points through a Gray code: in essence, feasible points are represented in the formulation by their L most significant digits in a binary expansion.

For our particular problem, define

$$\begin{aligned} P^{\text{LP}} &:= \{(x, y, \boldsymbol{\alpha}, \boldsymbol{g}) \in [0,1] \times \mathbf{R}^+ \times [0,1]^L \times [0,1]^{L+1} : (4.8)\} \\ P^{\text{IP}} &:= \{(x, y, \boldsymbol{\alpha}, \boldsymbol{g}) \in [0,1] \times \mathbf{R}^+ \times \{0,1\}^L \times [0,1]^{L+1} : (4.8)\} \end{aligned} \tag{4.19}$$

and

$$\begin{aligned} Q^{\text{LP}} &:= \text{proj}_{x,y}(P^{\text{LP}}) \\ Q^{\text{IP}} &:= \text{proj}_{x,y}(P^{\text{IP}}). \end{aligned} \tag{4.20}$$

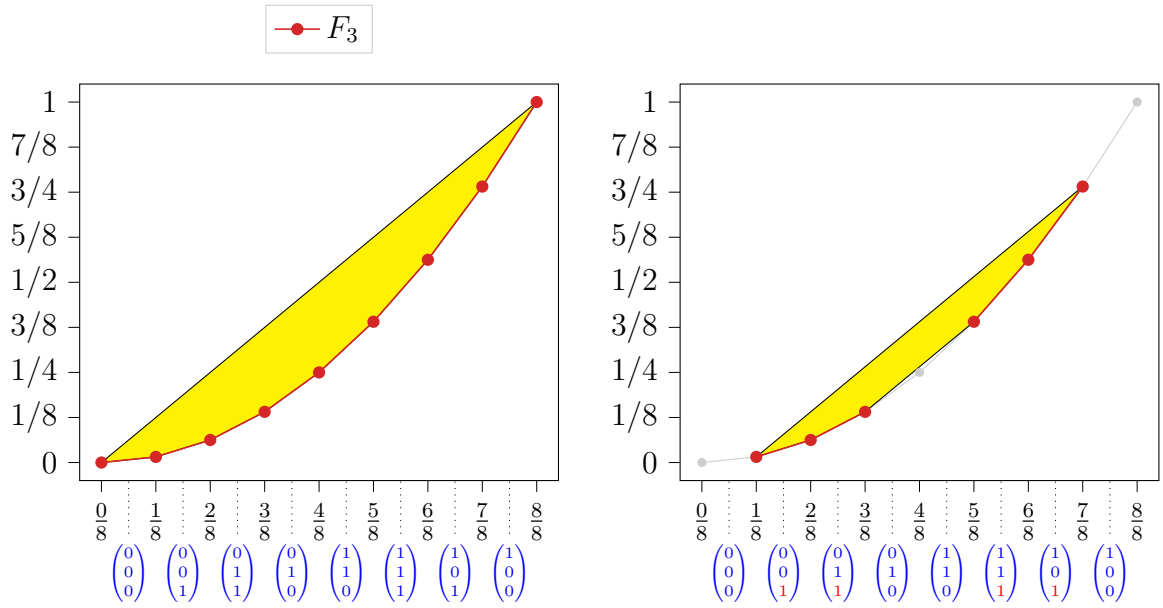


Figure 4.3: *Left:* The piecewise linear approximation Q^{IP} in red and the linear relaxation Q^{LP} in yellow filled in. The formulation is sharp because Q^{LP} is the convex hull of Q^{IP}. The vectors $\boldsymbol{\alpha} \in \{0,1\}^3$ below are the corresponding binary α variables for any x value on that interval $(\frac{i}{8}, \frac{i+1}{8})$. *Right:* The branch $Q^{\text{IP}}|_{\alpha_3=1}$ in red and the linear relaxation $Q^{\text{LP}}|_{\alpha_3=1}$ in yellow. This demonstrates hereditary sharpness since it holds that $Q^{\text{LP}}|_{\alpha_3=1}$ is the convex hull of $Q^{\text{IP}}|_{\alpha_3=1}$.

First, we show that Q^{IP} is, unfortunately, not ideal.

Example 4.6. The formulation P^{IP} approximating $y = x^2$ is not ideal.

Proof. Consider $L = 2$, $x = 0.25$, $\boldsymbol{\alpha} = [\frac{1}{2}, 1]$, $\boldsymbol{g} = [\frac{1}{2}, 1]$, and $y = 0.25 - 2^{-2}(\frac{1}{2}) - 2^{-4}(0.25) = \frac{11}{64}$. This point is chosen to maximize g_2 along a facet $g_2 \leq 2 \cdot (2_x - \alpha_1)$ of the convex hull. It is an extreme point because it is incident with six facets: $g_2 \leq 1$, $\alpha_2 \leq 1$, $g_2 \leq 2g_1$, $g_1 \leq 2x$, $\alpha_1 - x \leq g_2$, and $y = x - 2^{-2}g_1 + 2^{-4}g_2$. Thus, P^{LP} has a fractional extreme point, and so is not ideal. □

Next, we will show that Q^{IP} is sharp. To assist deriving this result, we define the generic sawtooth function $G\colon\ \mathbf{R} \to \mathbf{R}$ as

$$g_i = G(g_{i-1}) := \begin{cases} 2g_{i-1} & g_{i-1} < \frac{1}{2}, \\ 2(1 - g_{i-1}) & g_{i-1} \geq \frac{1}{2}. \end{cases} \tag{4.21}$$

Theorem 4.7. *The formulation P^{IP} is sharp for Q^{IP}.*

Proof. For sharpness, we wish to show that $Q^{\text{LP}} = \text{conv}(Q^{\text{IP}})$. Clearly $Q^{\text{LP}} \supseteq \text{conv}(Q^{\text{IP}})$ from validity of our formulation; therefore, we focus on showing that $Q^{\text{LP}} \subseteq \text{conv}(Q^{\text{IP}})$. We start by fixing some $\bar{x} \in [0,1]$. The result then follows if we can show that the "slice" of Q^{LP} at $\bar{x}$, $Q^{\text{LP}}|_{x=\bar{x}} \coloneqq \{\, y \mid (\bar{x}, y) \in Q^{\text{LP}} \,\}$, is contained in $\text{proj}_x(\text{conv}(Q^{\text{IP}}))$. Since $Q^{\text{LP}}|_{x=\bar{x}} \subset [0,1]$ and is convex, it suffices to show that both its maximum value and minimum value are contained in $\text{conv}(Q^{\text{IP}})$.

First, let $y^* = \max\{y \in Q^{\text{LP}}|_{x=\bar{x}}\}$. From Q^{LP}, we know that $y = x - \sum_{i=1}^{L} 2^{-2i} g_i \leq x$ since $g_i \geq 0$ for all i. Hence, $y^* \leq \bar{x}$. Next, we observe that, as $(0,0), (1,1) \in Q^{\text{IP}}$, convexity in turn implies that $(\bar{x}, \bar{x}) \in \text{conv}(Q^{\text{IP}}) \subseteq Q^{\text{LP}}$. Therefore, $y^* = \bar{x}$ by maximality, and so $(\bar{x}, y^*) \in \text{conv}(Q^{\text{IP}})$.

Next, let $y^* = \min\{y \in Q^{\text{LP}}|_{x=\bar{x}}\}$. From definition, it follows that there is some $\boldsymbol{g}^*$ and $\boldsymbol{\alpha}^*$ such that $(\bar{x}, y^*, \boldsymbol{g}^*, \boldsymbol{\alpha}^*) \in P^{\text{LP}}$. Define G as in (4.21). If $g_i^* = G(g_{i-1}^*)$ for each $i \in [\![L]\!]$, then we immediately conclude that there exists some $\tilde{\boldsymbol{\alpha}} \in \{0,1\}^L$ such that $(\bar{x}, y^*, \boldsymbol{g}^*, \tilde{\boldsymbol{\alpha}}) \in P^I$. This, in turn, implies that $(\hat{x}, y^*) \in Q^{\text{IP}}$.

Otherwise, take $i \in [\![L]\!]$ as the largest index such that $g_i^* > G(g_{i-1}^*)$. Then, recursively define $\tilde{\boldsymbol{g}}$ such that $\tilde{g}_j = \begin{cases} g_j^* & j < i \\ G(\tilde{g}_{j-1}) & j \geq i \end{cases}$ for each $j \in \{0, \ldots, L\}$. Further, take $\tilde{y} = \bar{x} - \sum_{i=1}^{L} 2^{-2i} \tilde{g}_i$, with $\tilde{\alpha}_i = g_{i-1}$ for all i, inducing only lower bounds of 0 on all g_j. Then $(\tilde{y}, \bar{x}, \tilde{\boldsymbol{g}}, \tilde{\boldsymbol{\alpha}}) \in Q^{\text{LP}}$. We now show that $\tilde{y} < y^*$, contradicting the minimality of y^*.

Let $\varepsilon = \tilde{g}_i - g_i^*$. Note that $G \colon \mathbf{R} \to \mathbf{R}$ is Lipschitz continuous with Lipschitz constant 2. That is, for any $g, g' \in [0,1]$, we have that $|G(g) - G(g')| \leq 2|g - g'|$. Hence, since $|g_i^* - \tilde{g}_i| \leq \varepsilon$ we conclude

inductively that $|g^*_{i+k} - \tilde{g}_{i+k}| \leq 2^k \varepsilon$ for each $k \in [\![L-i]\!]$. Thus, we have

$$\begin{aligned}
y^* - \tilde{y} &= \sum_{j=i}^{L} 2^{-2j}(\tilde{g}_j - g^*_j) \\
&\geq 2^{-2i}(\tilde{g}_i - g^*_i) + \sum_{j=i+1}^{L} 2^{-2j}|g^*_j - \tilde{g}_j| \\
&\geq 2^{-2i}\varepsilon - \textstyle\sum_{j=i+1}^{L} 2^{-2j}2^{j-i}\varepsilon \\
&= 2^{-2i}\varepsilon(1 - 2^i \sum_{j=i+1}^{L} 2^{-j}) \\
&= 2^{-2i}\varepsilon(1 - 2^i(2^{-i} - 2^{-L})) \\
&= \varepsilon(2^{-i-l}) > 0.
\end{aligned}$$

Therefore, $\tilde{y} < y^*$, contradicting the minimality of y^*. From this, we conclude that no such i exists, completing the proof. □

We now show the correspondence between our formulation and the Gray code. It is helpful to note that, in P^{IP}, we have $g_i = G(g_{i-1})$, where G is as defined in (4.21).

Theorem 4.8 (The MIP formulation follows a Gray code)**.** *Take* $(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in \text{proj}_{x,\boldsymbol{g},\boldsymbol{\alpha}}(P^{\text{IP}}|_{g_L \in (0,1)})$ *for some fixed value* $g_L \in (0,1)$*. Fix some* $j \in [\![0, L-1]\!]$*, and define* $i_j \coloneqq \lfloor 2^{L-j} g_j \rfloor$*. Then* $[\alpha_{j+1}, \ldots, \alpha_L]$ *is the* $L-j$*-bit Gray code* $\boldsymbol{\alpha}^{i_j}$ *for* i_j*, and* $g_j \in (\frac{i_j}{2^{L-j}}, \frac{i_j+1}{2^{L-j}})$*.*

Proof. First, we note that, for each $j \leq L+1$, we have

$$g_j = \begin{cases} \frac{1}{2} g_{j+1} & \alpha_{j+1} = 0, \\ 1 - \frac{1}{2} g_{j+1} & \alpha_{j+1} = 1. \end{cases}$$

We will proceed by induction in a manner similar to the proof for Lemma 4.3.

Base Case: $j = L-1$

In this case, if $\alpha_L = 0$, then $g_{L-1} = \frac{1}{2}g_L \in (0, \frac{1}{2})$, and so $i_{L-1} = \lfloor 2g_{L-1} \rfloor = 0$, whose 1-bit Gray code is $[0] = [\alpha_L]$, as desired. On the other hand, if $\alpha_L = 1$, then $g_{L-1} = 1 - \frac{1}{2}g_L \in (\frac{1}{2}, 1)$, and so $i_{L-1} = \lfloor 2g_{L-1} \rfloor = 1$, whose 1-bit Gray code is $[1] = [\alpha_L]$, as desired.

Inductive step: Let $j \in [\![0, L-2]\!]$ and suppose the statement holds $j+1$.

If $\alpha_{j+1} = 0$, then $g_j = \frac{1}{2}g_{j+1} \in \left(\frac{i_{j+1}}{2^{L-j}}, \frac{i_{j+1}+1}{2^{L-j}}\right)$ and $2^{L-j}g_j \in (i_{j+1}, i_{j+1}+1)$. In this case, we have $i_j = \left\lfloor 2^{L-j}g_j \right\rfloor = i_{j+1} \in [\![0, 2^{L-j-1}-1]\!]$. Thus, by Lemma 4.2, we have that the $L-j$-bit Gray code for i_j is given by $\boldsymbol{\alpha}^{i_j} = [0\ \tilde{\boldsymbol{\alpha}}^{i_{j+1}}]$, where $\tilde{\boldsymbol{\alpha}}^{i_j}$ is the $(L-j-1)$-bit Gray code for i_j, which is $[\alpha_{j+2} \dots \alpha_L]$ by the induction hypothesis. This yields $\boldsymbol{\alpha}^{i_j} = [0\ \alpha_{j+2} \dots \alpha_L] = [\alpha_{j+1} \dots \alpha_L]$ as desired. Further, observing the bounds we derived for g_j, we have $g_j \in (\frac{i_j}{2^{L-j}}, \frac{i_j+1}{2^{L-j}})$.

On the other hand, if $\alpha_{j+1} = 1$, then $g_j = 1 - \frac{1}{2}g_{j+1} \in (1 - \frac{i_{j+1}+1}{2^{L-j}}, 1 - \frac{i_{j+1}}{2^{L-j}})$ and $2^{L-j}g_j \in (2^{L-j} - i_{j+1} - 1, 2^{L-j} - i_{j+1})$. Thus, we have that $i_j = 2^{L-j} - i_{j+1} - 1 \in [\![2^{L-j-1}, 2^{L-j}-1]\!]$. Thus, by Lemma 4.2, computing $\tilde{i}_j = 2^{L-j} - 1 - i_j = i_{j+1}$, we have that the $(L-j)$-bit Gray code for i_j is $\boldsymbol{\alpha}^{i_j} = [1\ \tilde{\boldsymbol{\alpha}}^{\tilde{i}_j}]$, where $\tilde{\boldsymbol{\alpha}}^{\tilde{i}_j}$ is the $(L-j-1)$-bit Gray code for $\tilde{i}_j$, which by the induction hypothesis is given as $[\alpha_{j+2} \dots \alpha_L]$. This yields $\boldsymbol{\alpha}^{i_j} = [1\ \alpha_{j+2} \dots \alpha_L] = [\alpha_{j+1} \dots \alpha_L]$ as desired. Further, observing the bounds we derived for g_j, we have $g_j \in (\frac{i_j}{2^{L-j}}, \frac{i_j+1}{2^{L-j}})$.

Thus, with these cases, the result holds by induction. □

Note that, if $g_L \in \{0, 1\}$ (so that $x \in 2^{-L}\mathbb{Z} \cap (0, 1)$), the same Gray codes from Theorem 4.8 can be used as when $g_L \in (0, 1)$. The primary difference is that there two choices for this Gray code when $x \in 2^{-L}\mathbb{Z} \cap (0, 1)$. This dichotomy stems from the fact that we will obtain $g_j = \frac{1}{2}$ from some j, introducing ambiguity in the choice of α_j.

4.4.1 Hereditary Sharpness

In this section, we prove hereditary sharpness of the formulation P^{IP}.

Let L be a nonnegative integer, and define the sets P^{IP}, P^{LP}, Q^{IP}, and Q^{LP} as before. Suppose we fix

some subset of the binary variables to $\boldsymbol{\alpha}_I = \bar{\boldsymbol{\alpha}}_I \in \{0,1\}^{|I|}$ for some set $I \subseteq L$. Furthermore, define $P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I} := P^{\text{IP}}|_{\boldsymbol{\alpha}_I = \bar{\boldsymbol{\alpha}}_I} := \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) \in P^{\text{IP}} | \alpha_i = \bar{\alpha}_i \text{ for } i \in I\}$, and similarly for Q^{LP}, Q^{IP}, and P^{LP}. We then wish to show $Q^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I} = \text{conv}(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$. We will use this notation for the remainder of this section. An example demonstrating the hereditary sharpness of the formulation is shown in Figure 4.3

To study this relationship in more detail, we in particular wish to study $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$. Let $(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) \in P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$. For each $i \in I$, then g_{i-1} relates to g_i via the linear function

$$g_i = 2g_{i-1}(1 - \bar{\alpha}_i) + 2(1 - g_{i-1})\bar{\alpha}_i. \tag{4.22}$$

Alternatively, for each $i \in [\![L]\!] \backslash I$, the relationship can be written as

$$2|g_{i-1} - \alpha_i| \leq g_i \leq \min\{2g_{i-1}, 2(1 - g_{i-1})\}.$$

In this form, simply setting each $\alpha_i = g_{i-1}$ yields the least restrictive possible lower-bound of 0 on g_i in terms of g_{i-1}. Thus, making this choice, we find that $\text{proj}_{x,y,\boldsymbol{g}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$ can be expressed via the constraints

$$\begin{aligned}
y &= g_0 - \sum_{i=1}^{L} 2^{-2i} g_i \\
g_0 &= x \\
g_i &\leq 2g_{i-1} && i \in [\![L]\!], i \notin I \\
g_i &\leq 2(1 - g_{i-1}) && i \in [\![L]\!], i \notin I \\
g_i &= 2g_{i-1}(1 - \bar{\alpha}_i) + 2(1 - g_{i-1})\bar{\alpha}_i && i \in I \\
g_i &\in [0, 1] && i \in [\![L]\!]
\end{aligned} \tag{4.23}$$

while, after combining the linear constraints with the new constraints (4.22) fixing variables α_i for

$i \in I$, $P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$ can be written as

$$\begin{aligned}
y &= g_0 - \sum_{i=1}^{L} 2^{-2i} g_i \\
g_0 &= x \\
2(g_{i-1} - \alpha_i) &\le g_i \le 2g_{i-1} && i \in [\![L]\!], i \notin I \\
2(\alpha_i - g_{i-1}) &\le g_i \le 2(1 - g_{i-1}) && i \in [\![L]\!], i \notin I \\
g_i &= 2g_{i-1}(1 - \bar{\alpha}_i) + 2(1 - g_{i-1})\bar{\alpha}_i && i \in I \\
g_i &\in [0, 1] && i \in [\![L]\!] \\
\alpha_i &\in \{0, 1\} && i \in [\![L]\!], i \notin I
\end{aligned} \tag{4.24}$$

For convenience, for all $i \in I$, define

$$G_i(g_{i-1}, \alpha_i) := 2g_{i-1}(1 - \alpha_i) + 2(1 - g_{i-1})\alpha_i. \tag{4.25}$$

For $i \in [\![L]\!]$, define the shorthand $G_i(g_{i-1}) := G_i(g_{i-1}, \bar{\alpha}_i)$ if $i \in I$ and $G_i = G$ otherwise. Then by the construction of $P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$, we have that $\boldsymbol{g} \in \text{proj}_{\boldsymbol{g}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$ if and only if for all $i \in [\![L]\!]$, we have $g_i = G_i(g_{i-1})$ and $g_i \in [0, 1]$.

For all $i \in [\![0, L]\!]$, the below proposition explores how to compute the feasible region for g_i, $\text{proj}_{g_i}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) =: [a_i, b_i]$, while establishing that $\text{proj}_{g_i}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = \text{conv}(\text{proj}_{g_i}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}))$.

Lemma 4.9 (Bounds in Projection)**.** *For all $i \in [\![0, L]\!]$ and $I \subseteq [\![L]\!]$, we have* $\text{proj}_{g_i}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = \text{conv}(\text{proj}_{g_i}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) =: [a_i, b_i] \neq \emptyset$. *Furthermore, $[a_L, b_L] = [0, 1]$, and $[a_{i-1}, b_{i-1}]$ can be computed from $[a_i, b_i]$ as*

$$[a_{i-1}, b_{i-1}] = \begin{cases} [\frac{1}{2}a_i, \frac{1}{2}b_i] & \text{if } i \in I \text{ and } \bar{\alpha}_i = 0, \\ [1 - \frac{1}{2}b_i, 1 - \frac{1}{2}a_i] & \text{if } i \in I \text{ and } \bar{\alpha}_i = 1, \\ [\frac{1}{2}a_i, 1 - \frac{1}{2}a_i] & \text{if } i \notin I. \end{cases} \tag{4.26}$$

Note that in the last case $a_{i-1} \le \frac{1}{2}$ and $b_{i-1} \ge \frac{1}{2}$.

Proof. We proceed by induction.

Base Case: $i = L$

In this case, Theorem 4.8 establishes that, even if $\hat{I} = [\![L]\!]$ with some corresponding $\hat{\boldsymbol{\alpha}}_{\hat{I}} \in \{0,1\}^L$, we have $[0,1] = \text{proj}_{g_i}(P^{\text{IP}}_{\hat{\boldsymbol{\alpha}}_{\hat{I}}})$, yielding

$$\begin{aligned}[0,1] &= \text{proj}_{g_L}(P^{\text{IP}}_{\hat{\boldsymbol{\alpha}}_{\hat{I}}}) \subseteq \text{conv}(\text{proj}_{g_L}(P^{\text{IP}}_{\hat{\boldsymbol{\alpha}}_{\hat{I}}})) \\ &\subseteq \text{conv}(\text{proj}_{g_L}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) \subseteq \text{proj}_{g_L}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) \subseteq [0,1],\end{aligned}$$

and so $\text{conv}(\text{proj}_{g_L}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) = \text{proj}_{g_L}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = [0,1]$, as required.

Inductive step:

Let $i \in [\![L]\!]$, and suppose that $\text{conv}(\text{proj}_{g_i}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) = \text{proj}_{g_i}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = [a_i, b_i]$. Then, observing (4.23) and (4.24), we find that there are three cases.

Case 1: If $i \in I$ and $\bar{\alpha}_i = 0$, then in both $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$ and $P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$, we have $g_{i-1} = \frac{1}{2}g_i$, yielding

$$\text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) \subseteq \text{proj}_{g_{i-1}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = [\tfrac{1}{2}a_i, \tfrac{1}{2}b_i].$$

Furthermore, $a_i, b_i \in \text{proj}_{g_i}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$ yields $\frac{1}{2}a_i, \frac{1}{2}b_i \in \text{proj}_{g_{i-1}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$. This implies $[\frac{1}{2}a_i, \frac{1}{2}b_i] \subseteq \text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}))$, yielding

$$\text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) = \text{proj}_{g_{i-1}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = [1 - \tfrac{1}{2}a_i, 1\tfrac{1}{2}b_i]$$

Note that $[a_i, b_i] \neq \emptyset \Rightarrow [\frac{1}{2}a_i, \frac{1}{2}b_i] \neq \emptyset$, and that $[a_i, b_i] \subseteq [0,1] \Rightarrow [\frac{1}{2}a_i, \frac{1}{2}b_i] \subseteq [0,1]$.

Case 2: If $i \in I$ and $\bar{\alpha}_i = 1$, then in both $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$ and $P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$, we have $g_i = 1 - \frac{1}{2}g_i$, yielding

$$\text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) \subseteq \text{proj}_{g_{i-1}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = [1 - \tfrac{1}{2}b_i, 1 - \tfrac{1}{2}a_i].$$

Furthermore, $a_i, b_i \in \text{proj}_{g_i}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$ yields $1 - \frac{1}{2}b_i, 1 - \frac{1}{2}a_i \in \text{proj}_{g_{i-1}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$. This implies $[1 - \frac{1}{2}b_i, 1 -$

$\frac{1}{2}a_i] \subseteq \text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}))$, yielding

$$\text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I})) = \text{proj}_{g_{i-1}}(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}) = [1 - \tfrac{1}{2}b_i, 1 - \tfrac{1}{2}a_i].$$

Note that $[a_i, b_i] \neq \emptyset \Rightarrow [1 - \frac{1}{2}b_i, 1 - \frac{1}{2}a_i] \neq \emptyset$, and $[a_i, b_i] \subseteq [0, 1] \Rightarrow [1 - \frac{1}{2}b_i, 1 - \frac{1}{2}a_i] \subseteq [0, 1]$.

Case 3: If $i \notin I$, then $P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}$ and $P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}$ model different relations between g_i and g_{i-1}.

In $P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}$, we have

$$\begin{aligned} g_i &\leq 2g_{i-1} \\ g_i &\leq 2(1 - g_{i-1}), \end{aligned}$$

which can be written as

$$\begin{aligned} g_{i-1} &\geq \tfrac{1}{2}g_i \geq \tfrac{1}{2}a_i \\ g_{i-1} &\leq 1 - \tfrac{1}{2}g_i \leq 1 - \tfrac{1}{2}a_i. \end{aligned}$$

Further, as $a_i \in \text{proj}_{g_i}(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I})$, we have that

$$\text{proj}_{g_{i-1}}(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}) = [\tfrac{1}{2}a_i, 1 - \tfrac{1}{2}a_i]),$$

where $a_i \in [0, 1]$ implies $\frac{1}{2}a_i \in [0, \frac{1}{2}]$, so that $[\frac{1}{2}a_i, 1 - \frac{1}{2}a_i] \neq \emptyset$. Thus,

$$\text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I})) \subseteq \text{proj}_{g_{i-1}}(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}) = [\tfrac{1}{2}a_i, 1 - \tfrac{1}{2}a_i].$$

To show $[\frac{1}{2}a_i, 1 - \frac{1}{2}a_i] \subseteq \text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I})$, we have only to show that the endpoints are in $\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I})$. To this end, let $g_i = a_i$. In $P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}$, we can choose either $\alpha_i = 0$ or $\alpha_i = 1$. If $\alpha_i = 0$, we have that $g_{i-1} = \frac{1}{2}g_i = \frac{1}{2}a_i$, while if $\alpha_i = 1$, we have $g_{i-1} = 1 - \frac{1}{2}g_i = 1 - \frac{1}{2}a_i$. Thus, we have $[\frac{1}{2}a_i, 1 - \frac{1}{2}a_i] \subseteq \text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}))$, yielding

$$\text{conv}(\text{proj}_{g_{i-1}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I})) = \text{proj}_{g_{i-1}}(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}) = [\tfrac{1}{2}a_i, 1 - \tfrac{1}{2}a_i] \neq \emptyset,$$

as desired. □

Note that $\text{conv}(\text{proj}_x(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})) = \text{proj}_x(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$ is a direct corollary of the above lemma. This fact will be helpful during the proof of hereditary sharpness. Next, to prove hereditary sharpness, we want to show that if we fix some g_i to either a_i or b_i, then for each $j > i$, we have that g_j has only one feasible solution e_j in $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$, where $e_j \in \{a_i, b_i\}$.

Lemma 4.10 (Single solution at endpoints)**.** *For all $i \in [\![0, L]\!]$ and $\hat{g}_i \in \{a_i, b_i\}$, we have for all $j \geq i$ that $\text{proj}_{g_j}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{g_i=\hat{g}_i}) = \{e_j\}$, where $e_j \in \{a_j, b_j\}$.*

Proof. We will proceed by induction. Let $\hat{g}_0 \in \{a_0, b_0\}$. Fortunately, the base case is trivial, since we have chosen $g_i \in \{a_i, b_i\}$.

Thus, for an induction, let $j \in [\![i{+}1, L]\!]$, and assume that $\text{proj}_{g_{j-1}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{g_i=\hat{g}_i}) = \{e_{j-1}\} \subset \{a_{j-1}, b_{j-1}\}$. Then there are three cases.

Case 1: If $j \in I$ and $\bar{\alpha}_j = 0$, then we have $g_j = 2g_{j-1}$, so that $\text{proj}_{g_j}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{g_i=\hat{g}_i}) = \{2e_{j-1}\} =: \{e_j\}$. Now, by Lemma 4.9, we have that $a_j = 2a_{j-1}$ and $b_j = 2b_{j-1}$, so that $e_{j-1} \in \{a_{j-1}, b_{j-1}\} \Rightarrow e_j \in \{a_j, b_j\}$.

Case 2: If $j \in I$ and $\bar{\alpha}_j = 1$, then we have $g_j = 2(1 - g_{j-1})$, so that $\text{proj}_{g_j}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{g_j=\hat{g}_j}) = \{2(1 - e_{j-1})\} =: \{e_j\}$. Now, by Lemma 4.9, we have that $a_j = 2(1 - b_{j-1})$ and $b_j = 2(1 - a_{j-1})$, so that $e_{j-1} \in \{a_{j-1}, b_{j-1}\} \Rightarrow e_j \in \{a_j, b_j\}$.

Case 3: If $j \notin I$, then we have by Lemma 4.9 $a_j = 2a_{j-1} = 2(1 - b_{j-1})$, with $a_j \leq \frac{1}{2}$ and $b_j \geq \frac{1}{2}$. Further, we have in $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$ that

$$g_j \leq 2g_{j-1} = 2e_{j-1},$$
$$g_j \leq 2(1 - g_{j-1}) = 2(1 - e_{j-1}).$$

Now, if $e_{j-1} = a_{j-1}$, then $2e_{j-1} = 2a_{j-1} \leq 1$, while $2(1 - e_{j-1}) = 2(1 - a_{j-1}) \geq 1$. Thus, these bounds consolidate to $a_j \leq g_j \leq 2a_{j-1} = a_j$, which implies $g_j = a_j$.

On the other hand, if $e_{j-1} = b_{j-1}$, then $2e_{j-1} = 2b_{j-1} \geq 1$, while $2(1 - e_{j-1}) = 2(1 - b_{j-1}) \leq 1$. Thus, these bounds consolidate to $a_j \leq g_j \leq 2(1 - b_{j-1}) = a_j$, which implies $g_j = a_j$. Thus, in this

case, we have $\mathrm{proj}_{g_j}(P^{\mathrm{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{g_i=\hat{g}_i}) = \{a_j\} =: \{e_j\}$. This completes the proof. □

Now, computing all a_i and b_i via Lemma 4.9, we obtain the following form for $\mathrm{proj}_{x,y,\boldsymbol{g}}\{P^{\mathrm{LP}}|_{I,\bar{\boldsymbol{\alpha}}_I}\}$:

$$\begin{aligned}
y &= g_0 - \sum_{i=1}^{L} 2^{-2i} g_i \\
g_0 &= x \\
g_i &\le 2g_{i-1} && i \in [\![L]\!] \setminus I \\
g_i &\le 2(1-g_{i-1}) && i \in [\![L]\!] \setminus I \\
g_i &= 2g_{i-1}(1-\bar{\alpha}_i) + 2(1-g_{i-1})\bar{\alpha}_i && i \in I \\
g_i &\in [a_i, b_i] && i \in [\![L]\!].
\end{aligned} \tag{4.27}$$

Now, note that, for each $i \in [\![L]\!]$, g_i has negative coefficient in first equation of (4.27). Note also that this is the only constraint in (4.27) involving y. Thus, if we are to minimize over y, then we implicitly maximize g_i, and so g_i will tend towards its upper bound. Furthermore, it turns out that, in any y-minimal or y-maximal solution in $P^{\mathrm{LP}}_{\bar{\boldsymbol{\alpha}}_I}$ given some fixed value of x, each g_i can be explicitly computed from g_{i-1} using only the constraints from (4.27) directly connecting g_i and g_{i-1}. We refer to this as the *greedy* solution property described in Lemma 4.11 below, and it holds due to the rapid decay of coefficients of g_i's.

For each $i \in [\![L]\!] \setminus I$, let b_i be as in Lemma 4.9, and define

$$u_i(g_{i-1}) \coloneqq \min\{b_i, 2g_{i-1}, 2(1-g_{i-1})\}. \tag{4.28}$$

Lemma 4.11. *Let a_i, b_i as in Lemma 4.9, and let $\hat{x} \in [a_0, b_0]$. Define*

$$(y^*, \boldsymbol{g}^*) = \arg\min\{y \,:\, (y, \boldsymbol{g}) \in \mathrm{proj}_{y,\boldsymbol{g}}(P^{\mathrm{LP}}_{I,\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}})\}, \tag{4.29a}$$

$$(y^\star, \boldsymbol{g}^\star) = \arg\max\{y \,:\, (y, \boldsymbol{g}) \in \mathrm{proj}_{y,\boldsymbol{g}}(P^{\mathrm{LP}}_{I,\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}})\}. \tag{4.29b}$$

Then, for $i \notin I$, we have

$$g_i^* = u_i(g_{i-1})$$
$$g_i^\star = a_i.$$

That is, one of the upper bounds is tight for each g_i when maximizing y, while the domain lower bound is tight for g_i when minimizing y.

Proof. Let $\hat{x} \in [a_0, b_0]$. Then $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}} \neq \emptyset$ by Lemma 4.9.

We begin by proving that $g_i^* = u_i(g_{i-1}^*)$ for all $i \notin I$. Suppose for a contradiction that, for some subset $j \in J \subseteq [\![L]\!] \setminus I$ and $\varepsilon_j > 0$, we have $g_j^* = u_j(g_{j-1}^*) - \varepsilon_j$. Let i be maximal in J, so that $g_j^* = u_j(g_{j-1}^*)$ for all $j > i$. Then we have $i \notin I$, since otherwise we have $g_i^* = 2g_{i-1}(1-\bar{\alpha}_i) + 2(1-g_{i-1}) = u_j(g_{j-1}^*)$ from (4.27).

Let $\tilde{g}_j = u_j(g_{j-1})$ for all $j \geq i$. for convenience, define

$$\tilde{\boldsymbol{g}} = (g_0^*, \dots g_{i-1}^*, \tilde{g}_i, \dots, \tilde{g}_L).$$

To show that this choice is feasible, i.e., $\tilde{\boldsymbol{g}} \in \text{proj}_{\boldsymbol{g}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$, we make an inductive observation. First, note that $\tilde{g}_{i-1} = g_{i-1}^* \in [a_{i-1}, b_{i-1}]$ by Lemma 4.9, with $\tilde{\boldsymbol{g}}_{[\![0,i-1]\!]} \in \text{proj}_{\boldsymbol{g}_{[\![0,i-1]\!]}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$. Furthermore, for any j, $\tilde{\boldsymbol{g}}_{[\![0,j-1]\!]} \in \text{proj}_{\boldsymbol{g}_{[\![0,j-1]\!]}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$ implies that there exists some $g_j \in \text{proj}_{g_j}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{\boldsymbol{g}_{[\![0,j-1]\!]}=\tilde{\boldsymbol{g}}_{[\![0,j-1]\!]}}) \subseteq [a_j, b_j]$, and $\tilde{g}_j$ is the largest such value by construction, yielding $\tilde{g}_j \in [a_j, b_j]$ and $\tilde{g}_{[\![0,j]\!]} \in \text{proj}_{\boldsymbol{g}_{[\![0,j]\!]}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$.

Now, we have by definition that $\tilde{g}_i - g_i^* = \varepsilon_i$. Furthermore, observe that for all $j > i$, $u_j(g_{j-1})$ is Lipschitz continuous with Lipschitz constant 2, yielding

$$|\tilde{g}_j - g_j^*| = |u_j(\tilde{g}_{j-1}) - u_j(g_{j-1}^*)| \leq 2|\tilde{g}_{j-1} - g_{j-1}^*|$$

Applying this recursively yields, for all $j > i$,

$$|\tilde{g}_j - g_j^*| \leq 2^{j-i}\varepsilon.$$

Note that this implies that, for $j > i$, we have $\tilde{g}_j - g_j^* \geq -2^{j-i}\varepsilon$. Then we have

$$\begin{aligned} y^* - \tilde{y} &= 2^{-2i}(\tilde{g}_i - g_i^*) + \textstyle\sum_{j=i+1}^{L} 2^{-2j}(\tilde{g}_j - g_j^*) \\ &\geq 2^{-2i}\varepsilon + \textstyle\sum_{j=i+1}^{L} 2^{-2j}(-2^{j-i}\varepsilon) \\ &= 2^{-i}\varepsilon\left(2^{-i} - \textstyle\sum_{j=i+1}^{L} 2^{-j}\right) \\ &= 2^{-i}\varepsilon(2^{-i} - (2^{-i} - 2^{-L})) \\ &= 2^{-(i+L)}\varepsilon. \end{aligned} \tag{4.30}$$

However, this is a contradiction: it implies $\tilde{y} < y^*$, but y^* was optimal! Thus, all g_i must take on their upper-bounds given g_{i-1}.

Next, we prove that $g_i^\star = a_i$ for all $i \notin I$. The idea behind the proof is identical, with a notable simplifying difference: there is only one (constant) lower-bound a_i on each g_i for $i \notin I$. Thus, when enforcing that each $g_j^\star = a_i$ for $j > i$, $j \notin I$ with $g_i = a_i + \varepsilon$, the shift from $g_i^\star$ to $\tilde{g}_i$ effects no change in g_j, $j > i$. That is, if $i + 1 \notin I$, $\tilde{g}_{i+1} - g_{i+1}^\star = 0$, yielding $\tilde{g}_j - g_j^\star = 0$ for $j \geq i + 1$, so that (4.31) simplifies to

$$\tilde{y} - y^\star = 2^{-2i}(g_i^* - \tilde{g}_i) = 2^{-2i}\varepsilon, \tag{4.31}$$

implying $\tilde{y} > y^\star$, a contradiction. On the other hand, if $i+1 \in I$, let $k = \min\{j \in L - I : j \geq i+2\}$. Then, for each $j \in [\![i + 1, \ldots, k - 1]\!]$, we have $|\tilde{g}_i - g_i^\star| = 2^{j-i}\varepsilon$. Furthermore, as $k \notin I$, we have

$\tilde{g}_k = a_k = g_k^\star$, so that $\tilde{g}_k = g_k^\star$ for all $j > k$, yielding

$$\begin{aligned}
\tilde{y} - y^\star &= 2^{-2i}(g_i^\star - \tilde{g}_i) + \sum_{j=i+1}^{k-1} 2^{-2j}(\tilde{g}_j - g_j^*) \\
&\geq 2^{-2i}\varepsilon + \sum_{j=i+1}^{k-1} 2^{-2j}(-2^{j-i}\varepsilon) \\
&= 2^{-i}\varepsilon \left(2^{-i} - \sum_{j=i+1}^{k-1} 2^{-j}\right) \\
&= 2^{-i}\varepsilon(2^{-i} - (2^{-i} - 2^{-(k-1)})) \\
&= 2^{-(i+k-1)}, \varepsilon
\end{aligned} \tag{4.32}$$

again implying $\tilde{y} > y^\star$, a contradiction. □

Observation 4.12. Since each u_i, defined in (4.28), is a continuous piecewise linear function, Lemma 4.11 recursively implies that each g_i^* is continuous in x, and is a function of g_{i-1}^*.

As a corollary to Lemma 4.10 and Lemma 4.11, we have that, for the optimal solution to the minimization problem in Lemma 4.11, we have that $g_j = b_j < G(g_{j-1})$ for exactly one value of j if $\hat{x}$ is not feasible in $P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}$.

Corollary 4.13. *For all $\hat{x} \in \text{proj}_x(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}) \setminus \text{proj}_x(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I})$, define $(y^*, \boldsymbol{g}^*)$ as in (4.29a). Then we have for exactly one $j \in [\![L]\!] \setminus I$ that $g_j^* = b_j < G(g_{i-1}^*)$.*

Furthermore, let $x^1, x^2 \in \text{proj}_x P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}$ be such that for all $\lambda \in (0,1)$, we have $\hat{x} := \lambda x^1 + (1-\lambda)x^2 \notin \text{proj}_x P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I} = \emptyset$. Then the uniquely determined j discussed above is the same for all $\hat{x} \in (x^1, x^2)$ where (x^1, x^2) denotes the open interval between x^1 and x^2.

Proof. Consider the first $j \in [\![L]\!] \setminus I$ for which $g_j^* = b_j$. Such a j exists; otherwise, Lemma 4.11 would give $g_i^* = G_i(g_{i-1})$ for all $i \in [\![L]\!]$, a contradiction on the choice of $\hat{x}$ since there is a corresponding feasible choice of $\boldsymbol{\alpha}^* \in \{0,1\}^L$. Then, by Lemma 4.10, the set $\text{proj}_{\boldsymbol{g}_{[\![j+1,L]\!]}}(P^{\text{LP}}_{\tilde{\boldsymbol{\alpha}}_I}|_{g_j = g_j^*})$ consists of only a single point, for which each $g_i \in \{a_i, b_i\}$ for $i \geq j+1$. Furthermore, by Lemma 4.10, this point is also in $\text{proj}_{\boldsymbol{g}_{[\![j+1,L]\!]}}(P^{\text{IP}}_{\tilde{\boldsymbol{\alpha}}_I}|_{g_j = g_j^*})$ so that $g_i^* = G_i(g_{j-1}^*)$ for all $i \geq j+1$. Further, if $g_j^* = G(g_{j-1}^*) = b_j$,

then we would obtain $\boldsymbol{g}^* \in \operatorname{proj}_{\boldsymbol{g}} P^{\text{IP}}_{x=\hat{x}}$, a contradiction on the choice of $\hat{x}$. Thus, as $g^*_j \leq G(g^*_{j-1})$ by Lemma 4.11, we must have that $g^*_j = b_j < G(g^*_{j-1})$.

Now, let $x^1, x^2 \in \operatorname{proj}_x P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$ be such that $(x^1, x^2) \cap \operatorname{proj}_x P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I} = \emptyset$. Suppose that there is some $\hat{x} \in (x^1, x^2)$ such that, for all sufficiently small $\varepsilon > 0$, we have that the value j^1 in for $x - \varepsilon$ is different from the value j^2 for $x + \varepsilon$. In this case, since the g^*_j is continuous in x and g^*_{j-1}, we have at $\hat{x}$ that both $g^*_{j_1} = b_{j_1} = G_{j_1}(g^*_{j_1-1})$ and $g^*_{j_2} = b_{j_2} = G_{j_2}(g^*_{j_2})$. Furthermore, for all other $i \in [\![L]\!] \setminus I$, we have by continuity that $g^*_i = G_i(g^*_{i-1})$, yielding $g^*_i = G_i(g^*_{i-1})$ for all i. This yields $\boldsymbol{g}^* \in \operatorname{proj}_{\boldsymbol{g}} P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$, a contradiction on the choice of $\hat{x}$. □

With this greedy solution property and the following corollary, we are ready to prove the hereditary sharpness of P^{IP} as a formulation for Q^{IP}. It is helpful to note that, for the minimizer solutions in Lemma 4.11, $g^*_i < b_i$ implies $g^*_i = G_i(g_{i-1})$. Furthermore, if $\boldsymbol{g} \in \operatorname{proj}_{\boldsymbol{g}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$ and $g^*_i = G_i(g_{i-1})$ for all $i \in [\![L]\!] \setminus I$, then we have $\boldsymbol{g} \in \operatorname{proj}_{\boldsymbol{g}}(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$.

Theorem 4.14. *Q^{IP} is hereditarily sharp.*

Proof. Let a_0, b_0 as in Lemma 4.9, so $[a_0, b_0] = \operatorname{proj}_x(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}) = \operatorname{conv}(\operatorname{proj}_x(P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}))$.

Let $\hat{x} \in [a_0, b_0]$. We need to show that $Q^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I} = \operatorname{conv}(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$. Since both sets are compact convex sets in $\mathbf{R}^2$, it suffices to show that $Q^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}} = \operatorname{conv}(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})|_{x=\hat{x}}$. In this vein, define we define the upper and lower bounds in y for each set

$$[y^{\text{LP}}_l(\hat{x}), y^{\text{LP}}_u(\hat{x})] = \left[\min\{Q^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}}\}, \max\{Q^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}}\}\right],$$
$$[y^{\text{IP}}_l(\hat{x}), y^{\text{IP}}_u(\hat{x})] = \left[\min\{\operatorname{conv}(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})|_{x=\hat{x}}\}, \max\{\operatorname{conv}(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})|_{x=\hat{x}}\}\right].$$

We will show that the upper and lower bounds coincide.

Lower bounds We begin by showing that $y_l^{\text{IP}}(x) = y_l^{\text{LP}}(x)$. Since $\hat{x} \in \text{proj}_x(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$, we have two cases. If $\hat{x} \in \text{proj}_x(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$, then due to sharpness by Theorem 4.7, we have

$$\begin{aligned}\min\{Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}}\} &\geq \min\{Q^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}}\} \geq \min\{Q^{\text{LP}}|_{x=\hat{x}}\} \\ &= \min\{Q^{\text{IP}}|_{x=\hat{x}}\} = \min\{Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}}\},\end{aligned}$$

and so the result holds. In order, the four relations hold by: relaxation; relaxation; sharpness; and the fact that $P^{\text{IP}}|_{x=\hat{x}}$ consists of a single point for feasible $\hat{x}$, so that the restriction does not change the optimal solution.

Otherwise, $\hat{x} \notin \text{proj}_x(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$. Let

$$(y^*, \boldsymbol{g}^*) := \arg\min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x=\hat{x}})\}$$

and let $x^1 < \hat{x}$ and $x^2 > \hat{x}$ be the closest lower and upper bounds to $\hat{x}$ in $\text{proj}_x(Q^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I})$ (such points exist by Lemma 4.9). Let $(x^1, y^1, \boldsymbol{g}^1, \boldsymbol{\alpha}^1), (x^2, y^2, \boldsymbol{g}^2, \boldsymbol{\alpha}^2) \in P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}$ be chosen such that $\boldsymbol{\alpha}^1 \in \text{proj}_{\boldsymbol{\alpha}} P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x\in(x^1-2^{-L},x^1)}$ and $\boldsymbol{\alpha}^2 \in \text{proj}_{\boldsymbol{\alpha}} P^{\text{IP}}_{\bar{\boldsymbol{\alpha}}_I}|_{x\in(x^2,x^2+2^{-L})}$. According to Theorem 4.8, $\boldsymbol{\alpha}^1$ and $\boldsymbol{\alpha}^2$ are the Gray codes for some integers i_j and i_{j+1}. By Lemma 4.3, we know that there exists exactly one index $k \in [\![L]\!]$ such that $\alpha^1_i = \alpha^2_i$ for all $i \neq k$ and $\alpha^1_k = 1 - \alpha^2_k$.

It follows that $\boldsymbol{g}^1$ and $\boldsymbol{g}^2$ satisfy the equations

$$g_i = G_i(g_{i-1}, \alpha^1_i) = 2g_{i-1}(1 - \alpha^1_i) + 2(1 - g_{i-1})\alpha^1_i \qquad i \in [\![L]\!] \setminus k. \tag{4.33}$$

Furthermore, by Lemma 4.11, for the y-minimal solution at all three x-values, we have that all g_i, $i \in [\![L]\!] \setminus I$ take on $u_i(g_{i-1}) = \min\{2g_{i-1}, 2(1 - g_{i-1}), b_i\}$. This function is linear if $i \in I$; otherwise, it is the minimum of three functions: two linear, and one constant.

Choose $\lambda \in [0, 1]$ such that $\hat{x} = \lambda x^1 + (1 - \lambda)x^2$ and define $\hat{y}$ and $\hat{\boldsymbol{g}}$ by $(\hat{x}, \hat{y}, \hat{\boldsymbol{g}}) = \lambda(x^1, y^1, \boldsymbol{g}^1) + (1 - \lambda)(x^2, y^2, \boldsymbol{g}^2)$. By convexity of $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$, the point $(\hat{x}, \hat{y}, \hat{\boldsymbol{g}})$ is in $\text{proj}_{x,y,\boldsymbol{g}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$. We want to show

that $(\hat{x}, \hat{y}, \hat{\boldsymbol{g}}) = (\hat{x}, y^*, \boldsymbol{g}^*)$. To do so, by Lemma 4.11, we have only to show that all $\hat{g}_i = u_i(\hat{g}_{i-1})$.

By convexity, since $\boldsymbol{g}^1$ and $\boldsymbol{g}^2$ satisfy (4.33), it follows that $\hat{g}$ satisfies them as well. Hence, $\hat{g}_i = G_i(\hat{g}_{i-1}, \alpha_i^1)$ for all $i \neq k$. Furthermore, we have by induction that $g_i^* = \hat{g}_i$ for all $i \leq k-1$: (base case) $g_0^* = \hat{g}_0 = \hat{x}$; (inductive case) for $i \leq k-1$, if $g_{i-1}^* = \hat{g}_{i-1}$, then by Lemma 4.11, we have

$$\hat{g}_i \leq \max\{g_i : (x, y, \boldsymbol{g}, \boldsymbol{\alpha}) \in P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_{g_{i-1}=\hat{g}_{i-1}}\} = g_i^* = u_i(\hat{g}_{i-1}) \leq G_i(\hat{g}_{i-1}, \alpha_i^1) = \hat{g}_i,$$

so that $\hat{g}_i = g_i^* = G_i(\hat{g}_{i-1}, \alpha_i^1)$. Similarly, we have $\hat{g}_k \leq g_k^*$. We will show that $\hat{g}_k = b_k$.

To do so, we will first show that $g_k^* = b_k$. If this holds, then since $\hat{x}$ was arbitrary, and since g_k^* is continuous in $\hat{x}$ by Observation 4.12, we have that $g_k^1 = g_k^2 = b_k$. Since $\hat{g}_k$ is a convex combination of g_k^1 and g_k^2, this yields $\hat{g}_k = b_k$.

To show $g_k^* = b_k$, we first note that, by Corollary 4.13, there exists some $j \in [\![L]\!] \setminus I$ such that, for all $\hat{x} \in (x^1, x^2)$, we have $g_j^* = b_j < G(g_{j-1}^*)$, with $g_i^* < b_i$ for all $i < j$. Furthermore, since $g_i^* = G(g_{i-1}^*)$ for $i \in [\![k-1]\!] \setminus I$, we must have that $j \geq k$. To establish that $j = k$, we will show that $g_k^* = b_k$ when $\lambda = \frac{1}{2}$, implying that $j > k$ is impossible, since then we would have $g_k^* < b_k$.

Now, define $\tilde{I} = [\![L]\!] \setminus \{k\}$, and define $\tilde{\alpha}_I$ so that $\tilde{\alpha}_i = \alpha_i^1$ for all $i \in \tilde{I}$. Define $\tilde{a}_i$ and $\tilde{b}_i$ as in Lemma 4.9 for $P^{\text{LP}}_{\tilde{\alpha}_{\tilde{I}}}$. Let

$$(\tilde{y}, \tilde{\boldsymbol{g}}) := \arg\min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P^{\text{LP}}_{\tilde{\alpha}_{\tilde{I}}}|_{x=\hat{x}})\}.$$

Then by Lemma 4.11, we have $\tilde{g}_k = \min\{\tilde{b}_k, G(\tilde{g}_{k-1})\}$. However, $\tilde{g}_k = G(\tilde{g}_{k-1})$ would yield $\tilde{g} \in \text{proj}_{\boldsymbol{g}}(P^{\text{IP}}_{\tilde{\alpha}_{\tilde{I}}})$, a contradiction on the choice of $\hat{x}$. Thus, we have $\tilde{g}_k = \tilde{b}_k$. Since $\hat{x}$ was arbitrary and since $\tilde{g}_k$ is continuous in $\hat{x}$ by Observation 4.12, this implies that $g_k^1 = g_k^2 = \tilde{b}_k$. Now, this allows us to compute g_{k-1}^1 and g_{k-1}^2 given α_i^1, which will allow us to compute b_k^* for $\lambda = \frac{1}{2}$ via $g_{k-1}^* = \hat{g}_{k-1}$ and $g_k^* = u_k(g_{k-1}^*)$.

To compute $\hat{g}_{k-1}$, there are two cases. Either $\alpha^1 = 0$, yielding $g_{k-1}^1 = \frac{1}{2}\tilde{b}_k$ and $g_{k-1}^2 = 1 - \frac{1}{2}\tilde{b}_k$, or

$\alpha^1 = 1$, yielding $g^1_{k-1} = \frac{1}{2}(1 - \tilde{b}_k)$ and $g^2_{k-1} = \frac{1}{2}\tilde{b}_k$. In either case, we have $g^1_{k-1} + g^2_{k-1} = 1$, and so

$$g^*_{k-1} = \hat{g}_{k-1} = \tfrac{1}{2}(g^1_{k-1} + g^2_{k-1}) = \tfrac{1}{2},$$

yielding by Lemma 4.11 that

$$g^*_k = \min\{2g^*_{k-1}, 2(1 - g^*_{k-1}), b_k\} = \min\{1, 1, b_k\} = b_k,$$

as required. Thus, since $\hat{g}_k = b_k$ and $\hat{g}_i$ satisfies (4.33) for all $i \neq k$, it follows that $\hat{g}_i = u_i(\hat{g}_{i-1})$ for all $i \in [\![L]\!]$. Hence, by Lemma 4.11, we have $\boldsymbol{g}^* = \hat{\boldsymbol{g}}$.

Upper bounds For the upper bounds, note that $(x, y) \in Q^{\text{IP}}$ implies $y = F(x)$, where F is a convex function. Furthermore, by Lemma 4.10, we have that $y^{\text{IP}}_l(x) = y^{\text{IP}}_u(x) = y^{\text{LP}}_u(e_0) = y^{\text{LP}}_l(x)$ for $x \in \{a_0, b_0\}$ (as the extended-space solutions are unique in $P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}$ for $x \in \{a_0, b_0\}$). Thus, by the convexity of F, we have that $y^{\text{IP}}_u = y^{\text{LP}}_u$ if and only if for all $x \in [a_0, b_0]$, we have for some $\lambda \in [0, 1]$ that $(x, y^{\text{LP}}_u(x)) = \lambda(a_0, F(a_0)) + (1 - \lambda)(b_0, F(b_0))$. Alternatively, since $y_{\text{LP}_u}(x) = F(x)$ for $x \in [a_0, b_0]$, it suffices to show that $y^{\text{LP}}_u(a_0) = F(a_0)$, $y^{\text{LP}}_u(b_0) = F_{b_0}$, and that y^{LP}_u is a linear function of x.

To this end, consider any $x \in \text{proj}_x(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I})$, and consider $(y, \boldsymbol{g}) = \max_y\{(y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P^{\text{LP}}_{\bar{\boldsymbol{\alpha}}_I}|_x)\}$. Then by Lemma 4.11, we have for all $i \in [\![L]\!]$, $i \notin 1$ that $g_i = a_i$, which is a constant, while for all $i \in I$ we have that $g_i = G(g_{i-1}, \bar{\alpha}_i)$. Thus, eliminating all $g'_i s$, we find that, y^* is defined as $y^* = x - t(x)$, where $t(x)$ is some linear function of x, and thus so is $y^*(x)$, as required. □

4.4.2 A connection with existing MIP formulations

Interestingly, Gray codes also naturally appear in the "logarithmic" MIP formulations for general continuous univariate piecewise linear functions due to Vielma et al. [64], [156]. Consider applying

this existing formulation[3] to approximate the univariate quadratic term with the same $2^L + 1$ breakpoints as discussed in Section 4.3. The resulting MIP formulation uses L binary variables, which follow the same interpretation as the neural network formulation as discussed in Theorem 4.8. Moreover, it requires $\mathcal{O}(L)$ linear constraints (excluding variable bounds), and is ideal, a stronger property than the sharpness shown in Theorem 4.14. However, it comes at the price of an additional $2^L + 1$ auxiliary continuous variables, and so is unlikely to be practical without a careful handling through, e.g., column generation. Therefore, our formulation sacrifices strength to reduce this to $\mathcal{O}(L)$ auxiliary continuous variables.

4.5 Convex hull characterization

We explore a facet characterization of the convex hull of our model. Such a characterization could be used to improve and branch & cut scheme when solving MIPs with our model.

Although (4.7) offers a convex hull formulation for a single "layer" in our construction, the composition over multiple layers ($L > 1$) in (4.8) will in general fail describe the integer hull. We characterize additional valid inequalities for the integer hull of (4.8) that are derived via a connection with the parity polytope.

We begin by rewriting the relationship between the variables associated with each layer with the quadratic recurrence relation

$$g_i = (1 - 2\alpha_i)(2g_{i-1} - 1) + 1 \quad i \in [\![L]\!]. \tag{4.34}$$

For convenience, let $h_i := 2g_i - 1$ and $a_i := (1 - 2\alpha_i)$ for each $i \in [\![L]\!]$. Then after some simple

[3] In actuality, any Gray code, not just the reflected Gray code studied in this paper, yields a (potentially distinct) logarithmic formulation for a univariate function. Here, we mean the one constructed with the reflected Gray code, which is the most common choice regardless.

algebraic manipulation, (4.34) is equivalent to

$$h_i = 2a_i h_{i-1} + 1 \quad i \in [\![L]\!].$$

Expanding the recurrence relation, we have

$$h_L = 2^L \left(\prod_{i=1}^{L} a_i \right) \left(h_0 + \sum_{i=1}^{L-1} \frac{1}{2^i \prod_{j=1}^{i} a_j} \right) + 1.$$

Define $b_i := \prod_{j=1}^{i} a_j$ for each $i \in [\![L]\!]$. As each $a_j \in \{-1, +1\}$, each $b_i \in \{-1, +1\}$ as well, and so $b_i = 1/b_i$. Hence,

$$h_L = 2^L b_L \left(h_0 + \sum_{i=1}^{L-1} 2^{-i} b_i \right) + 1. \tag{4.35}$$

Multiplying both sides of (4.35) by $b_L \in \{-1, +1\}$ yields

$$h_L b_L = 2^L \left(h_0 + \sum_{i=1}^{L-1} 2^{-i} b_i \right) + b_L. \tag{4.36}$$

Combining this with the McCormick inequality $h_L + b_L - 1 \leq h_L b_L$ that is valid for the bilinear left-hand side of (4.36) (recall $h_L, b_L \in [-1, +1]$), we derive the following valid inequalities:

$$h_L \leq 2^L \left(h_0 + \sum_{i=1}^{L-1} 2^{-i} b_i \right) + 1, \tag{4.37a}$$

$$h_L \leq -2^L \left(h_0 + \sum_{i=1}^{L-1} 2^{-i} b_i \right) + 1. \tag{4.37b}$$

which can be readily mapped back to the original space of variables.

Proposition 4.15. *The following inequalities are valid for (4.8):*

$$g_L \leq 2^{L-1}\left(2g_0 - 1 + \sum_{i=1}^{L-1} 2^{-i} \prod_{j=1}^{i}(1-2\alpha_j)\right) + 1 \tag{4.38a}$$

$$g_L \leq -2^{L-1}\left(2g_0 - 1 + \sum_{i=1}^{L-1} 2^{-i} \prod_{j=1}^{i}(1-2\alpha_j)\right) + 1 \tag{4.38b}$$

If $L = 2$, then (4.37a) and (4.37b) are both linear inequalities in g and α.

Based on computational observations, for $L = 2$, these are exactly the nontrivial facet-defining linear inequalities for the integer hull of (4.8). For $L > 2$, we can produce a large class of valid inequalities by bounding the product variables b_i. In particular, bounds on these products can be derived from valid inequalities for the parity polytope.

4.5.1 Parity inequalities

The parity polytope is the convex hull of P_n^{even}, the set of all $\alpha \in \{0,1\}^n$ whose components sum to an even number. It has 2^{n-1} facets of the form

$$\sum_{i \in [\![n]\!] \setminus I} \alpha_i + \sum_{i \in I}(1-\alpha_i) \geq 1 \qquad\qquad I \subseteq [\![n]\!] \text{ s.t. } |I| \text{ is odd.} \tag{4.39}$$

Define $\beta_i \in \{0,1\}$, such that $b_i = (1-2\beta_i)$. Then

$$1 = b_j^2 = (1-2\beta_j)\prod_{i=1}^{j}(1-2\alpha_i) = (-1)^{\beta_j + \sum_{i=1}^{j}\alpha_i}. \tag{4.40}$$

Hence, for $\alpha_i \in \{0,1\}$, the sum $\beta_j + \sum_{i=1}^{j}\alpha_i$ is even and therefore $(\beta_j, \alpha_1, \ldots, \alpha_j) \in P_{j+1}^{\text{even}}$. Therefore,

we can apply (4.39) to derive valid inequalities for feasible solutions to (4.8) of the form

$$\beta_j + \sum_{i \in [\![j]\!] \setminus I} \alpha_i + \sum_{i \in I} (1 - \alpha_i) \geq 1 \qquad I \subseteq [\![j]\!] \text{ s.t. } |I| \text{ is odd}, \tag{4.41a}$$

$$(1 - \beta_j) + \sum_{i \in [\![j]\!] \setminus I} \alpha_i + \sum_{i \in I} (1 - \alpha_i) \geq 1 \qquad I \subseteq [\![j]\!] \text{ s.t. } |I| \text{ is even}. \tag{4.41b}$$

After recalling the definition $b_j = (1 - 2\beta_j)$ and rearranging, we are left with

$$b_j \leq 2 \left(\sum_{i \in [\![j]\!] \setminus I} \alpha_i + \sum_{i \in I} (1 - \alpha_i) \right) - 1 \qquad I \subseteq [\![j]\!] \text{ s.t. } |I| \text{ is odd}, \tag{4.42a}$$

$$-b_j \leq 2 \left(\sum_{i \in [\![j]\!] \setminus I} \alpha_i + \sum_{i \in I} (1 - \alpha_i) \right) - 1 \qquad I \subseteq [\![j]\!] \text{ s.t. } |I| \text{ is even}. \tag{4.42b}$$

Hence, combining these upper bounds on b_i with (4.37a) and the upper bounds on $-b_i$ with (4.37b) produces an exponential family of valid inequalities for feasible solutions to (4.8). We call these *parity inequalities*.

Example 4.16. For $L = 4$, we compute some of the nontrivial facet-defining inequalities of (4.8), written in terms of the original variables:

$$\begin{aligned}
g_4 &\leq \ 16g_0 - 14\alpha_1 + 6\alpha_2 + 2\alpha_3, \\
g_4 &\leq \ 16g_0 - 2\alpha_1 - 6\alpha_2 + 2\alpha_3, \\
g_4 &\leq -\ 16g_0 + 14\alpha_1 + 6\alpha_2 + 2\alpha_3 + 2, \\
g_4 &\leq -\ 16g_0 + 2\alpha_1 - 6\alpha_2 + 2\alpha_3 + 14.
\end{aligned}$$

Each of these inequalities is, in fact, a parity inequality, and can be constructed by a suitable combination of either (4.37a) with inequalities from (4.42a), or (4.37b) with inequalities from (4.42b).

For example, the facet $g_4 \leq 16g_0 - 14\alpha_1 + 6\alpha_2 + 2\alpha_3$ is equivalent to

$$h_4 \leq 16h_0 + 15 + 2(-14\alpha_1 + 6\alpha_2 + 2\alpha_3),$$

which can be produced as a conic combination of the inequality

$$h_4 \leq 2^4 \left(h_0 + \frac{1}{2}b_1 + \frac{1}{4}b_2 + \frac{1}{8}b_3 \right) + 1$$

from (4.37a) with the inequalities

$$\begin{aligned} b_1 &\leq 2\big((1-\alpha_1)\big) - 1, \\ b_2 &\leq 2\big((1-\alpha_1) + \alpha_2\big) - 1, \\ b_3 &\leq 2\big((1-\alpha_1) + \alpha_2 + \alpha_3)\big) - 1 \end{aligned}$$

from the family (4.42a) corresponding to $I = \{1\}$ for all $j = 1, 2, 3$, respectively.

4.5.2 Separation over exponentially many parity inequalities

Since there may be exponentially many parity inequalities, we provide an algorithm to separate over them. In particular, given a point $(g, \alpha) \subseteq [0,1]^{L+1} \times [0,1]^L$, we can determine if it lies in the intersection of the parity inequalities by computing the inequalities that give smallest upper bounds for b_j for each $j \in [\![L]\!]$. To do so, for each b_j, we need to determine the set $I_j \subseteq [\![j]\!]$ that minimizes the right-hand side of equation (4.42a) or (4.42b). This can be done, in fact, by optimizing over another parity polytope. That is, set $I_j := \{i \in [\![j]\!] : z_i^* = 1\}$, where

$$z^* \in \arg\min_{z \in \{0,1\}^j} \left\{ \sum_{i=1}^{j} z_i\alpha_i + (1-z_i)(1-\alpha_i) \;\middle|\; \sum_{i=1}^{j}(1-z_i) \text{ is odd} \right\}.$$

Linear functions can be optimized over the parity polytope in polynomial time via a linear size extended formulation [157], or by writing an integer program with a single integer variable [158], or by a simple greedy-like algorithm. An analogous approach can be taken if the sum of z_i should be odd.

4.6 Area comparisons

The approximation presented above is an over approximation of $y = x^2$. This is sufficient for providing dual bounds due how the approximation is applied using the diagonal perturbation. However, our formulation can also be altered slightly to provide an under approximation of $y = x^2$, and in particular, creates a covering of the curve with a union of polytopes. We describe two relaxations that are comparable to that of Dong and Luo [129]. We then compare these models based on the combined area of the covering to see how these methods converge.

We construct our first relaxation, named `NN-R1`, from the constraints (4.7) and

$$y \leq x - \sum_{i=1}^{L} \tfrac{g_i}{2^{2i}} \tag{4.43a}$$

$$y \geq \left(x - \sum_{i=1}^{j} \frac{g_i}{2^{2i}}\right) - \frac{1}{2^{2j+2}} \qquad j \in [\![L-1]\!] \tag{4.43b}$$

$$y \geq 2x - 1 \tag{4.43c}$$

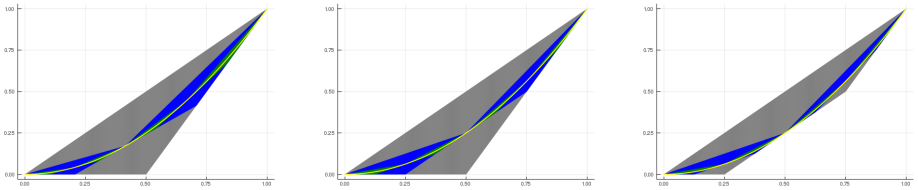
$$y \geq 0 \tag{4.43d}$$

We can form a tighter relaxation `NN-R2` by starting with `NN-R1`, then adding the cut (4.43b) with $j = L$:

$$y \geq \left(x - \sum_{i=1}^{j} \frac{g_i}{2^{2i}}\right) - \frac{1}{2^{2j+2}} \qquad j \in [\![L]\!]. \tag{4.44}$$

Tables 4.1 and 4.2 compare the volume of our relaxed method with the method of Dong and

Method	$L = 0$	$L = 1$	$L = 2$	$L = 3$	$L = 4$
`CDA`	0.25	0.0680 **(3.68)**	0.0177**(3.84)**	0.00448**(3.95)**	0.00112 **(3.994)**
`NN-R1`	0.25	0.0625**(4)**	0.0156**(4)**	0.00391**(4)**	0.000977**(4)**
`NN-R2`	0.188	0.0469**(4)**	0.0117**(4)**	0.00293**(4)**	0.000732**(4)**



`CDA` `NN-R1` `NN-R2`

Table 4.1: Area/ratio table $x \in [0, 1]$. The Gray area is for $L = 0$, the blue area is $L = 1$, the green area (very small) is $L = 2$, and the yellow curve is the curve $y = x^2$. The area for each method was approximated numerically using a Riemann sum.

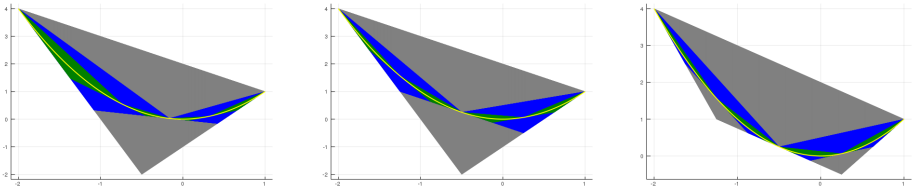
Luo [129] on the intervals $x \in [0, 1]$ and $x \in [-2, 1]$, respectively. As L increases, the volume of our relaxation consistently shrinks by a factor of 4, which is strictly greater by a fair margin to the improvement rate observed for the method of Dong and Luo. We can formalize our rate of improvement in the following proposition.

Proposition 4.17. *The volume of our approximation decreases by a factor of 4 with each subsequent layer (i.e. as L increases). Furthermore, the expected error at points x sampled uniformly at random from the input interval domain is proportional to the total volume.*

Proposition 4.17 relies on the characterization of `NN-R1` as the piecewise McCormick relaxation of $y = x^2$ at uniformly-spaced breakpoints. For one piece $[x_1, x_2]$, this relaxation consists of the tangent lines, or outer-approximation cuts, at x_1 and x_2 for the lower bound, and the secant line between x_1 and x_2 for the upper bound. We have already established that the upper bound, the `NN` approximation, is a piecewise interpolant to x^2 at the chosen breakpoints, yielding the secant line on each interval $[x_1, x_2]$ between interpolation points, and thus the upper-bound of the piecewise McCormick approximation. In Lemma 4.18 below, we show that the lower bound of the `NN-R1` and `NN-R2` approximations give the piecewise McCormick lower bounds for 2^L and 2^{L+1} uniformly-spaced breakpoints, respectively.

Lemma 4.18. *Define T as the lower-bounding set in (x, y) for the relaxation `NN-R2`, $T =$*

Method	$L=0$	$L=1$	$L=2$	$L=3$	$L=4$
`CDA`	6.75	1.94(**3.47**)	0.659(**2.95**)	0.197(**3.34**)	0.0530(**3.72**)
`NN-R1`	6.75	1.69(**4**)	0.422(**4**)	0.105(**4**)	0.0264(**4**)
`NN-R2`	5.06	1.27(**4**)	0.316(**4**)	0.0791(**4**)	0.0198(**4**)



`CDA` `NN-R1` `NN-R2`

Table 4.2: Area/ratio table $x \in [-2,1]$. Area/ratio table $x \in [0,1]$. The Gray area is for $L=0$, the blue area is $L=1$, the green area (very small) is $L=2$, and the yellow curve is the curve $y=x^2$. The area for each method was approximated numerically using a Riemann sum. Interestingly, the `CDA` model converges at different rates depending on the domain, whereas our methods always converges at the same rate.

$\text{proj}_{x,y}\{(x,y,\boldsymbol{g},\boldsymbol{\alpha}) \in [0,1]\times\mathbf{R}\times[0,1]^L\times\{0,1\}^L : (4.7),(4.43c),(4.43d) \textit{and } (4.44)\}$. *Then T can be constructed via $2^{L+1}+1$ uniformly-spaced outer-approximation cuts for $y \geq x^2$ on the interval $[0,1]$, based on the tangent lines to x^2 at $x_i = i\cdot 2^{-(L+1)}$, $i=0,\ldots,2^{(L+1)}$. That is, letting $p_{\hat{x}}(x)$ be the tangent line to $y=x^2$ at the point $\hat{x}$,*

$$p_{\hat{x}}(x) = 2\hat{x}(x-\hat{x}) + \hat{x}^2 = \hat{x}(2x-\hat{x}), \tag{4.45}$$

T is equivalent to $\left\{(x,y)\in[0,1]\times\mathbf{R} : y \geq p_{\hat{x}}(x) \quad \forall i\in\{0,\ldots,L\},\ \hat{x}=i\cdot 2^{-(L+1)}\right\}$.

Proof. To begin, we note that, as shown by Yarotsky [110], we have that for each $l=0,\ldots,L$, $F_l(x)$ gives a piecewise-linear interpolant in $y=x^2$ at 2^l+1 uniformly-spaced points on $x\in[0,1]$. That is, for each $i\in\{0,1,\ldots,2^l\}$, we have that $F_l(i\cdot 2^{-l}) = (i\cdot 2^{-l})^2$.

For $x_1 < x_2$, Consider a single linear interpolant to $y=x^2$ on the interval $[x_1,x_2]$, given as

$$\begin{aligned} h(x) &= \frac{x_2^2-x_1^2}{x_2-x_1}(x-x_1)+x_1^2 \\ &= (x_1+x_2)(x-x_1)+x_1^2. \end{aligned} \tag{4.46}$$

Since $h(x)-x^2$ is concave, the deviation $h(x)-x^2$ is maximized when $\frac{d}{dx}(h(x)-x^2) = x_1+x_2-2x = 0$,

yielding $x^* = \frac{x_1+x_2}{2}$. At this point, we have

$$\begin{aligned} h(x^*) - (x^*)^2 &= (x_1 + x_2)\left(\frac{x_1+x_2}{2} - x_1\right) + x_1^2 - (\frac{x_1+x_2}{2})^2 \\ &= \tfrac{1}{2}(x_1+x_2)^2 - \tfrac{1}{4}(x_1+x_2)^2 - (x_1+x_2)x_1 + x_1^2 \\ &= \tfrac{1}{4}(x_1+x_2)^2 - x_1x_2 \\ &= \tfrac{1}{4}x_1^2 + \tfrac{1}{2}x_1x_2 - x_1x_2 + \tfrac{1}{4}x_2^2 \\ &= \tfrac{1}{4}(x_2-x_1)^2. \end{aligned} \tag{4.47}$$

Thus, we have that $(x^*)^2 = h(x^*) - \frac{1}{4}(x_2-x_1)^2$. Since x^2 is a convex function, and since $h - \frac{1}{4}(x_2-x_1)^2$ is tangent to x^2 at x^* (as both slopes are $x_2 - x_1$), this implies that for all $x \in [0,1]$, we have $(x^*)^2 \geq h(x) - \frac{1}{4}(x_2-x_1)^2$.

Now, since $F_l(x)$ is a linear interpolant to $y = x^2$ on each interval $[i \cdot 2^{-l}, (i+1)\cdot 2^{-l}]$, with interval width 2^{-l}, we have that the cuts $y \geq F_l(x) - \frac{1}{4}2^{-2l} = F_l(x) - 2^{-2l-2}$ are valid for $y = x^2$, and furthermore are tangent to x^2 at the point $x = (i+\frac{1}{2})\cdot 2^{-l}$, the midpoints of all interpolants. Thus, we obtain outer-approximation cuts at $x = (i+\frac{1}{2})\cdot 2^{-l}$ for each $i = 0, \ldots, 2^l - 1$.

We obtain T by applying these cuts for $l = 0, \ldots, L$, combined with the outer-approximation cuts $y \geq 0$ (tangent at $x = 0$) and $y \geq 2x - 1$ (tangent at $x = 1$). We now show that this yields $2^{L+1} + 1$ uniformly-spaced outer-approximation cuts to $y = x^2$. This can be easily seen by expressing the outer-approximation points in binary. The points $x = (i+\frac{1}{2})\cdot 2^{-l}$ for each $i = 0, \ldots, 2^l - 1$ can be characterized as exactly the numbers in $(0,1)$ such that the $(l+1)$th binary decimal is a 1, and all later binary decimals are zero. Alternatively, it is the set of points

$$P_l := \left\{ x \in [0,1] : \exists\, \mathbf{a} \in \{0,1\}^l \text{ s.t. } x = 2^{-(l+1)} + \sum_{i=1}^{l} 2^{-i} a_i \right\}.$$

We then have that the set of outer-approximation points, $x \in \{0,1\} \cup \bigcup_{l \in [\![0,L]\!]} P_l$, is the set of all binary decimals in the interval $[0,1]$ for which the last 1 occurs no later than the $L+1$st decimal place, which has cardinality $2^{L+1} + 1$. This forms the set of uniformly-spaced points $i \cdot 2^{-(L+1)}$,

$i = 0, \ldots, 2^{L+1}$. Thus, we have that the set T consists of a set of $2^{L+1} + 1$ uniformly-spaced outer-approximation cuts to the function $y = x^2$, as required. □

With Lemma 4.18, we establish that the `NN-R1` relaxation is equivalent to the piecewise McCormick relaxation of $y = x^2$ on the uniformly-spaced breakpoints $x_i = 2^{-L}i$, $i \in [\![0, 2^L]\!]$. We now establish the area of any given piece of the relaxation.

Lemma 4.19. *The area of the McCormick Relaxation of $y = x^2$ on the interval $[x_1, x_2]$ is $\frac{1}{4}(x_2 - x_1)^3$.*

Proof. We wish to obtain a closed-form solution for the area of the resulting triangle region. To do so, we compute the vertices of this triangle, construct vectors between them, then compute the cross-product of these vectors.

The equations of the tangent lines are given by

$$y = 2x_1(x - x_1) + x_1^2 = 2x_1(x - \tfrac{x_1}{2}),$$
$$y = 2x_2(x - x_2) + x_2^2 = 2x_2(x - \tfrac{x_2}{2}).$$

We thus find that the intersection of these lines is given by the point $(\frac{x_1+x_2}{2}, x_1x_2)$. Thus, the three vertices are given as $v_1 = (x_1, x_1^2)$, $v_2 = (\frac{x_1+x_2}{2}, x_1x_2)$, and $v_3 = (x_2, x_2^2)$. From these vertices, we obtain two vectors

$$v_2 - v_1 = [\tfrac{1}{2}(x_2 - x_1), x_1(x_2 - x_1)] = (x_2 - x_1)[\tfrac{1}{2}, x_1],$$
$$v_3 - v_2 = [\tfrac{1}{2}(x_2 - x_1), x_2(x_2 - x_1)] = (x_2 - x_1)[\tfrac{1}{2}, x_2].$$

The area is given by the magnitude of the cross product as

$$A = \frac{1}{2}|(v_2 - v_1) \times (v_3 - v_2)| = \frac{1}{2}(x_2 - x_1)^2|\tfrac{1}{2}x_2 - \tfrac{1}{2}x_1| = \frac{1}{4}(x_2 - x_1)^3.$$

As required. □

With Lemma 4.19, we are now ready to show that the area of `NN-R1` is optimal among all piecewise

McCormick relaxations with a fixed number of pieces.

Proposition 4.20. *The minimum possible area covering $y = x^2$ on $x \in [a, b]$ with a sequence of n McCormick Relaxations is $\frac{1}{4}(b-a)^3/n^2$, and is achieved via uniformly spaced breakpoints.*

Proof. Consider a general piecewise McCormick relaxation of $y = x^2$ on the interval $[a, b]$ with consecutive breakpoints $a = x_0 \leq x_1 \leq \cdots \leq x_n = b$. For any segment of consecutive breakpoints x_i and x_{i+1}, the McCormick relaxation between those points is bounded by the secant line between (x_i, x_i^2) and (x_{i+1}, x_{i+1}^2), and the tangent lines to $y = x^2$ at x_i and x_{i+1}. For simplicity, let $i = 1$ for this discussion.

Thus, letting $y_i = x_i - x_{i-1},\ i = 1 \dots n$, the problem of choosing the area-optimal breakpoints $a \leq x_1, \leq x_2 \leq \cdots \leq x_{n-1} \leq x_n$ reduces to solving

$$\min_{y \in \mathbf{R}^n_+} \left\{ \frac{1}{4} \sum_{i=1}^{n} y_i^3 : \sum_{i=1}^{n} y_i = b - a \right\}. \tag{4.48}$$

It is then easy to show via the KKT conditions that, due to the convexity of $\sum_i y_i^3$ on positive support, all y_i must be equal, yielding the uniformly-spaced solution $y_i = \frac{b-a}{n}$. Thus, the choice of breakpoints induced by our algorithm is optimal. □ □

The `NN-R1` relaxation is exactly a union of McCormick Relaxations 2^n uniformly spaced breakpoints. Hence the total area is $\frac{1}{4}(b-a)^3 2^{-2n}$. We now show that adding the inequality (4.43b) with $j = L$ cuts off an extra fourth of the total area.

Proposition 4.21. *The area of the relaxation in (4.43) is $\frac{3}{16}(b-a)^3 2^{-2n}$.*

Proof. We will compute the area removed by the addition of this cut on a general interval $[x_1, x_2]$, where $x_1 = 2^{-L} i$ for some $i \in [\![0, 2^L - 1]\!]$. As shown in Lemma 4.18, the cut (4.43b) with $j = L$ intersects the curve at $x = \frac{x_1 + x_2}{2}$. Now, the area removed by the addition of this cut is the area of the triangle formed by the intersections of the tangent lines at x_1, x_2, and the midpoint $x_3 := \frac{x_1 + x_2}{2}$.

These vertices, given as intersection points v_{ij} for tangent lines for x_i and x_j, are derived following the process in Lemma 4.19 as:

$$\begin{aligned} v_{12} &= \left(\tfrac{x_1+x_2}{2}, x_1x_2\right) \\ v_{13} &= \left(\tfrac{1}{2}\cdot\left(\tfrac{x_1+x_2}{2}+x_1\right), \tfrac{1}{2}x_1(x_1+x_2)\right) = \left(\tfrac{1}{4}(3x_1+x_2), \tfrac{1}{2}x_1(x_1+x_2)\right) \\ v_{23} &= \left(\tfrac{1}{2}\cdot\left(\tfrac{x_1+x_2}{2}+x_2\right), \tfrac{1}{2}x_2(x_1+x_2)\right) = \left(\tfrac{1}{4}(x_1+3x_2), \tfrac{1}{2}x_2(x_1+x_2)\right) \end{aligned}$$

From these points, we obtain vectors

$$\begin{aligned} v_{12}-v_{13} &= \left(\tfrac{1}{4}(x_2-x_1), \tfrac{1}{2}x_1(x_2-x_1)\right) = (x_2-x_1)\left(\tfrac{1}{4}, \tfrac{1}{2}x_1\right) \\ v_{23}-v_{12} &= \left(\tfrac{1}{4}(x_2-x_1), \tfrac{1}{2}x_2(x_2-x_1)\right) = (x_2-x_1)\left(\tfrac{1}{4}, \tfrac{1}{2}x_2\right). \end{aligned}$$

Finally, we obtain cut area

$$A_{cut} = \frac{1}{2}|(v_{12}-v_{13})\times(v_{23}-v_{12})| = \frac{1}{2}(x_2-x_1)^2\cdot\frac{1}{8}(x_2-x_1) = \frac{1}{16}(x_2-x_1)^3.$$

The result easily follows. □

4.7 A computational study

We study the efficacy of our MIP relaxation approach on a family of nonconvex quadratic optimization problems. We compare 9 methods:

1. `GRB`: The native method in Gurobi v9.1.1 for nonconvex quadratic problems.
2. `GRB-S`: The native method in Gurobi v9.1.1, applied to the diagonalized shift reformulation of (4.2).
3. `BRN`: Baron v21.1.13, using CPLEX v12.10 for the MIP/LP solver.
4. `BRN-S`: Baron v21.1.13, using CPLEX v12.10 for the MIP/LP solver, applied to the diagonal-

ized shift reformulation of (4.2).

5. `CPLEX`: The native method in CPLEX v12.10 for nonconvex quadratic objectives.

6. `CDA`: The algorithm of Dong and Luo [129]. The number of layers L will correspond to the parameter ν appearing in their paper.

7. `NN`: The new formulation (4.8).

8. `NMDT`: The "normalized multi-parametric disaggregation technique" (NMDT) of Castro [1]. See Appendix 4.A for a restatement in terms of the number of levels L.

9. `T-NMDT`: A tightened variant of NMDT also described in Appendix 4.A.

We can cluster these methods into two families: five "native" methods (`GRB`, `GRB-S`, `BRN`, `BRN-S`, and `CPLEX`) that pass an exact representation of the quadratic problem to the solver, and four "relaxations" (`CDA`, `NN`, `NMDT`, and `T-NMDT`) which relax the quadratic problem using a MIP reformulation, which is then passed to an underlying solver. For each of these relaxations, we use Gurobi v9.1.1 as the underlying MIP solver. Note that `GRB`, `BRN`, and `CPLEX` are directly given (4.50), which is an optimization problem with linear constraints and a nonconvex quadratic objective. In contrast, `GRB-S` and `BRN-S` are given the diagonalized reformulation of (4.50) per (4.2), which is an optimization problem with a convex quadratic objective and a series of nonconvex quadratic constraints.[4]

Our objective in this computational study is to measure the quality of the dual bound provided by the different methods. To place the methods on an even footing on the primal side, as initialization we run the nonconvex quadratic optimization method in Gurobi v9.1.1 with "feasible solution emphasis" to produce a good starting feasible solution. We then inject this primal objective bound as a "cut-off" for each method.

We will consider 4 metrics, which will be applied with respect to a given family of instances:

[4]CPLEX does not support nonconvex quadratic constraints of this form, so we do not include a corresponding approach with the diagonal shift.

- time: The shifted geometric mean of the solve time in seconds (shift is minimum solve time in the family).

- gap: The shifted geometric mean of the final relative optimality gap $\frac{|\texttt{db}-\texttt{bpb}|}{|\texttt{bpb}|}$, where `db` is the dual bound provided by the method and `bpb` is the best observed primal solution for the instance across all methods. Shift is taken as $\max\{10^{-4}, \text{minimum gap observed in the family}\}$.

- BB: The number of instances in which the method either produced the best dual bound, or attained Gurobi's default optimality criteria of $\texttt{gap} < 10^{-4}$. Note that on a given instance, more than one method can potentially attain the best bound.

- TO: The number of instances in which the solver *times out* and terminates due to the time limit.

We note that even if the solver terminates within the time limit (with an "optimal" solver status), the optimality gap for `NN` or `CDA` as reported in Table 4.3 may be nonzero, due to the fact that these two methods serve as relaxations for the original boxQP problem.

We implement each model in the JuMP algebraic modeling language [159]. To compute the shift used by the four relaxations, `GRB-S`, and `BRN-S`, we use Mosek v9.2 to solve a semidefinite programming problem to produce the "tightest" diagonal matrix $D = \text{diag}(\delta)$ such that $Q + D$ is positive semidefinite as in Dong and Luo [129]:

$$\min_{\delta \in \mathbb{R}^n} \mathbf{e} \cdot \delta \quad \text{s.t.} \quad Q + \text{diag}(\delta) \succeq 0. \tag{4.49}$$

In Section 4.7.4 we study an alternative, simpler method for computing this shift and its computational implications. Note that this time to solve the SDP is not included in the solve time numbers, but is relatively small (on the order of a few seconds for the largest instances) and is computation that is shared by most of the approaches.

Each method is provided a time limit of 10 minutes. Computational experiments are performed

on a machine with a 3.8 GHz CPU with 24 cores and 128 GB of RAM. Each solver is restricted to one thread, and all experiments for a given instance are run concurrently. Our code and the corresponding problem instances are publicly available at https://github.com/joehuchette/quadratic-relaxation-experiments.

4.7.1 Baseline comparison

We start by comparing our nine methods on 99 box constrained quadratic objective ($boxQP$) optimization problem instances as studied in Chen and Burer [160] and Dong and Luo [129]:

$$\min_{0 \le x \le 1} \quad x'Qx + c \cdot x. \tag{4.50}$$

Despite its simple constraint structure, this is a nonconvex optimization problem when Q is not positive semidefinite, and is difficult from both a theoretical and a practical perspective.

For this baseline study, we fix each of the relaxations to use $L = 3$ layers; we will revisit this selection in Section 4.7.3. We leave as future work an implementation that iteratively refines the approximation to guarantee a pre-specified approximation error, as done by Dong and Luo [129].

We split these instances into three families: 63 "solved" instances on which each method terminates at optimality within the time limit, 18 "unsolved" instances on which each method terminates due to the time limit, and 18 "contested" instances on which some methods terminate and some do not. We present the computational results in Table 4.3, stratified by family. At a high level, we observe that `NN` attains the "best bound" on 87 of 99 instances. We now survey each family in more detail. Alternatively, we stratify the results based on the size of the instances in Appendix 4.B.

Solved instances On the solved instances, all methods are able to terminate quickly–all in under two seconds, on average. The native methods are able to meet the termination criteria on nearly all instances, while the relaxation methods lag behind. Our new `NN` method performs the

family	method	time (sec)	gap	BB	TO
solved	BRN	0.51	0.00%	$^{63}/_{63}$	-
	CPLEX	0.68	0.00%	$^{63}/_{63}$	-
	GRB	0.37	0.00%	$^{63}/_{63}$	-
	BRN-S	0.96	0.00%	$^{63}/_{63}$	-
	GRB-S	0.63	0.00%	$^{62}/_{63}$	-
	CDA	1.24	0.08%	$^{5}/_{63}$	-
	NN	0.39	0.01%	$^{26}/_{63}$	-
	NMDT	0.67	0.02%	$^{19}/_{63}$	-
	T-NMDT	1.07	0.01%	$^{24}/_{63}$	-
contested	BRN	66.1	0.00%	$^{12}/_{18}$	$^{6}/_{18}$
	CPLEX	46.0	0.00%	$^{17}/_{18}$	$^{1}/_{18}$
	GRB	34.4	0.00%	$^{15}/_{18}$	$^{3}/_{18}$
	BRN-S	154.6	0.02%	$^{10}/_{18}$	$^{8}/_{18}$
	GRB-S	450.1	1.22%	$^{2}/_{18}$	$^{16}/_{18}$
	CDA	429.0	1.49%	$^{0}/_{18}$	$^{15}/_{18}$
	NN	273.0	0.24%	$^{2}/_{18}$	$^{10}/_{18}$
	NMDT	318.0	0.54%	$^{1}/_{18}$	$^{11}/_{18}$
	T-NMDT	446.5	0.83%	$^{1}/_{18}$	$^{14}/_{18}$
unsolved	BRN	-	11.48%	$^{0}/_{18}$	-
	CPLEX	-	15.67%	$^{3}/_{18}$	-
	GRB	-	30.73%	$^{0}/_{18}$	-
	BRN-S	-	11.86%	$^{0}/_{18}$	-
	GRB-S	-	5.21%	$^{0}/_{18}$	-
	CDA	-	5.33%	$^{0}/_{18}$	-
	NN	-	4.31%	$^{15}/_{18}$	-
	NMDT	-	4.59%	$^{0}/_{18}$	-
	T-NMDT	-	5.04%	$^{0}/_{18}$	-

Table 4.3: Baseline computational results on 99 boxQP instances.

family	method	time (sec)	gap	BB	TO
solved	Baron	0.51	0.00%	63/63	-
	CPLEX	0.68	0.00%	63/63	-
	Gurobi	0.37	0.00%	63/63	-
	Gurobi+Shift	0.63	0.00%	62/63	-
	Dong-Luo[2]	1.24	0.08%	5/63	-
	Proposed	0.39	0.01%	26/63	-
	NMDT[5]	0.67	0.02%	19/63	-
contested	Baron	66.1	0.00%	12/18	6/18
	CPLEX	46.0	0.00%	17/18	1/18
	Gurobi	34.4	0.00%	15/18	3/18
	Gurobi+Shift	450.1	1.22%	2/18	16/18
	Dong-Luo	429.0	1.49%	0/18	15/18
	Proposed	273.0	0.24%	2/18	10/18
	NMDT	318.0	0.54%	1/18	11/18
unsolved	Baron	-	11.48%	0/18	-
	CPLEX	-	15.67%	3/18	-
	Gurobi	-	30.73%	0/18	-
	Gurobi+Shift	-	5.21%	0/18	-
	Dong-Luo	-	5.33%	0/18	-
	Proposed	-	4.31%	15/18	-
	NMDT	-	4.59%	0/18	-

Table 4.4: Baseline computational results on 99 boxQP instances.

best, attaining the termination gap criteria on roughly half of the instances, while `CDA` performs the worst, attaining it on only 5 of 63 instances. We stress that, for these experiments, L is set relatively low. In Section 4.7.3 we revisit this decision, and observe that this gap can be closed on these easy instances by increasing L at a nominal computational cost.

Contested instances On the contested instances the native solvers `BRN`, `CPLEX`, and `GRB` perform best, producing the best bound in a clear majority of the 18 instances. Interestingly, the shifted variants `BRN-S` and `GRB-S` perform worse, with Gurobi exhibiting a substantial degradation in performance as opposed to without the diagonal shift. In contrast, the relaxations time out on a majority of the instances. Taken together, we conclude that there is a transitional family of instances wherein the native solvers still excel, but which the more complex relaxations succumb to the curse of dimensionality.

Unsolved instances This family of instances tests the scenario where a method is given a fixed time budget and is asked to produce the best possible dual bound. On these 18 instances, `NN` is the clear winner, producing the best bounds on 15 and the lowest mean gap. The other relaxations come relatively close in terms of termination gap, but do not attain the best bound on any instance. The native `GRB` performs the worst of all methods in terms of gap closed, but applying the shift as in `GRB-S` helps tremendously, producing gaps than are much lower than the other native solver methods and close to what the relaxations are able to provide.

4.7.2 Varying the solver focus

Modern solvers such as Gurobi expose high-level parameters for configuring the search algorithm for different goals. In this subsection, we configure Gurobi to focus on the best objective bound (`MIPFocus=3`). We summarize our results in Table 4.5. The story on the "solved" and "contested" instances remains roughly the same as in Section 4.7.1. However, on the "unsolved" instances all

methods perform better, with the largest improvement coming from the native Gurobi methods. Nonetheless, the `NN` method is still attaining the best bound on 10 of 18 instances, with the `GRB-S` method coming close in terms of gap closed, and is able to produce the best bound on the remaining 8 instances.

family	method	time (sec)	gap	BB	TO
solved	`GRB`	0.70	0.00%	$^{63}/_{63}$	-
	`GRB-S`	0.64	0.00%	$^{63}/_{63}$	-
	`CDA`	1.71	0.08%	$^{5}/_{63}$	-
	`NN`	0.48	0.01%	$^{18}/_{63}$	-
	`NMDT`	1.45	0.04%	$^{17}/_{63}$	-
	`T-NMDT`	1.13	0.01%	$^{24}/_{63}$	-
contested	`GRB`	49.3	0.00%	$^{14}/_{18}$	$^{4}/_{18}$
	`GRB-S`	458.9	0.26%	$^{3}/_{18}$	$^{15}/_{18}$
	`CDA`	470.2	1.31%	$^{0}/_{18}$	$^{16}/_{18}$
	`NN`	301.9	0.18%	$^{4}/_{18}$	$^{10}/_{18}$
	`NMDT`	457.8	0.83%	$^{0}/_{18}$	$^{15}/_{18}$
	`T-NMDT`	436.6	0.34%	$^{1}/_{18}$	$^{14}/_{18}$
unsolved	`GRB`	-	6.13%	$^{0}/_{18}$	-
	`GRB-S`	-	3.58%	$^{8}/_{18}$	-
	`CDA`	-	4.71%	$^{0}/_{18}$	-
	`NN`	-	3.34%	$^{10}/_{18}$	-
	`NMDT`	-	4.50%	$^{0}/_{18}$	-
	`T-NMDT`	-	4.05%	$^{0}/_{18}$	-

Table 4.5: Computational results with Gurobi configured with `MIPFocus=3`.

4.7.3 Varying the relaxation resolution

In the previous experiments, we fixed the number of layers for each relaxation at $L = 3$. In this subsection, we study how varying this parameter affects each relaxation, in terms of both solve time and gap closed. In particular, we consider setting $L \in \{2, 4, 6, 8\}$ for each relaxation method, and experiment with the same set of 99 boxQP instances as before. We summarize the results in Table 4.6.

On the "solved" instances we observe that, unsurprisingly, increasing L allows us to reach the best bound criteria on far more instances. Moreover, we observe that this results in only a nominal increase in computational cost; all methods terminate with a mean solve time of seconds, even with the finest discretization. We observe that `NN` performs the best, in terms of mean solve time and "best bound" for each value of L considered. Morever, `NN` can attain the termination criteria on each instance with $L = 8$, which is not the case for any other method. These results indicate that, on easy instances, increasing the resolution is cheap, and can attain the same dual bound quality as the native solvers in roughly the same time. In contrast, on the "unsolved" instances we observe that increasing L results in *higher* gaps across the board. This is unsurprising–increasing L results in larger formulations, and on instances where the solver is already struggling this will quickly lead to performance degradation due to the "combinatorial explosion" effect. Moreover, even small values for L offer a nontrivial refinement in a branch-and-bound setting over a tight convex relaxation. This result suggests that, for instances known to be hard, a reasonable strategy would be to set L to a small value by default and then target finer discretizations on individual quadratic terms as-needed, through a dynamic refinement approach or otherwise.

4.7.4 Varying the diagonal perturbation

We now turn our attention to how the diagonal shift that is used by the relaxations, `BRN-S`, and `GRB-S` is computed. As discussed above, we may solve the SDP (4.49) to compute a valid shift that is "tightest" under some reasonable objective measure. While this approach is reasonable for the boxQP instances studied here, it may not be practical for larger-scale instances due to the scalability of the SDP solver. Therefore, we compare this shift against a simpler "eigenvalue" shift $D = -\lambda_{\min} I$, where $I \in \mathbb{R}^{n \times n}$ is the identity matrix and $\lambda_{\min}$ is the smallest eigenvalue of Q. This minimum eigenvalue can be readily computed, and the resulting shift is conceptually similar to the convexification process used in αBB [161], for example.

We perform a similar experiment as in Section 4.7.1, focusing on comparing our two shifts head-to-

family	method	L	time (sec)	gap	BB	TO
solved	CDA	2	0.39	0.79%	$^{0}/_{63}$	-
		4	1.64	0.01%	$^{24}/_{63}$	-
		6	2.84	0.00%	$^{52}/_{63}$	-
		8	3.74	0.00%	$^{61}/_{63}$	-
	NN	2	0.30	0.04%	$^{8}/_{63}$	-
		4	0.41	0.00%	$^{41}/_{63}$	-
		6	0.48	0.00%	$^{58}/_{63}$	-
		8	0.55	0.00%	$^{63}/_{63}$	-
	NMDT	2	0.41	0.12%	$^{8}/_{63}$	-
		4	0.84	0.01%	$^{31}/_{63}$	-
		6	1.17	0.00%	$^{41}/_{63}$	-
		8	1.41	0.00%	$^{46}/_{63}$	-
	T-NMDT	2	0.58	0.05%	$^{8}/_{63}$	-
		4	1.24	0.00%	$^{37}/_{63}$	-
		6	1.43	0.00%	$^{53}/_{63}$	-
		8	1.59	0.00%	$^{62}/_{63}$	-
contested	CDA	2	136.18	1.10%	$^{2}/_{13}$	$^{0}/_{13}$
		4	552.44	0.68%	$^{0}/_{13}$	$^{10}/_{13}$
		6	595.58	1.33%	$^{0}/_{13}$	$^{12}/_{13}$
		8	600.00	2.06%	$^{0}/_{13}$	$^{13}/_{13}$
	NN	2	206.03	0.16%	$^{1}/_{13}$	$^{3}/_{13}$
		4	268.56	0.04%	$^{4}/_{13}$	$^{3}/_{13}$
		6	293.47	0.07%	$^{4}/_{13}$	$^{5}/_{13}$
		8	319.46	0.08%	$^{8}/_{13}$	$^{5}/_{13}$
	NMDT	2	208.66	0.42%	$^{0}/_{13}$	$^{3}/_{13}$
		4	376.76	0.16%	$^{2}/_{13}$	$^{5}/_{13}$
		6	447.43	0.18%	$^{4}/_{13}$	$^{7}/_{13}$
		8	473.95	0.18%	$^{4}/_{13}$	$^{7}/_{13}$
	T-NMDT	2	344.76	0.33%	$^{0}/_{13}$	$^{6}/_{13}$
		4	511.43	0.36%	$^{1}/_{13}$	$^{9}/_{13}$
		6	505.40	0.24%	$^{3}/_{13}$	$^{8}/_{13}$
		8	534.02	0.26%	$^{4}/_{13}$	$^{8}/_{13}$
unsolved	CDA	2	-	3.98%	$^{0}/_{23}$	-
		4	-	4.72%	$^{0}/_{23}$	-
		6	-	5.02%	$^{0}/_{23}$	-
		8	-	5.29%	$^{0}/_{23}$	-
	NN	2	-	3.37%	$^{23}/_{23}$	-
		4	-	3.53%	$^{0}/_{23}$	-
		6	-	3.63%	$^{0}/_{23}$	-
		8	-	3.72%	$^{0}/_{23}$	-
	NMDT	2	-	3.55%	$^{0}/_{23}$	-
		4	-	3.92%	$^{0}/_{23}$	-
		6	-	4.05%	$^{0}/_{23}$	-
		8	-	4.16%	$^{0}/_{23}$	-
	T-NMDT	2	-	3.84%	$^{0}/_{23}$	-
		4	-	4.36%	$^{0}/_{23}$	-
		6	-	4.31%	$^{0}/_{23}$	-
		8	-	4.30%	$^{0}/_{23}$	-

Table 4.6: Computational results with varying discretization levels.

head for each method that utilizes it. We summarize our results in Table 4.7. We observe that the tighter shift provided by the SDP (4.49) offers a substantial improvement over the eigenvalue shift across the board. On the "solved" instances, we observe an order of magnitude reduction in solve time for all methods, as well as a significant increase in best bound attainment for the relaxation methods. On the "contested" instances, we observe a similar degradation when using the shift. The difference is perhaps most striking for `GRB`: on the 22 instances, produces the best bound on 16 of 22 with the SDP shift, but with the eigenvalue shift only attains it on only one, and times out on remaining 21. Finally, for the "unsolved" instances we observe that the SDP shift provides 2-3x smaller gaps than the eigenvalue shift for each method.

4.7.5 A (simple) problem with quadratic constraints

In this section, we present a unique class of instances on which our model displays suprisingly strong performance compared to Gurobi. In this model, we minimize a 1-norm with respect to box constraints and an additional quadratic constraint stating that the 2-norm is greater than some bound. The specific model considered is

$$\begin{aligned} \min \quad & \tfrac{100}{n} \textstyle\sum_{i=1}^{n} |x_i - \varepsilon_i| & \\ \text{s.t.} \quad & x_i \in [-1, 1] & i \in [\![n]\!] \\ & \textstyle\sum_{i=1}^{n} x_i^2 \geq n - 0.5 & \end{aligned} \tag{4.51}$$

where $\varepsilon_i = \text{rand}(-1, 1) \cdot 10^{-3}$, sorted in ascending order of $|\varepsilon_i|$. We note that this problem can be solved in closed form.

We compare against `GRB`, `GRB-S`, and `T-NMDT` for various values of n, with $L = 10$. The results are shown in Table 4.8 below.

The results indicate a strong performance advantage of `NN` above the competing methods shown. Note that the number of nodes for `GRB` and `GRB-S` are consistently close to 2^{n+1}, with computational times to match, while `T-NMDT` is even worse. On the other hand, while `NN` shows only moderate

family	method	shift	time	gap	BB	TO
solved	GRB	eigen	1.81	0.00%	51/51	-
		sdp	0.19	0.00%	51/51	-
	CDA	eigen	3.57	0.14%	17/51	-
		sdp	0.47	0.07%	34/51	-
	NN	eigen	1.22	0.01%	14/51	-
		sdp	0.14	0.00%	16/51	-
	NMDT	eigen	1.73	0.04%	12/51	-
		sdp	0.26	0.01%	19/51	-
	T-NMDT	eigen	3.85	0.02%	19/51	-
		sdp	0.40	0.00%	24/51	-
contested	GRB	eigen	582.97	3.52%	1/22	21/22
		sdp	130.76	0.05%	16/22	6/22
	CDA	eigen	600.00	5.02%	0/22	22/22
		sdp	126.65	0.23%	0/22	5/22
	NN	eigen	562.82	2.03%	0/22	19/22
		sdp	43.24	0.02%	6/22	0/22
	NMDT	eigen	577.73	2.84%	0/22	20/22
		sdp	62.64	0.11%	1/22	2/22
	T-NMDT	eigen	594.63	3.72%	0/22	22/22
		sdp	116.27	0.05%	1/22	4/22
unsolved	GRB	eigen	-	8.64%	0/26	-
		sdp	-	4.17%	0/26	-
	CDA	eigen	-	9.22%	0/26	-
		sdp	-	4.23%	0/26	-
	NN	eigen	-	8.31%	0/26	-
		sdp	-	3.12%	26/26	-
	NMDT	eigen	-	8.47%	0/26	-
		sdp	-	3.42%	0/26	-
	T-NMDT	eigen	-	9.01%	0/26	-
		sdp	-	3.95%	0/26	-

Table 4.7: Computational results with two different diagonal shifts.

n	method	time (sec)	gap	nodes
10	GRB	0.30	0.00%	2047
	GRB-S	0.08	0.00%	2047
	NN	0.10	0.00%	89
	T-NMDT	16.89	0.00%	45306
15	GRB	1.66	0.00%	66828
	GRB-S	1.22	0.00%	66828
	NN	0.72	0.00%	3477
	T-NMDT	318.95	0.00%	1494473
18	GRB	8.88	0.00%	528270
	GRB-S	8.08	0.00%	528210
	NN	1.20	0.00%	7757
	T-NMDT	(TO)	0.73%	3451026
20	GRB	37.15	0.00%	2099824
	GRB-S	35.49	0.00%	2099563
	NN	1.17	0.00%	9746
	T-NMDT	(TO)	1.04%	3976996
22	GRB	202.63	0.00%	8389900
	GRB-S	309.72	0.00%	8389887
	NN	1.66	0.00%	11226
	T-NMDT	(TO)	0.91%	2937766

Table 4.8: Computational results for a stylized quadratically constrained problem with $L = 10$.

increases in computational time, with a max of about $1.66s$, with a far smaller node count to match.

Upon deeper investigation, we found that the primary computational advantage of the NN method stems from Gurobi's Gomory cuts. In fact, turning off presolve, heuristics, and all cuts except for Gomory cuts, the performance significantly improves over the baseline performance. For $n = 22$ the problem solves in 0.09s with only 191 nodes and 39 Gomory cuts. For $n = 250$, over an order of magnitude higher, the problem solves in 10.65s with only 13016 nodes and 378 Gomory cuts.

We expect that Gurobi is performing a spatial branching algorithm. However, the problem was constructed so that the feasible solutions are near corners of a hypercube, while at spatial branching relaxations, optimal solutions to the relaxations are close to the center. Moreover, this property is likely to hold in a spatial branch-and-bound algorithm, meaning that you will likely need to branch on all variables in order to identify the correct corner. This behavior would yield at least $O(2^n)$ spatial branching nodes, and give poor bounds throughout the branching process, as observed in the computational results. On the other hand, the NN formulation provides an alternative branching structure that, when combined with Gomory cuts, provides excellent computational performance for this collection of instances.

4.7.6 More difficult problems with nonconvex quadratic constraints

To conclude our computational section, we study a "best nearest" variant of the boxQP problem that is in the spirit of the problem from Section 4.7.5. In more detail, for some fixed $\hat{\gamma} \in \mathbb{R}$ and $\hat{x} \in [0,1]^n$, we solve the problem

$$\begin{aligned} \min_{x} \quad & \|x - \hat{x}\|_1 \\ \text{s.t.} \quad & x'Qx + c \cdot x \leq 0.95\hat{\gamma} \\ & 0 \leq x \leq 1. \end{aligned}$$

Since each boxQP instance considered has negative objective cost, this will constrain the feasible region to those points which are "close" to optimal for the original boxQP instance. We construct 54 instances based on the the `basic` family of boxQP instances from Chen and Burer [160]. We set $\hat{x}$ as the vector of all 0.5s, and set $\hat{\gamma}$ to be the best primal cost on the underlying boxQP instance that is found by Gurobi after 10 minutes. We summarize the results in Table 4.9.

family	method	time (sec)	gap	BB	TO
solved	BRN	12.64	0.00%	8/8	-
	GRB	5.27	0.00%	8/8	-
	BRN-S	13.06	0.00%	8/8	-
	GRB-S	32.31	0.00%	8/8	-
	CDA	6.13	1.70%	0/8	-
	NN	4.50	0.06%	6/8	-
	NMDT	11.99	1.03%	0/8	-
	T-NMDT	22.66	0.07%	6/8	-
contested	BRN	176.89	0.00%	30/40	12/40
	GRB	66.53	0.04%	28/40	12/40
	BRN-S	173.21	0.01%	29/40	12/40
	GRB-S	566.15	10.07%	1/40	39/40
	CDA	412.17	7.12%	0/40	34/40
	NN	383.73	2.92%	5/40	33/40
	NMDT	481.24	6.10%	0/40	35/40
	T-NMDT	522.14	4.16%	4/40	36/40
unsolved	BRN	-	69.60%	0/6	-
	GRB	-	65.66%	0/6	-
	BRN-S	-	48.23%	0/6	-
	GRB-S	-	24.42%	0/6	-
	CDA	-	12.29%	0/6	-
	NN	-	8.98%	6/6	-
	NMDT	-	10.10%	0/6	-
	T-NMDT	-	10.31%	0/6	-

Table 4.9: Baseline computational results on 54 "best nearest" boxQP instances.

On the "solved" instances, we observe that our `NN` relaxation has the lowest mean solve time of all methods, and is able to prove optimality on 6 of 8 methods. We note that the optimality gaps for `CDA` and `NMDT` are nearly two orders of magnitude greater than what was observed on

the baseline boxQP instances in Table 4.3. This is in keeping with the common knowledge in the global optimization (e.g. Dey and Gupte [152]) that tight relaxations for quadratic functions in the constraints do not necessarily lead to tight relaxations in the objective.

Similar to the baseline boxQP instances, we observe that the "contested" instances are a transient class where the native methods are able to terminate within the time limit with greater frequency than the relaxations, leading to significantly smaller mean optimality gaps. On the hardest "unsolved" instances, we again see that our `NN` method produces the smallest optimality gap across all methods on each of the 6 instances, outperforming all other methods.

4.8 Conclusion

We present a simple MIP model for relaxing quadratic optimization problems that competes with robust commercial solvers in terms of solve time and bound quality. There are a number of ways that our method could be further improved. For example, we could follow the strategy of Dong and Luo [129] and implement an adaptive strategy that dynamically refines individual quadratic terms as-needed. Additionally, for boxQP instances we can potentially improve performance by leveraging the results of Hansen et al. [147], or applying existing cutting plane procedures [162]. Further, we could apply bound tightening on variables [146] or include a tail-end call to a nonlinear solver to produce an optimal primal feasible solution.

We also have performed preliminary analysis on a variant of this method to model higher-order monomials as opposed to quadratics. Fundamental results of Wei [163] show that our sawtooth functions form a basis for *any* continuous functions. Unfortunately, we have observed that a comparable approximation for x^3 seem to require a relatively large number of basis functions. We summarize our preliminary results in Appendix 4.C. We believe it would be interesting future work to observe if this seeming obstruction is fundamental, or if compact methods for higher-order monomials can be derived through our approach.

Appendix

4.A Normalized multi-parametric disaggregation technique

We present a standard approach to descretizing continuous variables for handling bilinear products in nonlinear models. This approach is perhaps the most straightforward way to convert bilinear problems to MILPs and has been referred to as *Normalized Multi-Parametric Disaggregation Technique* (NMDT) [1]. We adapt the bilinear approach here to a squaring a single variable.

Consider $x \in [0, 1]$, and let L be a positive integer. We then use the representation

$$x = \sum_{i=1}^{p} 2^{-i}\beta_i + \Delta x \tag{4.52a}$$

$$\beta_i \in \{0, 1\} \qquad i \in [\![L]\!] \tag{4.52b}$$

$$\Delta x \in [0, 2^{-L}], \tag{4.52c}$$

where L is the number of binary variables to use.

Multiplying (4.52a) by x, and substituting the representation into the $x\Delta x$ term, we obtain

$$\begin{aligned} y = x \cdot x \qquad &= \sum_{i=1}^{L} 2^{-i} x\beta_i + x\Delta x \qquad &&= \sum_{i=1}^{L} 2^{-i} x\beta_i + \left(\sum_{i=1}^{L} 2^{-i}\beta_i + \Delta x\right)\Delta x \\ &= \sum_{i=1}^{L} 2^{-i}(x + \Delta x)\beta_i + \Delta x^2 \end{aligned}$$

Now, using the fact that $x + \Delta x \in [0, 1 + 2^{-L}]$, first lift the model by adding variables u_i and Δu such that $u_i = (x + \Delta x)\beta_i$ and $\Delta u = \Delta x^2$, and then we relax these equations using McCormick Envelopes.

Given bounds $x \in [\underline{x}^{\min}, \underline{x}^{\max}]$ and $\beta \in [0, 1]$, The McCormick envelope $\mathcal{M}(x, \beta)$ is defined as the following relaxation of $u = x\beta$

$$\mathcal{M}(x, \beta) = \left\{(x, \beta, y) \in [\underline{x}^{\min}, \underline{x}^{\max}] \times [0, 1] \times \mathbf{R} \ : (4.55)\right\}. \tag{4.54}$$

$$\begin{aligned} \underline{x}^{\min} \cdot \beta \leq u \leq \underline{x}^{\max} \cdot \beta \\ x - \underline{x}^{\max} \cdot (1 - \beta) \leq u \leq x - \underline{x}^{\min} \cdot (1 - \beta) \end{aligned} \tag{4.55}$$

To approximate $u = x^2$ with $x \in [0, \underline{x}^{\max}]$, this becomes

$$\mathcal{M}(x) = \left\{ (x, u) \in [0, \underline{x}^{\max}] \times \mathbf{R} \ : \begin{array}{c} u \geq 0 \\ \underline{x}^{\max}(2x - \underline{x}^{\max}) \leq u \leq \underline{x}^{\max} \cdot x \end{array} \right\}. \tag{4.56}$$

We present two ways to use this approach. The first is the most direct use of `NMDT`, as used in [1]. This model is

$$x = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta x \tag{4.57a}$$

$$y = \sum_{i=1}^{L} 2^{-i}u_i + \Delta u \tag{4.57b}$$

$$(x, \beta_i, u_i) \in \mathcal{M}(x, \beta_i) \qquad i \in [\![L]\!] \tag{4.57c}$$

$$(\Delta x, x, \Delta u) \in \mathcal{M}(\Delta x, x) \tag{4.57d}$$

$$\beta_i \in \{0, 1\} \qquad i \in [\![L]\!] \tag{4.57e}$$

$$\Delta x \in [0, 2^{-L}] \tag{4.57f}$$

Here, the only error introduced in the relaxation is from $\Delta u = x\Delta x$, yielding a maximum error of

2^{-L-2}, again occurring when $\Delta x = 2^{-L-1}$.

Alternatively, we consider the expansion of the $x\Delta x$ term. We thus obtain the `T-NMDT` relaxation for $y = x^2$.

$$x = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta x \tag{4.58a}$$

$$y = \sum_{i=1}^{L} 2^{-i}u_i + \Delta u \tag{4.58b}$$

$$(x + \Delta x, \beta_i, u_i) \in \mathcal{M}(x + \Delta x, \beta_i) \qquad i \in [\![L]\!] \tag{4.58c}$$

$$(\Delta x, \Delta u) \in \mathcal{M}(\Delta x) \tag{4.58d}$$

$$\beta_i \in \{0, 1\} \qquad i \in [\![L]\!] \tag{4.58e}$$

$$\Delta x \in [0, 2^{-L}] \tag{4.58f}$$

Since β_i is binary, $u_i = \beta_i(x + \Delta x)$ is represented exactly. Thus, the only possible error is introduced in the relaxation of $\Delta y = \Delta x^2$, which yields a maximum error of 2^{-2L-2}, occurring when $\Delta x = 2^{-L-1}$.

Now, the expected error of `T-NMDT` is the expected error from the relaxation of $\Delta y = \Delta x^2$. Modeling Δx as a uniform random variable within its bounds $[0, 2^{-L}]$, and noting that the only overestimator from (4.56) is $y \leq 2^{-L}\Delta x$ we obtain expected overapproximation error

$$\begin{aligned}
\mathbb{E}(2^{-L}\Delta x - \Delta x^2) &= \textstyle\int_0^{2^{-L}} 2^L(2^{-L}\Delta x - \Delta x^2)\mathrm{d}\Delta x \\
&= 2^L \textstyle\int_0^{2^{-L}} (2^{-L}\Delta x - \Delta x^2)\mathrm{d}\Delta x \\
&= 2^L(\tfrac{1}{6}(2^{-L})^3) \\
&= \tfrac{1}{6}2^{-2L}.
\end{aligned} \tag{4.59}$$

Similarly, the expected underapproximation error can be computed as $\frac{1}{12}2^{-2L}$.

4.B Additional baseline computation summaries

In Table 4.B.1 we summarize the results of our baseline experiments stratified by the number of decision variables as in, e.g., Table 4 of Dey et al. [164].

4.C General representations with sawtooth bases

The premise of our formulation is that the function $y = x^2$ can be arbitrarily closely approximated by a series of sawtooth functions. We discuss here if such approximations could conveniently apply to other polynomials.

In [163], the authors present a Fourier series-like method that leverages orthogonal triangular functions to derive a convergent class of L_2-optimal approximations for general functions on the interval $[-\pi, \pi]$. Define the periodic triangular functions

$$X(x) = \begin{cases} \frac{\pi^2+2\pi x}{8} & -\pi < x + 2\pi k \leq 0, k \in \mathbb{Z} \\ \frac{\pi^2-2\pi x}{8} & 0 < x + 2\pi k \leq \pi, k \in \mathbb{Z} \end{cases}$$
$$Y(x) = \begin{cases} \frac{\pi x}{4} & -\frac{\pi}{2} < x + 2\pi k \leq \frac{\pi}{2}, k \in \mathbb{Z} \\ \frac{\pi^2-\pi x}{4} & \frac{\pi}{2} < x + 2\pi k \leq \frac{3\pi}{2}, k \in \mathbb{Z} \end{cases} \tag{4.60}$$

The authors then build their orthogonal basis functions using an orthogonal linear transformation of the basis

$$1, X(x), Y(x), X(2x), Y(2x), \ldots, X(nx), Y(nx).$$

However, as with Fourier series approximations, this method has the limitation that all approximating functions are equal at the endpoints of the interval, resulting in a poor approximation for functions at which the endpoints are not equal. Thus, to obtain good approximations for x^3 on $[-\pi, \pi]$, we first add the linear function $-\pi^2 x$ to enforce equality at the endpoints.

family	method	time (sec)	gap	BB	TO
$n \in [20, 30]$	BRN	0.19	0.00%	$^{18}/_{18}$	$^{0}/_{33}$
	CPLEX	0.20	0.00%	$^{18}/_{18}$	$^{0}/_{18}$
	GRB	0.14	0.00%	$^{18}/_{18}$	$^{0}/_{18}$
	BRN-S	0.34	0.00%	$^{18}/_{18}$	$^{0}/_{18}$
	GRB-S	0.05	0.00%	$^{18}/_{18}$	$^{0}/_{18}$
	CDA	0.16	0.06%	$^{2}/_{18}$	$^{0}/_{18}$
	NN	0.05	0.00%	$^{9}/_{18}$	$^{0}/_{18}$
	NMDT	0.09	0.01%	$^{7}/_{18}$	$^{0}/_{18}$
	T-NMDT	0.11	0.00%	$^{8}/_{18}$	$^{0}/_{18}$
$n \in [40, 50]$	BRN	0.46	0.00%	$^{33}/_{33}$	$^{0}/_{33}$
	CPLEX	0.70	0.00%	$^{33}/_{33}$	$^{0}/_{33}$
	GRB	0.37	0.00%	$^{33}/_{33}$	$^{0}/_{33}$
	BRN-S	0.87	0.00%	$^{33}/_{33}$	$^{0}/_{33}$
	GRB-S	0.54	0.00%	$^{33}/_{33}$	$^{0}/_{33}$
	CDA	1.00	0.07%	$^{3}/_{33}$	$^{0}/_{33}$
	NN	0.36	0.00%	$^{16}/_{33}$	$^{0}/_{33}$
	NMDT	0.61	0.03%	$^{11}/_{33}$	$^{0}/_{33}$
	T-NMDT	0.98	0.01%	$^{15}/_{33}$	$^{0}/_{33}$
$n \in [60, 80]$	BRN	13.66	0.00%	$^{15}/_{21}$	$^{6}/_{21}$
	CPLEX	15.10	0.00%	$^{19}/_{21}$	$^{3}/_{21}$
	GRB	9.99	0.00%	$^{16}/_{21}$	$^{5}/_{21}$
	BRN-S	25.29	0.00%	$^{15}/_{21}$	$^{6}/_{21}$
	GRB-S	112.85	0.00%	$^{12}/_{21}$	$^{8}/_{21}$
	CDA	112.31	0.30%	$^{0}/_{21}$	$^{7}/_{21}$
	NN	37.26	0.04%	$^{4}/_{21}$	$^{3}/_{21}$
	NMDT	53.73	0.12%	$^{2}/_{21}$	$^{4}/_{21}$
	T-NMDT	110.42	0.07%	$^{2}/_{21}$	$^{6}/_{21}$
$n \in [90, 125]$	BRN	261.48	0.24%	$^{9}/_{27}$	$^{18}/_{27}$
	CPLEX	218.22	0.13%	$^{13}/_{27}$	$^{16}/_{27}$
	GRB	170.56	0.20%	$^{11}/_{27}$	$^{16}/_{27}$
	BRN-S	375.45	0.55%	$^{7}/_{27}$	$^{20}/_{27}$
	GRB-S	578.95	3.43%	$^{1}/_{27}$	$^{26}/_{27}$
	CDA	569.53	3.84%	$^{0}/_{27}$	$^{26}/_{27}$
	NN	533.97	2.35%	$^{14}/_{27}$	$^{25}/_{27}$
	NMDT	543.72	2.84%	$^{0}/_{27}$	$^{25}/_{27}$
	T-NMDT	563.94	3.39%	$^{0}/_{27}$	$^{26}/_{27}$

Table 4.B.1: Computational results with instances stratified based on number of variables.

Then, applying this method to x^2 and $x^3 - \pi^2 x$ on the interval $x \in [-\pi, \pi]$, we obtain the following numbers for the (L_1-error). Note that almost all of the $Y(nx)$ functions are relevant for approximating $x^3 - \pi^2 x$ (and no $X(nx)$'s), while only a few $X(nx)$ functions (and no $Y(nx)'s$) are relevant for approximating x^2.

Function	$N=2$	$N=4$		$N=8$		$N=16$		$N=32$	
x^2	0.994	0.249	**(4)**	0.0622	**(4)**	0.0155	**(4)**	0.0039	**(3.97)**
$x^3 - \pi^2 x$	7.23	2.07	**(3.5)**	0.626	**(3.3)**	0.304	**(2.06)**	0.108	**(2.81)**

Table 4.C.1: Comparison of L_1-error. Factor of improvement over the previous value for L is shown in bold.

To investigate the outlook of sparsely approximating x^3 with triangular functions directly, we solved the following MIP to obtain the L_1-optimal triangular approximation to x^3 on the interval $[0, 1]$ using re-scaled versions of the basis functions above, and explicitly including a linear shift. We discretely approximate the $L - 1$ error via the error at uniformly-spaced points $x_1, \ldots, x_{N_p} \in [0, 1]$, allowing the inclusion of only N_f triangular functions.

$$
\begin{aligned}
\min \quad & \tfrac{1}{Np} \sum_{j=1}^{N_p} t_j & & \\
s.t. \quad & t_j \geq \textstyle\sum_{i=I} (\lambda_i f_i(x_j) + \lambda_0 x_j + f_c) - x_j^3 & & \forall j \\
& t_j \geq -(\textstyle\sum_{i \in I} (\lambda_i f_i(x_j) + \lambda_0 x_j + f_c) - x_j^3) & & \forall j \\
& -M \cdot \alpha_i \leq \lambda_i \leq M \cdot \alpha_i & & \forall i \geq 1 \\
& \sum_{i=1}^{N} \alpha_i \leq N_f & & \\
& \lambda_i \in [-M, M] & & \forall i \\
& \alpha_i \in \{0, 1\} & & \forall i
\end{aligned}
\tag{4.61}
$$

The result, shown in Figure 4.C.1, suggests that it is not possible to use this triangular basis to obtain a similar-quality sparse approximation for x^3 as for x^2: the best achievable error rate for x^3 is roughly $O(N_f^{-2})$, compared to $O(2^{-2N_f})$ for the quadratic. See also Table 4.C.1 where we compare the convergence of the two approximations.

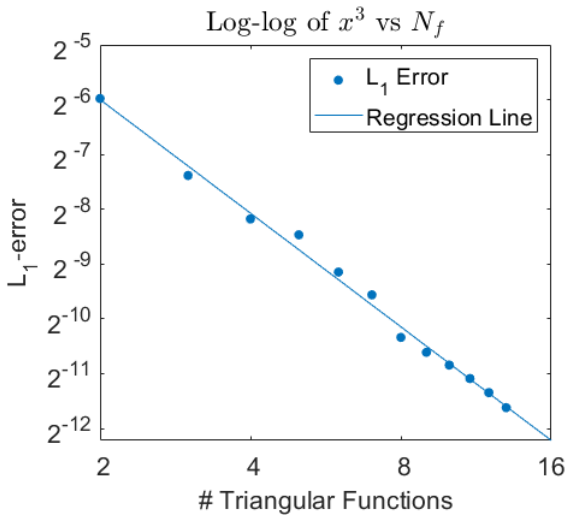


Figure 4.C.1: The L_1-error of x^3 vs. the number of approximating triangular functions N_f. The equation of the regression line suggests an asymtotic error rate of roughly $O(N_f^{-2})$, compared to $O(2^{-2N_f})$ for the quadratic.

Chapter 5

A Comparison of Discretization Approaches for Mixed Integer Nonconvex Quadratic Programming

5.1 Introduction

Abstract *In this work, we study piecewise linear relaxations for solving mixed-integer quadratically constrained quadratic programs (MIQCQP's). We introduce new univariate relaxation methods based on separable reformulations of nonconvex variable products combined with a logarithmic binarization. Further, we extend the well-known approach normalized multiparametric disaggregation technique (NMDT) by a double discretization (D-NMDT). We show in a broad computational study that these new methods are suitable to determine good dual bounds for MIQCQP's.*

We wish to solve general mixed-integer quadratically constrained quadratic programs (MIQCQP's),

defined as

$$\begin{aligned} \min \quad & x'Q_0x + c_0'x + d_0'y \\ \text{s.t.} \quad & x'Q_jx + c_j'x + d_j'y + b_j \leq 0 \quad j \in 1, \dots, m \\ & x_i \in [\underline{x}_i, \overline{x}_i] \qquad\qquad\quad i \in 1, \dots, n, \\ & y \in \{0,1\}^k \end{aligned} \tag{5.1}$$

MIQCQP's are broadly applicable to a wide variety of problems, including power supply [165], gas network [166], [167], water management [168], and pooling or blending problems [16], [22], [55], [169] and Chapter 2. See [111], [112] and references therein for more examples.

For solving such problems there are a number of different approaches, which mainly depend on the characteristics of the quadratic matrices involved. Within this work we focus on non-convex MIQCQP's with known finite upper and lower bounds for the variables as defined in (5.1).

In particular, we consider solution approaches for non-convex MIQCQP's which are based on piecewise linear (pwl.) approximations or relaxations. Such approaches can be divided into three classes of techniques with respect to how they linearize the non-convex quadratic terms.

The first class of techniques are *McCormick based approaches* [1], [141]–[145]. For each variable product, the idea is to partition the domain into smaller rectangles over which the graph is relaxed by its convex hull, the so-called McCormick relaxation, which is known to be linear [57]. Due to the binarization of the partition, the resulting problem is a pwl. relaxation of the original problem including binary variables. McCormick based methods can differ in the way the partition and the binarization is performed. The partition can be performed purely on one variable or on both variables, equidistantly or non-equidistantly. The binarization, can be done linearly or logarithmically in the number of partition elements [122], [170]. In a broader sense, (axial-)spatial branching for bilinear terms can also be seen as a piecewise McCormick linearization approach. Here, the partition is not performed a-priori, instead it is added implicitly via branching. An overview of spatial-branching techniques can be found in [171].

Another idea for linearizing variable products is to use *quadratic convex reformulations* [115], [129],

[131], [134] and Chapter 4. Here, all non-convexity of the problem is shifted to univariate terms via reformulations. In Chapter 4, we apply a *diagonal perturbation* to convexify the quadratic matrices. The resulting univariate quadratic correction terms are then linearized by introducing new variables and constraints of the form $y_i = x_i^2$, which are then approximated by pwl. functions. The binarization of the univariate pwl. functions is done logarithmically by using the so-called *sawtooth* function [110]. An advantage of this approach is that only linearly many expressions of the form $y_i = x_i^2$ have to be linearized instead of quadratically many terms of the form $z_{ij} = x_i x_j$, with the respect to the dimension of the original quadratic matrix. The cost of this approach is the use of convex MIQCQP relaxations instead of the MILP relaxations obtained via direct modelling of bilinear terms.

A further approach is *separable reformulation* [172], in which each bivariate term $x_i x_j$ is reformulated as a sum of separable univariate terms, for example $x_i x_j = 1/2(x_i^2 + x_j^2 - (x_i - x_j)^2)$ [173]. The univariate terms are then relaxed, here equations of the form $z_i = x_i^2$, $z_j = x_j^2$ and $z_{ij} = (x_i - x_j)^2$. Again, this approach can be combined with a logarithmic encoding of the univariate linear segments, as in Chapter 4 and [129]. In [172] the authors list the following possible reformulations:

- Bin1: $x_i x_j = \left(\frac{1}{2}(x_i + x_j)\right)^2 - \left(\frac{1}{2}(x_i - x_j)\right)^2$,
- Bin2: $x_i x_j = 1/2\left((x_i + x_j)^2 - x_i^2 - x_j^2\right)$,
- Bin3: $xy = 1/2\left(x_i^2 + x_j^2 - (x_i - x_j)^2\right)$,
- Log: $\ln z_{ij} = \ln x_i + \ln x_j$.

They prove that each of the reformulations requires fewer linear segments than a bivariate triangulation-based approximation to achieve the same level of precision. However, when modeled as MIP, this comes at the cost of weaker LP relaxations.

Finally, one can also approximate or relax $x_i x_j$ directly via a bivariate pwl. function [122], [167], [172], [174]. This technique starts with a triangulation of the domain, which defines a pwl. approximation

of the variable products. This pwl. approximation can then be easily converted into a relaxation by shifting it to become a pwl. underestimator and overestimator. Bivariate pwl. approximations can also be modeled using (logarithmically many) binary variables, see e.g. [122], [167], [170].

In this work, we focus on direct relaxations of the individual $x_i x_j$ terms. In Section 5.2, we introduce some useful concepts used throughout the work. In Section 5.3, we present core formulations used repeatedly in our quadratic relaxations, namely McCormick envelopes [57] and a strengthened form of the sawtooth-based relaxation for the univariate quadratic from Chapter 4. In Section 5.4, we introduce two new relaxations for terms of the form $x_i x_j$. These relaxations are extensions of existing formulations, modified to require significantly fewer binary variables to reach the same level of precision for problems with dense Q matrices. In Section 5.5, we prove various properties about the relative strengths of both the original formulations and their extensions. The topics we explore include MIP- and LP-relaxation areas, sharpness, and optimality of the choice of breakpoints in the underlying piecewise relaxations. Finally, in Section 5.6, we computationally compare the various methods on two families of problem instances: boxQP's and MIQCQP's, respectively.

5.2 Preliminaries

We start by introducing some convenient notation. We then introduce the notions of formulation strength explored in this work.

5.2.1 Notation

Throughout the paper, we use the following convenient notation: for any integers $i \leq j$, we define $[\![i, j]\!] := \{i, i+1, \ldots, j\}$, and for integers $i \geq 1$ we define $[\![i]\!] := [\![1, i]\!]$.

Furthermore, as we will introduce a number of models for the graphs of the functions $f(x) = x^2$ and $f(x, y) = xy$, we introduce the following notation. For a function $f \colon X \to \mathbb{R}$, let $\text{gra}_X(f)$,

$\text{epi}_X(f)$, and $\text{hyp}_X(f)$ denote the *graph*, *epigraph*, and *hypograph* of the function f over the set X, respectively. That is,

$$\begin{aligned}\text{gra}_X(f) &:= \{(x, y) \in X \times \mathbb{R} : y = f(x)\}\\ \text{epi}_X(f) &:= \{(x, y) \in X \times \mathbb{R} : y \geq f(x)\}\\ \text{hyp}_X(f) &:= \{(x, y) \in X \times \mathbb{R} : y \leq f(x)\}.\end{aligned}$$

5.2.2 Strength of MIP Formulations

We will work with the three following notions of formulation strength for MIP formulations. The first two notions are defined as in [155] and Section 4.4, and relate to the tightness of the LP relaxation of the MIP formulation.

Definition 5.1. Consider a set $U \subseteq \mathbb{R}^n$. For a formulation $P^{\text{IP}} = \{(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{z}) \in P^{\text{LP}} : \boldsymbol{z} \in \{0, 1\}^L\}$, where $P^{\text{LP}} \subseteq \mathbb{R}^{n+d} \times [0, 1]^L$ is a polyhedron and $\text{proj}_{\boldsymbol{u}}(P^{\text{IP}}) = U$, we say the the formulation P^{IP} is

- *sharp* if $\text{proj}_{\boldsymbol{u}}(P^{\text{LP}}) = \text{conv}(U)$.
- *hereditarily sharp* if, for all $I \subseteq [\![L]\!]$ and $\hat{\boldsymbol{z}} \in \{0, 1\}^{|I|}$, we have

$$\text{proj}_{\boldsymbol{u}}(P^{\text{LP}}|_{\boldsymbol{z}_I = \hat{\boldsymbol{z}}}) = \text{conv}(\text{proj}_{\boldsymbol{u}}(P^{\text{IP}}|_{\boldsymbol{z}_I = \hat{\boldsymbol{z}}}).$$

For a third notion of formulation strength, we further consider the *volume* of $U = \text{proj}_{\boldsymbol{u}}(P^{\text{IP}})$. If U is a relaxation of some zero-volume set, here the graph of $f(x, y) = xy$ or $f(x) = x^2$, the volume gives some idea of the 'average-case' approximation error.

In this way, given a MIP relaxation P^{IP} for the graph of a function $f(\boldsymbol{x})$, $x \in \mathbb{R}^{n-1}$, the *sharpness* of P^{IP} measures its formulation strength with respect to its LP-relaxation, which relates to its performance within a branch and bound tree. On the other hand, the *volume* measures its formulation strength with respect to the relaxation of the graph of f.

5.3 Core Formulations

Throughout this document, we repeatedly make use of the following formulations.

5.3.1 McCormick Envelopes

The convex hull of $z = xy$ is given by a set of linear equations known as the McCormick envelope [57]:

$$\mathcal{M}(x, y) = \left\{(x, y, z) \in [\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}] \times \mathbb{R} \ : (5.3)\right\}. \tag{5.2}$$

$$\begin{aligned} z &\geq \underline{x} \cdot y + x \cdot \underline{y} - \underline{x} \cdot \underline{y} \quad z \geq \overline{x} \cdot y + x \cdot \overline{y} - \overline{x} \cdot \overline{y} \\ z &\leq \overline{x} \cdot y + x \cdot \underline{y} - \overline{x} \cdot \underline{y} \quad z \leq \underline{x} \cdot y + x \cdot \overline{y} - \underline{x} \cdot \overline{y}. \end{aligned} \tag{5.3}$$

In case one of the variables is binary, here β, the McCormick envelope of $z = \beta x$ simplifies to

$$\mathcal{M}(x, \beta) = \{(x, \beta, z) \in [\underline{x}, \overline{x}] \times [0, 1] \times \mathbb{R} \ : (5.5)\}. \tag{5.4}$$

$$\begin{aligned} \underline{x} \cdot \beta \leq z \leq \overline{x} \cdot \beta \\ x - \overline{x} \cdot (1 - \beta) \leq z \leq x - \underline{x} \cdot (1 - \beta). \end{aligned} \tag{5.5}$$

For univariate continuous quadratic terms $y = x^2$, it simplifies to

$$\mathcal{M}(x) = \{(x, y) \in [\underline{x}, \overline{x}] \times \mathbb{R} \ : (5.7)\}. \tag{5.6}$$

$$\begin{aligned} y &\geq 2\underline{x} \cdot x - \underline{x}^2 \\ y &\geq 2\overline{x} \cdot x - \overline{x}^2 \\ y &\leq x(\overline{x} + \underline{x}) - \overline{x} \cdot \underline{x}. \end{aligned} \tag{5.7}$$

5.3.2 Sawtooth-Based MIP Formulations

We derive a MIP formulation for approximating terms of the form $y = x^2$ that requires only logarithmically many binary variables in the number of linear segments. We make use of an elegant pwl. formulation for $\text{gra}_{[0,1]}(x^2)$ by [110] using the recursively defined *sawtooth* function described in [175] to formulate the approximation of $\text{gra}_{[0,1]}(x^2)$, as described in Section 4.4. To this end, we define

$$S^L := \left\{(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1] \times [0,1]^{L+1} \times \{0,1\}^L : (5.9)\right\} \tag{5.8}$$

$$\begin{aligned} g_0 &= x \\ 2(g_{i-1} - \alpha_i) &\leq g_i \leq 2g_{i-1} && i \in 1, \dots, L \\ 2(\alpha_i - g_{i-1}) &\leq g_i \leq 2(1 - g_{i-1}) && i \in 1, \dots, L. \end{aligned} \tag{5.9}$$

Note that, by construction in Section 4.2, S^L is defined so that the relationship between each g_i is the 'tooth' function, $g_i = \min(2g_{i-1}, 2(1 - g_{i-1})) =: G(g_{i-1})$. In this way, g_i is a 'sawtooth' function w.r.t. x, as described in [110], [175],

$$g_i = G^i(x) := \underbrace{G \circ G \circ \cdots \circ G}_{i}(x),$$

illustrated in Figure 5.3.1. Given this above relationship with x and $\boldsymbol{g}$, the constraint $y = x^2$ can be approximated by

$$F^L(x, \boldsymbol{g}) := x - \sum_{i=1}^{L} 2^{-2i} g_i, \text{ with } F^0(x, \boldsymbol{g}) = x. \tag{5.10}$$

In this way, (5.10) is an approximation for x^2, $F^L(x) := x - \sum_{i=1}^{L} 2^{-2i} G^i(x)$. We use these definitions to give a MIP approximation for $y = x^2$.

Definition 5.2 (Sawtooth MIP Approximation)**.** Given a layer depth $L \in \mathbb{N}$, the L-layer sawtooth

approximation for $y = x^2$ on the interval $x \in [0, 1]$ is given by

$$\left\{(x, y) \in [0, 1]^2 : \exists(\boldsymbol{g}, \boldsymbol{\alpha}) \in [0, 1]^{L+1} \times \{0, 1\}^L : y = F^L(x, \boldsymbol{g}),\ (x, \boldsymbol{g}, \boldsymbol{\alpha}) \in S^L\right\}. \tag{5.11}$$

Alternatively, this approximation can be written as $\text{gra}_{S^L}(F^L(x, \boldsymbol{g}))$.

Next, we define the sawtooth relaxation for $y = x^2$ (`NN-R2` from Section 4.6) by shifting each approximating function F^j, $j = 0, \ldots, L$, down by its maximum error 2^{-2j-2} (established in [110, Proposition 2]), then adding additional outer-approximation cuts to x^2 at $x = 0$ and $x = 1$.

Definition 5.3 (Sawtooth Relaxation)**.** Given a layer depth $L \in \mathbb{N}$, the L-layer sawtooth relaxation for $y = x^2$ on the interval $x \in [0, 1]$ is given by

$$\left\{(x, y) \in [0, 1] \times \mathbb{R} : \exists(\boldsymbol{g}, \boldsymbol{\alpha}) \in [0, 1]^{L+1} \times \{0, 1\}^L : (5.13)\right\}, \tag{5.12}$$

$$\begin{aligned} y &\leq F^L(x, \boldsymbol{g}) \\ y &\geq F^j(x, \boldsymbol{g}) - 2^{-2j-2} \qquad j \in 0, \ldots, L \\ y &\geq 0, \quad y \geq 2x - 1 \\ (x, \boldsymbol{g}, \boldsymbol{\alpha}) &\in S^L. \end{aligned} \tag{5.13}$$

Alternatively, this relaxation can be written as

$$\text{hyp}_{S^L}(F^L(x, \boldsymbol{g})) \cap \text{epi}_{S^L}\left(\max\left\{0, 2x - 1, \max_{j \in [\![0, L]\!]}(F^j(x, \boldsymbol{g}) - 2^{-2j-2})\right\}\right).$$

Remark 5.4 (Transformation to general bounds)**.** On general intervals $x \in [\underline{x}, \overline{x}]$, the sawtooth over-approximation (5.11) can be defined by mapping x to $[0, 1]$ via $\hat{x} = \frac{x - \underline{x}}{\overline{x} - \underline{x}} \in [0, 1]$, applying the sawtooth formulation to model $\hat{y} = \hat{x}^2$, and computing y from x and $\hat{y}$ appropriately. Defining $l_x = \overline{x} - \underline{x}$, this yields

$$\hat{y} \ = \hat{x}^2 = \left(\tfrac{x - \underline{x}}{l_x}\right)^2 = \tfrac{x^2 - 2x\underline{x} + \underline{x}^2}{{l_x}^2} = \tfrac{y - 2x\underline{x} + \underline{x}^2}{{l_x}^2} = \tfrac{y - \underline{x}(2x - \underline{x})}{{l_x}^2}.$$

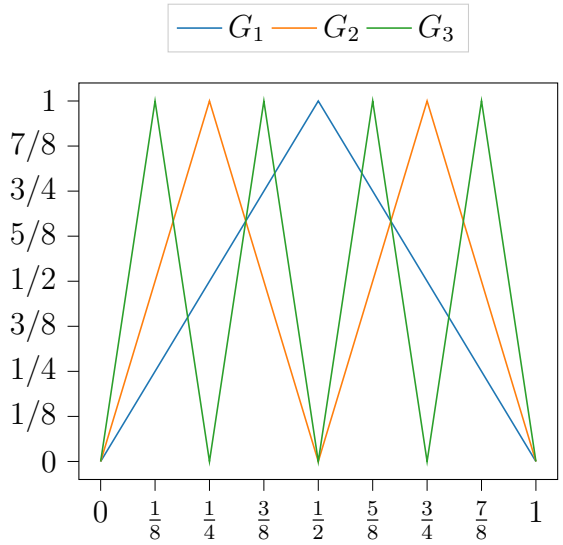


Figure 5.3.1: The *sawtooth* function $G^i(x)$.

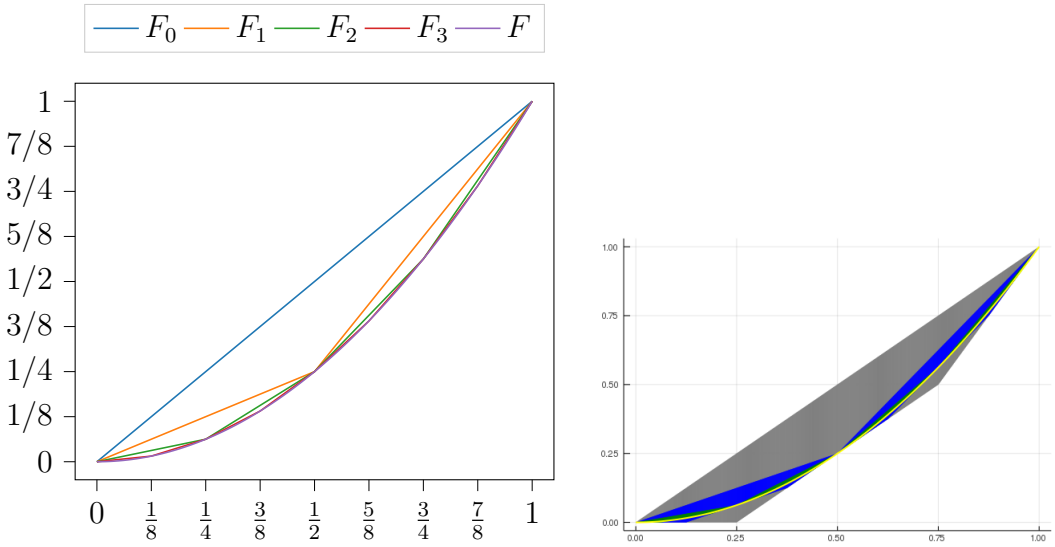


Figure 5.3.2: *Left*: The sawtooth approximation $F^L(x)$ modelled by (5.11). *Right*: The sawtooth relaxation modelled by (5.12), with $L = 0$ (grey), $L = 1$ (blue), and $L = 2$ (green).

Thus, to apply sawtooth-type formulations on general intervals for x, we first apply the formulations to $(\hat{x}, \hat{y})$, then add constraints

$$\hat{x} = \tfrac{x-\underline{x}}{l_x}, \quad \hat{y} = \tfrac{y-\underline{x}(2x-\underline{x})}{{l_x}^2}.$$

In the computations performed in this work, these constraints are implemented as defining expressions for $\hat{x}$ and $\hat{y}$, so that the definitions are programmatically substituted before the models are constructed.

Now, consider the LP-relaxation of S^L, where each α_i is relaxed to the interval $[0, 1]$. Then, observing the constraints (5.9), we see that the weakest lower-bounds on each g_i w.r.t. g_{i-1} can be attained via $\alpha_i = g_{i-1}$, yielding a lower-bound of 0. Thus, we have that, after projecting out $\boldsymbol{\alpha}$, the LP-relaxation of S^L, $T^L := \text{proj}_{x,\boldsymbol{g}}(\text{relax}(S^L))$, can be given as

$$T^L := \left\{(x, \boldsymbol{g}) \in [0,1] \times [0,1]^{L+1} : (5.14)\right\},$$

$$\begin{aligned} g_0 &= x \\ g_i &\leq 2(1 - g_{i-1}) \qquad i = 1, \ldots, L \\ g_i &\leq 2g_{i-1} \qquad\quad i = 1, \ldots, L. \end{aligned} \tag{5.14}$$

The sawtooth relaxation (5.12) is sharp by Theorem 5.18 (proven later in this work), which follows in much the same way as the sharpness of the sawtooth approximation (5.11), as established in Theorem 4.14. Thus, the LP relaxation of the sawtooth relaxation (5.12) yields the same lower-bound on y as the MIP version due to sharpness and the convexity of $F^L(x)$. This allows us to define an LP relaxation for the epigraph of a quadratic function $y \geq x^2$, $\text{epi}_{[0,1]}(x^2)$:

Definition 5.5 (Sawtooth Epigraph Relaxation, SER)**.** The L-layer sawtooth relaxation for $y \geq x^2$ on the interval $x \in [0, 1]$ is given by

$$Q^L := \left\{(x, y) \in [0,1] \times \mathbb{R} : \exists\, \boldsymbol{g} \in [0,1]^{L+1} : (5.16)\right\}, \tag{5.15}$$

$$\begin{aligned} y &\geq F^j(x, \boldsymbol{g}) - 2^{-2j-2} \qquad j \in 0, \dots, L \\ y &\geq 0, \quad y \geq 2x - 1 \\ (x, \boldsymbol{g}) &\in T^L. \end{aligned} \tag{5.16}$$

Alternatively, this relaxation can be written as

$$\text{epi}_{T^{L_1}} \left(\max \left\{ 0, 2x - 1, \max_{j \in [\![0, L_1]\!]} \left(F^j(x, \boldsymbol{g}) - 2^{-2j-2} \right) \right\} \right).$$

We prove in Proposition 5.17 that the maximum approximation error for the sawtooth epigraph relaxation is 2^{-2L-4}.

Finally, we combine the *depth-L* sawtooth relaxation (5.12) with the *depth-L_1* sawtooth epigraph relaxation (5.15) to obtain a strengthened sawtooth relaxation with the same number of binary variables as in (5.12).

Definition 5.6 (Strengthened Sawtooth Relaxation, SSR)**.** The strengthened sawtooth relaxation for $y = x^2$ on the interval $x \in [0, 1]$ with upper-bounding depth L and lower-bounding depth $L_1 \geq L$ is given by

$$R^{L,L_1} := \{(x, y) \in [0, 1] \times \mathbb{R} : \exists (\boldsymbol{g}, \boldsymbol{\alpha}) \in [0, 1]^{L_1+1} \times \{0, 1\}^L : (5.18)\}, \tag{5.17}$$

$$(x, \boldsymbol{g}_{[\![0,L]\!]}, \boldsymbol{\alpha}) \in S^L(x) \tag{5.18a}$$

$$(x, \boldsymbol{g}) \in T^{L_1}(x) \tag{5.18b}$$

$$y \leq F^L(x, \boldsymbol{g}_{[\![0,L]\!]}) \tag{5.18c}$$

$$y \geq F^j(x, \boldsymbol{g}) - 2^{-2j-2} \quad j \in 0, \dots, L_1 \tag{5.18d}$$

$$y \geq 0, \quad y \geq 2x - 1, \tag{5.18e}$$

Alternatively, this relaxation can be written as

$$\text{hyp}_{S^L}(F^L(x,\boldsymbol{g})) \cap \text{epi}_{T^{L_1}} \left(\max \left\{ 0, 2x-1, \max_{j \in [\![0,L_1]\!]} \left(F^j(x,\boldsymbol{g}) - 2^{-2j-2} \right) \right\} \right).$$

We prove in Theorem 5.18 that this strengthened sawtooth relaxation is sharp, and in Theorem 5.35 that it is hereditarily sharp. We also note that the sawtooth epigraph relaxation (5.15), which requires no binary variables, can be used to tighten relaxations of univariate quadratic terms when using any alternative methods. For example, we use this relaxation to tighten the univariate forms of the NMDT ([1], Section 5.4.2) and D-NMDT (Section 5.4.3) relaxations.

5.4 MIP Formulations for Non-Convex QCQP's

In this section, we focus on MIP formulations for relaxing bilinear terms of the form $z = xy$. The novel formulations presented herein, *D-NMDT* and *hybrid univariate*, are extensions of existing formulations, designed to significantly reduce the number of binary variables required to reach the same level of approximation accuracy compared to their original counterparts *NMDT, Bin2*, and *Bin3*.

5.4.1 Separable MIP Formulations

We present three MIP formulations based on separable reformulations. A separable reformulation turns a multivariate expression into a sum of univariate functions. We make use of the reformulations Bin2 and Bin3

$$\begin{aligned} \text{Bin2:} \quad xy &= 1/2((x+y)^2 - x^2 - y^2) \\ \text{Bin3:} \quad xy &= 1/2(x^2 + y^2 - (x-y)^2), \end{aligned}$$

from [172] and combine them with the sawtooth relaxation for graphs in (5.17) to derive MIP formulations for $z = xy$. While the Bin2 and Bin3 MIP formulations are just natural extensions

of MIP approximations in [172] to MIP relaxations, the third formulation combines the Bin2 and Bin3 reformulations to obtain a MIP relaxation for which the required number of binary variables is greatly reduced.

Definition 5.7 (Bin2)**.** The *Bin2* MIP relaxation of $z = xy$, $x, y \in [0,1]^2$ with a lower bounding depth L_1 and an upper bounding depth of L is defined as follows:

$$\begin{aligned} p &= x + y \\ z &= 1/2(z_p - z_x - z_y) \\ (x, y, z) &\in \mathcal{M}(x, y) \\ (x, z^x), &\quad (y, z^y), (p, z^p) \in R^{L,L_1} \\ x, y &\in [0,1], \quad p \in [0,2]. \end{aligned} \tag{5.19}$$

Definition 5.8 (Bin3)**.** The *Bin3* MIP relaxation of $z = xy$, $x, y \in [0,1]^2$ with a lower bounding depth L_1 and an upper bounding depth of L is defined as follows:

$$\begin{aligned} p &= x - y \\ z &= 1/2(z_x + z_y - z_p) \\ (x, y, z) &\in \mathcal{M}(x, y) \\ (x, z^x), &\quad (y, z^y), (p, z^p) \in R^{L,L_1} \\ x, y &\in [0,1], \quad p \in [-1,1]. \end{aligned} \tag{5.20}$$

Note that we apply the strengthened sawtooth relaxation R^{L,L_1} (5.17) on the variable p, with domain $[-1,1]$ or $[0,2]$. This is done by following the transformation in Remark 5.4 to map p and z^p to $[0,1]$ interval, then applying (5.17) to the transformed variables.

We now combine these two reformulations to describe a novel separable MIP formulation. We wish

to derive a MIP relaxation for $z = xy$ based on the following setting

$$\begin{aligned} z &\leq 1/2(x^2 + y^2 - (x-y)^2) \\ z &\geq 1/2((x+y)^2 - x^2 - y^2) \end{aligned}$$

by replacing each right hand side with the proper upper- and lower-bounds. We choose this setting so that we have only to model lower-bounds for the $(x-y)^2$ and $(x+y)^2$ terms, and can thus apply the sawtooth epigraph relaxation (5.15) to circumvent the use of binary variables for these terms. To this end, we introduce the continuous intermediate variables $p_1, p_2, z^x, z^y, z^{p_1}, z^{p_2}$ and z to obtain an equivalent model for $z = xy$:

$$p_1 = x + y, \quad p_2 = x - y \tag{5.21a}$$

$$z^x \leq x^2, \quad z^y \leq y^2, \tag{5.21b}$$

$$z^{p_1} \geq p_1^2, \quad z^{p_2} \geq p_2^2 \tag{5.21c}$$

$$z \leq z^x + z^y - z^{p_1},\ z \geq z^{p_2} - z^x - z^y. \tag{5.21d}$$

Finally, we replace x^2 and y^2 in the non-convex constraints (5.21b) with a sawtooth relaxation (5.18c) of depth L and p_1^2 and p_2^2 in the convex constraints (5.21c) by a sawtooth epigraph relaxation (5.18e) with depth L_1 to obtain a relaxation of $z = xy$ in (5.21d). This model is of special interest as, in contrast with Bin2 and Bin3, it does not require binary variables to model terms of the form $p_1^2 = (x+y)^2$ and $p_2^2 = (x-y)^2$. Note that, when each term $x_i x_j$ and x_i^2 appears in some constraint of the QCQP, for example if some Q_i is dense, the number of (5.21c)-type constraints is quadratic in the dimension of x. Thus, the number of binary variables for Bin2 and Bin3 is $O(n^2 L)$, while this hybrid formulation requires only nL binary variables.

As we show in Section 5.5, this hybrid formulation also has a smaller volume than either the Bin2 or Bin3 formulations, with a strictly tighter LP-relaxation. We also note, however, that the full MIP relaxation is not strictly tighter. For example, if $L = 1,\ L_1 = \infty$, the upper-bounding portion

of the Bin2 relaxation is exact on the line $x + y = 1$, while at e.g. $(x, y) = (\frac{1}{4}, \frac{3}{4})$, the HybU formulation reaches its maximum error.

Definition 5.9 (Hybrid Univariate, HybU)**.** Suppose $x, y \in [0, 1]$, and $L, L_1 \in \mathbb{N}$. We define a MIP relaxation for $z = xy$, combining the Bin2 and Bin3 relaxations to obtain a *hybrid univariate* MIP relaxation of $z = xy$, $x, y \in [0, 1]^2$ with a lower bounding depth L_1 and an upper bounding depth of L:

$$\begin{aligned} p_1 &= x + y, \quad p_2 = x - y \\ (x, \boldsymbol{z}^x), &\quad (y, \boldsymbol{z}^y) \in R^{L,L_1} \\ (p_1, z^{p_1}), &\quad (p_2, z^{p_2}) \in Q^L \\ z &\geq 1/2(z^{p_1} - z^x - z^y), \quad z \leq 1/2(z^x + z^y - z^{p_2}) \\ (x, y, z) &\in \mathcal{M}(x, y) \\ x, y &\in [0, 1], \quad p_1 \in [0, 2], \quad p_2 \in [-1, 1]. \end{aligned} \tag{5.22}$$

Remark 5.10. We can alternatively obtain a convex mixed-integer quadratic relaxation of $z = xy$ by modeling the constraints $z^{p_i} \geq p_i^2$, $z^x \geq x^2$, and $z^y \geq y^2$ exactly instead of using the pwl. relaxation Q^L.

When applying any of the three formulations to solve an MIQCQP, all univariate quadratic terms of the form x_i^2 are modelled via the strengthened sawtooth relaxation (5.17).

5.4.2 Base-2 NMDT

Here, we present the base-2 version of the Normalized multiparametric disaggregation technique (*NMDT*) by Castro [1], along with its univariate form Appendix 4.A. We then discuss a tightened version of the univariate form that takes advantage of the sawtooth epigraph relaxation (5.15). The key idea here to model $z = xy$ is to discretize one variable, e.g. x, using binary variables

$\boldsymbol{\alpha}_i \in \{0,1\}^L$ and a residual term Δ_x^L, then model the resulting products $\alpha_i y$ and $\Delta_x^L y$ using McCormick relaxations.

Definition 5.11 (NMDT [1])**.** The *NMDT* relaxation of $z = xy$ with $x \in [0,1]$, $y \in [0,1]$, and a depth of $L \in \mathbb{N}$ is defined as follows:

$$\begin{aligned} x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L \\ z &= \sum_{i=1}^{L} 2^{-i}u_i + \Delta_z^L \\ (y, \alpha_i, u_i) &\in \mathcal{M}(y, \alpha_i) && i \in 1, \dots, L \\ (\Delta_x^L, y, \Delta_z^L) &\in \mathcal{M}(\Delta_x^L, y) \\ \Delta_x^L &\in [0, 2^{-L}], \quad y \in [0,1], \quad \boldsymbol{\alpha} \in \{0,1\}^L. \end{aligned} \tag{5.23}$$

Since McCormick envelopes are exact if at least one of the variables is binary, all error for this formulation comes from the McCormick relaxation of $\Delta_z^L = \Delta_x^L \cdot y$, yielding a maximum possible error of 2^{-L-2}. An advantage of the NMDT approach compared to the other formulations in this section is that it requires fewer binary variables to reach a desired level of accuracy for *bipartite* QCQP's, for which the quadratic part in each constraint is of the form $\boldsymbol{x}^T Q_i \boldsymbol{y}$. This is due to the fact that one can discretize only the smaller vector $\boldsymbol{x} \in \mathbb{R}^n$ or $\boldsymbol{y} \in \mathbb{R}^m$. Thus, NMDT requires only $2L\min(m,n)$ binary variables instead of the $L(m+n)$ variables required by D-NMDT or HybU to reach a maximum error of 2^{-2L-2} for each bilinear term.

On the other hand, NMDT requires double the binary variables to reach the same level of precision if all quadratic terms $x_i x_j$ and x_i^2 must be modeled, for example if some Q_i is dense. We can also model univariate quadratic terms $y = x^2$ with NMDT technique:

Definition 5.12 (Univariate NMDT [1])**.** The *NMDT* relaxation of $y = x^2$ with $x \in [0,1]$ and a depth of $L \in \mathbb{N}$ is defined as follows:

$$
\begin{aligned}
x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L \\
y &= \sum_{i=1}^{L} 2^{-i}u_i + \Delta_y^L \\
(x, \alpha_i, u_i) &\in \mathcal{M}(x, \alpha_i) && i \in 1, \dots, L \\
(\Delta_x^L, x, \Delta_y^L) &\in \mathcal{M}(\Delta_x^L, x) \\
\Delta_x^L &\in [0, 2^{-L}], \quad x \in [0,1], \quad \boldsymbol{\alpha} \in \{0,1\}^L.
\end{aligned}
\tag{5.24}
$$

Note that this formulation yields a maximum linearization error of nearly 2^{-L-2} instead of the 2^{-2L-2} error bound of the sawtooth relaxation from (5.12). Further, the formulation is not sharp, as for example at $x = \frac{1}{2}$, its LP-relaxation admits the solution $\alpha_i = \frac{1}{2}\ \forall i$, $\Delta_x^L = 2^{-L-1}$, $u_i = 0\ \forall i$, $\Delta_y^L = 0$, $y = 0$. However, we can tighten (5.24) in the lower-bound, adding the depth-L_1 ($\geq L$) sawtooth epigraph relaxation (5.15) to the McCormick envelope, yielding $(x, y) \in Q^{L_1}$. We refer to NMDT with this lower-bounding tightening for quadratic terms as *T-NMDT*.

5.4.3 Double-Discretized NMDT

The key idea for this formulation is to further increase the precision of the NMDT technique by discretizing the second variable y as well, which leads to a *double* NMDT substition, namely in the $\Delta_x^L y$ term. In this way, for problems where NMDT would require discretizing all x_i variables, for example if every x_i^2 term appears in some constraint, we can double the accuracy of the formulation for $z_{ij} = x_i x_j$ without adding additional binary variables by taking advantage of the fact that both variables are discretized anyway.

Now, when applying this technique, we can choose whether we first substitute x or y for each term $z = xy$, leading to two different formulations. These formulations are equivalent in terms of accuracy, but differ in terms of the strength of the LP-relaxations. For example, when first substituting x, the LP-relaxation is weakest when $x = \frac{1}{2}$, $\Delta y = 2^{-L-1}$, whereas the roles of x and

y are reversed when first substituting y. By introducing a parameter $\lambda \in [0,1]$, we can model a hybrid version of the two formulations. We write

$$xy = \lambda xy + (1-\lambda)xy,$$

then discretize y first in the relaxation of λxy and x first in the relaxation of $(1-\lambda)xy$. Finally, the complete MIP formulation *D-NMDT* is obtained by relaxing the resulting products via McCormick envelopes.

Definition 5.13 (D-NMDT)**.** The *D-NMDT* relaxation of $z = xy$ with $x, y \in [0,1]$ and a depth of $L \in \mathbb{N}$ is defined as follows:

$$\begin{aligned}
x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L, \quad y = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta_y^L \\
z &= \sum_{i=1}^{L} 2^{-i}(u_i + v_i) + \Delta_z^L \\
(\lambda\Delta_y^L + (1-\lambda)y, \alpha_i, u_i) &\in \mathcal{M}(\lambda\Delta_y^L + (1-\lambda)y, \alpha_i) \qquad \forall i \\
((1-\lambda)\Delta_x^L + \lambda x, \beta_i, v_i) &\in \mathcal{M}((1-\lambda)\Delta_x^L + \lambda x, \beta_i) \qquad \forall i \\
(\Delta_x^L, \Delta_y^L, \Delta_z^L) &\in \mathcal{M}(\Delta_x^L, \Delta_y^L) \\
\Delta_x^L, \Delta_y^L &\in [0, 2^{-L}], \quad x, y \in [0,1], \quad \boldsymbol{\alpha}, \boldsymbol{\beta} \in \{0,1\}^L
\end{aligned} \tag{5.25}$$

As McCormick envelopes are exact if one of the variables is binary, all relaxation error stems from the relaxation of the continuous variable product $\Delta_x^L \Delta_y^L$, yielding an error bound of 2^{-2L-2}. For bounds on the terms $(1-\lambda)\Delta_x^L + \lambda x$ and $\lambda\Delta_y^L + (1-\lambda)y$, see Appendix 5.A.

Remark 5.14. We recommend $\lambda = \frac{1}{2}$ for the sake of formulation symmetry in x and y, or else adding the required constraints for both $\lambda = 0$ and $\lambda = 1$, which can be done without increasing the number of binary variables. Using the constraints for both $\lambda = 0$ and $\lambda = 1$ has the added benefit of yielding very strong LP-relaxations in the original space near the edges of the domain for x and y.

To model the univariate quadratic terms with this method, we substitute $y \to x$ and $z \to y$, to obtain the following relaxation for $y = x^2$, which is equivalent to the *T-NMDT* formulation from Appendix 4.A.

Definition 5.15 (Univariate D-NMDT)**.** The *D-NMDT* relaxation of $y = x^2$ with $x \in [0, 1]$ and a depth of $L \in \mathbb{N}$ is defined as follows:

$$
\begin{aligned}
x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L \\
y &= \sum_{i=1}^{L} 2^{-i}u_i + \Delta_y^L \\
(\Delta_x^L + x, \alpha_i, u_i) &\in \mathcal{M}(\Delta_x^L + x, \alpha_i) && i \in 1, \dots, L \\
(\Delta_x^L, \Delta_y^L) &\in \mathcal{M}(\Delta_x^L) \\
\Delta_x^L &\in [0, 2^{-L}], \quad x \in [0, 1], \quad \boldsymbol{\alpha} \in \{0, 1\}^L.
\end{aligned}
\tag{5.26}
$$

As with the sawtooth formulations (5.11), (5.12) and (5.17), this formulation yields an error bound of 2^{-2L-2}. The upper-bound of this formulation models exactly the same pwl. approximation for $y = x^2$ as the sawtooth formulations. Unfortunately, the formulation is not sharp; for example, at $x = \frac{1}{2}$, its LP-relaxation admits the solution $\alpha_i = \frac{1}{2}\ \forall i$, $\Delta_x^L = 2^{-L-1}$, $\Delta_y^L = 0$, $u_i = 0\ \forall i$, $y = 0$.

To form a strengthened form of D-NMDT, here TD-NMDT, we tighten (5.26) in the lower-bound, removing all McCormick lower-bounds and adding the depth-L_1 ($\geq L$) sawtooth epigraph relaxation (5.15).

Remark 5.16 (Improvements in binary variable counts)**.** The Bin2 and Bin3 formulations require binary variables for each univariate term x_i^2, plus L extra binary variables for each $x_i x_j$ term. If all product terms are present, this results in $\frac{1}{2}(n^2 + n)L$ total binary variables instead of just nL in the hybrid univariate approach, to reach a comparable approximation accuracy. Similarly NMDT requires twice the number of binary variables to reach the same level of precision as the hybrid univariate and D-NMDT formulations.

5.4.4 Derivations of NMDT-Based Formulations

Below, we derive the formulations for NMDT and D-NMDT. To start, let $x \in [0,1]$ and $y \in [0,1]$. The core idea behind the base-2 version of NMDT is to discretize x as $x = \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L$, then multiply by y to obtain the exact representation

$$\begin{aligned} x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L, \quad z = \sum_{i=1}^{L} 2^{-i}\alpha_i y + \Delta_x^L y \\ \Delta_x^L &\in [0, 2^{-L}], \quad y \in [0,1], \quad \boldsymbol{\alpha} \in \{0,1\}^L \end{aligned} \tag{5.27}$$

We then use McCormick envelopes to model all remaining product terms, $\alpha_i y$ and $\Delta_x^L \cdot y$, to obtain the final formulation. The derivation for D-NMDT is somewhat more involved. To begin, define $x = \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L$ and $y = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta_y^L$. We then start with the NMDT representation (5.27), expand the $\Delta_x^L y$ term, and obtain

$$\begin{aligned} z &= xy = y\left(\sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L\right) \\ &= \sum_{i=1}^{L} 2^{-i}\alpha_i y + y\Delta_x^L \\ &= \sum_{i=1}^{L} 2^{-i}\alpha_i y + \Delta_x^L\left(\sum_{i=1}^{L} 2^{-i}\beta_i + \Delta_y^L\right) \\ &= \sum_{i=1}^{L} 2^{-i}(\alpha_i y + \beta_i \Delta_x^L) + \Delta_x^L \Delta_y^L. \end{aligned}$$

Alternatively, if we discretize y first, then expand the $\Delta_y^L x$ term, we obtain

$$z = \sum_{i=1}^{L} 2^{-i}(\beta_i x + \alpha_i \Delta_y^L) + \Delta_x^L \Delta_y^L$$

Finally, to balance between the two formulations, we observe for any $\lambda \in [0,1]$ that

$$\begin{aligned} z &= xy = \lambda xy + (1-\lambda)xy \\ &= \lambda \left(\sum_{i=1}^{L} 2^{-i}(\beta_i x + \alpha_i \Delta_y^L) + \Delta_x^L \Delta_y^L \right) \\ &+(1-\lambda) \left(\sum_{i=1}^{L} 2^{-i}(\alpha_i y + \beta_i \Delta_x^L) + \Delta_x^L \Delta_y^L \right) \\ &= \sum_{i=1}^{L} 2^{-i}[\beta_i((1-\lambda)\Delta_x^L + \lambda x) + \alpha_i(\lambda \Delta_y^L + (1-\lambda)y)] + \Delta_x^L \Delta_y^L. \end{aligned}$$

This yields the formulation

$$\begin{aligned} x &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L, \quad y = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta_y^L \\ z &= \sum_{i=1}^{L} 2^{-i}[\beta_i((1-\lambda)\Delta_x^L + \lambda x) + \alpha_i(\lambda \Delta_y^L + (1-\lambda)y)] + \Delta_x^L \Delta_y^L \\ \Delta_x^L, \Delta_y^L &\in [0, 2^{-L}], \quad x, y \in [0,1], \quad \boldsymbol{\alpha}, \boldsymbol{\beta} \in \{0,1\}^L \end{aligned} \tag{5.28}$$

We then obtain the D-NMDT formulation by applying McCormick envelopes to the product terms $\beta_i((1-\lambda)\Delta_x^L + \lambda x)$, $\alpha_i(\lambda \Delta_y^L + (1-\lambda)y)$, and $\Delta_x^L \Delta_y^L$. For bounds on these terms, see Appendix 5.A.

5.5 Theoretical Results

In this section, we prove some theoretical results that allow a comparison of the presented MIP formulations.

5.5.1 Approximation error

Here, we discuss the relaxation errors for the individual formulations.

Core formulations

First, we discuss the relaxation errors for the core formulations from Section 5.3. For the sawtooth approximation, the maximum error is given by 2^{-2L-2}, see Section 4.2.1. The strengthened sawtooth graph relaxation stated in (5.17) has the same maximum error of 2^{-2L-2} as the sawtooth approximation in terms of overestimation. However, the lower-bound, which is incident with the sawtooth epigraph relaxation (5.15), gains an extra layer of precision, with a maximum error of 2^{-2L_1-4}, as we prove below.

Proposition 5.17 (Error of sawtooth epigraph relaxation)**.** *The maximum error for the sawtooth epigraph relaxation* (5.15) *is* 2^{-2L-4}*. That is, if* $(x, y) \in \text{proj}_{x,y}(Q^L)$*, then we have* $x^2 - y \leq 2^{-2L-4}$*.*

Proof. In a manner similar to Lemma 4.18, the lower-bounding set for the (x, y)-projection of the sawtooth epigraph relaxation, $\text{proj}_{x,y}(Q^L)$, is comprised of the outer-approximation cuts to $y \geq x^2$ at exactly the points $\frac{k}{2^{L+1}}$, $k = 0 \dots 2^L$. Further, the maximum error occurs at the intersection of consecutive cuts, which from the proof of Lemma 4.19 is the point $(x, y) = (\frac{x_1+x_2}{2}, x_1x_2)$ for any two tangent lines at $x = x_1$ and $x = x_2$, which in this case becomes $((k + \frac{1}{2})2^{-L-1}, k(k + 1)2^{-2L-2})$. Thus, the error is given by

$$x^2 - y = \left((k + \tfrac{1}{2})2^{-L-1}\right)^2 - k(k + 1)2^{-2L-2} = 2^{-2L-2}(k^2 + k + \tfrac{1}{4} - k^2 - k) = 2^{-2L-4}.$$

Since this maximum error applies to the interval $x \in [\frac{k}{2^{L+1}}, \frac{k+1}{2^{L+1}}]$ for any k, we conclude that the maximum error is 2^{-2L-4}. □

For McCormick envelopes on $z = xy$ over $[\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}]$, it is known that the maximum under- and overestimation is $\frac{1}{4}(\overline{x} - \underline{x})(\overline{y} - \underline{y})$, taken at $(x, y) = (\frac{1}{2}(\underline{x} + \overline{x}), \frac{1}{2}(\underline{y} + \overline{y}))$, see e.g. [176].

Full Formulations

We start with the separable formulations. In the Bin2, Bin3, and HybU approaches, increasing L_1 does not introduce any new binary variables. Therefore L_1 is usually chosen to be significantly larger than L, so that the approximation error depends primarily on L. In the following, we therefore assume an exact underestimation with $L_1 \to \infty$. Consider bounds on z for which x^2 and y^2 are overestimated. Then the corresponding p^2 term is underestimated exactly as $z^p \geq p^2$, so that the only source of error is the overestimation of the x^2 and y^2 terms. Thus, when overestimating x^2 and y^2, we have a maximum error of 2^{-2L-2}, which occurs at the grid centers $x, y = (k + \frac{1}{2})2^{-L}$, $k = 0, \ldots, 2^L - 1$. This applies to the underestimator for Bin2, the overestimator for Bin3, and both the underestimator and overestimator for the hybrid univariate approach.

Similarly, for the Bin2 and Bin3 methods, consider bounds on z for which p^2 is overestimated, $p = (x+y)$ or $p = (x-y)$, with a domain width of 2. Then the x^2 and y^2 terms are underestimated exactly as e.g. $z^x \geq x^2$, so that the only source of error is the overestimation of the p^2 term. Then we have a maximum error for p^2 of 2^{-2L} (twice the domain width $\Rightarrow$ quadruple the error), yielding a total maximum error on z of 2^{-2L-1}. Mapping to $\hat{p} \in [0, 1]$ as in Remark 5.4, this error which occurs whenever $p = (k + \frac{1}{2})2^{-L}$, $k = 0, \ldots, 2^L - 1$, with an overall maximum error of $\frac{1}{2}2^{-2L} = 2^{-2L-1}$. Thus, the overall maximum possible error for Bin2 and Bin3 is 2^{-2L-1}, while the maximum possible error for the hybrid univariate approach is 2^{-2L-2}.

To compute these points of maximum error, let $k \in [\![0, 2^L - 1]\!]$. Then for the overestimator for Bin2, where $p = x + y$, this occurs whenever $\frac{x+y}{2} = (k + \frac{1}{2})2^{-L} \Rightarrow x + y = (2k + 1)2^{-L}$. For Bin3, where $p = x - y$, this occurs whenever $\frac{x-y+1}{2} = (k + \frac{1}{2})2^{-L} \Rightarrow x - y = (2k + 1)2^{-L} - 1$.

Now, the base-2 NMDT and D-NMDT approaches for $z = xy$ yield piecewise McCormick relaxation over a uniform grid. For NMDT, the only source of error is the McCormick relaxation of the term $\Delta x \cdot y$, yielding a maximum error of $\frac{1}{4}(2^{-L} \cdot 1) = 2^{-L-2}$. Likewise, for D-NMDT, the only source of error is the McCormick relaxation of the term $\Delta x \cdot \Delta y$, yielding a maximum error of $\frac{1}{4}(2^{-L} \cdot 2^{-L}) = 2^{-2L-2}$.

For univariate terms, the maximum error for D-NMDT is still 2^{-2L-2}, from the McCormick relaxation of the term Δx^2. On the other hand, for NMDT, the error stems from the McCormick relaxation of the term $\Delta x \cdot x$, for which the usual maximizing point $x = \frac{1}{2}, \Delta x = 2^{-L-1}$ is not feasible, since $x = \frac{1}{2} \Rightarrow \Delta x = 0$. Thus, the maximum error is slightly less than 2^{-L-2}.

For $L \geq 1$, the maximum lower-bounding error for univariate NMDT is:

$$
\begin{aligned}
&\max_{\Delta x \in [0, 2^{-L}]} ((\tfrac{1}{2} + \Delta x)\Delta x - \max\{0, \Delta x - 2^{-L}(1 - (\tfrac{1}{2} + \Delta x))\} \\
=&\max_{\Delta x \in [0, 2^{-L}]} ((\tfrac{1}{2} + \Delta x)\Delta x + 2^{-L-1} - \max\{2^{-L-1}, \Delta x(1 + 2^{-L})\} \\
=&\max_{\Delta x \in [0, 2^{-L}]} \left((\tfrac{1}{2} + \Delta x)\Delta x + 2^{-L-1} - \begin{cases} 2^{-L-1} & \Delta x \in [0, \frac{1}{2(2^L+1)}] \\ (1 + 2^{-L})\Delta x & \Delta x \in [\frac{1}{2(2^L+1)}, 2^{-L}] \end{cases} \right) \\
=&2^{-L-2} - 2^{-3L-2}(1 + 2^{-L})^{-2} \quad \text{at } \Delta x = \tfrac{1}{2(2^L+1)}
\end{aligned}
$$

We compute the maximum upper-bounding error in a similar manner. For easy comparison, we provide the following error-maximal points for univariate NMDT, $L \geq 1$. We note that the upper-bound for univariate NMDT matches D-NMDT if $L = 1$.

Lower-bound: $\frac{1}{2^{L+2}} - \frac{1}{2^{3L+2}(1+2^{-L})^2}$, at $x = \frac{1}{2}\left(1 \pm \frac{1}{2(2^L+1)}\right)$,

Upper-bound: $\frac{1}{2^{L+2}} - \frac{1}{2^{3L+2}(1-2^{-L})^2}$, at $x = \frac{1}{2}\left(1 \pm \frac{1}{2(2^L-1)}\right)$ if $L \geq 2$.

5.5.2 Formulation strength

We start with sawtooth-relaxation, since we know from Theorem 4.7 that the sawtooth approximation is sharp.

Theorem 5.18. *For nonnegative integers $L \leq L_1$, define P^{IP} as the full formulation for the strengthened sawtooth relaxation (5.17), so that $R^{L,L_1} = \text{proj}_{x,y}(P^{\text{IP}})$. Then P^{IP} is sharp.*

Proof. First, we note that the upper- and lower-bounds on y in (5.17) do not interact. In P^{IP}, the

upper-bounds on y are always strictly greater than x^2, while the lower-bounds are always strictly smaller. Thus, we can consider sharpness with respect to upper- and lower-bounds independently. More formally, define

$$\begin{aligned} P_+^{\text{IP}} &:= \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1] \times \mathbb{R} \times [0,1]^{L_1+1} \times \{0,1\}^L : (5.18a) - (5.18c)\} \\ P_-^{\text{IP}} &:= \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1] \times \mathbb{R} \times [0,1]^{L_1+1} \times \{0,1\}^L : (5.18a), (5.18b), (5.18d), (5.18e)\}. \end{aligned}$$

Then P^{IP} is sharp if and only if both P_+^{IP} and P_-^{IP} are sharp: $\text{proj}_{x,y}(P_-^{\text{IP}}) \supseteq \text{epi}_{[0,1]}(x^2)$, while $\text{proj}_{x,y}(P_+^{\text{IP}}) \supseteq \text{hyp}_{[0,1]}(x^2)$, with $P^{\text{IP}} = P_+^{\text{IP}} \cap P_-^{\text{IP}}$. That is, in upper-bound, P_+^{IP} strictly overestimates x^2, while in lower-bound, P_-^{IP} strictly underestimates x^2, so that sharpness of the two can be considered separately.

Now, sharpness of P_+^{IP} follows directly from sharpness of the sawtooth approximation (5.11), which holds by Theorem 4.7. For sharpness of P_-^{IP}, the proof closely follows the proof of sharpness in Theorem 4.7, except that, after choosing some fixed $x \in [0,1]$ we frame the contradiction as follows.

1. Choose $\boldsymbol{g}^*$ as in Theorem 4.7, and choose the minimum possible value of y^* given $\boldsymbol{g}^*$, so that y^* is incident with one of its lower-bounds.

2. Observe that the chosen solution admits a feasible solution in P^{IP}, so that if it is minimal in the LP, then we are done.

3. Assume for a contradiction that there exists a better y-minimal solution $(\hat{y}, \hat{\boldsymbol{g}})$ than the proposed solution $(y^*, \boldsymbol{g}^*)$, so that some incident lower-bound must have been improved.

4. Observe that the improved incident lower-bound must be of the form $y \geq F^j(x, \boldsymbol{g}^*) - 2^{-2L-2}$ for some $j \geq 0$, as the lower-bounds 0 and $2x - 1$ do not change with the choice of g^*. Thus, $F^j(x, \boldsymbol{g}^*) - 2^{-2L-2} \geq F^j(x, \hat{\boldsymbol{g}}) - 2^{-2L-2}$

5. Show that $F^j(x, \boldsymbol{g}^*) - F^j(x, \boldsymbol{g}^*) < 0$, a contradiction on the choice of $(\hat{y}, \hat{\boldsymbol{g}})$. Thus, the solution $y^*, \boldsymbol{g}^*$ was optimal to begin with, and so sharpness must hold.

The proof that $F^j(x, \boldsymbol{g}^*) - F^j(x, \boldsymbol{g}^*) < 0$ follows in exactly the same manner as Theorem 4.7, and so is omitted here. □

Next, we analyse the separable formulations HybU, Bin2 and Bin3. We show that none of these approaches is sharp, implying that their projected LP relaxation does not correspond to $\mathcal{M}(x, y)$ in the convergence case L, $L_1 \to \infty$. We also compute the volume of projected LP relaxations and show that the hybrid univariate approach is strictly tighter than Bin2 and Bin3.

Proposition 5.19. *Let P^{IP}_{L,L_1} be the hybrid univariate MIP formulation stated in (5.22). Then, ignoring the McCormick envelope constraints, P^{IP}_{L,L_1} is not sharp for any L, $L_1 \in \mathbb{N}$. For the convergence case L, $L_1 \to \infty$ it holds that $P^{\text{IP}}_{\infty,\infty}$ is also not "sharp", i.e. $\mathcal{M}(x, y) \subset \text{proj}_{\boldsymbol{u}}(P^{\text{LP}}_{\infty,\infty})$.*

Proof. Ignoring the McCormick envelopes, the HybU MIP formulation P^{IP}_{L,L_1}, and its LP-relaxation P^{LP}_{L,L_1}, become strictly tighter as either L or L_1 increases. Thus, we have $\text{proj}_{(x,y,z)}(P^{\text{LP}}_{L,L_1}) \supseteq \text{proj}_{(x,y,z)}(P^{\text{LP}}_{\infty,\infty})$ and $\text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{1,1})) \supseteq \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{L,L_1}))$.

We show that $\text{proj}_{(x,y,z)}(P^{\text{LP}}_{\infty,\infty}) \setminus \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{1,1})) \neq \emptyset$ holds, implying $\text{proj}_{(x,y,z)}(P^{\text{LP}}_{L,L_1}) \setminus \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{L,L_1})) \neq \emptyset$, so that P^{IP}_{L,L_1} is not sharp:

$$
\begin{aligned}
&\text{proj}_{(x,y,z)}(P^{\text{LP}}_{L_1,L_1}) \setminus \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{L_1,L_1})) \\
\supseteq\ &\text{proj}_{(x,y,z)}(P^{\text{LP}}_{\infty,\infty}) \setminus \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{1,1})) \neq \emptyset \\
\Rightarrow\ &\text{proj}_{(x,y,z)}(P^{\text{LP}}_{L,L_1}) \setminus \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{L,L_1})) \neq \emptyset \\
\Rightarrow\ &\text{proj}_{(x,y,z)}(P^{\text{LP}}_{L,L_1}) \neq \text{conv}(\text{proj}_{(x,y,z)}(P^{\text{IP}}_{L,L_1})).
\end{aligned}
$$

To this end, we show that there exist points $(x, y, z) \in \text{proj}_{(x,y,z)}(P^{\text{LP}}_{\infty,\infty})$ with $(x, y, z) \notin \text{proj}_{(x,y,z)}(P^{\text{IP}}_{1,1})$. Now, observe that, for any L, the point (x, x) is feasible within the LP-relaxation of the strengthened sawtooth relaxation (5.17) for $f(x) = x^2$, with $\alpha_i = g_{i-1}$, $g_i = 0$. Thus, for all $L, L_1 \geq 0$ and for all $\hat{x}, \hat{y} \in [0, 1]^2$, we have that P^{LP}_{L,L_1}, and thus its limit $P^{\text{LP}}_{\infty,\infty}$, admits the values $z_x = \hat{x}, z_y = \hat{y}$, and

$z^{p_1} = (\hat{x} + \hat{y})^2$. Thus, for $x = 0,\ y = \frac{1}{4}$, we obtain

$$z = \tfrac{1}{2}((x+y)^2 - x - y) = -\tfrac{3}{16}$$

so that $(0, \frac{1}{4}, -\frac{3}{16}) \in P^{\text{LP}}_{\infty,\infty}$.

Now, to prove that $(0, \frac{1}{4}, -\frac{3}{16}) \notin \text{conv}(\text{proj}_{x,y,z}(P^{\text{IP}}_{1,1}))$, we show that $\min\{z : (y,z) \in \text{proj}_{y,z}(P^{\text{IP}}_{1,1}|_{x=0})\} = -\frac{1}{8}$. If this holds, then we have $\min\{z : (y,z) \in \text{conv}(\text{proj}_{y,z}(P^{\text{IP}}_{1,1}|_{x=0}))\} = -\frac{1}{8}$, so that $(0, \frac{1}{4}, -\frac{3}{16}) \notin \text{conv}(\text{proj}_{x,y,z}(P^{\text{IP}}_{1,1}))$.

We derive a representation of $\text{proj}_{y,z}(P^{\text{IP}}_{1,1}|_{x=0})$ that becomes an LP after branching spatially at $y = \frac{1}{2}$ to resolve the upper-bound on z_y. We then minimize z on both branches via solving a MIP.

Let $x = 0$. Then the bounds on z, z^x, z^y, z^{p_1} within $\text{proj}_{x,y,z} P^{\text{IP}}_{1,1}$, are:

$$\begin{aligned}
&z_x \le 0, \quad z_y \le y - \tfrac{1}{4}\min\{2y, 2(1-y)\} = \max\{\tfrac{y}{2}, \tfrac{3y-1}{2}\} \\
&z^{p_1} \ge 4\left(\tfrac{y}{2} - \tfrac{1}{4}\min\{2\tfrac{y}{2}, 2(1-\tfrac{y}{2}) - \tfrac{1}{16}\}\right) = \max\{y - \tfrac{1}{4}, 3y - \tfrac{9}{4}\} \\
&z^{p_1} \ge 4(\tfrac{y}{2} - \tfrac{1}{4}) = 2y - 1 \\
&z^{p_1} \ge 0 \\
&z^{p_1} \ge 4(2\tfrac{y}{2} - 1) = 4(y-1) \\
&z \ge z^{p_1} - z_x - z_y \\
&y \in [0,1]
\end{aligned}$$

Note that the two pieces of the upper-bound on z_y meet at $y = \frac{1}{2}$. Then, separately minimizing z over the above set given $y \in [0, \frac{1}{2}]$ and $[\frac{1}{2}, 1]$, e.g. with Gurobi, we obtain two globally minimizing solutions with $z = -\frac{1}{8}$, at $y = \frac{1}{4}$ and $y = \frac{3}{4}$. Thus, we conclude that $(0, \frac{1}{4}, -\frac{3}{16}) \notin \text{conv}(\text{proj}_{x,y,z}(P^{\text{IP}}_{1,1}))$, so that P^{IP}_{L,L_1} is not sharp for any $1 \le L \le L_1$. □

Proposition 5.20. *Let P^{IP}_{L,L_1} be a MIP formulation resulting from Bin2 (5.19) or Bin3 (5.20). Then, ignoring the McCormick envelope constraints, P^{IP}_{L,L_1} is not sharp for any L, $L_1 \in \mathbb{N}$. For the*

convergence case L, $L_1 \to \infty$ it holds that $P^{\text{IP}}_{\infty,\infty}$ is also not "sharp", i.e. $\mathcal{M}(x,y) \subset \text{proj}_{\boldsymbol{u}}(P^{\text{LP}}_{\infty,\infty})$.

Proof. Since Bin2 (5.19) has the same lower-bounding constraints as the HybU formulation, the proof follows directly from Proposition 5.19. Moreover, for Bin3 (5.20), the proof follows in exactly the same way as the proof of Proposition 5.19, except with the upper-bounding version of the same point, $(x,y,z) = (0, \frac{1}{4}, \frac{3}{8})$, and acting on the upper-bounding constraints from (5.22) and maximizing z instead. As the proof is very similar, with the corresponding upper-bound $z = \frac{1}{8}$ on $\text{proj}_{y,z}(P^{\text{IP}}_{1,1}|_{x=0})$, we omit it here. □

Having shown that none of the separable MIP formulations is sharp, which naturally implies that they are also not hereditarily sharp, we now turn to the volume of projected LP relaxations.

Proposition 5.21. *Let $P^{\text{LP}}_{\infty,\infty}$ be the LP relaxation of the HybU MIP formulation stated in (5.22) with $L, L_1 \to \infty$ over the domain $[\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}]$. Ignoring the McCormick envelope constraints, the volume of the projected LP relaxation $\text{proj}_{(x,y,z)}(P^{\text{LP}}_{\infty,\infty})$ is given as $\frac{1}{6}(l_x {l_y}^3 + l_y {l_x}^3)$, where $l_x = \overline{x} - \underline{x}$ and $l_y = \overline{y} - \underline{y}$.*

Proof. The z-values in the projected LP relaxation of (5.22) are bounded by the convex function C_2^L and concave function C_3^U, derived in [172],

$$z \geq C_2^L(x,y) = \frac{1}{2}((x+y)^2 - (\overline{x}+\underline{x})x + \overline{x}\underline{x} - (\overline{y}+\underline{y})y + \overline{y}\underline{y})$$
$$z \leq C_3^U(x,y) = \frac{1}{2}((\underline{x}+\overline{x})x - \underline{x}\overline{x} + (\underline{y}+\overline{y})y - \underline{y}\overline{y} - (x-y)^2).$$

The volume of the projected LP relaxation (5.22) is then calculated via integration,

$$\int_{\underline{x}}^{\overline{x}} \int_{\underline{y}}^{\overline{y}} C_3^U - C_2^L dy dx = \tfrac{1}{6}(l_x {l_y}^3 + l_y {l_x}^3).$$

□

Proposition 5.22. *Let $P^{\text{LP}}_{\infty,\infty}$ be the LP relaxation of either the Bin2 or Bin3 MIP formulation stated in (5.19) or (5.20) with $L, L_1 \to \infty$ over the domain $[\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}]$. Ignoring the McCormick envelope constraints, the volume of the projected LP relaxation $\text{proj}_{(x,y,z)}(P^{\text{LP}}_{\infty,\infty})$ is given as $\frac{1}{12} l_x l_y (2{l_x}^2 + 3 l_x l_y + 2{l_y}^2)$, where $l_x = \overline{x} - \underline{x}$ and $l_y = \overline{y} - \underline{y}$.*

Proof. The z-values in the projected LP relaxation of (5.19) and (5.20) are bounded the convex function C_2^L and concave function C_3^U, derived in [172],

$$
\begin{aligned}
Bin2: \; z \geq C_2^L(x,y) &= \frac{1}{2}((x+y)^2 - (\overline{x}+\underline{x})x + \overline{x}\underline{x} - (\overline{y}+\underline{y})y + \overline{y}\underline{y}) \\
z \leq C_2^U(x,y) &= \frac{1}{2}((\underline{x}+\overline{x}+\underline{y}+\overline{y})(x+y) - (\underline{x}+\underline{y})(\overline{x}+\overline{y}) - x^2 - y^2) \\
Bin3: \; z \geq C_3^L(x,y) &= \frac{1}{2}(x^2 + y^2 - (\overline{x}+\underline{x}-\overline{y}-\underline{y})(x-y) + (\underline{x}-\overline{y})(\overline{x}-\underline{y})) \\
z \leq C_3^U(x,y) &= \frac{1}{2}((\underline{x}+\overline{x})x - \underline{x}\overline{x} + (\underline{y}+\overline{y})y - \underline{y}\overline{y} - (x-y)^2).
\end{aligned}
$$

The volume proof is done via integration,

$$
\int_{\underline{x}}^{\overline{x}} \int_{\underline{y}}^{\overline{y}} C_3^U - C_3^L dy dx = \int_{\underline{x}}^{\overline{x}} \int_{\underline{y}}^{\overline{y}} C_2^U - C_2^L dy dx = \frac{1}{12}(l_x l_y (2{l_x}^2 + 3 l_x l_y + 2{l_y}^2).
$$

□ □

Lemma 5.23. *Ignoring the McCormick envelope constraints, the LP relaxation of HybU MIP formulation is strictly tighter than that of Bin2 or Bin3. Moreover, the volume of its projected LP relaxation is smaller by $\frac{1}{4}{l_x}^2{l_y}^2$.*

Proof. In [172, Appendix, Proposition 2] it is shown that C_2^L is a tighter convex underestimator than C_3^L and C_3^U is a tighter concave overestimator than C_2^U on $z = xy$. Thus, since the hybrid univariate approach uses C_2^L as an underestimator and C_3^L as an overestimator, it is strictly tighter

than either Bin2 or Bin3. The volume proof is again via integration,

$$\begin{aligned}&\int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} C_2^U - C_2^L dydx - \int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} C_3^U - C_2^L dydx \\ =&\int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} C_3^U - C_3^L dydx - \int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} C_3^U - C_2^L dydx = \frac{1}{4}{l_x}^2{l_y}^2 > 0.\end{aligned}$$

□

Crucially, we emphasize that the HybU MIP relaxation is not strictly tighter than the Bin2 or Bin3 formulations: For example, as noted in Section 5.4.1, with e.g. $L = 1, L_1 \to \infty$, the upper-bound for the Bin2 relaxation is exact on the line $x + y = 1$, while the HybU relaxation has maximal error at e.g. $(x, y) = \left(\frac{1}{4}, \frac{3}{4}\right)$.

Next, we show that the separable formulations yield the same relaxation volumes over each piece of a grid.

Proposition 5.24. *Let $P^{\text{IP}}_{(L^x,L^y),\infty}$ be the HybU formulation, with $L_1 \to \infty$, over the domain $[0, 1]$, and applying separate discritization depths L^x and L^y to model x^2 and y^2, respectively. Then $P^{\text{IP}}_{(L^x,L^y),\infty}$ has the same volume over each grid piece of the form $[k^x 2^{-L^x}, (k^x+1)2^{-L^x}] \times [k^y 2^{-L^y}, (k^y + 1)2^{-L^y}]$, where $k^x \in [\![0, 2^{L^x}]\!]$ and $k^y \in [\![0, 2^{L^y}]\!]$. Furthermore, the volume of $P^{\text{IP}}_{(L^x,L^y),\infty}$ is given as $\frac{1}{6}(2^{-2L_x} + 2^{-2L_y})$.*

Proof. To begin, let $P^{\text{IP}}_{L,\infty}$ be the Bin2 formulation. Now, recall that $F^L(x)$ is the sawtooth overapproximation of $y = x^2$, and consists of secant lines to x^2 between consecutive points $k2^{-L}, (k + 1)2^{-L}$. Further, applying the appropriate sawtooth overapproximations and exact underapproximations, we have that $\text{proj}_{x,y,z}(P^{\text{IP}}_{(L^x,L^y),\infty})$ is given via the envelopes on z,

$$\tfrac{1}{2}((x + y)^2 - F^{L^x}(x) - F^{L^y}(y)) \le z \le \tfrac{1}{2}((x - y)^2 - F^{L^x}(x) - F^{L^y}(y)).$$

Now, let $\underline{x}, \overline{x} = k^x 2^{-L^x}, (k^x + 1)2^{-L^x}$ and $\underline{y}, \overline{y} = k^y 2^{-L^y}, (k^y + 1)2^{-L^y}$, and define $l_x = \overline{x} - \underline{x} = 2^{-L^x}$ and $l_y = \overline{y} - \underline{y} = 2^{-L^y}$. Consider the restriction of x and y to the grid piece $[\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}]$. Then,

as $F^{L^x}(x)$ and $F^{L^y}(y)$ are secant lines between the domain endpoints, the bounds on z above yield exactly the envelopes $C_3^U(x,y)$ and $C_2^L(x,y)$ defined in Proposition 5.21. Thus, by the calculation in the proof of Proposition 5.21, we have the volume of $P^{\text{IP}}_{(L^x,L^y),\infty}$ over the grid piece is $\frac{1}{6}(l_x {l_y}^3 + l_y {l_x}^3) = \frac{1}{6}2^{-(L_x+L_y)}(2^{-2L_x} + 2^{-2L_y})$, which does not depend on the choice of k^x and k^y (and thus the choice of grid piece).

The total volume is then given as

$$V(P^{\text{IP}}_{(L^x,L^y),\infty}) = 2^{-(L_x+L_y)}(2^{-2L_x} + 2^{-2L_y})(2^{L_x}2^{L_y}) = \tfrac{1}{6}(2^{-2L_x} + 2^{-2L_y}),$$

as required. □

The following proposition establishes the MIP volumes and grid structure for the Bin2 and Bin3 formulations. As this computation is messy, we prove it in Appendix 5.B.

Proposition 5.25. *Let $P^{\text{IP}}_{L,\infty}$ be one of the Bin2 or Bin3 formulations, with $L \geq 1$ and $L_1 \to \infty$, over the domain $[0,1]$. Then $P^{\text{IP}}_{L,\infty}$ has the same volume over each grid piece of the form $[k^x 2^{-(L-1)}, (k^x+1)2^{-(L-1)}] \times [k^y 2^{-(L-1)}, (k^y+1)2^{-(L-1)}]$, $k^x, k^y \in [\![0, 2^{L-1}]\!]$. Furthermore, the total volume of each formulation is $\frac{1}{2}2^{-2L}$.*

Finally, we derive the volume of the NMDT and D-NMDT relaxations.

Proposition 5.26. *The volume of the NMDT relaxation (5.23) is $\frac{1}{6}2^{-L-2}$, while the volume of the D-NMDT relaxation (5.25) is $\frac{1}{6}2^{-2L-2}$.*

Proof. We begin by noting that the NMDT and D-NMDT relaxations yield piecewise McCormick relaxations over a uniformly-spaced grid, where each grid piece $[\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}]$ corresponds to some fixed integer solution $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \{0,1\}^L$, $\Delta_x^L, \Delta_y^L \in [0, 2^{-L}]$. Thus, letting $l_x = \overline{x} - \underline{x}$ and $l_y = \overline{y} - \underline{y}$, we have that the volume of the D-NMDT and NMDT relaxations over each grid piece is the volume of the McCormick envelope over the grid piece, which is given by $\frac{1}{6}{l_x}^2 {l_y}^2$ (see e.g. [176]).

Formulation	# Binary Variables	# Constraints	Error Bound	Volume
NMDT	nL	$n(\frac{1}{2}(5n+7)+2(n+1)L)$	2^{-L-2}	$\frac{1}{6}2^{-L}$
D-NMDT	nL	$n(\frac{1}{2}(5n+5)+4nL)$	2^{-2L-2}	$\frac{1}{6}2^{-2L}$
HybU	nL	$n(\frac{1}{2}(5n-3)+2n(L+L_1))$	2^{-2L-2}	$\frac{1}{3}2^{-2L}$
Bin2	$nL(\frac{n+1}{2})$	$n(\frac{1}{2}(3n-1)+(n+1)(L+L_1))$	2^{-2L-1}	$\frac{1}{2}2^{-2L}$
Bin3	$nL(\frac{n+1}{2})$	$n(\frac{1}{2}(3n-1)+(n+1)(L+L_1))$	2^{-2L-1}	$\frac{1}{2}2^{-2L}$

Table 5.5.1: Summarizing formulation characteristics. Binary variables and constraints are given in the worst-case, in which every possible quadratic term must be modeled, for example if some Q matrix is dense. Error bounds and volumes are given for $L_1 \to \infty$, and ignore the McCormick envelopes added to HybU, Bin2, and Bin3. Finally, the volumes for Bin2 and Bin3 apply only to $L \geq 1$; those volumes are $\frac{7}{12}$ for $L = 0$. Finite L_1 leads to increased error bounds for the Bin2, Bin3, and HybU methods.

Now, for the NMDT relaxation, we have 2^L grid pieces with $l_y = 1$ and $l_x = 2^{-L}$, yielding a volume per grid piece $\frac{1}{6}2^{-2L}$ and thus a total volume of $\frac{1}{6}2^{-L}$. Similarly, for the D-NMDT relaxation, we have 2^{2L} grid pieces with $l_y = l_x = 2^{-L}$, yielding a volume per grid piece $\frac{1}{6}2^{-4L}$ and thus a total volume of $\frac{1}{6}2^{-2L}$. □

We summarize the key results for tightness, volumes, and approximation errors in the remark below, and in Table 5.5.1 and Table 5.5.2.

Remark 5.27 (Tightness of MIP Relaxations)**.** For pure quadratic terms, the strengthened sawtooth relaxation (5.17), and the formulations that employ it, are the tightest among the formulations: they are equivalent in upper-bound, with a strengthened lower-bound, compared to univariate NMDT and D-NMDT. For bivariate terms, D-NMDT is the tightest formulation, as it is yields the convex hull of the graph of $z = xy$ on each grid piece $[k^x 2^{-L}, (k^x+1)2^{-L}] \times [k^y 2^{-L}, (k^y+1)2^{-L}]$, $k^x, k^y \in [\![0, 2^L - 1]\!]$. Combining these facts, TD-NMDT is the tightest relaxation presented for the full MIQCQP.

Formulation	Error Bound	Sharpness
SSR	2^{-2L-2}	Hereditarily Sharp
SER	2^{-2L-4}	Hereditarily Sharp
NMDT	$2^{-L-2} - \varepsilon_L$	None
D-NMDT	2^{-2L-2}	None

Table 5.5.2: Summarizing formulation characteristics for quadratic terms. Error bounds given for $L_1 \to \infty$. For the NMDT maximum error, $L \geq 1$, we have $\varepsilon_L \in [\frac{4}{9}2^{-3L-2}, 2^{-3L-2}]$.

5.5.3 Optimal breakpoint placement

In this section, we discuss the optimal placement of breakpoints for the underlying discretization in each formulation. We show that in all formulations an equidistant placement is optimal, in the sense that the volume of the projected MIP relaxation for $z = xy$ is minimal. For $y = x^2$ and the sawtooth formulation this has already been shown in Proposition 4.20.

Proposition 5.28. *Let $P^{\text{IP}}_{(n,m),\infty}$ be a formulation equivalent to the HybU MIP formulation with $L_1 = \infty$ on some rectangular grid in (x, y), where we can choose n and m linear segments to linearize x^2 and y^2, respectively. Then the volume-optimal placement of breakpoints is uniform, with an optimal volume for $\text{proj}_{(x,y,z)}(P^{\text{IP}}_{n,m,\infty})$ of $\frac{1}{6}(\frac{1}{n^2} + \frac{1}{m^2})$.*

Proof. Let $L_1 = \infty$. From the proof of Proposition 5.24, we know that the volume of each grid piece of $\text{proj}_{(x,y,z)}(P^{\text{IP}})$ is $\frac{1}{6}(l_x {l_y}^3 + l_y {l_x}^3)$. Thus, if we assume a grid of n x-steps and m y-steps, the optimal placement of the gridpoints results from solving the following optimization problem

$$\begin{aligned} \min \quad & \tfrac{1}{6}\textstyle\sum_{i=1}^{n}\sum_{j=1}^{m}(l_{xi}{l_{y_j}}^3 + l_{y_j}{l_{xi}}^3) \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{n} l_{xi} = 1 \\ & \textstyle\sum_{j=1}^{m} l_{y_j} = 1 \\ & l_{xi} \geq 0 \qquad\qquad i \in 1, \ldots, n \\ & l_{y_j} \geq 0 \qquad\qquad j \in 1, \ldots, m. \end{aligned} \tag{5.29}$$

We can rewrite the objective function in (5.29):

$$\begin{aligned}\tfrac{1}{6}\sum_{i=1}^{n}\sum_{j=1}^{m}(l_{xi}l_{y_j}^{\,3}+l_{y_j}l_{xi}^{\,3}) &= \tfrac{1}{6}(\sum_{i=1}^{n}\sum_{j=1}^{m}(l_{xi}l_{y_j}^{\,3})+\sum_{i=1}^{n}\sum_{j=1}^{m}(l_{y_j}l_{xi}^{\,3}))\\ \tfrac{1}{6}(\sum_{i=1}^{n}l_{xi}\sum_{j=1}^{m}l_{y_j}^{\,3}+\sum_{j=1}^{m}l_{y_j}\sum_{i=1}^{n}l_{xi}^{\,3}) &= \tfrac{1}{6}(1\cdot\sum_{j=1}^{m}l_{y_j}^{\,3}+1\cdot\sum_{i=1}^{n}l_{xi}^{\,3})\\ &= \tfrac{1}{6}\sum_{j=1}^{m}l_{y_j}^{\,3}+\tfrac{1}{6}\sum_{i=1}^{n}l_{xi}^{\,3}\end{aligned}$$

Thus, (5.29) decomposes into two independent problems where the optimal solutions are also optimal for (5.29):

$$\begin{aligned}\min\quad & \textstyle\sum_{i=1}^{n} l_{xi}^{\,3}\\ \text{s.t.}\quad & \textstyle\sum_{i=1}^{n} l_{xi} = 1\\ & l_{xi}\geq 0 \qquad i\in 1,\dots,n\end{aligned}\tag{5.30}$$

and

$$\begin{aligned}\min\quad & \textstyle\sum_{j=1}^{m} l_{y_i}^{\,3}\\ \text{s.t.}\quad & \textstyle\sum_{j=1}^{m} l_{y_i} = 1\\ & l_{y_i}\geq 0 \qquad j\in 1,\dots,m.\end{aligned}\tag{5.31}$$

Since all constraints in (5.30) are linear and the objective function is convex over the positive orthand, we know that the KKT conditions are sufficient. Moreover, these are exactly the sawtooth-area optimization problems from Proposition 4.20. The KKT conditions for l_x are given as:

$$\begin{aligned}\textstyle\sum_{i=1}^{n} l_{xi} &= 1\\ l_{xi} &\geq 0 \quad i\in 1,\dots,n\\ 3l_{xi}^{\,2}-v_i &= u_1 \quad i\in 1,\dots,n\\ \textstyle\sum_{i=1}^{n} v_i l_{xi} &= 0\\ v &\geq 0.\end{aligned}\tag{5.32}$$

From the KKT system it follows directly that an uniform placement of the breakpoints where each $l_{xi}=\frac{1}{n}$ is optimal for (5.30). With the same reasoning we get $l_{y_j}=\frac{1}{m}$ for (5.31). Consequently, a

uniform placement of grid points is optimal for (5.29) and the total volume is $\frac{1}{6}\frac{1}{m^2} + \frac{1}{n^2} = \frac{1}{6nm}(\frac{n}{m}+\frac{m}{n})$. Note that, if $n = m = 2^L$ as with the HybU formulation, then this volume becomes $\frac{1}{3}2^{-2L}$, as in Proposition 5.24. □

Proposition 5.29. *In NMDT and D-NMDT the volume-optimal placement of breakpoints is uniform. The volume of the NMDT formulation is $\frac{1}{6}2^{-L}$, while the volume of NMDT approach is $\frac{1}{6}2^{-2L}$.*

Proof. We assume that we discretize along the x-axis by n pieces and by m pieces along the y-axis, then apply a McCormick relaxation on each grid piece. The optimal breakpoint problem for D-NMDT can then be stated as

$$\begin{aligned} \tfrac{1}{6}\min \quad & \textstyle\sum_{i=1}^{n}\sum_{j=1}^{m} l_{xi}^2 l_{yj}^2 & \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{n} l_{xi} = 1 & \\ & \textstyle\sum_{j=1}^{m} l_{yj} = 1 & \\ & l_{xi} \geq 0 & i \in 1,\dots,n \\ & l_{yj} \geq 0 & j \in 1,\dots,m. \end{aligned} \tag{5.33}$$

Again, we rewrite the objective function

$$\begin{aligned} \tfrac{1}{6}\min \quad & (\textstyle\sum_{i=1}^{n} l_{xi}^2)\cdot(\sum_{j=1}^{m} l_{yj}^2) & \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{n} l_{xi} = 1 & \\ & \textstyle\sum_{j=1}^{m} l_{yj} = 1 & \\ & l_{xi} \geq 0 & i \in 1,\dots,n \\ & l_{yj} \geq 0 & j \in 1,\dots,m. \end{aligned} \tag{5.34}$$

and see that the problem can be decomposed into two independent convex subproblems:

$$\begin{aligned} \tfrac{1}{6}\min \quad & \textstyle\sum_{i=1}^{n} l_{xi}^2 \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{n} l_{xi} = 1 \\ & l_{xi} \geq 0 \qquad i \in 1, \dots, n \end{aligned} \tag{5.35}$$

$$\begin{aligned} \tfrac{1}{6}\min \quad & \textstyle\sum_{j=1}^{m} l_{y_j}^2 \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{m} l_{y_j} = 1 \\ & l_{y_j} \geq 0 \qquad j \in 1, \dots, m. \end{aligned} \tag{5.36}$$

Applying the KKT conditions it follows directly that an uniform placement of the breakpoints with $l_{xi} = \frac{1}{n}$ and $l_{y_j} = \frac{1}{m}$ is optimal for (5.33). The total optimal volume is then $\frac{1}{6nm}$. Then, for D-NMDT, we have $n = m = 2^L$, so that the optimal volume becomes $\frac{1}{6}2^{-2L}$. Since NMDT equals D-NMDT without a discretization of the y-axis, the same proof holds with $m = 1$ and we obtain a volume of $\frac{1}{6n} = \frac{1}{6}2^{-L}$ in the base-2 NMDT approach. □

5.5.4 Hereditary sharpness of sawtooth relaxation

In this section, we prove hereditary sharpness of the strengthened sawtooth relaxation (5.17). Before we begin the proof, we first introduce our driving notation, then give some key results from Section 4.4.1, along with some other helpful results. For convenience, to avoid variable redundancy throughout this section with $g_0 = x$, we will use shorthand $\boldsymbol{g} = \boldsymbol{g}_{[\![1,L]\!]}$.

Let $L_1 \geq L \geq 0$, and let P^{IP}_{L,L_1} be the strengthened sawtooth relaxation (5.17), so that $\text{proj}_{x,y}(P^{\text{IP}}_{L,L_1}) = R_{L,L_1}$. For shorthand, define $P^{\text{IP}} = P^{\text{IP}}_{L,L_1}$. Let I be the index set of binary variables to fix to values $\hat{\boldsymbol{\alpha}} \in \{0,1\}^{|I|}$. We wish to show that the strengthened sawtooth relaxation in (5.17) is sharp under the restriction $\boldsymbol{\alpha}_I = \hat{\boldsymbol{\alpha}}$, where we use the shorthand $\boldsymbol{\alpha}_I = [\alpha_{i_1}, \dots, \alpha_{i_{|I|}}]^\top$. That is, letting P^{LP} be the LP relaxation of P^{IP}, so that each α_i variable is relaxed to the interval $[0,1]$, we wish to show

that $P^{\text{IP}}|_{\boldsymbol{\alpha}_I=\hat{\boldsymbol{\alpha}}}$ is sharp, i.e.,

$$\operatorname{conv}(\operatorname{proj}_{x,y} P^{\text{IP}}|_{\boldsymbol{\alpha}_I=\hat{\boldsymbol{\alpha}}}) = \operatorname{proj}_{x,y}(P^{\text{LP}}|_{\boldsymbol{\alpha}_I=\hat{\boldsymbol{\alpha}}}).$$

To this end, we first split P^{IP} into two sets, P^{IP}_+ and P^{IP}_-, encapsulating the upper and lower bounds w.r.t. y, respectively. For ease of notation, we define the underlying set D for $(x, \boldsymbol{g}, \boldsymbol{\alpha})$ with the given bit fixing, and its LP relaxation, as

$$\begin{aligned} D &:= \{(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1] \times [0,1]^{L_1} \times \{0,1\}^L : \boldsymbol{\alpha}_I = \hat{\boldsymbol{\alpha}}, (5.18a), (5.18b)\} \\ D^{\text{LP}} &:= \{(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1] \times [0,1]^{L_1} \times [0,1]^L : \boldsymbol{\alpha}_I = \hat{\boldsymbol{\alpha}}, (5.18a), (5.18b)\} \end{aligned} \tag{5.37}$$

For convenience, define $X = \operatorname{proj}_x(D)$ and $X^{\text{LP}} = \operatorname{proj}_x(D^{\text{LP}})$. We note that $X^{\text{LP}} = \operatorname{conv}(X)$, which is shown as a corollary of Lemma 5.32. The epigraph and hypograph relaxations P^{IP}_- and P^{IP}_+ can then be expressed as

$$\begin{aligned} P^{\text{IP}}_+ &= \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) : (x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D,\ y \in \mathbb{R},\ (5.18c)\} \\ P^{\text{IP}}_- &= \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) : (x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D,\ y \in \mathbb{R},\ (5.18d), (5.18e)\}. \end{aligned} \tag{5.38}$$

In this way, we have that $P^{\text{IP}}|_{\boldsymbol{\alpha}_I=\hat{\boldsymbol{\alpha}}} = P^{\text{IP}}_- \cap P^{\text{IP}}_+$, and that the relaxation P^{IP} is sharp if and only if both the epigraph relaxation P^{IP}_- and the hypograph relaxation P^{IP}_+ are sharp. Now, the sharpness of P^{IP}_+ holds directly from Theorem 4.14: The theorem establishes hereditary sharpness of the sawtooth approximation (5.11), which has the same upper-bounding constraints on y as (5.17). Thus, we focus on P^{IP}_-.

Now, to establish sharpness of P^{IP}_-, we construct a separate epigraph set P^{IP}_j for each lower-bounding constraint on y in P^{IP}_-, along with their LP relaxations P^{LP}_j. For convenience, we define

the lower-bounding functions $\ell^j(x, \boldsymbol{g})$ as

$$\begin{aligned} \ell^j(x, \boldsymbol{g}) &:= F^j(x, \boldsymbol{g}) - 2^{-2j-2} \qquad j = 0, \dots, L_1, \\ \ell^{-1}(x, \boldsymbol{g}) &:= 2x - 1 \\ \ell^{-2}(x, \boldsymbol{g}) &:= 0 \end{aligned} \tag{5.39}$$

Further, for $j \geq 0$ define $\ell^j(x) := F^j(x) - 2^{-2j-2}$ as the jth piecewise linear underestimator to $y = x^2$ in the construction of the sawtooth relaxation, as defined in Section 5.3.2. Note that, by construction in [110], each $F^j(x)$ is incident to $y = x^2$ at exactly the points $\frac{k}{2^j}$, $k \in [\![0, 2^j]\!]$, and is linear between these points.

Then we define each P_j^{IP} and its LP relaxation P_j^{LP} as

$$\begin{aligned} P_j^{\text{IP}} &= \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) : (x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D,\ y \in \mathbb{R}, y \geq \ell^j(x, \boldsymbol{g})\} \qquad j = -2, \dots, L_1 \\ P_j^{\text{LP}} &= \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) : (x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D^{\text{LP}},\ y \in \mathbb{R}, y \geq \ell^j(x, \boldsymbol{g})\} \qquad j = -2, \dots, L_1. \end{aligned} \tag{5.40}$$

Then we have $P_-^{\text{IP}} = \bigcap_{j=-2}^{L_1} P_j^{\text{IP}}$, and equivalently,

$$P_-^{\text{IP}} = \{(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) : (x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D,\ y \in \mathbb{R}, y \geq \max_{j \in -2, \dots, L_1} \ell^j(x, \boldsymbol{g})\}.$$

Similarly, define $P_-^{\text{LP}} := \bigcap_{j=-2}^{L_1} P_j^{\text{LP}}$. Then P_-^{LP} is the LP-relaxation of P_-^{IP}, as relaxing a binary MIP formulation corresponds with relaxing the underlying alpha-variables to the interval $[0, 1]$ (the order of constraint intersection and LP-relaxation does not matter). We are left to prove that P_-^{LP} is sharp, i.e.

$$\text{proj}_{x,y}(P_-^{\text{LP}}) = \text{conv}(\text{proj}_{x,y}(P_-^{\text{IP}})).$$

To assist with our proof, we observe the following fact. Suppose we have for some $\hat{x} \in [0, 1]$ that $\hat{x}^2 = \hat{x} - \Delta$. Then we have

$$\hat{x}^2 = \hat{x} - \Delta \Rightarrow (1 - \hat{x})^2 = \hat{x}^2 - 2\hat{x} + 1 = \hat{x} - \Delta\hat{x} - 2\hat{x} + 1 = (1 - \hat{x}) - \Delta \tag{5.41}$$

We use this fact to prove the following facts about reflections about $x = \frac{1}{2}$ in P_-^{IP} and P_j^{IP}.

Lemma 5.30. *Let $L \in \mathbb{Z}_+$, let $\hat{x} \in X$, and suppose $1 \notin I$, so that α_1 is not fixed. Then*

$$\text{proj}_{\boldsymbol{g},\boldsymbol{\alpha}_{[\![2,L]\!]}}(D|_{x=\hat{x}}) = \text{proj}_{\boldsymbol{g},\boldsymbol{\alpha}_{[\![2,L]\!]}}(D|_{x=1-\hat{x}}). \tag{5.42}$$

That is, the fibers $D|_{x=\hat{x}}$ and $D|_{x=1-\hat{x}}$ coincide after ignoring α_1. Furthermore,

$$\hat{x}^2 - \ell^j(\hat{x}, \boldsymbol{g}^*) = (1-\hat{x})^2 - \ell^j(1-\hat{x}, \boldsymbol{g}^*) \qquad \text{for all } j \in [\![0, L_1]\!]. \tag{5.43}$$

That is, the relaxation errors from the lower bounds coincide. Similarly,

$$\hat{x}^2 - \ell^{-2}(\hat{x}, \boldsymbol{g}^*) = (1-\hat{x})^2 - \ell^{-1}(1-\hat{x}, \boldsymbol{g}^*) \tag{5.44}$$

$$\hat{x}^2 - \ell^{-1}(\hat{x}, \boldsymbol{g}^*) = (1-\hat{x})^2 - \ell^{-2}(1-\hat{x}, \boldsymbol{g}^*) \tag{5.45}$$

Proof. To begin, let $\hat{x} \in X$. Now, recall that D is formed from the constraints in S^L and T^{L_1}, along with the bit fixing $\boldsymbol{\alpha}_I = \hat{\boldsymbol{\alpha}}$. It is easy to check that $(\hat{x}, \boldsymbol{g}, \boldsymbol{\alpha}) \in D$ if an only if $(1-\hat{x}, \boldsymbol{g}, \bar{\boldsymbol{\alpha}}) \in D$ where $\bar{\alpha}_1 := 1 - \alpha_1$ and $\bar{\alpha}_i := \alpha_i$ otherwise. Thus, (5.42) holds due this correspondence.

For $j \in [\![0, L_1]\!]$,

$$\begin{aligned}
\hat{x}^2 - \ell^j(\hat{x}, \boldsymbol{g}^*) &= \hat{x}^2 - \left(\hat{x} - \sum_{i=1}^{j} 2^{-2i}\boldsymbol{g}_i^* - 2^{-2j-2}\right) \\
&= (1-2\hat{x}) + \hat{x}^2 - \left((1-2\hat{x}) + \hat{x} - \sum_{i=1}^{j} 2^{-2i}\boldsymbol{g}_i^* - 2^{-2j-2}\right) \\
&= (1-\hat{x})^2 - \left(1 - \hat{x} - \sum_{i=1}^{j} 2^{-2i}\boldsymbol{g}_i^* - 2^{-2j-2}\right) \\
&= (1-\hat{x})^2 - \ell^j(1-\hat{x}, \boldsymbol{g}^*).
\end{aligned}$$

Similarly,

$$\begin{aligned}\hat{x}^2 - \ell^{-1}(\hat{x}, \boldsymbol{g}^*) &= \hat{x}^2 - (2\hat{x} - 1) \\ &= (1 - \hat{x})^2 \\ &= (1 - \hat{x})^2 - \ell^{-2}(1 - \hat{x}, \boldsymbol{g}^*).\end{aligned}$$

The last equation in the lemma statement follows from the reverse calculation of above. □

Remark 5.31. As a direct corollary to (5.11), the same secondary result holds if ℓ^j is replaced with $\ell = \max_{j \in [\![-2, L]\!]} \ell^j$: since each constituting function (or the pair $j = -1, j = -2$) is symmetric about $x = \frac{1}{2}$ w.r.t. approximation error, the pointwise maximum over the functions retains the same symmetry. Further, we can extend this result on to $\ell(x)$ on D by minimizing w.r.t. $\boldsymbol{g}$, as both the roles and feasible regions of $\boldsymbol{g}$ are identical at $\hat{x}$ as at $1 - \hat{x}$. Similarly, the same result holds if $I = \emptyset$, so that $X = [0, 1]$.

Now, to assist in establishing sharpness for P^{IP}_-, we first introduce some important results from Section 4.4.1. These results establish bounds on each g_i variable within in D^{LP} and establish a closed-form optimal solution for $\boldsymbol{g}$ when minimizing y within P^{IP}_- or any P^{IP}_j. The bounds on each g_i are established in the lemma below.

Lemma 5.32 (Bounds in Projection; Lemma 4.9)**.** *For all $i \in [\![0, L]\!]$, we have* $\text{proj}_{g_i}(D^{\text{LP}}) = \text{conv}(\text{proj}_{g_i}(D)) =: [a_i, b_i] \neq \emptyset$. *Furthermore, $[a_L, b_L] = [0, 1]$, and $[a_{i-1}, b_{i-1}]$ can be computed from $[a_i, b_i]$ as*

$$[a_{i-1}, b_{i-1}] = \begin{cases} [\frac{1}{2}a_i, \frac{1}{2}b_i] & \text{if } i \in I \text{ and } \bar{\alpha}_i = 0, \\ [1 - \frac{1}{2}b_i, 1 - \frac{1}{2}a_i] & \text{if } i \in I \text{ and } \bar{\alpha}_i = 1, \\ [\frac{1}{2}a_i, 1 - \frac{1}{2}a_i] & \text{if } i \notin I. \end{cases} \tag{5.46}$$

Note that in the last case $a_{i-1} \leq \frac{1}{2}$ and $b_{i-1} \geq \frac{1}{2}$.

Note that Lemma 5.32 with $i = 0$ yields $X^{\text{LP}} = \text{conv}(X)$, via

$$X^{\text{LP}} = \text{proj}_x(D^{\text{LP}}) = \text{conv}(\text{proj}_x(D)) = \text{conv}(X).$$

Next, we generalize Lemma 4.11, which establishes that, when minimizing or maximizing y within the LP-relaxation of $P^{\text{IP}}_{L,L}|_{\boldsymbol{\alpha}_I = \hat{\boldsymbol{\alpha}}}$ given a fixed value for x, each g_i can be directly computed from g_{i-1} and the bounds established in Lemma 5.32. We first define

$$u^i(g_{i-1}) := \min\{b_i, 2g_{i-1}, 1 - 2g_{i-1}\}.$$

We then introduce the modified result below.

Lemma 5.33 (Adapted from Lemma 4.11)**.** *Let a_i, b_i be defined as in Lemma 5.32, and let $\hat{x} \in [a_0, b_0]$. Define $\boldsymbol{g}^*$ as*

$$\begin{aligned} g^*_0 &= \hat{x} \\ g^*_i &= u^i(g^*_{i-1}) \quad i \in [\![L_1]\!] \setminus I \\ g^*_i &= G^i(g_{i-1}) \qquad\quad i \in I \end{aligned}$$

where, for $i \in I$, $G^i(g_{i-1}) = 2g_{i-1}$ if α_i is fixed to 0 and $G^i(g_{i=1}) = 2(1 - g_{i-1})$ otherwise. Then we have

$$\boldsymbol{g}^* \in \arg\min\{y \,:\, (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P^{\text{LP}}_-|_{x=\hat{x}})\}, \tag{5.47a}$$

$$\boldsymbol{g}^* \in \arg\min\{y \,:\, (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P^{\text{LP}}_j|_{x=\hat{x}})\} \quad \forall j \in [\![-2, L_1]\!] \tag{5.47b}$$

That is, each g_i with unfixed α_i can take on one of its upper-bounds w.r.t. g_{i-1} when minimizing y within $P^{\text{LP}}_-|_{x=\hat{x}}$ and $P^{\text{LP}}_j|_{x=\hat{x}}$. Furthermore, this choice is unique for all $i \leq j$, i.e.

$$\left|\arg\min\{y \,:\, (y, \boldsymbol{g}_{[\![j]\!]}) \in \text{proj}_{y,\boldsymbol{g}_{[\![j]\!]}}(P^{\text{LP}}_j)|_{x=\hat{x}})\}\right| = 1.$$

Finally, there exists some $j \in [\![-2, L_1]\!]$ *for which*

$$\ell^j(\hat{x}, \boldsymbol{g}^*) = \min\{y \,:\, (y, \boldsymbol{g}) \in \operatorname{proj}_{y,\boldsymbol{g}}(P_-^{\mathrm{LP}})|_{x=\hat{x}})\}. \tag{5.48}$$

Proof. The proof of the optimality result for $j \geq 1$ follows the structure of the proof of Theorem 5.18, with the same concluding math as the proof of Lemma 4.11. Thus, the details are omitted here. To establish the result for $j \leq 0$, we observe that, for $j \leq 0$, ℓ^j is purely a function of x, so that the choice of $\boldsymbol{g}$ has no effect on ℓ^j, and $\boldsymbol{g}^*$ is thus still optimal.

To establish the uniqueness result, we observe for any j that, by same proof, any perturbation of some g_i, $1 \leq i \leq j$, will force an increase in y. Finally, to show (5.48), we have for some $\hat{j}$ that

$$\begin{aligned}\min\{y \,:\, (y, \boldsymbol{g}) \in \operatorname{proj}_{y,\boldsymbol{g}}(P_-^{\mathrm{LP}})|_{x=\hat{x}})\} &\leq \max_{j \in [\![-2, L_1]\!]} \ell^j(\hat{x}, \boldsymbol{g}^*) = \ell^{\hat{j}}(\hat{x}, \boldsymbol{g}^*) \\ = \min\{y \,:\, (y, \boldsymbol{g}) \in \operatorname{proj}_{y,\boldsymbol{g}}(P_{\hat{j}}^{\mathrm{LP}})|_{x=\hat{x}})\} &\leq \min\{y \,:\, (y, \boldsymbol{g}) \in \operatorname{proj}_{y,\boldsymbol{g}}(P_-^{\mathrm{LP}})|_{x=\hat{x}})\}\end{aligned}$$

as required. □

Now, to formalize the convex hull of convex functions over gaps, we introduce the following lemma. This lemma is proven in Appendix 5.B. We denote the boundary of the set X by ∂X.

Lemma 5.34. *Let* $X \subseteq \mathbb{R}$ *be closed and bounded, and let* $f \colon \operatorname{conv}(X) \to \mathbb{R}$ *be a convex function. For any* $\bar{x} \in \operatorname{conv}(X) \setminus X$*, define*

$$\bar{x}_- = \max\{x \in X : x < \bar{x}\}, \quad \text{and} \quad \bar{x}_+ = \min\{x \in X : x > \bar{x}\}.$$

Now define $g\colon \operatorname{conv}(X) \to \mathbb{R}$ *as*

$$g(x) := \begin{cases} f(x) & x \in X, \\ \lambda f(x_-) + (1-\lambda) f(x_+) & x \notin X, \text{ and} \\ & x = \lambda x_- + (1-\lambda)x_+, \lambda \in (0,1). \end{cases} \tag{5.49}$$

Then

$$\operatorname{conv}(\operatorname{epi}_X(f)) = \operatorname{epi}_{\operatorname{conv}(X)}(g).$$

We are now ready to prove hereditary sharpness of the strengthened sawtooth relaxation (5.17).

Theorem 5.35. *The sawtooth formulation* P^{IP}_{L,L_1} (5.6) *is hereditarily sharp.*

Proof. As in the preceding discussion, we begin by choosing a fixing $I, \hat{\boldsymbol{\alpha}}$, and noting that P^{IP}_+ is sharp by Theorem 4.14. We wish to show that P^{IP}_- is sharp. We prove this by induction on L and L_1, first incrementing L_1, then incrementing L and L_1 jointly. Since we need only consider P^{IP}_-, to assist with notation, we will henceforth redefine $P^{\text{IP}}_{L,L_1} := P^{\text{IP}}_-$; that is, we redefine P^{IP}_{L,L_1} to remove the upper-bounding constraint on y.

Base case

For the base case, let $L = L_1 = 0$. Then there are no binary variables and hence, nothing to branch on, so the result holds trivially.

Induction on L_1 For the first inductive step, suppose P^{IP}_{L,L_1-1} is hereditarily sharp. To construct P^{IP}_{L,L_1} from P^{IP}_{L,L_1-1}, we simply maintain the same fixing $I, \hat{\boldsymbol{\alpha}}$, then add on a new variable $g_{L_1} \geq 0$, with the new constraints

$$g_{L_1} \leq 2g_{L_1-1}, \quad g_{L_1} \leq 2(1 - g_{L_1-1}), \qquad \text{(from (5.18b) via (5.14))}$$

$$y \geq x - \sum_{i=1}^{L_1} 2^{-2i} g_i - 2^{-2L_1-2}. \qquad \text{(from (5.18d))}$$

We then note three facts.

1) We have simply added a new lower-bound on y, without modifying the feasible space for $(x, \boldsymbol{g}_{[\![L_1-1]\!]})$. Thus, within the LP-relaxation, the new line between boundary points can never lie strictly *below* the previous one (since we've only added a new constraint).

2) As the lower-bounds at any gap endpoints x_1, x_2 are incident with x^2, the new constraint, which is an under-estimator for x^2, has no impact at those endpoints.

3) By 2), and due to the convexity of $\ell(x)$ in the (x, y)-space, we have by Lemma 5.34 that the convex hull of the projection is still the same straight line across the gap.

Thus, the convex hull remains unchanged across the gap, and since the LP-relaxation does not weaken, sharpness in lower-bound is maintained. Thus, we conclude that P^{IP}_{L,L_1} is sharp.

Joint Induction on L, L_1

For the second inductive step, we suppose $\tilde{P}^{\text{IP}}_{-} := P^{\text{IP}}_{L-1,L-1}$ is hereditarily sharp for all possible bit fixings, and show that $P^{\text{IP}}_{-} := P^{\text{IP}}_{L,L}$ is hereditarily sharp. Without loss of generality, we can assume $L_1 = L$, as we can reach any larger L_1 by repeatedly applying the previous inductive step.

Problem simplification

We wish to show that $\text{conv}(\text{proj}_{x,y}(P^{\text{IP}}_{-})) = \text{proj}_{x,y}(P^{\text{LP}}_{-})$. Now, since $\text{proj}_{x,y}(P^{\text{IP}}_{-}) = \text{epi}_X(\ell(x))$, and by Lemma 5.34, we have by convexity of $\ell(x)$ that $\text{conv}(\text{epi}_X(\ell(x))) = \text{epi}_{\text{conv}\, X}(\underline{\ell}(x))$, where $\underline{\ell}(x) = \ell(x)$ if $x \in X$, and otherwise, $\underline{\ell}(x)$ is on the line between $(x_-, \underline{\ell}(x_-))$ and $(x_+, \underline{\ell}(x_+))$, where x_- and x_+ are the closest points x_-, x_+ to x in X with $x_- < x < x_+$. Thus, we have only to show that $\text{proj}_{x,y}(P^{\text{LP}}_{-}) = \text{epi}_X(\underline{\ell}(x))$.

Furthermore, by the sharpness of the P^{IP} before branching (Theorem 5.18), we have that the LP lower-bounds on y are incident with the MIP lower-bounds for MIP-feasible points $x \in X$, so that we have $\text{proj}_{x,y}(P^{\text{LP}}_{-})|_{x\in X} = \text{epi}_X(\underline{\ell}(x))|_{x\in X}$. Thus, we have only to show that $\text{proj}_{x,y}(P^{\text{LP}}_{-})|_{x\in \text{conv}(X)\setminus X} =$

$\text{epi}_X(\underline{\ell}(x))|_{x\in\text{conv}(X)\setminus X}$. Equivalently, let fixed $\hat{x} \in \text{conv}(X) \setminus X$ and $y^*(\hat{x}) = \min_{\boldsymbol{g}\in P_-^{\text{LP}}|_{x=\hat{x}}}(\ell(\hat{x}, \boldsymbol{g}))$. Then we show that $y^*(\hat{x}) = \underline{\ell}(\hat{x})$, or equivalently, that $(\hat{x}, y^*)$ lies on the line between the two endpoints across the gap containing $\hat{x}$, $(x_-, \ell(x_-))$ and $(x_+, \ell(x_+))$, where $x_-, x_+ \in X$ and $(x_-, x_+) \cap X = \emptyset$.

Map Φ for $1 \notin I$

To establish this fact, suppose $1 \notin I$, and define $\tilde{P}_-^{\text{IP}} = P_{L-1,L-1}^{\text{IP}}$ so that $\tilde{\hat{\boldsymbol{\alpha}}} = \hat{\boldsymbol{\alpha}}$ and $\tilde{I} = I - 1 := \{i - 1 : i \in I\}$; i.e. the bit fixings are the same, but with indices decremented by 1. Now, define the map $\Phi\colon \tilde{P}_-^{\text{LP}} \to P_-^{\text{LP}}|_{x\in\text{conv}(X\cap[0,1/2])}$, so that for $(\tilde{x}, \tilde{y}, \tilde{\boldsymbol{g}}, \tilde{\boldsymbol{\alpha}}) \in \tilde{P}_-^{\text{LP}}$, $(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) = \Phi(\tilde{x}, \tilde{y}, \tilde{\boldsymbol{g}}, \tilde{\boldsymbol{\alpha}})$ is defined via

$$\begin{aligned} x &= \tfrac{\tilde{x}}{2}, \quad y = \tfrac{\tilde{y}}{4}, \\ g_1 &= \tilde{x}, \quad \boldsymbol{g}_{[\![2,L]\!]} = \tilde{\boldsymbol{g}}, \\ \alpha_1 &= x, \quad \boldsymbol{\alpha}_{[\![2,L]\!]} = \tilde{\boldsymbol{\alpha}}. \end{aligned} \tag{5.51}$$

For convenience, under the definitions above we write $x = \Phi(\tilde{x})$, $y = \Phi(\tilde{y})$, $\boldsymbol{g} = \Phi(\tilde{\boldsymbol{g}})$, and $\boldsymbol{\alpha} = \phi(\tilde{\boldsymbol{\alpha}})$, and noting that $g_0 = x$ and $\tilde{g}_0 = \tilde{x}$. We first show that the mapped values are feasible in $P_-^{\text{LP}}|_{x\in\text{conv}(X\cap[0,1/2])}$, so that Φ maps onto the proper set. Then, to demonstrate sharpness of P_-^{LP} on $\Phi(\tilde{D}^{\text{LP}})$, we show two facts:

1) Let $\tilde{x} \in \tilde{X}^{\text{LP}}$ and $\tilde{y}^* \in \arg\min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(\tilde{P}_-^{\text{LP}}|_{x=\tilde{x}})\}$ with the corresponding solution $\tilde{\boldsymbol{g}^*}$ defined in Lemma 5.33. Then $(\frac{1}{4}\tilde{y}^*, \Phi(\tilde{g}^*)) \in \arg\min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P_-^{\text{LP}}|_{x=\Phi(\tilde{x})})\}$.

2) $\tilde{y} = \tilde{\underline{\ell}}(\tilde{x}) \Leftrightarrow \Phi(\tilde{y}) = \underline{\ell}(\Phi(\tilde{x}))$, so that $\underline{\ell}(\Phi(\tilde{x})) = 4\tilde{\underline{\ell}}(\tilde{x})$

By the sharpness of $\tilde{P}_-^{\text{IP}}$, these facts then imply that

$$\underline{\ell}(\Phi(\tilde{x})) = 4\tilde{\underline{\ell}}(\tilde{x}) = 4 \min_{\boldsymbol{g}\in\tilde{P}_-^{\text{LP}}|_{x=\tilde{x}}} (\ell(\hat{x}, \boldsymbol{g})) = \min_{\boldsymbol{g}\in P_-^{\text{LP}}|_{x=\Phi(\tilde{x})}} (\ell(\hat{x}, \boldsymbol{g})),$$

so that P_-^{LP} is sharp for $x \in \Phi(\tilde{D}^{\text{LP}})$.

Proof that $\Phi\colon \tilde{P}^{\text{LP}}_{-} \to P^{\text{LP}}_{-}|_{x\in\text{conv}(X\cap[0,1/2])}$

To begin, let $(\tilde{x},\tilde{y},\tilde{\boldsymbol{g}},\tilde{\boldsymbol{\alpha}}) \in \tilde{P}^{\text{LP}}$ be minimal in y, and let $(x,y,\boldsymbol{g},\boldsymbol{\alpha}) = \Phi(\tilde{x},\tilde{y},\tilde{\boldsymbol{g}},\tilde{\boldsymbol{\alpha}})$. Then we have $\tilde{y} = \ell^j(\tilde{x},\tilde{\boldsymbol{g}})$ for some j.

Now, suppose $j \geq 0$. Then, noting that

$$\tfrac{1}{4}\tilde{x} = \tfrac{1}{2}\tilde{x} - \tfrac{1}{4}\tilde{x} = x - \tfrac{1}{4}g_1,$$

we have

$$\begin{aligned} y &= \Phi(\tilde{y}) \\ &= \tfrac{1}{4}(\ell^j(\tilde{x},\tilde{\boldsymbol{g}})) \\ &= \tfrac{1}{4}(\tilde{x} - \textstyle\sum_{i=1}^{j} 2^{-2i}\tilde{g}_i - 2^{-2j-2}) \\ &= x - \tfrac{1}{4}g_1 - \tfrac{1}{4}(\textstyle\sum_{i=1}^{j} 2^{-2i}\tilde{g}_i - 2^{-2j-2}) \\ &= x - \textstyle\sum_{i=1}^{j+1} 2^{-2i}g_i - 2^{-2(j+1)-2} = \ell^{j+1}(x,\boldsymbol{g}). \end{aligned}$$

Furthermore, if $j = -1$, we have

$$\begin{aligned} y &= \Phi(\tilde{y}) \\ &= \tfrac{1}{4}(G^{-1}(\tilde{x},\tilde{\boldsymbol{g}})) \\ &= \tfrac{1}{4}(2\tilde{x} - 1) \\ &= x - \tfrac{1}{4} = \ell^{j+1}(x,\boldsymbol{g}) \\ &= x - \textstyle\sum_{i=1}^{j+1} 2^{-2i}g_i - 2^{-2(j+1)-2} = \ell^{0}(x,\boldsymbol{g}), \end{aligned}$$

while finally, $j = -2$, yields

$$\begin{aligned} y &= \Phi(\tilde{y}) \\ &= \tfrac{1}{4}(G^{-1}(\tilde{x},\tilde{\boldsymbol{g}})) \\ &= 0 \\ &= \ell^{-2}(x,\boldsymbol{g}). \end{aligned}$$

Thus, we have that $\Phi(\tilde{y}) \geq \ell^j(\Phi(x),\Phi(\boldsymbol{g}))$ for all j, where the absence of $G^{-1}(x,\boldsymbol{g})$ is due to the fact that $G^{-1}(x,\boldsymbol{g}) \leq 0$ for $x \in [0,\frac{1}{2}]$, so that the bound is inactive on $\Phi(\tilde{X}^{\text{LP}})$. Note that the above

calculations also imply that, for all $\tilde{j} \in [\![-2, L-1]\!]$ and for all $(\tilde{x}, \tilde{g}) \in \text{proj}_{x,\boldsymbol{g}}(\tilde{D}^{\text{LP}})$, we have for some $j \in [\![-2, L]\!]$ that $\Phi(\ell^{\tilde{j}}(\tilde{x}, \tilde{\boldsymbol{g}})) = \ell^{j}(\Phi(\tilde{x}), \Phi(\boldsymbol{g}))$. Further, since each $\tilde{j}$ maps to a unique j (with only the inactive $j = -1$ skipped), this implies that $\Phi(\tilde{\ell}(\tilde{x}, \tilde{\boldsymbol{g}})) = \ell(\Phi(\tilde{x}), \Phi(\boldsymbol{g}))$.

Now, to establish that $\Phi(\tilde{x}, \tilde{\boldsymbol{g}}, \tilde{\boldsymbol{\alpha}}) \in \text{proj}_{x,\boldsymbol{g},\boldsymbol{\alpha}_{2,L}}(D^{\text{LP}})$, since $g_1 = \tilde{x} = 2x$, we observe that D^{LP} can be written as the set of points $(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1]^{1+L+L}$ such that:

$$
\begin{aligned}
g_0 &= x \\
g_i &= 2g_{i-1} && i = 1 \text{ or } i \in I, \hat{\alpha}_i = 0 \\
g_i &= 2(1 - g_{i-1}) && i \in I, \hat{\alpha}_i = 1 \\
|g_{i-1} - \alpha_i| &\leq g_i \leq \min(2g_{i-1}, 2(1 - g_{i-1})) && i \in [\![L]\!] \setminus I,\ i \geq 2 \\
\boldsymbol{\alpha}_I &= \hat{\boldsymbol{\alpha}}_I \\
x, g_i, \alpha_i &\in [0,1] && i \in [\![L]\!]
\end{aligned}
$$

In this form, it is straightforward to confirm that $(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D^{\text{LP}}$ from the corresponding form for $\tilde{D}^{\text{LP}}$: since the indices for both the map on $\boldsymbol{g}$ and on the shift from $\tilde{I}$ to I are shifted by 1 in the same direction, with the same choice of $\hat{\boldsymbol{\alpha}}$, all equality constraints on g_i, $i \in \tilde{I}$ are preserved through the mapping. Further, the relationship between each g_i and g_{i-1} is likewise preserved, as the corresponding α_i is the same, and finally the choice of g_1 is feasible given x. Thus, all constraints are satisfied, so that $(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D^{\text{LP}}$, yielding for the choice of y above that $(x, y, \boldsymbol{g}, \boldsymbol{\alpha}) \in P_{-}^{\text{LP}}|_{x \in \text{conv}(X \cap [0, 1/2])}$.

Further, from the form for D^{LP} above, we observe that $\Phi(\tilde{X}) = X \cap [0, \frac{1}{2}]$ and $\Phi(\tilde{X}^{\text{LP}}) = \text{conv}(X \cap [0, \frac{1}{2}])$. To show the first portion, we have already shown that $\Phi(\tilde{X}) \subseteq X \cap [0, \frac{1}{2}]$. To prove the other direction, we simply reverse the map for any $(x, \boldsymbol{g}, \boldsymbol{\alpha}) \in D|_{x \in [0, 1/2]}$, ignoring α_1: letting $\tilde{x} = g_1 = \frac{x}{2}$, $\tilde{\boldsymbol{g}} = \boldsymbol{g}_{[\![2,L]\!]}, \tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha}_{[\![2,L]\!]}$, it is easy to confirm that $(\tilde{x}, \tilde{\boldsymbol{g}}, \tilde{\boldsymbol{\alpha}}) \in \tilde{D}$.

To show that $\Phi(\tilde{X}^{\text{LP}}) = \text{conv}(X \cap [0, \frac{1}{2}])$, we observe that $\text{conv}(X)|_{x \in [0, 1/2]}$ is a closed interval with boundary points in $X \cap [0, \frac{1}{2}] = \Phi(\tilde{X})$, so that $\text{conv}(X \cap [0, \frac{1}{2}]) = \text{conv}(\Phi(\tilde{X})) = \Phi(\text{conv}(\tilde{X})) = \Phi(\tilde{X}^{\text{LP}})$ (since Φ is linear in x).

Sharpness on $\Phi(\tilde{X}^{\text{LP}})$

We now establish that P_-^{IP} is sharp on $\Phi(\tilde{X}^{\text{LP}}) = \text{conv}(X \cap [0, \frac{1}{2}])$ by establishing the two facts 1) and 2) outlined above. We begin with fact 1). Let $\tilde{x} \in \tilde{X}^{\text{LP}}$ and $\tilde{y}^* = \min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(\tilde{P}_-^{\text{LP}}|_{x=\tilde{x}})\}$, and let $\tilde{\boldsymbol{g}}^*$ be the optimizing solution from Lemma 5.33. For convenience, let $\hat{x} = \Phi(x)$ and $\boldsymbol{g}^* = \Phi(\tilde{\boldsymbol{g}}^*)$. Then $\boldsymbol{g}^*$ takes on the optimal form from Lemma 5.33, with $\tilde{y}^* = \tilde{\ell}(\tilde{x}, \tilde{\boldsymbol{g}}^*)$, yielding

$$y^* := \Phi(\tilde{y}^*) = \Phi(\tilde{\ell}(\tilde{x}, \tilde{\boldsymbol{g}}^*)) = \ell(\hat{x}, \boldsymbol{g}^*) = \min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P_-^{\text{LP}}|_{x=\hat{x}})\},$$

so that $(\frac{1}{4}\tilde{y}^*, \Phi(\tilde{\boldsymbol{g}}^*)) \in \arg\min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P_-^{\text{LP}}|_{x=\hat{x}})\}$, as required. As a corollary, observing that $\tilde{\ell}(\tilde{x}) = \min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P_-^{\text{LP}}|_{x=\hat{x}})\}$, and likewise for $\ell(\hat{x})$, we have that $\Phi(\tilde{\ell}(\tilde{x})) = \frac{1}{4}\tilde{\ell}(\tilde{x}) = \ell(\hat{x})$.

Next, to show fact 2), i.e. $\tilde{y} = \tilde{\underline{\ell}}(\tilde{x}) \Leftrightarrow \Phi(\tilde{y}) = \underline{\ell}(\Phi(\tilde{x}))$, we observe that, for any $\tilde{x} \in \tilde{X}$, we have $\Phi(\tilde{x}) \in X$, and so

$$\Phi(\tilde{\underline{\ell}}(\tilde{x})) = \Phi(\tilde{\ell}(\tilde{x})) = \ell(\Phi(\tilde{x})) = \underline{\ell}(\Phi(\tilde{x})).$$

Thus, $\Phi(\tilde{\underline{\ell}}(\tilde{x})) = \underline{\ell}(\Phi(\tilde{x}))$ on $\tilde{X}$. Now, by Lemma 5.34, across any gap $\tilde{x}_-, \tilde{x}_+ \in \tilde{X}$ for which $(\tilde{x}_-, \tilde{x}_+) \cap \tilde{X} = \emptyset$ and $\tilde{x} \in [\tilde{x}_-, \tilde{x}_+]$, we have that $\tilde{\underline{\ell}}(\tilde{x})$ is on the line between the points $(\tilde{x}_-, \tilde{\ell}(\tilde{x}_-))$ and $(\tilde{x}_+, \tilde{\ell}(\tilde{x}_+))$. Thus, defining $\hat{x} := \Phi(\tilde{x})$, since Φ is linear in x and y, $\ell(\hat{x})$ lies on the line between the points $(\Phi(\tilde{x}_-), \tilde{\ell}(\tilde{x}_-)))$ and $\Phi((\tilde{x}_+), \tilde{\ell}(\tilde{x}_+))$.

Now, observe that, since $\Phi(\tilde{X}) = X \cap [0, \frac{1}{2}]$, we have that $(x_-, x_+) := (\Phi(\tilde{x}_-), \Phi(\tilde{x}_+))$ is a gap in X, with $x_-, x_+ \in X$ and $(x_-, x_+) \cap X = \emptyset$. Furthermore, as $x_+, x_- \in X$, we have that $\underline{\ell}(\hat{x}) = \Phi(\tilde{\underline{\ell}}(\tilde{x}_-))$, and similarly for x_+. Thus, by Lemma 5.34, we have for $x \in (x_+, x_-)$ that $\underline{\ell}(\Phi(\tilde{x})) = \underline{\ell}(x) = \Phi(\underline{\ell}(\tilde{x}))$, as required. Therefore, since facts 1) and 2) hold, we have that P_-^{IP} is sharp on $\Phi(\tilde{X}^{\text{LP}})$.

Sharpness on $1 - \Phi(\tilde{X}^{\text{LP}})$

Applying Lemma 5.30 to P_-^{IP}, we immediately recover sharpness on $1 - \Phi(\tilde{X}^{\text{LP}}) = \text{conv}(X \cap [\frac{1}{2}, 1])$. To see this, let $x \in \Phi(\tilde{X}^{\text{LP}})$. Then, via Lemma 5.30, we obtain exactly the same feasible regions for $\boldsymbol{g}, \boldsymbol{\alpha}$ with $x = 1 - \hat{x}$ as with $x = \hat{x}$, i.e., $\text{proj}_{\boldsymbol{g},\boldsymbol{\alpha}_{[\![2,L]\!]}}(D|_{x=\hat{x}}) = \text{proj}_{\boldsymbol{g},\boldsymbol{\alpha}_{[\![2,L]\!]}}(D|_{x=1-\hat{x}})$, and moreover (by the succeeding remark), we have $\hat{x}^2 - \underline{\ell}(\hat{x}) = (1-\hat{x})^2 - \underline{\ell}(1-\hat{x})$. Thus, we have that both $\underline{\ell}(1-\hat{x})$ and $\min_{\boldsymbol{g} \in P_-^{\text{LP}}|_{x=\hat{x}}}(\ell(1-\hat{x}, \boldsymbol{g}))$ maintain the same distance below $(1-\hat{x})^2$ as $\underline{\ell}(\hat{x})$ and $\min_{\boldsymbol{g} \in P_-^{\text{LP}}|_{x=\hat{x}}}(\ell(\hat{x}, \boldsymbol{g}))$, respectively. Thus, since the second pair coincides, so must the first pair, so that

$$\underline{\ell}(1-\hat{x}) = \min_{\boldsymbol{g} \in P_-^{\text{LP}}|_{x=\hat{x}}} (\ell(1-\hat{x}, \boldsymbol{g})),$$

and so sharpness holds on $1 - \Phi(\tilde{X}^{\text{LP}})$.

Sharpness on the gap containing $x = \frac{1}{2}$

Now, since we have shown sharpness on $\text{conv}(X \cap [0, \frac{1}{2}])$ and $\text{conv}(X \cap [\frac{1}{2}, 1])$, we have only to show sharpness on the gap between the two regions. Now, if $\frac{1}{2} \in X$, then we are done, as there is no gap between the regions. Otherwise, let (x_-, x_+) be the gap between the regions, so that $x_-, x_+ \in X$, $(x_-, x_+) \cap X = \emptyset$, and $\frac{1}{2} \in (x_-, x_+)$. We wish to show that $\min_{\boldsymbol{g} \in P_-^{\text{LP}}|_{x=\hat{x}}}(\ell(\hat{x}, \boldsymbol{g}))$ coincides with the line between $(x_-, \ell(x_-))$ and $(x_+, \ell(x_+))$.

To show this, we first note that both endpoints coincide with $\ell^{\hat{j}}(x, \boldsymbol{g}^*)$ for some $\hat{j}$, and by Lemma 5.30, both this value of j and corresponding solution $\boldsymbol{g}^*$ must be the same for both gap endpoints. Further, since x_-, x_+ are the endpoints of a gap, we have that $\ell(x_-) = x_-^2$ and $\ell(x_+) = x_+^2$. This holds as follows: First, by Lemma 4.18, we have that each ℓ^j, $j \geq 0$, is incident with x^2 exactly at the points $x = \frac{k}{2^j} + \frac{1}{2^{j+1}}$, $k = 0, \ldots, 2^j - 1$. Furthermore, the points at which the $\boldsymbol{\alpha}$-vector changes, and thus the possible gaps, are exactly the points $x = k2^{-L}$, which must take the form above for some $j \in [\![0, L-1]\!]$, so that $\ell^{j-1}(x) = x^2$ for $x \in \{x_-, x_+\}$. Since each other $\ell^j(x) \leq x^2$ at these

points, this yields $\ell(x) = x^2$ for $x \in \{x_-, x_+\}$.

Now, let $[a_1, b_1]$ be the bounds on g_1 from Lemma 5.32. Then we have $g_1^* = b_1$: through the mapping Φ, we have $g_1^* = \tilde{x} = \tilde{b}_0$ at both x_- and x_+, where $\tilde{b}_0$ is defined in the manner of Lemma 5.32. Thus, since g_1 is subject to every constraint in D that $\tilde{x}$ is in $\tilde{D}$, we have that $b_1 \leq \tilde{b}_0 = g_1^* \leq b_1$, so that $g_1^* = b_1$.

Furthermore, by the convexity of $\text{proj}_{x,\boldsymbol{g}} D^{\text{LP}}$, since $(x_-, \boldsymbol{g}^*), (x_+, \boldsymbol{g}^*) \in \text{proj}_{x,\boldsymbol{g}} D^{\text{LP}}$, we have that $(\hat{x}, \boldsymbol{g}^*) \in \text{proj}_{x,\boldsymbol{g}} D^{\text{LP}}$ for all $\hat{x} \in (x_-, x_+)$. Thus, we have for any such $\hat{x}$ that

$$g_1^* = b_1 \geq \min(2\hat{x}, 2(1-\hat{x}), b_1) = u(x) \geq g_1^*,$$

yielding by Lemma 5.33 that $g^* \in \arg\min\{y : (y, \boldsymbol{g}) \in \text{proj}_{y,\boldsymbol{g}}(P_-^{\text{LP}})|_{x=\hat{x}}\}$.

Thus, we have that each minimizing solution $\ell(\hat{x}, g^*) = \min_{\boldsymbol{g} \in P_-^{\text{LP}}|_{x=\hat{x}}}(\ell(\hat{x}, \boldsymbol{g})) = \ell(\hat{x})$ is linear in $\hat{x}$ across the gap $[x_-, x_+]$, and coincides with $\tilde{G}(\hat{x})$ at the endpoints, as required. Therefore, we have that P_-^{LP} is sharp across the gap. We have now established sharpness of P_-^{LP} over all of $\text{conv}(D)$, and so the proof is complete for $1 \notin I$.

Sharpness if $1 \in I$

Finally, to recover sharpness if $1 \in I$, we have only to observe that inserting 1 into I, thereby restricting $\alpha_1 = 1$ or $\alpha_1 = 0$, simply restricts P_-^{IP} to either $x \in \Phi(X)$ or $x \in 1 - \Phi(X)$, on which sharpness holds exactly as the sharpness result on the image of Φ (or its reflection) with $1 \in I$, with one difference: we define Φ so that $\alpha_1 = \hat{\alpha}_1$. However, this difference has no effect on the y-minimal solutions for g_1^* within X^{LP}, and thus no effect on sharpness.

□

5.6 Computational Results

In this section, we test the various methods presented in this work against each other on two problem sets: The nonconvex BoxQP instances from Section 4.7 and [129], [160] and earlier works, and the AC optimal power flow (ACOPF) instances from [165].

The BoxQP instances are nonconvex QP's of the form

$$\min_{x\in[0,1]^n} x^\top Q x + c^\top x$$

where Q has multiple negative eigenvalues and $n \in [20, 125]$.

The ACOPF instances are nonconvex MIQCQP's, in which integer variables do not appear in quadratic terms. Two examples of quadratic terms include approximations of trigonometric functions in computing complex voltages, modelling of complex power transmission. For more information, see [165].

All instances were solved in Python using Gurobi 9.1.1, on the 'Woody' cluster, using the "Kaby Lake" nodes with two Xeon E3-1240 v6 chips (4 cores, HT disabled), running at 3.7 GHZ with 32 GB of RAM. The global relative optimality gap with Gurobi is set to the default value of 0.01%. For more information, see the Woody Cluster Website.

For each instance, we solve the problem with each method using various approximation depths $L = 1, 2, 3, 5, 6$, with a time limit of 2 hours for $L \leq 2$ and 4 hours otherwise. All sawtooth and univariate formulation variants are solved with $L_1 = 10$. For the D-NMDT approach, we add the linear constraints for both $\lambda = 0$ and $\lambda = 1$. For the NMDT approach, and for each given variable product $z = xy$, as represented by Gurobi after reading in the .lp files, we choose to discretize x. The T-NMDT and TD-NMDT approaches are the NMDT and D-NMDT approaches, respectively, when applied to bilinear terms, but with all McCormick lower-bounding-constraints for

the univariate quadratic terms replaced with the sawtooth epigraph relaxation (5.15), as described in Sections 5.4.2 and 5.4.3.

Finally, we solve all instances with two different settings in Gurobi: 1) using the default settings to focus on best bound improvement, and 2) using MIPFocus=1, and applying a custom callback function. Given any MIP-feasible solution, this callback function freezes any integer variables from the original problem (before applying any of the discretization techniques from this work), then solves the resulting NLP locally via IPOPT in an attempt to find a feasible solution in the original QCQP.

For each trial run, we divide the instances into families based on performance. For the 'solved' instances, all methods terminate at optimality. For the 'contested' instances, some methods terminate at optimality, while others time out before reaching optimality. Finally, for the 'unsolved' instances, all methods time out before reaching optimality.

1. Time: The shifted geometric mean of the solve time in seconds, using the minimum computation time in the family for the shift.

2. Gap/bnd gap: The shifted geometric mean of the final relative optimality gap $\frac{|\texttt{db}-\texttt{bps}|}{|\texttt{bps}|}$, where `db` is the dual bound provided by the method and `bps` is the best known primal solution for the instance. Shift is taken as
   $\max\{10^{-4}, \text{minimum gap observed in the family}\}$.

3. Sol gap: The shifted geometric mean of the final relative optimality gap $\frac{|\texttt{bdb}-\texttt{ps}|}{|\texttt{ps}|}$, where `ps` is the primal solution provided by the method for the original problem using the custom callback function, and `bdb` is the best known dual bound for the instance. Shift is taken as $\max\{10^{-4}, \text{minimum gap observed in the family}\}$.

4. BB: Number of instances for which the method yielded the best upper-bound to the original problem (Up to an optimality-gap relative tolerance of 1e-4)

5. Timeout: Number of instances for which the method timed out.

For the ACOPF instances, the best known solutions were provided by the authors, while the 'sol gap' field is not reported. For the boxQP instances, the best known solution is computed from the results in Section 4.7.

The results from the ACOPF differ greatly than those from the BoxQP instances. For the ACOPF instances, the choice of best method is generally very unclear, with all but NMDT showing advantages in different cases. For example, in Table 5.6.1, the Bin2 approach finishes far faster on the contested instances; D-NMDT and T-NMDT give the best time and gap results, respectively for the solved instances; and T-NMDT gives the tightest bounds after 2 hours for the unsolved instances. Moreover, the optimality gaps are very high, even for solved instances. The $L = 2$ results are similarly murky, suggesting in turn advantages to each of the D-NMDT, T-NMDT, and TD-NMDT approaches.

However, for $L = 5$, the univariate approach gains a small but distinct advantage for the more difficult problems. For example, in Table 5.6.3, which maintains the instance classes from $L = 2$, substantial gains in optimality gap are made uniformly across all methods and all problem classes when compared to $L = 2$. However, the univariate approach pulls distinctly ahead of the other methods for the more difficult instance classes, for which some approaches did not yield a globally optimal solution. A similar story is reflected in Table 5.6.4.

However, the BoxQP results are more clear, and moreover benefit strongly from the IPOPT callback function. Even using $L = 1$ (Table 5.6.5), all methods are able to find solutions very close to global optimality in almost all cases, and these results only strengthen with $L = 2$ (Table 5.6.6). However, the NMDT-based approaches show a very strong advantage of more than an order of magnitude in terms of solve times compared to the univariate approaches. Further, with both $L = 1$ and $L = 2$, the double-discretized variants D-NMDT and TD-NMDT solve much faster, particularly for the contested cases, than the NMDT and T-NMDT approaches, with D-NMDT as by far the fastest method.

family	method	time (sec)	gap	BB	TO
solved	Bin2	3.55	15.81%	7/26	-
	Bin3	4.50	14.95%	8/26	-
	HybU	5.04	14.95%	7/26	-
	NMDT	3.84	15.70%	6/26	-
	D-NMDT	2.52	15.12%	6/26	-
	T-NMDT	16.52	13.63%	23/26	-
	TD-NMDT	6.41	14.30%	8/26	-
contested	Bin2	156.8	38.46%	2/15	2/15
	Bin3	917.0	37.59%	2/15	6/15
	HybU	837.8	37.59%	2/15	7/15
	NMDT	1733.8	39.25%	2/15	9/15
	D-NMDT	1203.8	38.01%	2/15	6/15
	T-NMDT	4239.3	35.90%	13/15	12/15
	TD-NMDT	2401.9	37.03%	4/15	8/15
unsolved	Bin2	-	36.14%	0/18	-
	Bin3	-	35.06%	0/18	-
	HybU	-	35.07%	0/18	-
	NMDT	-	36.52%	0/18	-
	D-NMDT	-	35.46%	1/18	-
	T-NMDT	-	33.72%	16/18	-
	TD-NMDT	-	34.60%	1/18	-

Table 5.6.1: Performance comparison of methods on ACOPF instances with $L = 1$, no callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	4.63	7.43%	5/17	-
	Bin3	4.55	6.96%	6/17	-
	HybU	4.63	6.93%	5/17	-
	NMDT	6.95	5.79%	2/17	-
	D-NMDT	2.14	7.27%	2/17	-
	T-NMDT	12.27	3.86%	12/17	-
	TD-NMDT	4.73	6.04%	10/17	-
contested	Bin2	1458.7	32.66%	0/13	3/13
	Bin3	1349.1	30.53%	0/13	5/13
	HybU	2477.4	30.43%	1/13	4/13
	NMDT	7200.0	35.67%	0/13	13/13
	D-NMDT	841.6	32.35%	0/13	4/13
	T-NMDT	6250.1	29.39%	6/13	12/13
	TD-NMDT	4773.1	29.60%	6/13	8/13
unsolved	Bin2	-	35.72%	1/29	-
	Bin3	-	34.94%	4/29	-
	HybU	-	34.97%	5/29	-
	NMDT	-	36.28%	1/29	-
	D-NMDT	-	35.58%	2/29	-
	T-NMDT	-	34.84%	10/29	-
	TD-NMDT	-	34.81%	12/29	-

Table 5.6.2: Performance comparison of methods on ACOPF instances with $L = 2$, no callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved-$L = 2$	Bin2	79.69	0.266%	4/17	3/17
	Bin3	84.54	0.204%	4/17	4/17
	HybU	80.37	0.188%	4/17	4/17
	NMDT	494.64	0.752%	1/17	7/17
	DNMDT	99.90	0.552%	1/17	4/17
	T-NMDT	233.57	0.151%	5/17	4/17
	T-DNMDT	89.06	0.136%	5/17	4/17
contested-$L = 2$	Bin2	7200.0	26.663%	0/13	13/13
	Bin3	7200.0	26.042%	2/13	13/13
	HybU	7200.0	25.799%	11/13	13/13
	NMDT	7200.1	27.872%	0/13	13/13
	DNMDT	7200.0	27.920%	0/13	13/13
	T-NMDT	775.0	27.187%	0/13	9/13
	T-DNMDT	7200.0	27.019%	0/13	13/13
unsolved-$L = 2$	Bin2	-	32.648%	0/29	-
	Bin3	-	32.333%	5/29	-
	HybU	-	32.099%	21/29	-
	NMDT	-	33.167%	0/29	-
	DNMDT	-	33.041%	2/29	-
	T-NMDT	-	32.857%	0/29	-
	T-DNMDT	-	32.759%	1/29	-

Table 5.6.3: Performance comparison of methods on ACOPF instances with $L = 5$, no callback function, 4-hour time limit, instance classes for $L = 2$.

family	method	time (sec)	gap	BB	TO
solved	Bin2	12.90	0.13%	2/10	-
	Bin3	11.80	0.09%	1/10	-
	HybU	11.68	0.08%	2/10	-
	NMDT	75.11	0.29%	1/10	-
	DNMDT	9.93	0.26%	1/10	-
	T-NMDT	36.90	0.05%	5/10	-
	T-DNMDT	9.74	0.04%	3/10	-
contested	Bin2	2243.6	0.06%	0/11	7/11
	Bin3	3511.8	0.05%	3/11	8/11
	HybU	2994.9	0.00%	8/11	7/11
	NMDT	14400.0	0.43%	0/11	11/11
	DNMDT	5797.9	0.22%	0/11	7/11
	T-NMDT	610.0	0.05%	0/11	4/11
	T-DNMDT	1549.6	0.03%	2/11	5/11
unsolved	Bin2	-	28.21%	2/38	-
	Bin3	-	27.71%	7/38	-
	HybU	-	27.63%	26/38	-
	NMDT	-	29.13%	0/38	-
	DNMDT	-	29.11%	2/38	-
	T-NMDT	-	28.71%	0/38	-
	T-DNMDT	-	28.56%	1/38	-

Table 5.6.4: Performance comparison of methods on ACOPF instances with $L = 5$, no callback function, 4-hour time limit.

family	method	time (sec)	bnd gap	sol gap	BB	TO
solved	Bin2	624.69	0.00%	0.000%	5/6	-
	Bin3	420.95	0.00%	0.000%	5/6	-
	HybU	68.56	0.00%	0.000%	6/6	-
	NMDT	1.98	0.00%	0.000%	5/6	-
	DNMDT	1.24	0.00%	0.000%	5/6	-
	T-NMDT	2.79	0.00%	0.000%	6/6	-
	T-DNMDT	1.55	0.00%	0.000%	6/6	-
contested	Bin2	7200.0	32.09%	0.000%	0/38	38/38
	Bin3	7080.5	15.98%	0.000%	2/38	36/38
	HybU	4937.6	3.08%	0.002%	7/38	30/38
	NMDT	1001.5	0.05%	0.000%	15/38	15/38
	DNMDT	95.3	0.00%	0.000%	26/38	0/38
	T-NMDT	1212.2	0.05%	0.000%	19/38	18/38
	T-DNMDT	188.4	0.00%	0.000%	32/38	4/38
unsolved	Bin2	-	113.25%	0.016%	0/55	-
	Bin3	-	111.79%	0.023%	0/55	-
	HybU	-	98.46%	1.010%	0/55	-
	NMDT	-	86.75%	0.004%	0/55	-
	DNMDT	-	54.35%	0.005%	38/55	-
	T-NMDT	-	87.93%	0.007%	0/55	-
	T-DNMDT	-	59.85%	0.005%	17/55	-

Table 5.6.5: Performance comparison of methods on BoxQP instances with $L = 1$, with callback function, 2-hour time limit.

family	method	time (sec)	bnd gap	sol gap	BB	TO
solved	Bin2	236.45	0.00%	0.000%	6/6	-
	Bin3	205.74	0.00%	0.000%	6/6	-
	HybU	126.49	0.00%	0.000%	6/6	-
	NMDT	4.40	0.00%	0.000%	5/6	-
	DNMDT	4.05	0.00%	0.000%	5/6	-
	T-NMDT	4.73	0.00%	0.000%	6/6	-
	T-DNMDT	5.34	0.00%	0.000%	5/6	-
contested	Bin2	7200.0	19.35%	0.000%	0/25	25/25
	Bin3	6131.8	4.85%	0.000%	3/25	22/25
	HybU	4545.7	0.93%	0.000%	7/25	18/25
	NMDT	752.5	0.01%	0.000%	12/25	9/25
	DNMDT	291.3	0.00%	0.000%	19/25	0/25
	T-NMDT	893.5	0.01%	0.000%	15/25	9/25
	T-DNMDT	418.1	0.00%	0.000%	22/25	3/25
unsolved	Bin2	-	92.84%	0.007%	0/68	-
	Bin3	-	90.91%	0.004%	0/68	-
	HybU	-	86.33%	0.038%	0/68	-
	NMDT	-	76.55%	0.003%	0/68	-
	DNMDT	-	63.20%	0.003%	40/68	-
	T-NMDT	-	76.84%	0.002%	0/68	-
	T-DNMDT	-	66.38%	0.004%	28/68	-

Table 5.6.6: Performance comparison of methods on BoxQP instances with $L = 2$, with callback function, 2-hour time limit.

5.7 Conclusion

We have presented two new discretization approaches, HybU and D-NMDT, for solving QCQP's that substantially reduce the total required number of integer variables to reach a desired level of precision compared to prior approaches. Moreover, these approaches are competitive with existing approaches when applied to the tested MIQCQP AC optimal power flow and boxQP problems. At high precision for the ACOPF instances, the proposed hybrid univariate approach narrowly edges out the others in terms of the best obtainable bounds. Moreover, for the boxQP instances, the proposed double-discretized NMDT approach yields substantially faster solution times compared to the other approaches, with all approaches finding near-optimal solutions via local solves from approximate MIP solutions even at low levels of precision. Moreover, we show that the hybrid univariate approach is substantially tighter than either prior univariate approach, but less tight than D-NMDT, and give the relative total relaxation volumes of each.

We note that, when applied to boxQP instances, the MIP discretization approaches considered in this work are significantly less performant than the convex MIQP diagonalization-based approaches in e.g. Chapter 4 and [129], applied to the same instances. However, our goal in this work is to compare the performance of different discretization approaches applied directly to bilinear terms, without incorporating other techniques. In particular, a promising approach for problems such as the AC optimal powerflow instances in [165] could be to combine an adaptive approach, such as the one outlined in [174], with the compact univariate-reformulation approach, HybU, or its underlying reformulation.

Appendix

5.A Formulations on general intervals

In this appendix, we generalize our formulations to the intervals $x \in [\underline{x}, \overline{x}]$ and $y \in [\underline{y}, \overline{y}]$.

5.A.1 Direct Formulations

Univariate (5.22):

$$
\begin{aligned}
p_1 &= x + y \\
p_2 &= x - y \\
(x, \boldsymbol{g}^x, \boldsymbol{\alpha}^x) &\in S^L \\
(y, \boldsymbol{g}^y, \boldsymbol{\alpha}^y) &\in S^L \\
(p_1, \boldsymbol{g}^{p_1}) &\in T^{L_1} \\
(p_2, \boldsymbol{g}^{p_2}) &\in T^{L_1} \\
z_{p_1} &\geq (l_x + l_y)^2 F^j(\tfrac{p_1 - \underline{x} - \underline{y}}{l_x + l_y}, \boldsymbol{g}^{p_1}) + (\underline{x} + \underline{y})(2p_2 - \underline{x} - \underline{y}) \qquad j \in 0, \ldots, L_1 \\
z_{p_2} &\geq (l_x + l_y)^2 F^j(\tfrac{p_2 - \underline{x} + \overline{y}}{l_x + l_y}, \boldsymbol{g}^{p_2}) + (\underline{x} - \overline{y})(2p_2 - \underline{x} + \overline{y}) \qquad j \in 0, \ldots, L_1 \\
z_x &\leq {l_x}^2 F^L(\tfrac{x - \underline{x}}{l_x}, \boldsymbol{g}^x) + \underline{x}(2x - \underline{x}) \\
z_y &\leq {l_y}^2 F^L(\tfrac{y - \underline{y}}{l_y}, \boldsymbol{g}^y) + \underline{y}(2y - \underline{y}) \\
z &\geq \tfrac{1}{2}(z_{p_1} - z_x - z_y) \\
z &\leq \tfrac{1}{2}(z_x + z_y - z_{p_2}) \\
(x, y, z) &\in \mathcal{M}(x, y) \\
x &\in [\underline{x}, \overline{x}] \\
y &\in [\underline{y}, \overline{y}] \\
p_1 &\in [\underline{x} + \underline{y}, \overline{x} + \overline{y}] \\
p_2 &\in [\underline{x} - \overline{y}, \overline{x} - \underline{y}].
\end{aligned}
\tag{5.52}
$$

NMDT (5.23):

$$\begin{aligned} x &= l_x \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L + \underline{x} \\ z &= l_x \sum_{i=1}^{L} 2^{-i}u_i + \Delta_z^L + \underline{x} \cdot y \\ (x, \alpha_i, u_i) &\in \mathcal{M}(x, \alpha_i) \qquad\qquad i \in 1, \dots, L \\ (\Delta_x^L, y, \Delta_z^L) &\in \mathcal{M}(\Delta_x^L, y) \\ \Delta_x^L &\in [0, 2^{-L} l_x], \quad y \in [\underline{y}, \overline{y}], \quad \boldsymbol{\alpha} \in \{0,1\}^L \end{aligned} \tag{5.53}$$

Do derive this formulation, we define $\hat{x} \in [0,1]$ and $\hat{z} \approx \hat{x}y$, then use the definitions $x = l_x\hat{x} + \underline{x}$ and

$$z = xy = (l_x\hat{x} + \underline{x})y \approx l_x\hat{z} + \underline{x} \cdot y$$

to obtain

$$\begin{aligned} x &= l_x \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L + \underline{x} \\ z &= l_x \sum_{i=1}^{L} 2^{-i}\alpha_i y + \Delta_x^L \cdot y + \underline{x} \cdot y \\ \Delta_x^L &\in [0, 2^{-L}(\overline{x} - \underline{x})], \quad y \in [\underline{y}, \overline{y}], \quad \boldsymbol{\alpha} \in \{0,1\}^L. \end{aligned} \tag{5.54}$$

As with NMDT, we then apply McCormick envelopes to model all remaining product terms, $\alpha_i y$ and $\Delta_x^L \cdot y$, to obtain the final formulation.

D-NMDT (5.25):

$$
\begin{aligned}
x &= l_x\hat{x} + \underline{x}, \quad y = l_y\hat{y} + \underline{y} \\
z &= l_x l_y \hat{z} + l_x\hat{x}\underline{y} + l_y\hat{y}\underline{x} + \underline{x}\underline{y} \\
\hat{x} &= \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_{\hat{x}}^L, \quad \hat{y} = \sum_{i=1}^{L} 2^{-i}\beta_i + \Delta_{\hat{y}}^L \\
\hat{z} &= \sum_{i=1}^{L} 2^{-i}(u_i + v_i) + \Delta_{\hat{z}}^L \\
(\lambda\Delta_{\hat{y}}^L + (1-\lambda)\hat{y}, \alpha_i, u_i) &\in \mathcal{M}(\lambda\Delta_{\hat{y}}^L + (1-\lambda)\hat{y}, \alpha_i) \qquad i \in 1, \dots, L \\
((1-\lambda)\Delta_{\hat{x}}^L + \lambda\hat{x}, \beta_i, v_i) &\in \mathcal{M}((1-\lambda)\Delta_{\hat{x}}^L + \lambda\hat{x}, \beta_i) \qquad i \in 1, \dots, L \\
(\Delta_{\hat{x}}^L, \Delta_{\hat{y}}^L, \Delta_{\hat{z}}^L) &\in \mathcal{M}(\Delta_{\hat{x}}^L, \Delta_{\hat{y}}^L) \\
\Delta_{\hat{x}}^L, \Delta_{\hat{y}}^L &\in [0, 2^{-L}], \quad \hat{x}, \hat{y} \in [0,1], \quad \boldsymbol{\alpha}, \boldsymbol{\beta} \in \{0,1\}^L
\end{aligned}
\tag{5.55}
$$

To derive this generalized formulation, we define $\hat{x} \in [0,1]$ and $\hat{y} \in [0,1]$. We then map to the original space by using definitions $x = l_x\hat{x} + \underline{x}$ and $y = l_y\hat{y} + \underline{y}$, and

$$
\begin{aligned}
z &= xy = (l_x\hat{x} + \underline{x})(l_y\hat{y} + \underline{y}) \\
&= l_x l_y \hat{x}\hat{y} + l_x\hat{x}\underline{y} + l_y\hat{y}\underline{x} + \underline{x}\underline{y} \\
&\approx l_x l_y \hat{z} + l_x\hat{x}\underline{y} + l_y\hat{y}\underline{x} + \underline{x}\underline{y},
\end{aligned}
$$

then following the original derivation on $\hat{x}, \hat{y}, \hat{z}$. As in the derivation of (5.25), we then obtain the D-NMDT formulation by applying McCormick envelopes to the product terms $\beta_i((1-\lambda)\Delta_{\hat{x}}^L + \lambda\hat{x})$, $\alpha_i(\lambda\Delta_{\hat{y}}^L + (1-\lambda)\hat{y})$, and $\Delta_{\hat{x}}^L\Delta_{\hat{y}}^L$. As in (5.25), for use in creating McCormick envelopes, the more involved product terms have bounds

$$
\begin{aligned}
(1-\lambda)\Delta_{\hat{x}}^L + \lambda\hat{x} &\in [0, (1-\lambda)2^{-L} + \lambda] \\
\lambda\Delta_{\hat{y}}^L + (1-\lambda)\hat{y} &\in [0, \lambda 2^{-L} + (1-\lambda)].
\end{aligned}
$$

5.A.2 Formulations for Univariate Quadratic

To generalize the formulations in this section, we introduce intermediate variables $\hat{x} \in [0,1]$, then apply each original formulation to model $\hat{y} = \hat{x}^2$. We then map $\hat{x}$ and $\hat{y}$ back to the original space, yielding

$$\hat{x} = \tfrac{x-\underline{x}}{l_x}, \quad \hat{y} = \tfrac{y-\underline{x}(2x-\underline{x})}{{l_x}^2}$$

For the sawtooth relaxation, this yields the final formulation. For NMDT and D-NMDT, we obtain the final formulation by substituting these equations, rearranging, and algebraically mapping the domain of Δ_x^L to $[0, 2^L]$. Alternatively, one can derive these formulations in the manner of Section 5.4.4, but with $x = y$.

Sawtooth (5.17):

$$\{(x,y) \in [\underline{x}, \overline{x}] \times \mathbb{R} : \exists (\hat{x}, \hat{y}, \boldsymbol{g}, \boldsymbol{\alpha}) \in [0,1] \times \mathbb{R} \times [0,1]^{L_1+1} \times \{0,1\}^L : (5.57)\}, \tag{5.56}$$

$$\begin{aligned}
\hat{x} &= \tfrac{x-\underline{x}}{l_x} \\
\hat{y} &= \tfrac{y-\underline{x}(2x-\underline{x})}{{l_x}^2} \\
(\hat{x}, \boldsymbol{g}_{[\![0,L]\!]}, \boldsymbol{\alpha}) &\in S^L(\hat{x}) \\
(\hat{x}, \boldsymbol{g}) &\in T^{L_1}(\hat{x}) \\
\hat{y} &\leq F^L(\hat{x}, \boldsymbol{g}_{[\![0,L]\!]}) \\
\hat{y} &\geq F^j(\hat{x}, \boldsymbol{g}) - 2^{-2j-2} \qquad j \in 0, \ldots, L_1 \\
\hat{y} &\geq 0 \\
\hat{y} &\geq 2\hat{x} - 1,
\end{aligned} \tag{5.57}$$

NMDT (5.24):

$$\begin{aligned} x &= l_x \sum_{i=1}^{L} 2^{-i}\alpha_i + \Delta_x^L + \underline{x} \\ y &= l_x \sum_{i=1}^{L} 2^{-i}u_i + \Delta_y^L + \underline{x} \cdot x \\ (x, \alpha_i, u_i) &\in \mathcal{M}(x, \alpha_i) \qquad\qquad i \in 1, \dots, L \\ (\Delta_x^L, x, \Delta_y^L) &\in \mathcal{M}(\Delta_x^L, x) \\ \Delta_x^L &\in [0, 2^{-L} l_x], \quad x \in [\underline{x}, \overline{x}], \quad \boldsymbol{\alpha} \in \{0,1\}^L. \end{aligned} \tag{5.58}$$

D-NMDT (5.26):

$$\begin{aligned} x &= l_x \sum_{i=1}^{L} 2^{-i}\alpha_i + l_x\Delta_x^L + \underline{x} \\ z &= l_x \sum_{i=1}^{L} 2^{-i}u_i + {l_x}^2\Delta_z^L + \underline{x}(x + l_x\Delta_x^L) \\ (l_x\Delta_x^L + x, \alpha_i, u_i) &\in \mathcal{M}(l_x\Delta_x^L + x, \alpha_i) \qquad\qquad i \in 1, \dots, L \\ (\Delta_x^L, \Delta_z^L) &\in \mathcal{M}(\Delta_x^L) \\ \Delta_x^L &\in [0, 2^{-L}], \quad x \in [\underline{x}, \overline{x}], \quad \boldsymbol{\alpha} \in \{0,1\}^L \end{aligned} \tag{5.59}$$

For use in creating McCormick envelopes, note that $l_x\Delta_x^L + x \in [\underline{x}, l_x 2^{-L} + \overline{x}]$.

5.B Auxiliary Results and Proofs

5.B.1 Epigraphs Over Gappy Domains

Here, we present the proof of Lemma 5.34 and a subsequent result, Lemma 5.36.

Proof of Lemma 5.34. We first note that we have $g(x) \geq f(x)$ for all $x \in \text{conv}(X)$: for all $x \in \text{conv}(X)$, we have that either $g(x) = f(x)$ or that $g(x)$ is the line between two points on the graph of f, which must lie above the graph of f by the convexity of f. Further, we have that g is convex, as it is a maximum between the convex function f and some of its secant lines, which are

also convex.

Now, trivially, by the convexity of g, we have

$$\operatorname{conv}(\operatorname{epi}_X(f)) = \operatorname{conv}(\operatorname{epi}_X(g)) \subseteq \operatorname{conv}(\operatorname{epi}_{\operatorname{conv}(X)}(g)) = \operatorname{epi}_{\operatorname{conv}(x)}(g)$$

To show that $\operatorname{epi}_{\operatorname{conv}(X)}(g) \subseteq \operatorname{conv}(\operatorname{epi}_X(f))$, let $(x, y) \in \operatorname{epi}_{\operatorname{conv}(X)}(g)$. Then if $x \in X$, $y \geq g(x) = f(x)$, so that $(x, y) \in \operatorname{epi}_X(f) \subseteq \operatorname{conv}(\operatorname{epi}_X(f))$. On the other hand, if $x \in \operatorname{conv}(X) \setminus X$, then by definition of g we have that there exist some $\lambda \in [0, 1]$ and $x_1, x_2 \in X$ such that $x = \lambda x_1 + (1 - \lambda)x_2$ and $g(x) = \lambda f(x_1) + (1 - \lambda)f(x_2)$. Then we have that (x, y) is a convex combination of the points $(x_1, f(x_1) + (y - g(x)))$ and $(x_2, f(x_2) + (y - g(x)))$, which are in $\operatorname{epi}_X(f)$ (since $y - g(x) \geq 0$), yielding $(x, y) \in \operatorname{conv}(\operatorname{epi}_X(f))$ as required. □

Lemma 5.34 supports the proof of the following lemma, which was used to support the original, non-inductive proof of hereditary sharpness of the strengthened sawtooth relaxation (5.17).

Lemma 5.36. *Let $X \subseteq \mathbb{R}$ be closed and bounded. Let $f_i \colon \operatorname{conv}(X) \to \mathbb{R}$, $i = 1, \ldots, t$, be convex functions, and define $f \colon \operatorname{conv}(X) \to \mathbb{R}$ by $f(x) := \max_i(f_i(x))$.*

$$\operatorname{conv}\left(\operatorname{epi}_X(f)\right) = \bigcap_{j=1}^{t} \operatorname{conv}\left(\operatorname{epi}_X(f_i)\right). \tag{5.60}$$

if and only if, for any distinct $x_1, x_2 \in \partial X$ such that $(x_1, x_2) \cap X = \emptyset$, there exists some i such that $f(x_1) = f_i(x_1)$ and $f(x_2) = f_i(x_2)$.

Proof. Since the functions f_i are convex, we have that f is convex and

$$\operatorname{epi}_X(f) = \bigcap_{i=1}^{t} \operatorname{epi}_X(f_i).$$

This holds as a standard result in optimization theory, and we leave the proof to the reader.

Now, let g and g_i be defined for f and f_i, respectively, as in Lemma 5.34. Then we have by the

same lemma that

$$\operatorname{conv}\left(\operatorname{epi}_X(f)\right) = \operatorname{epi}_{\operatorname{conv}(X)}(g)$$
$$\operatorname{conv}\left(\operatorname{epi}_X(f_i)\right) = \operatorname{epi}_{\operatorname{conv}(X)}(g_i)$$

Thus, as $\bigcup_{j=1}^{t} \operatorname{epi}_{\operatorname{conv}(X)}(g_i) = \operatorname{epi}_{\operatorname{conv}(X)}(\max_i g_i)$, (5.60) holds if and only if $\operatorname{epi}_{\operatorname{conv}(X)}(g) = \operatorname{epi}_{\operatorname{conv}(X)}(\max_i g_i)$, which holds if and only if for all $x \in \operatorname{conv}(X)$, we have

$$g(x) = \max_i \{g_i(x)\}.$$

Now, this equality is trivial for $x \in X$, as in that case $g(x) = f(x) = \max_i f_i(x) = \max_i g_i(x)$. Thus, any difference must lie in $\operatorname{conv}(X) \setminus X$. We begin by proving the 'if' direction.

Suppose $x \in \operatorname{conv}(X) \setminus X$. Then, by the definition of g and g_i, we have for the $\lambda \in [0,1], x_1, x_2 \in X$ such that $(x_1, x_2) \cap X = \emptyset$ and $x = \lambda x_1 + (1-\lambda)x_2$, that $g(x) = \lambda f(x_1) + (1-\lambda) f(x_2)$ and $g_i(x) = \lambda f_i(x_1) + (1-\lambda) f_i(x_2)$.

Now, clearly, we have $g(x) \geq \max_i\{g_i(x)\}$:

$$\begin{aligned} g(x) &= \lambda f(x_1) + (1-\lambda) f(x_2) = \lambda \max_i(f_i(x_1)) + \lambda \max_i(f_i(x_2)) \\ &\geq \max_i(\lambda f_i(x_1) + \lambda f_i(x_2)) = \max_i(g_i(x)) \end{aligned}$$

Furthermore, we have by the choice of f_i that there exists some $\hat{i}$ such that $f(x_1) = f_{\hat{i}}(x_1)$ and $f(x_2) = f_{\hat{i}}(x_2)$. Thus,

$$\begin{aligned} g(x) &= \lambda f(x_1) + (1-\lambda) f(x_2) = \lambda f_{\hat{i}}(x_1) + (1-\lambda) f_{\hat{i}}(x_2) \\ &\leq \max_i(\lambda f_i(x_1) + \lambda f_i(x_2)) = \max_i(g_i(x)) \end{aligned}$$

Thus, we have $g(x) = \max_i\{g_i(x)\}$, as required.

To prove the 'only if' direction, we have clearly that if no such $\hat{i}$ exists, then for all i, we have either

that $\max_i(f_i(x_1)) > f_i(x_1)$ or $\max_i(f_i(x_2)) > f_i(x_2)$, and so

$$\begin{aligned} g(x) &= \lambda f(x_1) + (1-\lambda) f(x_2) = \lambda \max_i(f_i(x_1)) + \lambda \max_i(f_i(x_2)) \\ &> \max_i(\lambda f_i(x_1) + (1-\lambda) f_i(x_2)) = \max_i(g_i(x)). \end{aligned}$$

Thus, we have $g(x) \neq \max_i\{g_i(x)\}$, as required. □

5.B.2 Volume Proof for Bin2 and Bin3

Proof of Proposition 5.25. To begin, let $P^{\text{IP}}_{L,\infty}$ be the Bin2 formulation. Now, recall that $F^L(x)$ is the sawtooth overapproximation of $y = x^2$, and consists of secant lines to x^2 between consecutive points $k2^{-L}, (k+1)2^{-L}$. Further, applying the appropriate sawtooth overapproximations and exact underapproximations (and applying the appropriate map for $F(x+y)$), we have that for any given $\hat{x}, \hat{y}$ that $\text{proj}_z(P^{\text{IP}}_{L,\infty})|_{x=\hat{x},y=\hat{y}}$ is given as

$$\tfrac{1}{2}\left((x+y)^2 - F^L(x) - F^L(y)\right) \leq z \leq \tfrac{1}{2}\left(4F^L\left(\tfrac{x+y}{2}\right) - x^2 - y^2\right).$$

Now, let $\underline{x}, \overline{x} = k^x 2^{-(L-1)}, (k^x+1)2^{-(L-1)}$ and $\underline{y}, \overline{y} = k^y 2^{-(L-1)}, (k^y+1)2^{-(L-1)}$, and define $l_x = \overline{x} - \underline{x} = 2^{-(L-1)}$ and $l_y = \overline{y} - \underline{y} = 2^{-(L-1)}$. Consider the restriction of x and y to the grid piece $[\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}]$. Then the volume of $P^{\text{IP}}_{L,\infty}$ over the grid piece is given as

$$\begin{aligned} & \tfrac{1}{2}\int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} \left(4F^L\left(\tfrac{x+y}{2}\right) - x^2 - y^2 - \left((x+y)^2 - F^L(x) - F^L(y)\right)\right) dydx \\ = \; & \tfrac{1}{2}\int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} \left(\left(4F^L\left(\tfrac{x+y}{2}\right) - (x+y)^2\right) + \left(F^L(x) - x^2\right) + \left(F^L(y) - y^2\right)\right) dydx \\ = \; & \tfrac{l_y}{2}\int_{\underline{x}}^{\overline{x}} (F^L(x) - x^2)dx + \tfrac{l_x}{2}\int_{\underline{y}}^{\overline{y}} (F^L(y) - y^2)dy \\ & + 2\int_{\underline{x}}^{\overline{x}}\int_{\underline{y}}^{\overline{y}} \left(F^L\left(\tfrac{x+y}{2}\right) - \left(\tfrac{x+y}{2}\right)^2\right) dydx \end{aligned}$$

Now, note that the first two integrals are each the overapproximation volumes for the sawtooth approximation over two consecutive univariate domain segments, each of which has area $\frac{1}{6}2^{-3L}$. Thus, since $l_x = l_y = 2 \cdot 2^{-L}$, we have that the first two integrals (and thus the relaxation volumes from the relaxations of x and y) add to $4 \cdot (\frac{1}{6}2^{-4L}) = \frac{2}{3}2^{-4L}$.

To process the third integral, we first apply a substitution $u = \frac{(x-\underline{x})+(y-\underline{y})}{2}$, $v = \frac{(x-\underline{x})-(y-\underline{y})}{2}$, yielding a Jacobian magnitude of $\frac{1}{2}$. The integral then becomes

$$
\begin{aligned}
& 2\int_0^{2^{-L}} (F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2) \int_{-u}^{u} 1 dv du \\
+\ & 2\int_{2^{-L}}^{2\cdot 2^{-L}} (F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2) \int_{-(2\cdot 2^{-L}-u)}^{2\cdot 2^{-L}-u} 1 dv du \\
=\ & 4\int_0^{2^{-L}} u(F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2) du \\
+\ & 4\int_{2^{-L}}^{2\cdot 2^{-L}} (2\cdot 2^{-L} - u)(F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2) du \\
=\ & 8\int_0^{2^{-L}} u(F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2) du && \text{(J1)} \\
=\ & 8\int_0^{2^{-L}} u(u(2^{-L} - u)) du && \text{(J2)} \\
=\ & 8\int_0^{2^{-L}} (2^{-L}u^2 - u^3) du \\
=\ & 8(\tfrac{1}{3}2^{-4L} - \tfrac{1}{4}2^{-4L}) = \tfrac{2}{3}2^{-4L}
\end{aligned}
$$

The steps J1 and J2 rely on the observation that F^L is the secant line to x^2 across the intervals $[\frac{\underline{x}+\underline{y}}{2}, \frac{\underline{x}+\underline{y}}{2} + 2^{-2L}]$ and $[\frac{\underline{x}+\underline{y}}{2} + 2^{-2L}, \frac{\underline{x}+\underline{y}}{2} + 2 \cdot 2^{-2L}]$, due to the form of $\underline{x}$ and $\underline{y}$. In addition, for some $x \in [x_1, x_2]$, the error between and x^2 and the secant line to x^2 at points x_1 and x_2 is given by $(x - x_1)(x_2 - x)$ - the product of distances to each endpoint. Thus, for $u \in [0, 2^{-L}]$, we have

$$F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2 = u(2^L - u),$$

yielding the step J2. On the other hand, to show the step J1, we have for $u \in [0, 2^{-L}]$ that

$$F^L\left(u + \tfrac{\underline{x}+\underline{y}}{2}\right) - (u + \tfrac{\underline{x}+\underline{y}}{2})^2 = (u - 2^L)(2^{-2L} - u),$$

so that the second integral becomes the first integral under the substitution $\tilde{u} = 2^{-L} - u$, since the secant-error portion of the integrand is symmetric about $u = 2^{-L}$. Thus, the volume related to the second integral is $\frac{4}{3}2^{-4L}$ The volume of $P^{\text{IP}}_{L,\infty}$ over each grid piece is thus $2 \cdot 2^{-4L}$, yielding a total volume for $P^{\text{IP}}_{L,\infty}$ of $2^{2(L-1)}(2 \cdot 2^{-4L}) = \frac{1}{2}2^{-2L}$. The proof for Bin3 is similar, and so is omitted here. □

5.B.3 Additional Reflective Gray Code Result

To support our original, non-inductive proof of hereditary sharpness for the sawtooth relaxation, we introduced the reflected Gray code (RGC) and some of its useful properties, as in Section 4.3, then proved an additional property of the RGC. We include the details here for posterity.

In the remainder of this section, we will work with two notions of expressing integers as vectors in $\{0,1\}^*$. First, we consider the standard binarization with L bits. That is, for an integer $i \in [\![0, 2^L - 1]\!]$ we define $\boldsymbol{\beta}^i \in \{0,1\}^L$ such that

$$i = \sum_{j=1}^{L} 2^{L-j}\beta^i_j. \tag{5.61}$$

Next, we define the *reflected Gray code* sequence, which is a sequence of binary representations of integers that is extremely well-studied in electrical engineering and engineering [153]. Notably, each adjacent pair in the sequence differs in exactly one bit. As presented in [154] and references therein, the L-bit reflected Gray code $\boldsymbol{\alpha}^i \in \{0,1\}^L$ representing the integer i can be described by

the recursion

$$\alpha_1^i = \beta_1^i \tag{5.62}$$

$$\alpha_j^i := \beta_j^i \oplus \beta_{j-1}^i \qquad \text{for all } j = 2, \dots, L, \tag{5.63}$$

where we use $\oplus$ to denote addition modulo 2. By inverting the relation, we obtain the formula

$$\beta_j^i = \alpha_1^i \oplus \alpha_2^i \oplus \cdots \oplus \alpha_j^i \qquad \text{for } j = 1, \dots, L. \tag{5.64}$$

In this way, flipping any α_j bit implies that we flip all less significant bits β_k for $k \geq j$. For convenience, we will refer to the 'reflected Gray code' as the 'Gray code'.

One key property of any Gray code is that successive integer representations differ by only 1 bit, i.e.,

$$\|\boldsymbol{\alpha}^i - \boldsymbol{\alpha}^{i+1}\|_1 = 1. \tag{5.65}$$

That is, only one bit changes between adjacent binary vectors in the sequence. In Lemma 4.2, it is shown that a similar property holds if we restrict the set of integers we work with by fixing some of the bits in the Gray code vector. This property relies on the following well-known property of the reflected Gray code.

Lemma 5.37 (Lemma 4.2)**.** *For each $i \in [\![0, 2^L - 1]\!]$, let $\tilde{\boldsymbol{\alpha}}^i$ be the L-bit Gray code for i, and let $\boldsymbol{\alpha}^j$ be the $L+1$-bit Gray code for some $i \in [\![0, 2^{L+1} - 1]\!]$.*

1. *If $j \in [\![0, 2^L - 1]\!]$, then $\boldsymbol{\alpha}^j = [0, \tilde{\boldsymbol{\alpha}}^j]$.*
2. *If $j \in [\![2^L, 2^{L+1} - 1]\!]$, then $\boldsymbol{\alpha}^j = [1, \tilde{\boldsymbol{\alpha}}^i]$, where $i = 2^{L+1} - j - 1$.*

Another key property of the reflective Gray code sequence is that, after fixing some bits, the remaining sequence is itself a reflective Gray code, but with some bits flipped. For convenience in defining this lemma, we introduce the following shorthand: For a vector $\boldsymbol{v} \in \mathbb{R}^n$ and an index set $I = \{i_1, \dots, i_m\} \subseteq [\![n]\!]$, we define $\boldsymbol{v}_I = [v_{i_1}, \dots, v_{i_m}]^\top$.

Lemma 5.38 (Lemma 4.3)**.** *Let $J \subseteq [\![L]\!]$ and $\overline{\boldsymbol{\alpha}} \in \{0,1\}^{|J|}$. Let $X = \{x \in [\![0, 2^L - 1]\!] : \boldsymbol{\alpha}^x_J = \bar{\boldsymbol{\alpha}}\}$. We will write X as $X = \{x_1, \ldots, x_t\}$, ordered such that $x_j < x_{j+1}$. Let $I = [\![L]\!] \setminus J$. Then $\boldsymbol{\alpha}_I^{x_j}$ is a reflected Gray code for the indices j over X. That is, for any $j \in 1, \ldots, t$, we have*

$$\|\boldsymbol{\alpha}_I{}^{x_j} - \boldsymbol{\alpha}_I{}^{x_{j+1}}\|_1 = 1. \tag{5.66}$$

Furthermore, if $|I| \geq 1$, there exists a $\boldsymbol{\gamma} \in \{0,1\}^{|I|}$ such that, for all $j \in [\![0, t]\!]$, we have

$$\boldsymbol{\alpha}_I^{x_j} \oplus \boldsymbol{\gamma} = \boldsymbol{\alpha}^j_{[\![L-|I|+1, L]\!]}. \tag{5.67}$$

That is, the modified Gray code after fixing some bits is the original reflected Gray code on $|I|$ bits, with some bits flipped.

To support the hereditary sharpness proof, we prove an additional result related to the behavior of the Gray code with fixed bits across gaps: after fixing some bits within a Gray code, consider a gap between integers x_1 and x_2, with some excluded integers in between. Then, according to Lemma 5.38, exactly one bit $\boldsymbol{\alpha}_{k_1}$ flips across the gap, while some other bit α_{k_2} must flip while incrementing x_1 by 1 to enter the gap. We will show that $k_2 \geq k_1 + 1$, so that the bit that flipped across the gap is strictly earlier than the frozen bit that flipped to enter the gap. We first give an example of how this works.

Example 5.39. Consider the gray code with 3 bits. The Sawtooth Approximation model, after fixing the third bit $\alpha_3 = 1$, appears in Figure 5.B.1.

Let $X = \{1, 2, 5, 6\}$ be the set of values represented by the gray code on three bits after fixing $\alpha_3 = 1$.

Notice that $\boldsymbol{\alpha}^2$ and $\boldsymbol{\alpha}^3$ differ by a single bit, which is α_2. Whereas, $\boldsymbol{\alpha}^2$ and $\boldsymbol{\alpha}^5$ differ by a single bit, which is α_1. Thus, we see that the next gray code vector in the restricted set of integers X has the property that the bit flip between subsequent vectors is always an earlier bit than the subsequent vector in the full gray code sequence. In the following lemma, we show that this property holds in

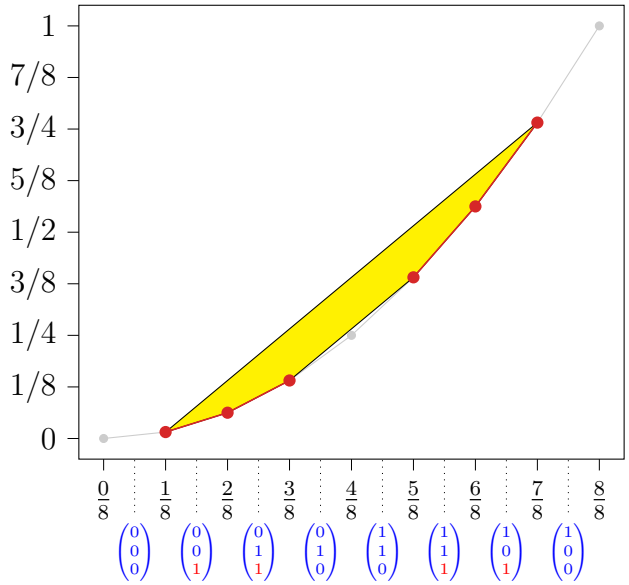


Figure 5.B.1: Mixed integer hull of the sawtooth approximation in the projected xy space. Below, the binary gray code vectors $\boldsymbol{\alpha}^i$ are labeling the intervals that they apply to. In particular, the interval $[i/8, (i+1)/8]$ has the gray code vector $\boldsymbol{\alpha}^i$ applying there. This plot shows sharpness after fixing the α_3 bit.

general.

Lemma 5.40. *Let L, J, $\overline{\boldsymbol{\alpha}}$, X, and I be defined as in Lemma 5.38. Let j be such that x_j and $x_{j+1} \in X$, with $x_{j+1} \geq x_j + 2$, so that there are some intermediate integers that are not in X. Now, let k_1 and k_2 be such that $\boldsymbol{\alpha}^{x_j} \oplus \boldsymbol{\alpha}^{x_{j+1}} = \hat{\boldsymbol{e}}_{k_1}$ and $\boldsymbol{\alpha}^{x_j} \oplus \boldsymbol{\alpha}^{x_j+1} = \hat{\boldsymbol{e}}_{k_2}$. Then*

$$k_2 \geq k_1 + 1. \tag{5.68}$$

Proof. By definition, there is a bijective correspondence between the first k bits of a gray code and a binary expansion: $(\beta_1^i, \ldots, \beta_k^i) \leftrightarrow (\alpha_1^i, \ldots, \alpha_k^i)$. Hence, fixing the first k bits of the gray code corresponds to fixing the first k bits of the binary expansion. Thus, we have

$$\boldsymbol{\alpha}^{x_j} \oplus \boldsymbol{\alpha}^{x_{j+1}} = \hat{\boldsymbol{e}}_{k_1} \Rightarrow \boldsymbol{\alpha}^{x_j}_{[\![k_1-1]\!]} = \boldsymbol{\alpha}^{x_{j+1}}_{[\![k_1-1]\!]} \Rightarrow \boldsymbol{\beta}^{x_j}_{[\![k_1-1]\!]} = \boldsymbol{\beta}^{x_{j+1}}_{[\![k_1-1]\!]}.$$

Now, for a binary bit fixing of the form $\boldsymbol{\beta}_{[\![k]\!]}$, we have that each x with $\boldsymbol{\beta}^x_{[\![k]\!]} = \boldsymbol{\beta}_{[\![k]\!]}$ follows

$$x = \sum_{j=1}^{k} 2^{L-j}\beta_j + \sum_{j=k+1}^{L} 2^{L-j}\beta_j = \bar{x} + \sum_{j=k+1}^{L} 2^{L-j}\beta_j \in \bar{x} + \{0, \dots, 2^{L-k-1+1} - 1\}.$$

Consequently, if two values x_i and x_j share the same $\boldsymbol{\beta}$-fixing, then all values $x \in [\![x_i, x_j]\!]$ must share that fixing, as well. Thus, since x_j and x_{j+1} share the same fixing $\boldsymbol{\beta}_{[\![k_1-1]\!]}$, we have every intermediate x, and in particular $x_j + 1$, also shares the same fixing. This yields $\beta^x_{[\![k_1-1]\!]} = \beta^{x_{j+1}}_{[\![k_1-1]\!]}$, and thus $\alpha^x_{[\![k_1-1]\!]} = \alpha^{x_{j+1}}_{[\![k_1-1]\!]}$, and so $k_2 \geq k_1$. However, note that x_j and x_{j+1} are not consecutive integers, i.e. $x_{j+1} \neq x_j + 1$. Thus, since $\boldsymbol{\alpha}^{x_j}$ differs from both $\boldsymbol{\alpha}^{x_j+1}$ and $\boldsymbol{\alpha}^{x_{j+1}}$ by exactly one bit by the definition of the reflective Gray code, $k_1 = k_2$ would imply that $x_j + 1 = x_{j+1}$ since the same bit would be flipped twice, a contradiction. Hence, we have $k_1 \neq k_2$, yielding $k_2 \geq k_1 + 1$ as required. □

5.C Additional Numerical Results

family	method	time (sec)	gap	BB	TO
solved	Bin2	1.79	8.66%	7/17	-
	Bin3	1.97	8.20%	8/17	-
	HybU	1.99	8.20%	7/17	-
	NMDT	2.31	8.42%	6/17	-
	D-NMDT	1.69	8.14%	6/17	-
	T-NMDT	4.74	7.50%	14/17	-
	TD-NMDT	2.41	7.71%	8/17	-
contested	Bin2	82.6	38.99%	2/20	0/20
	Bin3	529.8	37.52%	2/20	6/20
	HybU	528.0	37.51%	2/20	7/20
	NMDT	1086.3	40.37%	2/20	10/20
	D-NMDT	2705.5	39.30%	2/20	14/20
	T-NMDT	4637.5	34.88%	16/20	15/20
	TD-NMDT	1273.5	36.61%	5/20	8/20
unsolved	Bin2	-	33.40%	0/22	-
	Bin3	-	32.47%	0/22	-
	HybU	-	32.47%	0/22	-
	NMDT	-	34.09%	0/22	-
	D-NMDT	-	33.86%	0/22	-
	T-NMDT	-	31.39%	21/22	-
	TD-NMDT	-	32.30%	1/22	-

Table 5.C.1: Performance comparison of methods on ACOPF instances with $L = 1$, with callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	9.23	7.21%	4/16	-
	Bin3	9.95	6.72%	4/16	-
	HybU	9.63	6.69%	5/16	-
	NMDT	24.70	5.52%	1/16	-
	D-NMDT	5.64	6.98%	2/16	-
	T-NMDT	20.99	3.58%	12/16	-
	TD-NMDT	10.36	5.78%	9/16	-
contested	Bin2	1198.5	29.52%	1/9	4/9
	Bin3	4535.7	27.60%	1/9	7/9
	HybU	3584.0	27.52%	2/9	4/9
	NMDT	7200.0	32.12%	0/9	9/9
	D-NMDT	2208.2	30.16%	0/9	5/9
	T-NMDT	5976.1	26.95%	6/9	7/9
	TD-NMDT	6477.4	27.13%	1/9	8/9
unsolved	Bin2	-	35.27%	2/34	-
	Bin3	-	34.35%	3/34	-
	HybU	-	34.27%	11/34	-
	NMDT	-	36.68%	2/34	-
	D-NMDT	-	35.53%	2/34	-
	T-NMDT	-	34.18%	11/34	-
	TD-NMDT	-	34.13%	16/34	-

Table 5.C.2: Performance comparison of methods on ACOPF instances with $L = 2$, with callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	12.92	0.13%	2/10	-
	Bin3	11.81	0.09%	1/10	-
	HybU	11.64	0.08%	2/10	-
	NMDT	75.14	0.29%	1/10	-
	DNMDT	9.92	0.26%	1/10	-
	T-NMDT	36.93	0.05%	5/10	-
	T-DNMDT	9.74	0.04%	3/10	-
contested	Bin2	1451.7	0.06%	1/11	7/11
	Bin3	2133.2	0.05%	2/11	8/11
	HybU	1840.9	0.04%	8/11	8/11
	NMDT	7200.0	0.61%	0/11	11/11
	DNMDT	3548.2	0.23%	0/11	8/11
	T-NMDT	480.1	0.06%	0/11	4/11
	T-DNMDT	1136.3	0.04%	2/11	5/11
unsolved	Bin2	-	28.75%	1/38	-
	Bin3	-	28.39%	8/38	-
	HybU	-	28.21%	26/38	-
	NMDT	-	29.53%	0/38	-
	DNMDT	-	29.51%	2/38	-
	T-NMDT	-	29.11%	0/38	-
	T-DNMDT	-	28.97%	1/38	-

Table 5.C.3: Performance comparison of methods on ACOPF instances with $L = 5$, no callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	13.46	0.04%	1/8	-
	Bin3	7.24	0.02%	0/8	-
	HybU	11.78	0.02%	0/8	-
	NMDT	33.70	0.09%	0/8	-
	DNMDT	10.61	0.08%	0/8	-
	T-NMDT	11.14	0.01%	4/8	-
	T-DNMDT	7.49	0.01%	3/8	-
contested	Bin2	2644.3	0.20%	2/11	6/11
	Bin3	2323.0	0.12%	4/11	6/11
	HybU	2096.2	0.13%	8/11	6/11
	NMDT	7200.0	4.75%	1/11	11/11
	DNMDT	3333.3	1.30%	1/11	7/11
	T-NMDT	562.0	2.80%	1/11	6/11
	T-DNMDT	1391.8	0.47%	4/11	5/11
unsolved	Bin2	-	28.27%	2/40	-
	Bin3	-	27.71%	7/40	-
	HybU	-	27.46%	30/40	-
	NMDT	-	29.58%	2/40	-
	DNMDT	-	29.59%	0/40	-
	T-NMDT	-	28.97%	0/40	-
	T-DNMDT	-	28.84%	2/40	-

Table 5.C.4: Performance comparison of methods on ACOPF instances with $L = 5$, with callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	13.43	0.04%	1/8	-
	Bin3	7.20	0.02%	0/8	-
	HybU	11.81	0.02%	0/8	-
	NMDT	33.55	0.09%	0/8	-
	DNMDT	10.60	0.08%	0/8	-
	T-NMDT	11.14	0.01%	4/8	-
	T-DNMDT	7.43	0.01%	3/8	-
contested	Bin2	4262.5	0.36%	2/12	6/12
	Bin3	3815.1	0.25%	3/12	7/12
	HybU	3463.2	0.24%	8/12	7/12
	NMDT	14259.6	6.33%	1/12	11/12
	DNMDT	5407.2	0.81%	1/12	7/12
	T-NMDT	1016.2	3.76%	1/12	7/12
	T-DNMDT	2231.2	0.83%	3/12	6/12
unsolved	Bin2	-	28.98%	1/39	-
	Bin3	-	28.46%	7/39	-
	HybU	-	28.25%	27/39	-
	NMDT	-	29.50%	1/39	-
	DNMDT	-	30.05%	0/39	-
	T-NMDT	-	29.22%	2/39	-
	T-DNMDT	-	29.21%	1/39	-

Table 5.C.5: Performance comparison of methods on ACOPF instances with $L = 5$, with callback function, 4-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	8.42	0.011%	1/8	0/8
	Bin3	7.21	0.008%	3/8	0/8
	Univar	5.75	0.007%	3/8	0/8
	NMDT	39.52	0.027%	0/8	0/8
	DNMDT	7.80	0.028%	0/8	0/8
	T-NMDT	13.88	0.007%	4/8	0/8
	T-DNMDT	4.77	0.008%	3/8	0/8
contested	Bin2	879.1	0.027%	5/8	2/8
	Bin3	977.5	0.206%	2/8	3/8
	Univar	1279.2	0.032%	4/8	2/8
	NMDT	14400.8	1.139%	0/8	8/8
	DNMDT	3490.7	0.409%	2/8	3/8
	T-NMDT	5025.3	0.188%	4/8	4/8
	T-DNMDT	994.1	0.181%	4/8	1/8
unsolved	Bin2	-	28.222%	1/43	-
	Bin3	-	27.816%	8/43	-
	Univar	-	26.990%	33/43	-
	NMDT	-	29.171%	1/43	-
	DNMDT	-	29.133%	0/43	-
	T-NMDT	-	28.988%	0/43	-
	T-DNMDT	-	28.805%	0/43	-

Table 5.C.6: Performance comparison of methods on ACOPF instances with $L = 6$, no callback function, 4-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	301.74	0.00%	6/8	-
	Bin3	161.90	0.00%	6/8	-
	HybU	104.93	0.00%	8/8	-
	NMDT	0.65	0.00%	7/8	-
	DNMDT	0.51	0.00%	7/8	-
	T-NMDT	1.05	0.00%	8/8	-
	T-DNMDT	0.86	0.00%	7/8	-
contested	Bin2	7200.0	24.09%	0/41	41/41
	Bin3	6043.4	5.92%	5/41	36/41
	HybU	5787.4	6.25%	5/41	35/41
	NMDT	211.1	0.00%	23/41	2/41
	DNMDT	62.1	0.00%	23/41	2/41
	T-NMDT	372.8	0.00%	34/41	3/41
	T-DNMDT	103.6	0.00%	35/41	3/41
unsolved	Bin2	-	98.22%	0/50	-
	Bin3	-	94.11%	0/50	-
	HybU	-	114.11%	0/50	-
	NMDT	-	44.35%	18/50	-
	DNMDT	-	46.91%	19/50	-
	T-NMDT	-	53.67%	0/50	-
	T-DNMDT	-	48.62%	13/50	-

Table 5.C.7: Performance comparison of methods on BoxQP instances with $L = 1$, with no callback function, 2-hour time limit.

family	method	time (sec)	gap	BB	TO
solved	Bin2	562.37	0.00%	9/9	-
	Bin3	286.08	0.00%	9/9	-
	HybU	149.59	0.00%	9/9	-
	NMDT	1.83	0.00%	8/9	-
	DNMDT	1.47	0.00%	8/9	-
	T-NMDT	2.45	0.00%	9/9	-
	T-DNMDT	2.09	0.00%	9/9	-
contested	Bin2	7200.0	25.80%	0/34	34/34
	Bin3	7186.2	17.01%	1/34	33/34
	HybU	6243.3	4.83%	5/34	29/34
	NMDT	686.9	0.01%	17/34	8/34
	DNMDT	288.9	0.00%	22/34	0/34
	T-NMDT	1024.4	0.00%	21/34	10/34
	T-DNMDT	443.1	0.00%	33/34	1/34
unsolved	Bin2	-	90.88%	0/56	-
	Bin3	-	86.09%	2/56	-
	HybU	-	103.25%	0/56	-
	NMDT	-	61.89%	15/56	-
	DNMDT	-	49.55%	20/56	-
	T-NMDT	-	65.50%	4/56	-
	T-DNMDT	-	56.37%	15/56	-

Table 5.C.8: Performance comparison of methods on BoxQP instances with $L = 2$, with no callback function, 2-hour time limit.

family	method	time (sec)	bnd gap	sol gap	BB	TO
solved	Bin2	603.24	0.00%	0.000%	7/7	0/7
	Bin3	382.47	0.00%	0.000%	6/7	0/7
	Univar	149.79	0.00%	0.000%	7/7	0/7
	NMDT	11.13	0.00%	0.000%	7/7	0/7
	DNMDT	12.26	0.00%	0.000%	7/7	0/7
	T-NMDT	15.08	0.00%	0.000%	7/7	0/7
	T-DNMDT	18.44	0.00%	0.000%	7/7	0/7
contested	Bin2	14400.0	12.72%	0.000%	0/15	15/15
	Bin3	11750.7	2.41%	0.000%	2/15	13/15
	Univar	6850.6	0.15%	0.000%	6/15	9/15
	NMDT	888.0	0.00%	0.000%	13/15	0/15
	DNMDT	807.0	0.00%	0.000%	15/15	0/15
	T-NMDT	775.9	0.00%	0.000%	15/15	0/15
	T-DNMDT	877.7	0.00%	0.000%	15/15	0/15
unsolved	Bin2	-	83.43%	0.002%	7/77	-
	Bin3	-	79.98%	0.004%	11/77	-
	Univar	-	85.99%	0.004%	2/77	-
	NMDT	-	71.16%	0.003%	3/77	-
	DNMDT	-	66.60%	0.002%	15/77	-
	T-NMDT	-	70.02%	0.003%	7/77	-
	T-DNMDT	-	66.19%	0.002%	33/77	-

Table 5.C.9: Performance comparison of methods on BoxQP instances with $L = 5$, with callback function, 4-hour time limit.

family	method	time (sec)	bnd gap	sol gap	BB	TO
solved	Bin2	459.48	0.00%	0.000%	7/7	0/7
	Bin3	294.60	0.00%	0.000%	6/7	0/7
	Univar	139.26	0.00%	0.000%	7/7	0/7
	NMDT	18.41	0.00%	0.000%	6/7	0/7
	DNMDT	14.45	0.00%	0.000%	6/7	0/7
	T-NMDT	14.85	0.00%	0.000%	7/7	0/7
	T-DNMDT	12.66	0.00%	0.000%	7/7	0/7
contested	Bin2	14400.0	12.75%	0.000%	0/15	15/15
	Bin3	12998.6	4.83%	0.000%	1/15	14/15
	Univar	8058.9	0.16%	0.000%	6/15	9/15
	NMDT	1071.2	0.00%	0.000%	14/15	0/15
	DNMDT	1178.1	0.00%	0.000%	15/15	0/15
	T-NMDT	947.3	0.00%	0.000%	15/15	0/15
	T-DNMDT	1147.2	0.00%	0.000%	15/15	0/15
unsolved	Bin2	-	83.92%	0.002%	6/77	-
	Bin3	-	80.93%	0.002%	11/77	-
	Univar	-	86.53%	0.003%	3/77	-
	NMDT	-	73.37%	0.002%	0/77	-
	DNMDT	-	68.93%	0.002%	18/77	-
	T-NMDT	-	71.68%	0.002%	9/77	-
	T-DNMDT	-	68.39%	0.002%	30/77	-

Table 5.C.10: Performance comparison of methods on BoxQP instances with $L = 6$, with callback function, 4-hour time limit.

Bibliography

[1] P. M. Castro, "Normalized multiparametric disaggregation: An efficient relaxation for mixed-integer bilinear problems," *Journal of Global Optimization*, vol. 64, no. 4, pp. 765–784, Jul. 2015. DOI: 10.1007/s10898-015-0342-z. [Online]. Available: https://doi.org/10.1007/s10898-015-0342-z.

[2] B. Beach, R. Hildebrand, K. Ellis, and B. Lebreton, "An approximate method for the optimization of long-horizon tank blending and scheduling operations," *Computers & Chemical Engineering*, vol. 141, p. 106 839, Oct. 2020. DOI: 10.1016/j.compchemeng.2020.106839. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2020.106839.

[3] D. Balaban, J. Cooper, and E. Komendera, "Inverse kinematics and sensitivity minimization of an n-stack stewart platform," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Nov. 2019, pp. 6794–6799. DOI: 10.1109/iros40897.2019.8968190. [Online]. Available: https://doi.org/10.1109/iros40897.2019.8968190.

[4] L. H. Andreas Bärmann Robert Burlacu and T. Kleinert, *On piecewise linear approximations of bilinear terms: Structural comparison of univariate and bivariate mixed-integer programming formulations*, 2021. eprint: Optimization-online. [Online]. Available: %5Curl%7Bhttp://www.optimization-online.org/DB_HTML/2021/08/8536.html%7D.

[5] M. W. J. Rohde Herbert Meyr, "Die supply chain planning matrix," *PPS Management*, vol. 5, no. 1, Jan. 2000.

[6] J. P. Teles, P. M. Castro, and H. A. Matos, "Multi-parametric disaggregation technique for global optimization of polynomial programming problems," *Journal of Global Optimization*, vol. 55, no. 2, pp. 227–251, Nov. 2011. DOI: 10.1007/s10898-011-9809-8. [Online]. Available: https://doi.org/10.1007/s10898-011-9809-8.

[7] J. Li, I. A. Karimi, and R. Srinivasan, "Recipe determination and scheduling of gasoline blending operations," *AIChE Journal*, NA–NA, 2009. DOI: 10.1002/aic.11970. [Online]. Available: https://doi.org/10.1002/aic.11970.

[8] Y. Zhang, D. Monder, and J. F. Forbes, "Real-time optimization under parametric uncertainty: A probability constrained approach," *Journal of Process Control*, vol. 12, no. 3, pp. 373–389, Apr. 2002. DOI: 10.1016/s0959-1524(01)00047-6. [Online]. Available: https://doi.org/10.1016/s0959-1524(01)00047-6.

[9] A. Singh, J. Forbes, P. Vermeer, and S. Woo, "Model-based real-time optimization of automotive gasoline blending operations," *Journal of Process Control*, vol. 10, no. 1, pp. 43–58, Feb. 2000. DOI: 10.1016/s0959-1524(99)00037-2. [Online]. Available: https://doi.org/10.1016/s0959-1524(99)00037-2.

[10] R. Misener and C. A. Floudas, "Advances for the pooling problem: Modeling, global optimization, and computational studies," *Applied and Computational Mathematics*, vol. 8, no. 1, pp. 3–22, 2009.

[11] C. Cuiwen, G. Xingsheng, and X. Zhong, "A data-driven rolling-horizon online scheduling model for diesel production of a real-world refinery," *AIChE Journal*, vol. 59, no. 4, pp. 1160–1174, Sep. 2012. DOI: 10.1002/aic.13895. [Online]. Available: https://doi.org/10.1002/aic.13895.

[12] S. Mouret, "Optimal scheduling of refinery crude-oil operations," Ph.D. dissertation, Carnegie Mellon University, Mar. 2010. [Online]. Available: https://figshare.com/articles/Optimal_Scheduling_of_Refinery_Crude-Oil_Operations/6720965.

[13] T. A. Oddsdottir, M. Grunow, and R. Akkerman, "Procurement planning in oil refining industries considering blending operations," *Computers & Chemical Engineering*, vol. 58, pp. 1–13, Nov. 2013. DOI: 10.1016/j.compchemeng.2013.05.006. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2013.05.006.

[14] X. Chen, S. Huang, D. Chen, *et al.*, "Hierarchical decomposition approach for crude oil scheduling: A SINOPEC case," *Interfaces*, vol. 44, no. 3, pp. 269–285, Jun. 2014. DOI: 10.1287/inte.2014.0744. [Online]. Available: https://doi.org/10.1287/inte.2014.0744.

[15] J. Cerdá, P. C. Pautasso, and D. C. Cafaro, "Efficient approach for scheduling crude oil operations in marine-access refineries," *Industrial & Engineering Chemistry Research*, vol. 54, no. 33, pp. 8219–8238, Aug. 2015. DOI: 10.1021/acs.iecr.5b01461. [Online]. Available: https://doi.org/10.1021/acs.iecr.5b01461.

[16] P. M. Castro, "Source-based discrete and continuous-time formulations for the crude oil pooling problem," *Computers & Chemical Engineering*, vol. 93, pp. 382–401, Oct. 2016. DOI: 10.1016/j.compchemeng.2016.06.016. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2016.06.016.

[17] J. Xu, S. Zhang, J. Zhang, S. Wang, and Q. Xu, "Simultaneous scheduling of front-end crude transfer and refinery processing," *Computers & Chemical Engineering*, vol. 96, pp. 212–236, Jan. 2017. DOI: 10.1016/j.compchemeng.2016.09.019. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2016.09.019.

[18] J. Pinto and L. L. Moro, "A planning model for petroleum refineries," *Brazilian Journal of Chemical Engineering*, vol. 17, no. 4-7, pp. 575–586, Dec. 2000. DOI: 10.1590/s0104-66322000000400022. [Online]. Available: https://doi.org/10.1590/s0104-66322000000400022.

[19] P. C. Castillo, P. M. Castro, and V. Mahalec, "Global optimization algorithm for large-scale refinery planning models with bilinear terms," *Industrial & Engineering Chemistry Research*,

vol. 56, no. 2, pp. 530–548, Jan. 2017. DOI: 10.1021/acs.iecr.6b01350. [Online]. Available: https://doi.org/10.1021/acs.iecr.6b01350.

[20] L. F. L. Moro and J. M. Pinto, "Mixed-integer programming approach for short-term crude oil scheduling," *Industrial & Engineering Chemistry Research*, vol. 43, no. 1, pp. 85–94, Jan. 2004. DOI: 10.1021/ie030348d. [Online]. Available: https://doi.org/10.1021/ie030348d.

[21] C. A. Méndez, I. E. Grossmann, I. Harjunkoski, and P. Kaboré, "A simultaneous optimization approach for off-line blending and scheduling of oil-refinery operations," *Computers & Chemical Engineering*, vol. 30, no. 4, pp. 614–634, Feb. 2006. DOI: 10.1016/j.compchemeng.2005.11.004. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2005.11.004.

[22] S. P. Kolodziej, I. E. Grossmann, K. C. Furman, and N. W. Sawaya, "A discretization-based approach for the optimization of the multiperiod blend scheduling problem," *Computers & Chemical Engineering*, vol. 53, pp. 122–142, Jun. 2013. DOI: 10.1016/j.compchemeng.2013.01.016. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2013.01.016.

[23] C. Luo and G. Rong, "A strategy for the integration of production planning and scheduling in refineries under uncertainty," *Chinese Journal of Chemical Engineering*, vol. 17, no. 1, pp. 113–127, Feb. 2009. DOI: 10.1016/s1004-9541(09)60042-2. [Online]. Available: https://doi.org/10.1016/s1004-9541(09)60042-2.

[24] S. Lu, H. Su, Y. Wang, L. Xie, and Q. Zhang, "Multi-product multi-stage production planning with lead time on a rolling horizon basis," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 1162–1167, 2015. DOI: 10.1016/j.ifacol.2015.09.125. [Online]. Available: https://doi.org/10.1016/j.ifacol.2015.09.125.

[25] S. Torkaman, S. F. Ghomi, and B. Karimi, "Multi-stage multi-product multi-period production planning with sequence-dependent setups in closed-loop supply chain," *Computers & Industrial Engineering*, vol. 113, pp. 602–613, Nov. 2017. DOI: 10.1016/j.cie.2017.09.040. [Online]. Available: https://doi.org/10.1016/j.cie.2017.09.040.

[26] L. S. Dias and M. G. Ierapetritou, "Integration of scheduling and control under uncertainties: Review and challenges," *Chemical Engineering Research and Design*, vol. 116, pp. 98–113, Dec. 2016. DOI: 10.1016/j.cherd.2016.10.047. [Online]. Available: https://doi.org/10.1016/j.cherd.2016.10.047.

[27] M. A. Gutiérrez-Limón, A. Flores-Tlacuahuac, and I. E. Grossmann, "MINLP formulation for simultaneous planning, scheduling, and control of short-period single-unit processing systems," *Industrial & Engineering Chemistry Research*, vol. 53, no. 38, pp. 14 679–14 694, Sep. 2014. DOI: 10.1021/ie402563j. [Online]. Available: https://doi.org/10.1021/ie402563j.

[28] ——, "A reactive optimization strategy for the simultaneous planning, scheduling and control of short-period continuous reactors," *Computers & Chemical Engineering*, vol. 84, pp. 507–515, Jan. 2016. DOI: 10.1016/j.compchemeng.2015.09.017. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2015.09.017.

[29] P. C. P. Reddy, I. A. Karimi, and R. Srinivasan, "Novel solution approach for optimizing crude oil operations," *AIChE Journal*, vol. 50, no. 6, pp. 1177–1197, 2004. DOI: 10.1002/aic.10112. [Online]. Available: https://doi.org/10.1002/aic.10112.

[30] ——, "A new continuous-time formulation for scheduling crude oil operations," *Chemical Engineering Science*, vol. 59, no. 6, pp. 1325–1341, Mar. 2004. DOI: 10.1016/j.ces.2004.01.009. [Online]. Available: https://doi.org/10.1016/j.ces.2004.01.009.

[31] J. Li, W. Li, I. A. Karimi, and R. Srinivasan, "Improving the robustness and efficiency of crude scheduling algorithms," *AIChE Journal*, vol. 53, no. 10, pp. 2659–2680, 2007. DOI: 10.1002/aic.11280. [Online]. Available: https://doi.org/10.1002/aic.11280.

[32] S. P. Kolodziej, I. E. Grossmann, K. C. Furman, and N. W. Sawaya, "A novel global optimization approach to the multiperiod blending problem," in *Computer Aided Chemical Engineering*, Elsevier, 2012, pp. 1492–1496. DOI: 10.1016/b978-0-444-59506-5.50129-2. [Online]. Available: https://doi.org/10.1016/b978-0-444-59506-5.50129-2.

[33] P. M. Castro, "New MINLP formulation for the multiperiod pooling problem," *AIChE Journal*, vol. 61, no. 11, pp. 3728–3738, Sep. 2015. DOI: 10.1002/aic.15018. [Online]. Available: https://doi.org/10.1002/aic.15018.

[34] I. Lotero, F. Trespalacios, I. E. Grossmann, D. J. Papageorgiou, and M.-S. Cheon, "An MILP-MINLP decomposition method for the global optimization of a source based model of the multiperiod blending problem," *Computers & Chemical Engineering*, vol. 87, pp. 13–35, Apr. 2016. DOI: 10.1016/j.compchemeng.2015.12.017. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2015.12.017.

[35] P. M. Castro and I. E. Grossmann, "Global optimal scheduling of crude oil blending operations with RTN continuous-time and multiparametric disaggregation," *Industrial & Engineering Chemistry Research*, vol. 53, no. 39, pp. 15 127–15 145, Sep. 2014. DOI: 10.1021/ie503002k. [Online]. Available: https://doi.org/10.1021/ie503002k.

[36] X. Gao, Y. Jiang, T. Chen, and D. Huang, "Optimizing scheduling of refinery operations based on piecewise linear models," *Computers & Chemical Engineering*, vol. 75, pp. 105–119, Apr. 2015. DOI: 10.1016/j.compchemeng.2015.01.022. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2015.01.022.

[37] Z. Li and M. G. Ierapetritou, "Rolling horizon based planning and scheduling integration with production capacity consideration," *Chemical Engineering Science*, vol. 65, no. 22, pp. 5887–5900, Nov. 2010. DOI: 10.1016/j.ces.2010.08.010. [Online]. Available: https://doi.org/10.1016/j.ces.2010.08.010.

[38] J. Li and I. A. Karimi, "Scheduling gasoline blending operations from recipe determination to shipping using unit slots," *Industrial & Engineering Chemistry Research*, vol. 50, no. 15, pp. 9156–9174, Aug. 2011. DOI: 10.1021/ie102321b. [Online]. Available: https://doi.org/10.1021/ie102321b.

[39] L. S. de Assis, E. Camponogara, B. Zimberg, E. Ferreira, and I. E. Grossmann, "A piecewise McCormick relaxation-based strategy for scheduling operations in a crude oil terminal,"

Computers & Chemical Engineering, vol. 106, pp. 309–321, Nov. 2017. DOI: 10.1016/j.compchemeng.2017.06.012. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2017.06.012.

[40] S. Neiro and J. Pinto, "Lagrangean decomposition applied to multiperiod planning of petroleum refineries under uncertainty," *Latin American Applied Research*, vol. 36, pp. 213–220, Oct. 2006.

[41] V. Gupta and I. E. Grossmann, "Multistage stochastic programming approach for offshore oilfield infrastructure planning under production sharing agreements and endogenous uncertainties," *Journal of Petroleum Science and Engineering*, vol. 124, pp. 180–197, Dec. 2014. DOI: 10.1016/j.petrol.2014.10.006. [Online]. Available: https://doi.org/10.1016/j.petrol.2014.10.006.

[42] Y. Chu, F. You, J. M. Wassick, and A. Agarwal, "Integrated planning and scheduling under production uncertainties: Bi-level model formulation and hybrid solution method," *Computers & Chemical Engineering*, vol. 72, pp. 255–272, Jan. 2015. DOI: 10.1016/j.compchemeng.2014.02.023. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2014.02.023.

[43] C. Ning and F. You, "A data-driven multistage adaptive robust optimization framework for planning and scheduling under uncertainty," *AIChE Journal*, vol. 63, no. 10, pp. 4343–4369, May 2017. DOI: 10.1002/aic.15792. [Online]. Available: https://doi.org/10.1002/aic.15792.

[44] H. Lee, J. M. Pinto, I. E. Grossmann, and S. Park, "Mixed-integer linear programming model for refinery short-term scheduling of crude oil unloading with inventory management," *Industrial & Engineering Chemistry Research*, vol. 35, no. 5, pp. 1630–1641, Jan. 1996. DOI: 10.1021/ie950519h. [Online]. Available: https://doi.org/10.1021/ie950519h.

[45] S. Mouret, I. E. Grossmann, and P. Pestiaux, "A novel priority-slot based continuous-time formulation for crude-oil scheduling problems," *Industrial & Engineering Chemistry Research*,

vol. 48, no. 18, pp. 8515–8528, Sep. 2009. DOI: 10.1021/ie8019592. [Online]. Available: https://doi.org/10.1021/ie8019592.

[46] L. Wenkai, C.-W. Hui, B. Hua, and Z. Tong, "Scheduling crude oil unloading, storage, and processing," *Industrial & Engineering Chemistry Research*, vol. 41, no. 26, pp. 6723–6734, Dec. 2002. DOI: 10.1021/ie020130b. [Online]. Available: https://doi.org/10.1021/ie020130b.

[47] J. D. Kelly and J. L. Mann, "Crude oil blend scheduling optimization: An application with multimillion dollar benefits - part 1," *Hydrocarbon Processing, International Edition*, vol. 82, no. 6, pp. 47–53, Jun. 2003. [Online]. Available: https://www.tib.eu/en/search/id/ceaba%5C%3ACEAB20030701296/Crude-oil-blend-scheduling-optimization-An-application.

[48] R. Karuppiah, K. C. Furman, and I. E. Grossmann, "Global optimization for scheduling refinery crude oil operations," *Computers & Chemical Engineering*, vol. 32, no. 11, pp. 2745–2766, Nov. 2008. DOI: 10.1016/j.compchemeng.2007.11.008. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2007.11.008.

[49] J. Silvente, G. M. Kopanos, E. N. Pistikopoulos, and A. Espuña, "A rolling horizon optimization framework for the simultaneous energy supply and demand planning in microgrids," *Applied Energy*, vol. 155, pp. 485–501, Oct. 2015. DOI: 10.1016/j.apenergy.2015.05.090. [Online]. Available: https://doi.org/10.1016/j.apenergy.2015.05.090.

[50] R. Stolletz and E. Z. de Acha, "A rolling planning horizon heuristic for scheduling agents with different qualifications," *SSRN Electronic Journal*, 2014. DOI: 10.2139/ssrn.2401045. [Online]. Available: https://doi.org/10.2139/ssrn.2401045.

[51] J. F. Marquant, R. Evins, and J. Carmeliet, "Reducing computation time with a rolling horizon approach applied to a MILP formulation of multiple urban energy hub system," *Procedia Computer Science*, vol. 51, pp. 2137–2146, 2015. DOI: 10.1016/j.procs.2015.05.486. [Online]. Available: https://doi.org/10.1016/j.procs.2015.05.486.

[52] B. R. Champion and S. A. Gabriel, "A multistage stochastic energy model with endogenous probabilities and a rolling horizon," *Energy and Buildings*, vol. 135, pp. 338–349, Jan. 2017. DOI: 10.1016/j.enbuild.2016.11.058. [Online]. Available: https://doi.org/10.1016/j.enbuild.2016.11.058.

[53] J. Silvente, G. M. Kopanos, V. Dua, and L. G. Papageorgiou, "A rolling horizon approach for optimal management of microgrids under stochastic uncertainty," *Chemical Engineering Research and Design*, vol. 131, pp. 293–317, Mar. 2018. DOI: 10.1016/j.cherd.2017.09.013. [Online]. Available: https://doi.org/10.1016/j.cherd.2017.09.013.

[54] E. Zondervan, M. Kaland, M. A. van Elzakker, J. C. Fransoo, and J. Meuldijk, "Integrating planning and scheduling in an oil refinery with a rolling horizon approach," in *Computer Aided Chemical Engineering*, vol. 33, Elsevier, 2014, pp. 439–444. DOI: 10.1016/b978-0-444-63456-6.50074-0. [Online]. Available: https://doi.org/10.1016/b978-0-444-63456-6.50074-0.

[55] M. Joly and J. Pinto, "Mixed-integer programming techniques for the scheduling of fuel oil and asphalt production," *Chemical Engineering Research and Design*, vol. 81, no. 4, pp. 427–447, Apr. 2003. DOI: 10.1205/026387603765173691. [Online]. Available: https://doi.org/10.1205/026387603765173691.

[56] V. Pham, C. Laird, and M. El-Halwagi, "Convex hull discretization approach to the global optimization of pooling problems," *Industrial & Engineering Chemistry Research*, vol. 48, no. 4, pp. 1973–1979, Feb. 2009. DOI: 10.1021/ie8003573. [Online]. Available: https://doi.org/10.1021/ie8003573.

[57] G. P. McCormick, "Computability of global solutions to factorable nonconvex programs: Part i — convex underestimating problems," *Mathematical Programming*, vol. 10, no. 1, pp. 147–175, Dec. 1976. DOI: 10.1007/bf01580665. [Online]. Available: https://doi.org/10.1007/bf01580665.

[58] M. L. Bergamini, P. Aguirre, and I. Grossmann, "Logic-based outer approximation for globally optimal synthesis of process networks," *Computers & Chemical Engineering*, vol. 29, no. 9, pp. 1914–1933, Aug. 2005. DOI: 10.1016/j.compchemeng.2005.04.003. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2005.04.003.

[59] S. Mouret, I. E. Grossmann, and P. Pestiaux, "Tightening the linear relaxation of a mixed integer nonlinear program using constraint programming," in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer Berlin Heidelberg, 2009, pp. 208–222. DOI: 10.1007/978-3-642-01929-6_16. [Online]. Available: https://doi.org/10.1007/978-3-642-01929-6_16.

[60] P. M. Castro, "Tightening piecewise McCormick relaxations for bilinear problems," *Computers & Chemical Engineering*, vol. 72, pp. 300–311, Jan. 2015. DOI: 10.1016/j.compchemeng.2014.03.025. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2014.03.025.

[61] C. E. Gounaris, R. Misener, and C. A. Floudas, "Computational comparison of piecewise-linear relaxations for pooling problems," *Industrial & Engineering Chemistry Research*, vol. 48, no. 12, pp. 5742–5766, Jun. 2009. DOI: 10.1021/ie8016048. [Online]. Available: https://doi.org/10.1021/ie8016048.

[62] S. Kolodziej, P. M. Castro, and I. E. Grossmann, "Global optimization of bilinear programs with a multiparametric disaggregation technique," *Journal of Global Optimization*, vol. 57, no. 4, pp. 1039–1063, Jan. 2013. DOI: 10.1007/s10898-012-0022-1. [Online]. Available: https://doi.org/10.1007/s10898-012-0022-1.

[63] P. M. Castro and I. E. Grossmann, "Optimality-based bound contraction with multiparametric disaggregation for the global optimization of mixed-integer bilinear problems," *Journal of Global Optimization*, vol. 59, no. 2-3, pp. 277–306, Mar. 2014. DOI: 10.1007/s10898-014-0162-6. [Online]. Available: https://doi.org/10.1007/s10898-014-0162-6.

[64] J. P. Vielma and G. L. Nemhauser, "Modeling disjunctive constraints with a logarithmic number of binary variables and constraints," *Mathematical Programming*, vol. 128, no. 1-2, pp. 49–72, Jul. 2009. DOI: 10.1007/s10107-009-0295-4. [Online]. Available: https://doi.org/10.1007/s10107-009-0295-4.

[65] R. Misener, J. P. Thompson, and C. A. Floudas, "APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes," *Computers & Chemical Engineering*, vol. 35, no. 5, pp. 876–892, May 2011. DOI: 10.1016/j.compchemeng.2011.01.026. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2011.01.026.

[66] A. Gupte, S. Ahmed, M. S. Cheon, and S. Dey, "Solving mixed integer bilinear problems using MILP formulations," *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 721–744, Jan. 2013. DOI: 10.1137/110836183. [Online]. Available: https://doi.org/10.1137/110836183.

[67] A. Gupte, S. Ahmed, S. S. Dey, and M. S. Cheon, "Relaxations and discretizations for the pooling problem," *Journal of Global Optimization*, vol. 67, no. 3, pp. 631–669, Apr. 2016. DOI: 10.1007/s10898-016-0434-4. [Online]. Available: https://doi.org/10.1007/s10898-016-0434-4.

[68] I. E. Grossmann and J. P. Ruiz, "Generalized disjunctive programming: A framework for formulation and alternative algorithms for MINLP optimization," in *Mixed Integer Nonlinear Programming*, Springer New York, Nov. 2011, pp. 93–115. DOI: 10.1007/978-1-4614-1927-3_4. [Online]. Available: https://doi.org/10.1007/978-1-4614-1927-3_4.

[69] J. P. Ruiz and I. E. Grossmann, "Strengthening of lower bounds in the global optimization of bilinear and concave generalized disjunctive programs," *Computers & Chemical Engineering*, vol. 34, no. 6, pp. 914–930, Jun. 2010. DOI: 10.1016/j.compchemeng.2009.10.016. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2009.10.016.

[70] H. Chen, W. Chen, and J. Liu, "Optimal design of stewart platform safety mechanism," *Chinese Journal of Aeronautics*, vol. 20, no. 4, pp. 370–377, Aug. 2007. DOI: 10.1016/s1000-

9361(07)60057-0. [Online]. Available: https://doi.org/10.1016/s1000-9361(07)60057-0.

[71] L. Bangjun, P. Likun, and M. Tingtao, "Improving dynamic performance of stewart platforms through optimal design based on evolutionary multi-objective optimization algorithms," in *Proceedings of the 1st International Conference on Mechanical Engineering and Material Science*, Atlantis Press, 2012, pp. 294–298, ISBN: 978-90-78677-59-8. DOI: 10.2991/mems.2012.78. [Online]. Available: https://doi.org/10.2991/mems.2012.78.

[72] M. Toz and S. Kucuk, "Dexterous workspace optimization of an asymmetric six-degree of freedom stewart–gough platform type manipulator," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1516–1528, Dec. 2013. DOI: 10.1016/j.robot.2013.07.004. [Online]. Available: https://doi.org/10.1016/j.robot.2013.07.004.

[73] Z. Lei and D. Xiaolin, "Optimize the redundant 6-DOF stewart platform based on ant colony optimization," in *Proceedings of 2013 3rd International Conference on Computer Science and Network Technology*, IEEE, Oct. 2013, pp. 1238–1241. DOI: 10.1109/iccsnt.2013.6967326. [Online]. Available: https://doi.org/10.1109/iccsnt.2013.6967326.

[74] Z. Xie, G. Li, G. Liu, and J. Zhao, "Optimal design of a stewart platform using the global transmission index under determinate constraint of workspace," *Advances in Mechanical Engineering*, vol. 9, no. 10, p. 1 687 814 017 720 880, Oct. 2017. DOI: 10.1177/1687814017720880. [Online]. Available: https://doi.org/10.1177/1687814017720880.

[75] T. Sun and B. Lian, "Stiffness and mass optimization of parallel kinematic machine," *Mechanism and Machine Theory*, vol. 120, pp. 73–88, Feb. 2018. DOI: 10.1016/j.mechmachtheory.2017.09.014. [Online]. Available: https://doi.org/10.1016/j.mechmachtheory.2017.09.014.

[76] A. Ríos, E. E. Hernández, and S. I. Valdez, "A two-stage mono- and multi-objective method for the optimization of general UPS parallel manipulators," *Mathematics*, vol. 9, no. 5,

p. 543, Mar. 2021. DOI: 10.3390/math9050543. [Online]. Available: https://doi.org/10.3390/math9050543.

[77] B. Zhang, "Design and implementation of a 6 dof parallel manipulator with passive force control," Ph.D. dissertation, University of Florida, 2005.

[78] K. Yokoi, K. Komoriya, and K. Tanie, "A method for solving inverse kinematics of variable structure truss arm with high redundancy," *Journal of Intelligent Material Systems and Structures*, vol. 3, no. 4, pp. 631–645, Oct. 1992. DOI: 10.1177/1045389x9200300406. [Online]. Available: https://doi.org/10.1177/1045389x9200300406.

[79] R. L. Williams, "Survey of active truss modules," in *Volume 1: 21st Design Automation Conference*, American Society of Mechanical Engineers, Sep. 1995. DOI: 10.1115/detc1995-0118. [Online]. Available: https://doi.org/10.1115/detc1995-0118.

[80] J. Dorsey, T. Sutter, and K. Wu, "Structurally adaptive space crane concept for assembling space systems on orbit," NASA Langley Research Center, Tech. Rep., Nov. 1992.

[81] K. Miura and H. Furuya, "Adaptive structure concept for future space applications," *AIAA Journal*, vol. 26, no. 8, pp. 995–1002, Aug. 1988. DOI: 10.2514/3.10002. [Online]. Available: https://doi.org/10.2514/3.10002.

[82] S. M. LaValle, "Rapidly-exploring random trees : A new tool for path planning," *The annual research report*, 1998.

[83] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, IEEE, 2000, 995–1001 vol.2. DOI: 10.1109/robot.2000.844730. [Online]. Available: https://doi.org/10.1109/robot.2000.844730.

[84] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," vol. 30, no. 7, pp. 846–894, Jun. 2011. DOI: 10.1177/0278364911406761. [Online]. Available: https://doi.org/10.1177/0278364911406761.

[85] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, "RRT*-Smart: Rapid convergence implementation of RRT* towards optimal solution," IEEE, Aug. 2012. DOI: 10.1109/icma.2012.6284384. [Online]. Available: https://doi.org/10.1109/icma.2012.6284384.

[86] M. Zucker, N. Ratliff, A. Dragan, *et al.*, "Chomp: Covariant hamiltonian optimization for motion planning," *International Journal of Robotics Research*, vol. 32, no. 9, pp. 1164–1193, Aug. 2013.

[87] J. Schulman, Y. Duan, J. Ho, *et al.*, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014. DOI: 10.1177/0278364914528132. eprint: https://doi.org/10.1177/0278364914528132. [Online]. Available: https://doi.org/10.1177/0278364914528132.

[88] C. Quintero-Pena, A. Kyrillidis, and L. E. Kavraki, "Robust optimization-based motion planning for high-DOF robots under sensing uncertainty," IEEE, May 2021. DOI: 10.1109/icra48506.2021.9560917. [Online]. Available: https://doi.org/10.1109/icra48506.2021.9560917.

[89] C. C. Nguyen and S. Antrazi, "Trajectory planning and control of a 6 dof manipulator with stewart platform-based mechanism," NASA Goddard Space Flight Center, Tech. Rep., 1990. [Online]. Available: https://ntrs.nasa.gov/citations/19900016294.

[90] C. Nguyen, S. Antrazi, Z.-L. Zhou, and C. Campbell, "Experimental study of motion control and trajectory planning for a stewart platform robot manipulator," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, vol. 2, IEEE Comput. Soc. Press, 1991, 1873–1878 vol.2. DOI: 10.1109/robot.1991.131898. [Online]. Available: https://doi.org/10.1109/robot.1991.131898.

[91] J. Cortes and T. Simeon, "Probabilistic motion planning for parallel mechanisms," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*,

IEEE, 2003, pp. 4354–4359. DOI: 10.1109/robot.2003.1242274. [Online]. Available: https://doi.org/10.1109/robot.2003.1242274.

[92] P. Grosch, R. D. Gregorio, J. Lopez, and F. Thomas, "Motion planning for a novel reconfigurable parallel manipulator with lockable revolute joints," in *2010 IEEE International Conference on Robotics and Automation*, IEEE, May 2010, pp. 4697–4702. DOI: 10.1109/robot.2010.5509305. [Online]. Available: https://doi.org/10.1109/robot.2010.5509305.

[93] P. Wang, H. Yang, and K. Xue, "Jerk-optimal trajectory planning for stewart platform in joint space," in *2015 IEEE International Conference on Mechatronics and Automation (ICMA)*, IEEE, Aug. 2015, pp. 1932–1937. DOI: 10.1109/icma.2015.7237781. [Online]. Available: https://doi.org/10.1109/icma.2015.7237781.

[94] R. Ernandis, "Sampling based motion planning for minimizing position uncertainty with stewart platforms," Ph.D. dissertation, University of Maryland, College Park, 2021. DOI: 10.13016/P2OU-3IX9. [Online]. Available: https://drum.lib.umd.edu/handle/1903/26708.

[95] T. Osa, A. M. G. Esfahani, R. Stolkin, R. Lioutikov, J. Peters, and G. Neumann, "Guiding trajectory optimization by demonstrated distributions," vol. 2, no. 2, pp. 819–826, Apr. 2017. DOI: 10.1109/lra.2017.2653850. [Online]. Available: https://doi.org/10.1109/lra.2017.2653850.

[96] J. Ichnowski, Y. Avigal, V. Satish, and K. Goldberg, "Deep learning can accelerate grasp-optimized motion planning," vol. 5, no. 48, Nov. 2020. DOI: 10.1126/scirobotics.abd7710. [Online]. Available: https://doi.org/10.1126/scirobotics.abd7710.

[97] A. Dragan and S. Srinivasa, "Integrating human observer inferences into robot motion planning," vol. 37, no. 4, pp. 351–368, Aug. 2014. DOI: 10.1007/s10514-014-9408-x. [Online]. Available: https://doi.org/10.1007/s10514-014-9408-x.

[98] A. Volz and K. Graichen, "An optimization-based approach to dual-arm motion planning with closed kinematics," IEEE, Oct. 2018. DOI: 10.1109/iros.2018.8593927. [Online]. Available: https://doi.org/10.1109/iros.2018.8593927.

[99] W. Szynkiewicz and J. Błaszczyk, "Optimization-based approach to path planning for closed chain robot systems," vol. 21, no. 4, pp. 659–670, Dec. 2011. DOI: 10.2478/v10006-011-0052-8. [Online]. Available: https://doi.org/10.2478/v10006-011-0052-8.

[100] J. Ao, J. Santos, and M. Silva, "Investigation of motion planning methods with a kinematically redundant manipulator," Mar. 2017.

[101] Z. Bingul and O. Karah, "Dynamic modeling and simulation of stewart platform," in *Serial and Parallel Robot Manipulators - Kinematics, Dynamics, Control and Optimization*, InTech, Mar. 2012, pp. 19–42. DOI: 10.5772/32470. [Online]. Available: https://doi.org/10.5772/32470.

[102] D. Lazard and J.-P. Merlet, "The (true) stewart platform has 12 configurations," vol. 3, Jun. 1994, 2160–2165 vol.3, ISBN: 0-8186-5330-2. DOI: 10.1109/ROBOT.1994.350969.

[103] R. R. E. T. Charters and P. Freitas, "Detecting singularities of stewart platforms," 2009.

[104] K. M. Lynch and F. C. Park, *Modern robotics: mechanics, planning, and control*. Cambridge, UK: Cambridge University Press, 2017, pp. 140–152, OCLC: ocn983881868, ISBN: 978-1-107-15630-2.

[105] J. P. Merlet, *Parallel Robots*. Springer, 2006.

[106] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, 2020. DOI: https://doi.org/10.1038/s41592-019-0686-2.

[107] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," vol. 106, no. 1, pp. 25–57, Apr. 2005. DOI: 10.1007/s10107-004-0559-y. [Online]. Available: https://doi.org/10.1007/s10107-004-0559-y.

[108] W. E. Hart, J.-P. Watson, and D. L. Woodruff, "Pyomo: Modeling and solving mathematical programs in python," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 219–260, 2011.

[109] M. L. Bynum, G. A. Hackebeil, W. E. Hart, *et al.*, *Pyomo–optimization modeling in python*, Third. Springer Science & Business Media, 2021, vol. 67.

[110] D. Yarotsky, "Error bounds for approximations with deep relu networks," *Neural Networks*, vol. 94, pp. 103–114, 2017.

[111] F. Furini, E. Traversi, P. Belotti, *et al.*, "QPLIB: A library of quadratic programming instances," *Mathematical Programming Computation*, vol. 11, no. 2, pp. 237–265, Jun. 2019.

[112] E. Phan-huy-Hao, "Quadratically constrained quadratic programming: Some applications and a method for solution," *Zeitschrift für Operations Research*, vol. 26, no. 1, pp. 105–119, Jun. 1982.

[113] P. Pardalos and S. Vavasis, "Quadratic programming with one negative eigenvalue is NP-hard," *Journal of Global Optimization*, vol. 1, no. 1, pp. 15–22, Jun. 1991.

[114] S. Burer and A. Saxena, "The MILP road to MIQCP," in *Mixed Integer Nonlinear Programming*, J. Lee and S. Leyffer, Eds., Springer New York, 2012, pp. 373–405.

[115] H. Dong, "Relaxing nonconvex quadratic functions by multiple adaptive diagonal perturbations," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1962–1985, 2016.

[116] K. L. Croxton, B. Gendron, and T. L. Magnanti, "A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems," *Management Science*, vol. 49, no. 9, pp. 1268–1273, Sep. 2003.

[117] G. B. Dantzig, "On the significance of solving linear programming problems with some integer variables," *Econometrica, Journal of the Econometric Society*, pp. 30–44, 1960.

[118] J. Huchette and J. P. Vielma, "Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools," *Operations Research*, 2019, https://arxiv.org/abs/1708.00050.

[119] J. Lee and D. Wilson, "Polyhedral methods for piecewise-linear functions I: The lambda method," *Discrete Applied Mathematics*, vol. 108, pp. 269–285, 2001.

[120] T. L. Magnanti and D. Stratila, "Separable concave optimization approximately equals piecewise linear optimization," in *Lecture Notes in Computer Science*, D. Bienstock and G. Nemhauser, Eds., vol. 3064, Springer, 2004, pp. 234–243.

[121] M. Padberg, "Approximating separable nonlinear functions via mixed zero-one programs," *Operations Research Letters*, vol. 27, pp. 1–5, 2000.

[122] J. P. Vielma, S. Ahmed, and G. Nemhauser, "Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions," *Operations Research*, vol. 58, no. 2, pp. 303–315, 2010.

[123] R. Anderson, J. Huchette, C. Tjandraatmadja, and J. P. Vielma, "Strong mixed-integer programming formulations for trained neural networks," in *Proceedings of the 20th Conference on Integer Programming and Combinatorial Optimization*, A. Lodi and V. Nagarajan, Eds., https://arxiv.org/abs/1811.08359, Cham: Springer International Publishing, 2019, pp. 27–42.

[124] R. Bunel, J. Lu, I. Turkaslan, P. H. Torr, P. Kohli, and M. P. Kumar, "Branch and bound for piecewise linear neural network verification," https://arxiv.org/abs/1909.06588, 2019.

[125] T. Serra and S. Ramalingam, "Empirical bounds on linear regions of deep rectifier networks," https://arxiv.org/abs/1810.03370, 2018.

[126] T. Serra, C. Tjandraatmadja, and S. Ramalingam, "Bounding and counting linear regions of deep neural networks," in *Thirty-fifth International Conference on Machine Learning*, 2018.

[127] V. Tjeng, K. Xiao, and R. Tedrake, "Verifying neural networks with mixed integer programming," in *International Conference on Learning Representations*, 2019.

[128] R. Anderson, J. Huchette, C. Tjandraatmadja, and J. P. Vielma, "Strong mixed-integer programming formulations for trained neural networks," in *Integer Programming and Combinatorial Optimization*, A. Lodi and V. Nagarajan, Eds., Cham: Springer International Publishing, 2019, pp. 27–42, ISBN: 978-3-030-17953-3.

[129] H. Dong and Y. Luo, *Compact disjunctive approximations to nonconvex quadratically constrained programs*, 2018. eprint: arXiv:1811.08122.

[130] A. Saxena, P. Bomani, and J. Lee, "Convex relaxations of non-convex mixed integer quadratically constrained programs: Projected formulations," *Mathematical Programming*, vol. 130, pp. 359–413, 2011.

[131] A. Billionnet, S. Elloumi, and A. Lambert, "Extending the QCR method to general mixed-integer programs," *Mathematical Programming*, vol. 131, no. 1-2, pp. 381–401, May 2012. DOI: 10.1007/s10107-010-0381-7. [Online]. Available: https://doi.org/10.1007/s10107-010-0381-7.

[132] ——, "Exact quadratic convex reformulations of mixed-integer quadratically constrained problems," *Mathematical Programming*, vol. 158, no. 1, pp. 235–266, Jul. 2016, ISSN: 1436-4646. DOI: 10.1007/s10107-015-0921-2. [Online]. Available: https://doi.org/10.1007/s10107-015-0921-2.

[133] S. Elloumi and A. Lambert, "Global solution of non-convex quadratically constrained quadratic programs," *Optimization Methods and Software*, vol. 34, no. 1, pp. 98–114, 2019. DOI: 10.1080/10556788.2017.1350675. eprint: https://doi.org/10.1080/10556788.2017.1350675. [Online]. Available: https://doi.org/10.1080/10556788.2017.1350675.

[134] L. Galli and A. N. Letchford, "A compact variant of the qcr method for quadratically constrained quadratic 0–1 programs," *Optimization Letters*, vol. 8, no. 4, pp. 1213–1224, Apr. 2014, ISSN: 1862-4480. DOI: 10.1007/s11590-013-0676-8. [Online]. Available: https://doi.org/10.1007/s11590-013-0676-8.

[135] S. Wiese, *A computational practicability study of MIQCQP reformulations*, https://docs.mosek.com/whitepapers/miqcqp.pdf, Accessed: 2021-02-22, 2021.

[136] A. Frangioni and C. Gentile, "Perspective cuts for a class of convex $0-1$ mixed integer programs," *Math. Program., Ser. A*, vol. 106, pp. 225–236, 2006.

[137] A. Frangioni and C. Gentile, "SDP diagonalizations and perspective cuts for a class of nonseparable MIQP," *Operations Research Letters*, vol. 35, no. 2, pp. 181–185, 2007.

[138] C. S. Adjiman, I. P. Androulakis, and C. A. Floudas, "A global optimization method, αbb, for general twice-differentiable constrained NLPs—II. Implementation and computational results," *Computers and Chemical Engineering*, vol. 22, no. 9, pp. 1159–1179, 1998.

[139] C. S. Adjiman, S. Dallwig, C. A. Floudas, and A. Neumaier, "A global optimization method, αBB, for general twice-differentiable constrained NLPs—I. Theoretical advances," *Computers and Chemical Engineering*, vol. 22, no. 9, pp. 1137–1158, 1998.

[140] I. Androulakis and C. D. Maranas, "αBB: A global optimization method for general constrained nonconvex problems," *Journal of Global Optimization*, vol. 7, no. 4, pp. 337–363, 1995.

[141] P. A. C. Castillo, P. M. Castro, and V. Mahalec, "Global optimization of MIQCPs with dynamic piecewise relaxations," *Journal of Global Optimization*, vol. 71, no. 4, pp. 691–716, Feb. 2018. DOI: 10.1007/s10898-018-0612-7. [Online]. Available: https://doi.org/10.1007/s10898-018-0612-7.

[142] P. M. Castro, "Tightening piecewise McCormick relaxations for bilinear problems," *Computers & Chemical Engineering*, vol. 72, pp. 300–311, Jan. 2015. DOI: 10.1016/j.compchemeng.2014.03.025. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2014.03.025.

[143] R. Misener and C. A. Floudas, "Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations," *Mathematical Programming*, vol. 136, no. 1, pp. 155–182, Dec. 2012, ISSN: 1436-4646. DOI: 10.1007/s10107-012-0555-6. [Online]. Available: https://doi.org/10.1007/s10107-012-0555-6.

[144] H. Nagarajan, M. Lu, S. Wang, R. Bent, and K. Sundar, "An adaptive, multivariate partitioning algorithm for global optimization of nonconvex programs," *Journal of Global Optimization*, vol. 74, pp. 639–675, 4 2019.

[145] P. M. Castro, Q. Liao, and Y. Liang, "Comparison of mixed-integer relaxations with linear and logarithmic partitioning schemes for quadratically constrained problems," *Optimization and Engineering*, Mar. 2021. DOI: 10.1007/s11081-021-09603-5. [Online]. Available: https://doi.org/10.1007/s11081-021-09603-5.

[146] L. Galli and A. N. Letchford, "A binarisation heuristic for non-convex quadratic programming with box constraints," *Operations Research Letters*, vol. 46, no. 5, pp. 529–533, Sep. 2018. DOI: 10.1016/j.orl.2018.08.005. [Online]. Available: https://doi.org/10.1016/j.orl.2018.08.005.

[147] P. Hansen, B. Jaumard, M. Ruiz, and J. Xiong, "Global minimization of indefinite quadratic functions subject to box constraints," *Naval Research Logistics (NRL)*, vol. 40, no. 3, pp. 373–392, 1993. DOI: https://doi.org/10.1002/1520-6750(199304)40:3<373::AID-NAV3220400307>3.0.CO;2-A. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/1520-6750%28199304%2940%3A3%3C373%3A%3AAID-NAV3220400307%3E3.0.CO%3B2-A. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/1520-6750%28199304%2940%3A3%3C373%3A%3AAID-NAV3220400307%3E3.0.CO%3B2-A.

[148] R. Fortet, "L'algebre de boole et ses applications en recherche operationnelle," *Trabajos de Estadistica*, vol. 11, no. 2, pp. 111–118, Jun. 1960. DOI: 10.1007/bf03006558. [Online]. Available: https://doi.org/10.1007/bf03006558.

[149] F. Glover, "Improved linear integer programming formulations of nonlinear integer problems," *Management Science*, vol. 22, no. 4, pp. 455–460, Dec. 1975. DOI: 10.1287/mnsc.22.4.455. [Online]. Available: https://doi.org/10.1287/mnsc.22.4.455.

[150] P. Hammer and A. Ruben, "Some remarks on quadratic programming with 0-1 variables," *Revue Francaise D Automatique Informatique Recherche Operationnelle*, vol. 4, no. 3, pp. 67–79, 1970.

[151] W. Xia, J. C. Vera, and L. F. Zuluaga, "Globally solving nonconvex quadratic programs via linear integer programming techniques," *INFORMS Journal on Computing*, vol. 32, no. 1, pp. 40–56, Jan. 2020. DOI: 10.1287/ijoc.2018.0883. [Online]. Available: https://doi.org/10.1287/ijoc.2018.0883.

[152] S. S. Dey and A. Gupte, "Analysis of milp techniques for the pooling problem," *Operations Research*, vol. 63, no. 2, pp. 412–427, 2015.

[153] C. Savage, "A survey of combinatorial gray codes," *SIAM Review*, vol. 39, no. 4, pp. 605–629, 1997.

[154] F. A. Foss, "The use of a reflected code in digital control systems," *Transactions of the I.R.E. Professional Group on Electronic Computers*, vol. EC-3, no. 4, pp. 1–6, Dec. 1954. DOI: 10.1109/irepgelc.1954.6499244. [Online]. Available: https://doi.org/10.1109/irepgelc.1954.6499244.

[155] J. A. Huchette, "Advanced mixed-integer programming formulations : Methodology, computation, and application," Ph.D. dissertation, Massachusetts Institute of Technology, 2018.

[156] J. P. Vielma, S. Ahmed, and G. Nemhauser, "Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions," *Operations Research*, vol. 58, no. 2, pp. 303–315, Apr. 2010. DOI: 10.1287/opre.1090.0721. [Online]. Available: https://doi.org/10.1287/opre.1090.0721.

[157] V. Kaibel and K. Pashkovich, "Constructing extended formulations from reflection relations," in *Facets of Combinatorial Optimization: Festschrift for Martin Grötschel*, M. Jünger and G. Reinelt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 77–100.

[158] J. Bader, R. Hildebrand, R. Weismantel, and R. Zenklusen, "Mixed integer reformulations of integer programs and the affine tu-dimension of a matrix," *Mathematical Programming*, vol. 169, no. 2, pp. 565–584, Jun. 2018.

[159] I. Dunning, J. Huchette, and M. Lubin, "JuMP: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[160] J. Chen and S. Burer, "Globally solving nonconvex quadratic programming problems via completely positive programming," *Mathematical Programming Computation*, vol. 4, no. 1, pp. 33–52, 2012.

[161] I. P. Androulakis, C. D. Maranas, and C. A. Floudas, "αBb: A global optimization method for general constrained nonconvex problems," *Journal of Global Optimization*, vol. 7, no. 4, pp. 337–363, 1995.

[162] P. Bonami, O. Günlük, and J. Linderoth, "Globally solving nonconvex quadratic programming problems with box constraints via integer programming methods," *Mathematical Programming Computation*, vol. 10, no. 3, pp. 333–382, Feb. 2018. DOI: 10.1007/s12532-018-0133-x. [Online]. Available: https://doi.org/10.1007/s12532-018-0133-x.

[163] Y. Wei, "Triangular function analysis," *Computers & Mathematics with Applications*, vol. 37, no. 6, pp. 37–56, Mar. 1999. DOI: 10.1016/s0898-1221(99)00075-9. [Online]. Available: https://doi.org/10.1016/s0898-1221(99)00075-9.

[164] S. S. Dey, A. M. Kazachkov, A. Lodi, and G. Mu, *Cutting plane generation through sparse principal component analysis*, 2021. [Online]. Available: http://www.optimization-online.org/DB_HTML/2021/02/8259.html.

[165] K.-M. Aigner, R. Burlacu, F. Liers, and A. Martin, "Solving ac optimal power flow with discrete decisions to global optimality," 2020.

[166] C. M. Correa-Posada and P. Sánchez-Martín, "Gas network optimization: A comparison of piecewise linear models," *Optimization Online*, 2014.

[167] B. Geißler, A. Martin, A. Morsi, and L. Schewe, "Using piecewise linear functions for solving minlp s," in *Mixed integer nonlinear programming*, Springer, 2012, pp. 287–314.

[168] D. C. Faria and M. J. Bagajewicz, "Novel bound contraction procedure for global optimization of bilinear minlp problems with applications to water management problems," *Computers & Chemical Engineering*, vol. 35, no. 3, pp. 446–455, 2011.

[169] A. Bärmann and O. Schneider, "Set characterizations and convex extensions for geometric convex-hull proofs," *Mathematical Programming*, pp. 1–41, 2021.

[170] K. Kutzer, "Using piecewise linear approximation techniques to handle bilinear constraints," Ph.D. dissertation, 2020.

[171] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter, "Branching and bounds tightening-techniques for non-convex minlp," *Optimization Methods & Software*, vol. 24, no. 4-5, pp. 597–634, 2009.

[172] L. H. Andreas Bärmann Robert Burlacu and T. Kleinert, *On piecewise linear approximations of bilinear terms: Structural comparison of univariate and bivariate mixed-integer programming formulations*, 2021. eprint: Optimization-online. [Online]. Available: %5Curl%7Bhttp://www.optimization-online.org/DB_HTML/2021/08/8536.html%7D.

[173] G. M. Appa, L. Pitsoulis, and H. P. Williams, *Handbook on modelling for discrete optimization*. Springer Science & Business Media, 2006, vol. 88.

[174] R. Burlacu, B. Geißler, and L. Schewe, "Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes," *Optimization Methods and Software*, vol. 35, no. 1, pp. 37–64, 2020.

[175] M. Telgarsky, *Representation benefits of deep feedforward networks*, 2015. DOI: 10.48550/ARXIV.1509.08101. [Online]. Available: https://arxiv.org/abs/1509.08101.

[176] J. Linderoth, "A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs," *Mathematical programming*, vol. 103, no. 2, pp. 251–282, 2005.