

Mixed Integer Nonlinear Programming

Lecture Notes Compilation

Robert Hildebrand

Version Compilation date: February 28, 2024

Preface

This entire book is a working manuscript that is being put together for the sake of a course on Mixed Integer Nonlinear Programming.

This book is being written and compiled using a number of materials, most of which are NOT OPEN SOURCE! Thus, we do not claim an open source access to this material. Much of this material is HIGHLY PLAGARIZED! The material is mainly taken from research papers, surveys, and slides.

This manuscript is being written purely to collect all the information that is planned to be presented in the course.

These notes assume familiarity with Linear Programming and Convexity Theory, and some Integer Programming and Convex Optimization. We will try to develop background as needed.

Mixed Integer Nonlinear Programming is an extremely broad field. We will only be able to touch on a few different topics over the course of a semester.

MAJOR ACKNOWLEDGEMENTS

I will add a number of citations here with which this material is based on. I will try to keep references throughout the book as pointers of where to look for this material.

Contents

Contents	5
1 Resources and Notation	3
I Introduction	7
2 Mathematical Programming	11
2.1 Why study operations research?	11
2.2 What is Mathematical Programming?	11
2.3 Applications	11
2.4 Types of Optimization problems	11
2.5 Linear Programming (LP)	12
2.6 Mixed-Integer Linear Programming (MILP)	13
2.7 Non-Linear Programming (NLP)	15
2.7.1 Convex Programming	16
2.7.2 Non-Convex Non-linear Programming	16
2.7.3 Machine Learning	17
2.8 Mixed Integer Non-Linear Programming (MINLP)	18
2.8.1 Convex MINLP	19
2.8.2 Non-Convex MINLP	19
3 Background	21
3.1 LP	21
3.2 Convex Optimization	21
3.3 Integer Programming	21
4 MINLP Applications	23
4.0.1 Operations Research	25
4.1 Pooling Problems	25
4.2 Unit Commitment Problem	27
4.3 Machine Learning and Integer Programming	29
4.3.1 Outlier Detection in Regression	29
4.3.2 Quadratic constraints with indicator variables	30
4.3.3 Control theory and others	31
4.3.4 Pure Mathematics	31
4.4 Sparse Portfolio optimization	32
4.5 Portfolio Optimization with Higher Order Moments	32

6 ■ CONTENTS

4.6	Generalizations of Knapsack Problem	33
4.7	Machine Learning	34
4.8	Airline scheduling	35
4.9	Inverse Optimization	35
4.10	Wireless Networks	36
4.11	Watershed Management	38
4.12	AC Optimal Power Flow Problem	40
4.13	Other applications	43
4.14	Distance Geometry Problems	45
4.14.1	Molecular Distance Geometry Problem	45
4.14.2	Kissing Number	46
4.15	Obnoxious Facility Location Problem	48
4.16	AC Optimal Power Flow Problem	51
4.17	Multi Player Nash Equilibrium	54
4.18	Pricing Problem	56
5	Software	59
5.1	List of Leading MINLP Software	59
5.2	List of (some) MINLP Researchers	61
II	MINLP Theory and Algorithms	65
5.3	AC Optimal Power Flow Problem	67
5.4	Distance Geometry Problems	70
5.4.1	Molecular Distance Geometry Problem	70
5.4.2	Kissing Number	71
5.5	Obnoxious Facility Location Problem	73
5.6	Multi Player Nash Equilibrium	75
6	Optimization approaches	77
6.1	The general global optimization paradigm	77
6.1.1	Heuristic Approach	78
6.1.2	Relaxations and Convex Hulls	81
6.1.3	Solving using Branch-and Bound	82
6.1.4	Discussion of Branch-and-bound algorithm	86
6.1.4.1	Spatial B&B: Pruning	93
6.1.5	Spatial B&B: General idea	94
6.1.6	Spatial B&B: Exact Reformulation to Standard Form	94
6.1.7	Spatial B&B: convexification	94
6.1.7.1	Spatial B&B: Examples of Convexifications	95
6.1.7.2	Example: Standard Form Reformulation	95
6.1.8	Expression trees	96
6.1.9	Variable ranges	97
6.2	Convex Envelopes	102

6.2.0.1	Another way to think about convex envelope	103
6.2.1	Sufficient condition for polyhedral convex envelope of $f(x)$	106
6.3	MIQCQP	107
6.4	Convex hull of simple sets	108
6.4.1	McCormick envelope	108
6.4.2	Our first convex relaxation of QCQP	110
6.4.3	Semi-definite programming (SDP) relaxation of QCQPs	111
6.4.4	Extending the McCormick envelope ideas	112
6.4.5	Reformulation Linearization Technique (RLT)	117
6.4.6	Boolean Quadric Polytope (BQP)	118
6.5	Exercises	122
7	Convex Hulls with Special Structures	125
7.1	Convex Hulls with Special Structures	125
7.2	Convex hulls of structured sets	125
7.2.1	A packing-type bilinear knapsack set	125
7.2.2	Product of a simplex and a polytope	126
7.3	Dey recent results	127
7.4	To do - Add remaining Dey Slides	127
8	Perspective Reformulations	129
8.1	Perspective Reformulations	129
8.1.1	Perspective of a function	129
8.2	Perspective Reformulation and Applications - Oktay Günlük & Jeff Linderoth	131
8.2.1	A quadratic set with variable bounds	133
8.2.2	An extended formulation for Q	135
8.2.2.1	Convex hull description in the original space	137
8.2.3	More things....	137
8.2.3.1	Other examples	139
8.3	Convexification On Off	143
8.3.1	Relationships to previous theoretical results	150
8.3.2	Application: the delay-constrained routing problem	151
9	Convex Quadratic Reformulation	157
9.1	Tools: Semidefinite Programming Duality	157
9.2	Lagrangian Dual of QCQP	160
9.3	Convex Quadratic Reformulations	162
9.3.1	Convex Quadratic Reformulation by Diagonal Perturbation	162
9.3.2	Choosing the Diagonal Perturbation Matrix D	163
9.3.2.1	Using the Smallest Eigenvalue of Q	163
9.3.2.2	Solving a Semidefinite Programming (SDP) Problem	164
9.3.3	Diagonalization of Q and Variable Rotation for Perturbation	164
9.3.4	Optimizing choice of α and u	166

10 Discretization Approaches	169
10.1 Complete discretization - all binary variables	169
10.2 Products of continuous variables - NMDT	169
10.2.1 Discretizing a continuous variable + small error	170
10.2.2 Binary and continuous variables	170
10.2.3 Application to Pooling Problems	171
10.3 QCR + Discretization	172
10.4 Discretizations for QCQP	172
10.5 Adaptive Refinement Methods	172
10.5.1 References	172
11 Advanced Piecewise Linear Formulations	173
11.1 Standard Approaches	173
11.2 Compact Formulations	177
11.3 Modern Formulations	180
11.4 (Section 3.) Formulations for univariate piecewise linear functions	184
11.4.0.1 (subsection 3.1) The embedding approach	184
11.4.1 (Subsection 3.3.) New zig-zag formulations for the SOS2 constraint	186
III Working Material to be added	189
12 Other things...	191
12.1 Extended Space and Spatial Branch and Bound	191
12.1.0.1 Extended Space	191
12.1.1 Spatial Branch and Bound	192
12.2 McCormick Envelope	195
12.3 Relaxation Linearization Technique (RLT)	196
12.3.1 Boolean Quadric Polytope (BQP)	196
12.3.2 SDP Relaxation	198
12.4 Lagrangian Relaxation	198
12.5 New Results	198
13 Approximations of SOCP and SDP	199
13.1 Outer linear approximation of SDP	199
14 S-Lemma	201
15 Fractional MINLP	203
16 Binary Polynomial Optimization	205
17 Hierarchies	207
17.1 Lovasz-Schrijver, Sherali-Adams, Lasserre Hierarchies	207
18 Generalized Benders Decomposition	209

19 Convexification of Nonconvex Compositions with Norms	215
20 Polynomial Optimization via Convex Optimization	217
20.1 Sums of Squares (SOS) Programming	217
20.1.1 Putinar	217
20.2 Sums of nonnegative circuit polynomials (SONC)	217
21 Polynomial Optimization via Algebraic Techniques	219
21.1 Polynomial Optimization - Algebraic Techniques	219
21.2 Cylindrical Algebraic Decomposition	219
21.3 Gröbner Bases	219
21.4 Quantifier Elimination and Semi-algebraic Sets	219
21.5 Polynomial Optimization	219
21.5.1 Sums of Squares	224
21.5.2 Nullstellensatz	263
21.5.3 Positivstellensatz	263
21.5.4 Putinar's Positivstellensatz	263
21.5.5 Handelman Decompositions	263
22 Some notes	265
22.0.1 Semi-Definite Programming (SDP)	265
22.0.2 Application in Combinatorial optimization	266
22.0.3 Max-cut	268
22.0.4 SOS Theory	268
22.0.5 Deciding complexity of a polynomial function	269
23 Algebraic and Geometric Ideas	271
24 Complexity	279
24.1 Main references	279
24.2 Other references	279
25 Recent topics	281
25.1 Exactness in SDP Relaxations	281
25.2 Quadratics and polytopes	281
25.2.1 Hidden Hyperplane Convexity	281
26 Decision Diagrams	283
27 Sequential Quadratic Programming Methods	285
28 Possible Projects	287
28.1 List	287
29 (Some) Topics Not covered	289

Index	291
--------------	------------

Introduction

Letter to instructors

This is an advanced book on Mixed Integer Nonlinear Programming. We do not claim that things are well cited or that this material should be distributed freely. This material is designed solely for this course on MINLP as lecture notes.

Letter to students

These are developing lectures notes. Please take them as such. You are welcome and encouraged to contribute edits, comments, sections, and code to this book.

How to use this book

Skim ahead. We recommend that before you come across a topic in lecture, that you skim the relevant sections ahead of time to get a broad overview of what is to come. This may take only a fraction of the time that it may take for you to read it.

Read the expected outcomes. At the beginning of each section, there will be a list of expected outcomes from that section. Review these outcomes before reading the section to help guide you through what is most relevant for you to take away from the material. This will also provide a brief look into what is to follow in that section.

Read the text. Read carefully the text to understand the problems and techniques. We will try to provide a plethora of examples, therefore, depending on your understanding of a topic, you may need to go carefully over all of the examples.

Explore the resources. Lastly, we recognize that there are many alternative methods of learning given the massive amounts of information and resources online. Thus, at the end of each section, there will be a number of superb resources that are available on the internet in different formats. There are other free textbooks, informational websites, and also a number of fantastic videos posted to youtube. We encourage to explore the resources to get another perspective on the material or to hear/read it taught from a different point of view or in presentation style.

1. Resources and Notation

Here are a list of resources that may be useful as alternative references or additional references.

FREE NOTES AND TEXTBOOKS

- Mathematical Programming with Julia by Richard Lusby & Thomas Stidsen
- Linear Programming by K.J. Mtetwa, David
- A first course in optimization by Jon Lee
- Introduction to Optimization Notes by Komei Fukuda
- Convex Optimization by Boyd and Vandenberghe
- LP notes of Michel Goemans from MIT
- Understanding and Using Linear Programming - Matousek and Gärtner [Downloadable from Springer with University account]
- Operations Research Problems Statements and Solutions - Raúl Poler Josefa Mula Manuel Díaz-Madroñero [Downloadable from Springer with University account]

NOTES, BOOKS, AND VIDEOS BY VARIOUS SOLVER GROUPS

- AIMMS Optimization Modeling
- Optimization Modeling with LINGO by Linus Schrage
- The AMPL Book
- Microsoft Excel 2019 Data Analysis and Business Modeling, Sixth Edition, by Wayne Winston - Available to read for free as an e-book through Virginia Tech library at Orieilly.com.
- Lesson files for the Winston Book
- Video instructions for solver and an example workbook
- youtube-OR-course

4 ■ Resources and Notation

GUROBI LINKS

- Go to <https://github.com/Gurobi> and download the example files.
- Essential ingredients
- Gurobi Linear Programming tutorial
- Gurobi tutorial MILP
- GUROBI - Python 1 - Modeling with GUROBI in Python
- GUROBI - Python II: Advanced Algebraic Modeling with Python and Gurobi
- GUROBI - Python III: Optimization and Heuristics
- Webinar Materials
- GUROBI Tutorials

HOW TO PROVE THINGS

- Hammack - Book of Proof

STATISTICS

- Open Stax - Introductory Statistics

LINEAR ALGEBRA

- Beezer - A first course in linear algebra
- Selinger - Linear Algebra
- Cherney, Denton, Thomas, Waldron - Linear Algebra

REAL ANALYSIS

- Mathematical Analysis I by Elias Zakon

DISCRETE MATHEMATICS, GRAPHS, ALGORITHMS, AND COMBINATORICS

- Levin - Discrete Mathematics - An Open Introduction, 3rd edition
- Github - Discrete Mathematics: an Open Introduction CC BY SA
- Keller, Trotter - Applied Combinatorics (CC-BY-SA 4.0)
- Keller - Github - Applied Combinatorics

MIXED INTEGER NONLINEAR PROGRAMMING

- Mixed-Integer Nonlinear Optimization Pietro Belotti, Christian Kirches, Sven Leyffer, Jeff Linderoth, Jim Luedtke, and Ashutosh Mahajan

PROGRAMMING WITH PYTHON

- A Byte of Python
- Github - Byte of Python (CC-BY-SA)

Also, go to <https://github.com/open-optimization/open-optimization-or-examples> to look at more examples.

Part I

Introduction

Notation

- $\mathbf{1}$ - a vector of all ones (the size of the vector depends on context)
- \forall - for all
- \exists - there exists
- \in - in
- \therefore - therefore
- \Rightarrow - implies
- s.t. - such that (or sometimes "subject to".... from context?)
- $\{0,1\}$ - the set of numbers 0 and 1
- \mathbb{Z} - the set of integers (e.g. $1, 2, 3, -1, -2, -3, \dots$)
- \mathbb{Q} - the set of rational numbers (numbers that can be written as p/q for $p, q \in \mathbb{Z}$ (e.g. $1, 1/6, 27/2$)
- \mathbb{R} - the set of all real numbers (e.g. $1, 1.5, \pi, e, -11/5$)
- \setminus - setminus, (e.g. $\{0, 1, 2, 3\} \setminus \{0, 3\} = \{1, 2\}$)
- \cup - union (e.g. $\{1, 2\} \cup \{3, 5\} = \{1, 2, 3, 5\}$)
- \cap - intersection (e.g. $\{1, 2, 3, 4\} \cap \{3, 4, 5, 6\} = \{3, 4\}$)
- $\{0, 1\}^4$ - the set of 4 dimensional vectors taking values 0 or 1, (e.g. $[0, 0, 1, 0]$ or $[1, 1, 1, 1]$)
- \mathbb{Z}^4 - the set of 4 dimensional vectors taking integer values (e.g., $[1, -5, 17, 3]$ or $[6, 2, -3, -11]$)
- \mathbb{Q}^4 - the set of 4 dimensional vectors taking rational values (e.g. $[1.5, 3.4, -2.4, 2]$)
- \mathbb{R}^4 - the set of 4 dimensional vectors taking real values (e.g. $[3, \pi, -e, \sqrt{2}]$)
- $\sum_{i=1}^4 i = 1 + 2 + 3 + 4$
- $\sum_{i=1}^4 i^2 = 1^2 + 2^2 + 3^2 + 4^2$
- $\sum_{i=1}^4 x_i = x_1 + x_2 + x_3 + x_4$
- \square - this is a typical Q.E.D. symbol that you put at the end of a proof meaning "I proved it."
- For $x, y \in \mathbb{R}^3$, the following are equivalent (note, in other contexts, these notations can mean different things)
 - $x^\top y$ *matrix multiplication*
 - $x \cdot y$ *dot product*
 - $\langle x, y \rangle$ *inner product*

and evaluate to $\sum_{i=1}^3 x_i y_i = x_1 y_1 + x_2 y_2 + x_3 y_3$.

A sample sentence:

$$\forall x \in \mathbb{Q}^n \exists y \in \mathbb{Z}^n \setminus \{0\}^n s.t. x^\top y \in \{0, 1\}$$

"For all non-zero rational vectors x in n -dimensions, there exists a non-zero n -dimensional integer vector y such that the dot product of x with y evaluates to either 0 or 1."

2. Mathematical Programming

Outcomes

- Identify reasons for studying operations research
- Define "Mathematical Programming"
- Learn about different applications of the tools in this book
- Explore the different types of optimization models and what types we will see in this book.

2.1 Why study operations research?

2.2 What is Mathematical Programming?

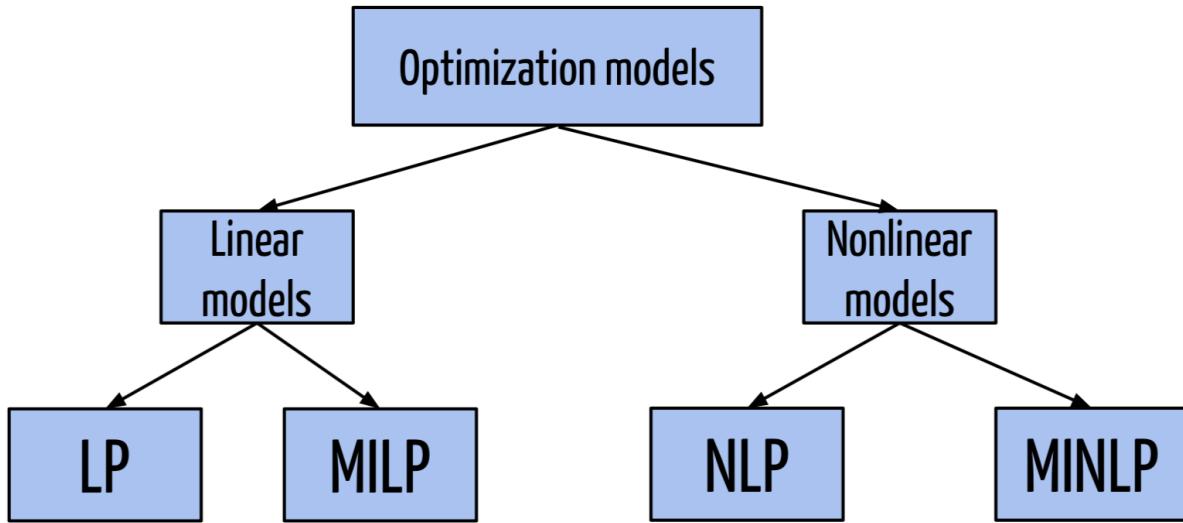
2.3 Applications

2.4 Types of Optimization problems

We will state main general problem classes to be associated with in these notes. These are Linear Programming (LP), Mixed-Integer Linear Programming (MILP), Non-Linear Programming (NLP), and Mixed-Integer Non-Linear Programming (MINLP).

Along with each problem class, we will associate a complexity class for the general version of the problem. See ?? for a discussion of complexity classes. Although we will often state that input data for a problem comes from \mathbb{R} , when we discuss complexity of such a problem, we actually mean that the data is rational, i.e., from \mathbb{Q} , and is given in binary encoding.

¹*Tree of optimization problems.* from. Diego Moran CC0., 2017.

© Diego Moran CC0.¹**Figure 2.1: Tree of optimization problems.**

2.5 Linear Programming (LP)

Some linear programming background, theory, and examples will be provided in ??.

Linear Programming (LP):

Polynomial time (P)

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *linear programming* problem is

$$\begin{aligned}
 & \max \quad c^\top x \\
 & \text{s.t.} \quad Ax \leq b \\
 & \quad \quad \quad x \geq 0
 \end{aligned} \tag{2.1}$$

Linear programming can come in several forms, whether we are maximizing or minimizing, or if the constraints are \leq , $=$ or \geq . One form commonly used is *Standard Form* given as

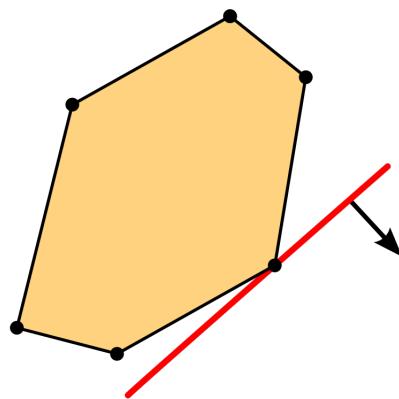
Linear Programming (LP) Standard Form:

Polynomial time (P)

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *linear programming* problem in

standard form is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{2.2}$$



© Yllooh CC0.²

Figure 2.2: Linear programming constraints and objective.

Figure 2.2

Exercise 2.1:

Start with a problem in form given as (2.1) and convert it to standard form (2.2) by adding at most m many new variables and by enlarging the constraint matrix A by at most m new columns.

2.6 Mixed-Integer Linear Programming (MILP)

Mixed-integer linear programming will be the focus of Sections ??, ??, ??, and ?. Recall that the notation \mathbb{Z} means the set of integers and the set \mathbb{R} means the set of real numbers. The first problem of interest here is a *binary integer program* (BIP) where all n variables are binary (either 0 or 1).

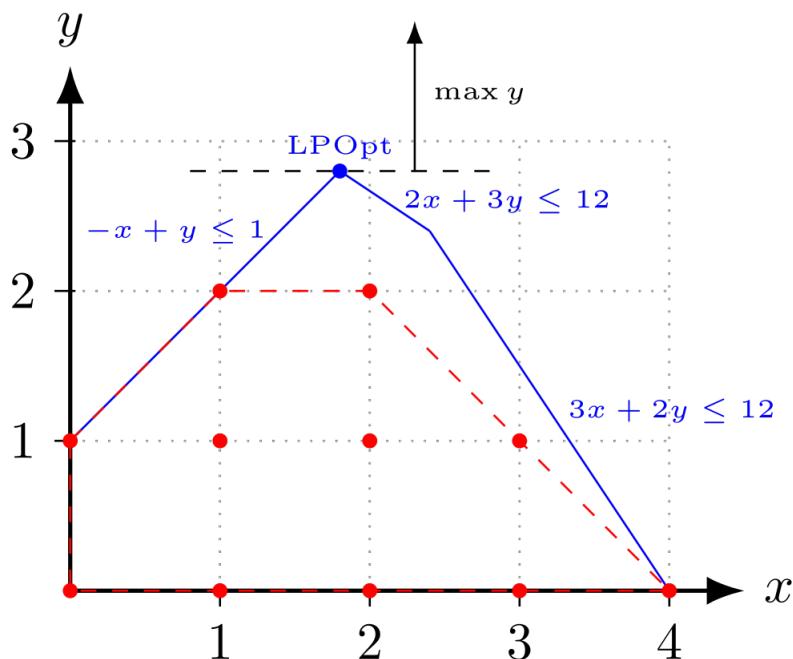
²A pictorial representation of a simple linear program with two variables and six inequalities. The set of feasible solutions is depicted in yellow and forms a polygon, a 2-dimensional polytope. The linear cost function is represented by the red line and the arrow: The red line is a level set of the cost function, and the arrow indicates the direction in which we are optimizing. from https://en.wikipedia.org/wiki/Linear_programming#/media/File:Linear_optimization_in_a_2-dimensional_polytope.svg. Yllooh CC0., 2012.

Binary Integer programming (BIP):*NP-Complete*

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *binary integer programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \{0, 1\}^n \end{aligned} \tag{2.1}$$

A slightly more general class is the class of *Integer Linear Programs* (ILP). Often this is referred to as *Integer Program* (IP), although this term could leave open the possibility of non-linear parts.



© Fanosta CC BY-SA 4.0.³
Figure 2.3: Comparing the LP relaxation to the IP solutions.

Figure 2.3

Integer Linear Programming (ILP):*NP-Complete*

³IP polytope with LP relaxation, from https://en.wikipedia.org/wiki/Integer_programming#/media/File:IP_polytope_with_LP_relaxation.svg. Fanosta CC BY-SA 4.0., 2019.

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *integer linear programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^n \end{aligned} \tag{2.2}$$

An even more general class is *Mixed-Integer Linear Programming (MILP)*. This is where we have n integer variables $x_1, \dots, x_n \in \mathbb{Z}$ and d continuous variables $x_{n+1}, \dots, x_{n+d} \in \mathbb{R}$. Succinctly, we can write this as $x \in \mathbb{Z}^n \times \mathbb{R}^d$, where \times stands for the *cross-product* between two spaces.

Below, the matrix A now has $n + d$ columns, that is, $A \in \mathbb{R}^{m \times n+d}$. Also note that we have not explicitly enforced non-negativity on the variables. If there are non-negativity restrictions, this can be assumed to be a part of the inequality description $Ax \leq b$.

Mixed-Integer Linear Programming (MILP):

NP-Complete

Given a matrix $A \in \mathbb{R}^{m \times (n+d)}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^{n+d}$, the *mixed-integer linear programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^n \times \mathbb{R}^d \end{aligned} \tag{2.3}$$

2.7 Non-Linear Programming (NLP)

NLP:

NP-Hard

Given a function $f(x): \mathbb{R}^d \rightarrow \mathbb{R}$ and other functions $f_i(x): \mathbb{R}^d \rightarrow \mathbb{R}$ for $i = 1, \dots, m$, the *nonlinear programming* problem is

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad \text{for } i = 1, \dots, m \\ & x \in \mathbb{R}^d \end{aligned} \tag{2.1}$$

Nonlinear programming can be separated into convex programming and non-convex programming. These two are very different beasts and it is important to distinguish between the two.

2.7.1. Convex Programming

Here the functions are all **convex!**

Convex Programming:

Polynomial time (P) (typically)

Given a convex function $f(x): \mathbb{R}^d \rightarrow \mathbb{R}$ and convex functions $f_i(x): \mathbb{R}^d \rightarrow \mathbb{R}$ for $i = 1, \dots, m$, the *convex programming* problem is

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad \text{for } i = 1, \dots, m \\ & x \in \mathbb{R}^d \end{aligned} \tag{2.2}$$

Observe that convex programming is a generalization of linear programming. This can be seen by letting $f(x) = c^\top x$ and $f_i(x) = A_i x - b_i$.

2.7.2. Non-Convex Non-linear Programming

When the function f or functions f_i are non-convex, this becomes a non-convex nonlinear programming problem. There are a few complexity issues with this.

IP AS NLP As seen above, quadratic constraints can be used to create a feasible region with discrete solutions. For example

$$x(1-x) = 0$$

has exactly two solutions: $x = 0, x = 1$. Thus, quadratic constraints can be used to model binary constraints.

Binary Integer programming (BIP) as a NLP:

NP-Hard

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *binary integer programming* problem

is

$$\begin{aligned}
 & \max \quad c^\top x \\
 \text{s.t.} \quad & Ax \leq b \\
 & x \in \{0, 1\}^n \\
 & x_i(1 - x_i) = 0 \quad \text{for } i = 1, \dots, n
 \end{aligned} \tag{2.3}$$

Alternatively, consider the transformation where $x_i \in \{-1, 1\}$.

$$\begin{aligned}
 & \min c^\top x \\
 \text{s.t.} \quad & Ax \leq b \\
 & x \in \{-1, 1\}^n
 \end{aligned}$$

This can be reformulated with a single nonconvex constraint as

$$\begin{aligned}
 & \min \quad c^\top x \\
 \text{s.t.} \quad & Ax \leq b \\
 & -1 \leq x_j \leq 1, \quad 1 \leq j \leq n, \\
 & \|x\|^2 \geq n.
 \end{aligned}$$

2.7.3. Machine Learning

Machine learning problems are often cast as continuous optimization problems, which involve adjusting parameters to minimize or maximize a particular objective. Frequently they are convex optimization problems, but many turn out to be nonconvex. Here are two examples of how these problems arise at a glance. We will see examples in greater detail later in the book.

Loss Function Minimization

In supervised learning, this objective is typically a loss function L that quantifies the discrepancy between the predictions of a model and the true data labels. The aim is to adjust the parameters θ of the model to minimize this loss. Mathematically, this can be represented as:

$$\min_{\theta} L(\theta) = \min_{\theta} \frac{1}{N} \sum_{i=1}^N l(y_i, f(x_i; \theta)) \tag{2.4}$$

where N is the number of data points, l is a per-data-point loss (e.g., squared error for regression or cross-entropy for classification), y_i is the true label for the i -th data point, and $f(x_i; \theta)$ is the model's prediction for the i -th data point with parameters θ .

Clustering Formulation

Clustering, on the other hand, seeks to group or partition data points such that data points in the same group are more similar to each other than those in other groups. One popular method is the k-means clustering algorithm. The objective of k-means is to partition the data into k clusters by minimizing the within-cluster sum of squares (WCSS). The mathematical formulation can be given as:

$$\min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \sum_{j=1}^k \sum_{x_i \in C_j} \|x_i - \mathbf{c}_j\|^2 \quad (2.5)$$

where C_j represents the j -th cluster and \mathbf{c}_j is the centroid of that cluster.

This encapsulation presents a glimpse into how ML problems are framed mathematically. In practice, numerous algorithms, constraints, and regularizations add complexity to these basic formulations.

2.8 Mixed Integer Non-Linear Programming (MINLP)

MINLP:

NP-Hard

Mixed Integer Nonlinear Programming (MINLP) combines elements of integer programming and nonlinear programming. In MINLP, some or all of the decision variables are constrained to be integers, and the objective function or constraints are nonlinear. A general MINLP problem can be formulated as:

$$\begin{aligned} \min \quad & f(x, y) \\ \text{s.t.} \quad & g_i(x, y) \leq 0 \quad \text{for } i = 1, \dots, m \\ & h_j(x, y) = 0 \quad \text{for } j = 1, \dots, p \\ & x \in \mathbb{R}^n, y \in \mathbb{Z}^k \end{aligned} \quad (2.1)$$

where f is the objective function, g_i and h_j are constraint functions, x is a vector of continuous variables, and y is a vector of integer variables.

MINLP is particularly challenging due to the nonlinearity in the objective and/or constraints and the discrete nature of some decision variables. It finds applications in various fields such as industry for process optimization, computational geometry, and machine learning for hyperparameter tuning.

2.8.1. Convex MINLP

In the convex case, both the objective function and the constraints are convex functions. This subclass is easier to solve compared to its non-convex counterpart.

Convex MINLP:

NP-Hard, but *Polynomial time (P)* in fixed dimension (typically)

For a convex MINLP, the problem is defined as:

$$\begin{aligned} \min \quad & f(x, y) \quad (\text{convex}) \\ \text{s.t.} \quad & g_i(x, y) \leq 0 \quad (\text{convex constraints}) \\ & x \in \mathbb{R}^n, y \in \mathbb{Z}^k \end{aligned} \tag{2.2}$$

Convex MINLPs, while still challenging, are typically more tractable due to the properties of convexity, which allow for more efficient solution methods.

2.8.2. Non-Convex MINLP

Non-convex MINLPs are significantly harder due to the presence of non-convex functions, which can lead to multiple local minima.

Non-Convex MINLP:

NP-Hard(in fact, undecidable)

The non-convex MINLP problem is formulated as:

$$\begin{aligned} \min \quad & f(x, y) \quad (\text{non-convex}) \\ \text{s.t.} \quad & g_i(x, y) \leq 0 \quad (\text{possibly non-convex constraints}) \\ & x \in \mathbb{R}^n, y \in \mathbb{Z}^k \end{aligned} \tag{2.3}$$

Non-convex MINLPs pose significant computational challenges due to the possibility of multiple local optima and the inherent complexity of integer constraints. These problems are common in real-world applications where decisions are discrete, and the system behavior is non-linear and complex.

Complexity and Applications

MINLP problems are known for their computational complexity, primarily due to the combination of non-linearity and integrality. The non-convex variants, in particular, are *NP-Hard*, making them some of the most challenging problems in optimization.

In practice, MINLP models find extensive applications across various domains. In industry, they are used for complex decision-making processes like supply chain optimization and production planning. In computational geometry, MINLP techniques help in solving problems like optimal packing or layout design. Furthermore, in the realm of machine learning, MINLPs are crucial for tasks like feature selection and hyperparameter optimization where discrete choices and nonlinear relationships are involved.

The versatility of MINLP models, combined with their inherent complexity, makes them a fascinating and essential area of study in the field of optimization.

3. Background

3.1 LP

- LP
- Duality, KKT
- Fourier Motzkin (Projections)

3.2 Convex Optimization

- Convex Sets, Convex Functions
- Subgradients
- Conic Programming, SOCP, SDP
- KKT Conditions, Slaters Conditions
- Duality
- Krein Milman Theorem

3.3 Integer Programming

- Relaxations
- Cutting Planes
- Branch and Bound

4. MINLP Applications

MINLP is such a broad area that can encompass almost any type of optimization. But we will focus on applications that require the tools described in this course.

FROM GUROBI

NON-CONVEX QP, QCP, MIQP, AND MIQCP APPLICATIONS

- Pooling problem
- Petrochemical industry
- Wastewater treatment
- Emissions regulation
- Agricultural / food industry
- Mining
- Energy
- Production planning
- Logistics
- Water distribution
- Engineering design
- Finance
- Blending problem is LP, pooling introduces intermediate pools → bilinear.
- Oil refinery: constraints on ratio of components in tanks.
- Blending based on pre-mix products.
- Constraints on ratio between internal and external workforce.
- Restrictions from free trade agreements.
- Darcy-Weisbach equation for volumetric flow.
- Constraints on exchange rates.

GENERAL MINLP

- Non-convex MIQCP solves (in theory) polynomial problems of arbitrary degree.
- Solve general MINLPs by approximating as a polynomial problem.
- However, will often fail for higher degrees due to numerical issues.

FROM MINLP LIB Agriculture Alkylation Argentina utility plant Asset Management Autocorrelated Sequences Batch processing Breeding Cascading Tanks Catalyst Mixing Catalytic Cracking of Gas Oil Chain Optimization Chemical Equilibrium Coil Compression String Design Coloring Computational geometry Constraint Satisfaction Cross-dock Door Assignment Crude Oil Scheduling Cutting Stock Cyclic multiproduct scheduling on parallel lines Cyclic Scheduling of Continuous Parallel Units Density modification based on single-crystal X-ray diffraction data Design of Just-in-Time Flowshops Deterministic Security Constrained Unit Commitment Edge-crossing minimization in bipartite graphs Elastic-plastic torsion Electricity generation Electricity Networks Electricity Storage Electrons on a Sphere Energy Facility Location Farming Feed Mix feed plate location Feed Plate Location Financial Optimization four membrane pipe modules in feed-and-bleed coupling Frequency Assignment Gas Transmission Gas Transmission Network Design Gear Train Design General Equilibrium Geometry Graph Partitioning Hang Glider Hanging Chain Heat Exchanger Network Heat Integrated Distillation Sequences Hybrid Dynamic Systems Hydro Energy System Scheduling Hydrodealkylation of Toluene Isometrization Job Scheduling Kissing Number Problem Kriging Launch Vehicle Design Layout Linear Algebra Location Item Planning Marine Population Dynamics Market Equilibrium Matrix Eigenvalues Max Cut Methanol to Hydrocarbons Minimizing Total Average Cycle Stock Molecular Design Multi-commodity facility location-allocation Multi-Product Batch Plant Design Multiperiod Blend Scheduling Multiproduct CSTR Natural Gas Production Network Design Neural Networks Nuclear Reactor Core Reload Pattern Optimal Control Optimal vehicle allocation for minimizing greenhouse gas emissions Parameter estimation in quantitative IR spectroscopy Particle Steering Periodic Scheduling of Continuous Multiproduct Plants Pipeline design Pooling problem Pooling Problem Portfolio Optimization power plant operation Process Flowsheets Process Networks Process selection Product Portfolio Optimization Product positioning in a multiattribute space Production pseudo components properties Pump configuration problem Quantum Mechanics Radiation therapy Rail Line Optimization Retrofit Planning Rockets Sensor Placement Separation Sequences Based on Distillation Service System Design Shape Optimization Shortest Path Simultaneous Optimization for HEN Synthesis Social Accounting Matrix Balancing Spacecraft Landing Sports Tournament Statistics Structural Optimization Supply Chain Design with Stochastic Inventory Management Synthesis of General Distillation Sequences Synthesis of processing system Synthesis of Space Truss Tank Size Design Telecommunication Test Problem Topology Optimization Traveling Salesman Problem with Neighborhoods Trim loss minimization problem Unit Commitment Waste paper treatment Waste Water Treatment Water Network Contamination Water Network Design Water Network Operation Water Resource Management Winding Factor of Electrical Machines

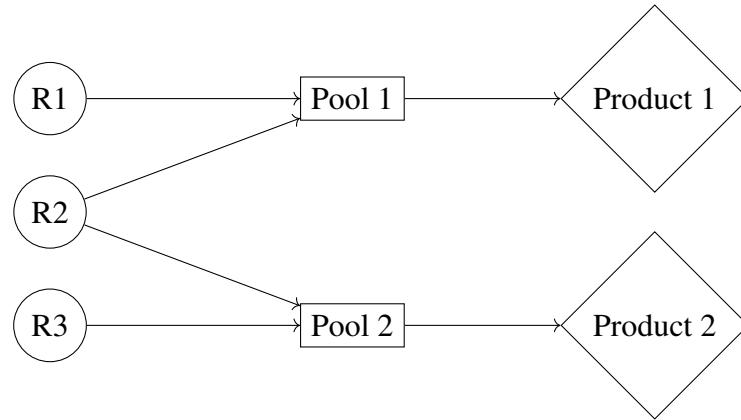
4.0.1. Operations Research

1. **Supply Chain Optimization:** Managing and optimizing the flow of goods, services, and information from origin to consumption.
2. **Production Planning:** Determining the optimal production schedule and mix to maximize profit or minimize costs in manufacturing processes.
3. **Energy Management:** Optimizing the operation and planning of energy production and distribution systems.
4. **Facility Location and Layout:** Determining optimal location for facilities and their layout for efficient operations.
5. **Transportation and Logistics:** Planning and optimizing routing, scheduling, and loading in transportation systems.
6. **Financial Optimization:** Portfolio optimization and risk management in finance.
7. **Telecommunications Network Design:** Designing and optimizing the layout and operation of telecommunications networks.
8. **Project Scheduling:** Planning and scheduling complex projects with multiple tasks and dependencies.
9. **Process Engineering:** Optimization of processes in chemical and process industries.
10. **Healthcare Management:** Planning and scheduling of healthcare services, resource allocation in hospitals.
11. **Environmental Engineering:** Addressing problems in environmental engineering, such as waste management and pollution control.
12. **Agricultural Planning:** Optimizing the use of resources in agriculture for crop and livestock management.

4.1 Pooling Problems

The pooling problem is a well-known Mixed Integer Nonlinear Programming (MINLP) problem, particularly prevalent in various industrial processes. It involves the optimal blending of different raw materials to meet certain quality specifications at the minimum cost.

FORMULATION There are various formulations for this problem and many variations based on operational needs. Here is one basic formulation.



SETS AND INDICES

- $i \in I$: set of raw materials.
- $j \in J$: set of pools where materials are mixed.
- $k \in K$: set of final products.

PARAMETERS

- a_i : cost of raw material i .
- q_{ik} : quality of raw material i for product k .
- Q_k : required quality for product k .
- D_k : demand for product k .

DECISION VARIABLES

- x_{ij} : amount of raw material i to pool j .
- y_{jk} : amount transferred from pool j to product k .
- z_k : amount of product k produced.

OBJECTIVE Minimize the total cost of raw materials:

$$\min \sum_{i \in I} \sum_{j \in J} a_i x_{ij}$$

CONSTRAINTS

- Material balance for pools:

$$\sum_{i \in I} x_{ij} = \sum_{k \in K} y_{jk}, \quad \forall j \in J$$

- Quality constraints for products:

$$\sum_{j \in J} q_{ik} y_{jk} / z_k \geq Q_k, \quad \forall k \in K$$

- Demand satisfaction:

$$\sum_{j \in J} y_{jk} = D_k, \quad \forall k \in K$$

- Non-negativity and integrality:

$$x_{ij}, y_{jk}, z_k \geq 0, \quad x_{ij} \in \mathbb{Z}$$

APPLICATIONS The pooling problem finds applications in various industries such as:

- Petroleum refining: Blending crude oils for specific fuel properties.
- Food and beverage industry: Mixing ingredients for desired nutritional values.
- Chemical industry: Creating mixtures with specific chemical properties.
- Agricultural sector: Blending fertilizers for optimal nutrient composition.
- Water resources management: Combining water from different sources to meet quality standards.

The pooling problem is a complex and significant challenge in the field of operations research and mathematical optimization, due to its nonlinear and non-convex nature.

4.2 Unit Commitment Problem

Conic relaxations of the unit commitment problem Author links open overlay panel - Salar Fattahi, Morteza Ashraphijuo, Javad Lavaei, Alper Atamtürk

INTRODUCTION The Unit Commitment Problem (UCP) in power system operations involves determining the on/off status of power generation units and their output levels over a time horizon, typically 24 hours. The aim is to meet electricity demand at minimum cost while respecting operational constraints.

OPTIMIZATION MODEL

VARIABLES

- Binary Variables:
 - u_{gt} : 1 if generation unit g is on at time t ; 0 otherwise.
- Continuous Variables:
 - p_{gt} : Power output of unit g at time t .
 - s_{gt} : Start-up costs for unit g at time t .

OBJECTIVE Minimize the total operational cost:

$$\text{Minimize} \sum_{g,t} (\text{FuelCost}(p_{gt}) + u_{gt} \cdot s_{gt})$$

CONSTRAINTS

- Demand Constraint:

$$\sum_g p_{gt} = \text{Demand}_t, \quad \forall t$$
- Generation Limits:

$$\text{MinGen}_g \cdot u_{gt} \leq p_{gt} \leq \text{MaxGen}_g \cdot u_{gt}, \quad \forall g, t$$
- Ramp Rate Constraints, Minimum Up/Down Time Constraints, Start-up/Shut-down Constraints.

NONLINEARITY Nonlinearity arises from start-up costs and fuel costs, making the UCP a Mixed Integer Nonlinear Programming (MINLP) problem.

FUEL COST MODELING Fuel costs are modeled as a function of power output:

$$\text{FuelCost}(p_{gt}) = a_g + b_g \cdot p_{gt} + c_g \cdot p_{gt}^2$$

where a_g , b_g , and c_g are cost coefficients for unit g .

RAMPING RATE CONSTRAINTS Ramping constraints limit output changes between periods:

- Ramp Up: $p_{gt} - p_{g,t-1} \leq \text{RampUp}_g$
- Ramp Down: $p_{g,t-1} - p_{gt} \leq \text{RampDown}_g$

for all g and t .

MINIMUM UP/DOWN TIME CONSTRAINTS Constraints for minimum operational times:

- Minimum Up Time: If $u_{gt} - u_{g,t-1} = 1$, then $u_{g\tau} = 1$ for $\tau = t, t+1, \dots, t + \text{MinUpTime}_g - 1$
- Minimum Down Time: If $u_{g,t-1} - u_{gt} = 1$, then $u_{g\tau} = 0$ for $\tau = t, t+1, \dots, t + \text{MinDownTime}_g - 1$

START-UP/SHUT-DOWN CONSTRAINTS Modeling transitions between on and off states:

- Start-up: $y_{gt} = u_{gt} - u_{g,t-1}$ indicates a start-up at time t for unit g .
- Shut-down: $z_{gt} = u_{g,t-1} - u_{gt}$ indicates a shut-down at time t for unit g .

4.3 Machine Learning and Integer Programming

4.3.1. Outlier Detection in Regression

Several statistical and machine learning problems can be formulated as optimization problems of the form

$$\begin{aligned} & \min_{x,z} \sum_{i=1}^m \left(y_i - a_i^\top x \right)^2 (1 - z_i) \\ & \text{s.t. } (x,z) \in F \subseteq \mathbb{R}^n \times \{0,1\}^m, \end{aligned}$$

where $(a_i, y_i) \in \mathbb{R}^{n+1}$ for all $i \in \{1, \dots, m\}$ are given data and F is the feasible region. Problem (1) includes the least trimmed squares as a special case, which is a focus of this paper and discussed at length in §1.1, but also includes regression trees [17] (where $1 - z_i = 1$ indicates that a given datapoint is routed to a given leaf), regression problems with mismatched data [38] (where variables z indicate the datapoint/response pairs) and k-means [33] (where variables z represent assignment of datapoints to clusters). We point out that few or no mixed-integer optimization (MIO) approaches exist in the literature for (1), as the problems are notoriously hard to solve to optimality, and heuristics are preferred in practice.¹

¹https://optimization-online.org/wp-content/uploads/2023/05/Least_Trimmed_Squares-4.pdf

4.3.2. Quadratic constraints with indicator variables

2

SPARSE PCA Set X arises directly in sparse principal component analysis problems [11, 12, 17, 25], a fundamental problem in statistics which can be formulated as

$$\begin{aligned} \max \quad & x' \Sigma x \\ \text{s.t.} \quad & \|x\|_2^2 \leq 1, \|z\|_1 \leq k, x \circ (e - z) = 0, \end{aligned}$$

where $\Sigma \succeq 0$ and $k \in \mathbb{Z}_+$ is a parameter controlling the sparsity of the solution. Observe that the feasible region given by constraints (2b) corresponds exactly to X with set $Z = \{z \in \{0, 1\}^n : \|z\|_1 \leq k\}$. Thus, understanding $\text{conv}(X)$ is critical to designing better convex approximations of (2).

GENERAL CONVEX QUADRATIC CONSTRAINTS Given $\Sigma \succeq 0$, consider the system of inequalities

$$y' \Sigma y \leq b, y \circ (e - z) = 0, y \in \mathbb{R}^n, z \in Z \subseteq \{0, 1\}^n.$$

System (3) arises for example in mean-variance optimization problems [5], where the quadratic constraint is used to impose an upper bound on the risk (variance) of the solution. While system (3) involves a non-separable quadratic constraint, a study of set $\text{conv}(X)$ can be still used to construct strong convex relaxations. Indeed, if $\Sigma = D + R$ where $R \succeq 0, D \succ 0$ and diagonal, then we can reformulate system (3) by introducing additional variables $(x_0, x) \in \mathbb{R}^{n+1}$ as

$$\begin{aligned} \sum_{i=0}^n x_i^2 \leq 1, x \circ (e - z) = 0, x_0(1 - z_0) = 0, z \in Z \\ z_0 = 1, \sqrt{(y'(R/b)y)} \leq x_0, \sqrt{(D_{ii}/b)} |y_i| \leq x_i \text{ for } i = 1, \dots, n, \end{aligned}$$

where constraints (4a) correspond precisely to X and constraints (4b) are convex and SOCRepresentable. Therefore, convex relaxations for system (3) can be obtained by strengthening constraints (4a) using $\text{conv}(X)$.

ROBUST OPTIMIZATION Consider a robust optimization problem of the form

$$\min_{y \in Y} \max_{a \in \mathcal{U}} a'y,$$

where vector y are the decision variables, set $Y \subseteq \mathbb{R}^n$ is the (possibly non-convex) feasible region and set $\mathcal{U} \subseteq \mathbb{R}^n$ is an uncertainty set corresponding to the objective coefficients. Robust optimization (5) is a fundamental tool to tackle decision-making under uncertainty problems. Two popular choices for the uncertainty set \mathcal{U} , each with its own merits and disadvantages, are: the approach of Ben-Tal and Nemirovski [7], where \mathcal{U} is an ellipsoid; and the approach of Bertsimas and Sim [8], where only a small subset of the coefficients a are allowed to change while satisfying box constraints.

²A note on quadratic constraints with indicator variables: Convex hull description and perspective relaxation - Andrés Gómez, Weijun Xie

Thus, a natural uncertainty set inspired by the aforementioned two approaches allows few coefficients to change and imposes ellipsoidal constraint on the changing coefficients, that is, set

$$\mathcal{U} \stackrel{\text{def}}{=} \left\{ a \in \mathbb{R}^n : \exists (x, z) \in \mathbb{R}^n \times \{0, 1\}^n \text{ s.t. } a = \tilde{a} + x, \sum_{i=1}^n (d_i x_i)^2 \leq b, \|z\|_1 \leq k, x \circ (e - z) = 0 \right\},$$

where \tilde{a} are the nominal values for the coefficients. The uncertainty set \mathcal{U} is appropriate for example when changes in coefficients a are caused by rare events, and the change in the coefficients (when such changes occur) can be accurately modeled with a Gaussian distribution. Constraint $\|z\|_1 \leq k$ could be replaced by other constraints to capture more sophisticated relationships on the support of the perturbed coefficients.

Since set \mathcal{U} is non-convex, solving (5) can be difficult and require sophisticated approaches [9]. Nonetheless, understanding $\text{conv}(X)$ may lead to the possibility of using standard duality approaches to obtain deterministic counterparts of (5).

OTHER APPLICATIONS Various other applications exist in the field.³

Given model matrix $F \in \mathbb{R}^{m \times n}$ and responses $\beta \in \mathbb{R}^m$, setting $a = -\beta^\top F, Q = F^\top F, b = 0$ and $Z = \{z \in \{0, 1\}^n : \sum_{i=1}^n z_i \leq r\}$ in (1) is equivalent to the best subset selection problem with a given cardinality r [10, 16] :

$$\min_{x, z} \|\beta - Fx\|_2^2 \quad \text{s.t.} \quad x \circ (e - z) = 0, \sum_{i=1}^n z_i \leq r.$$

Other constraints defining Z that have been considered in statistical learning applications include multicollinearity [10], cycle prevention [28, 30], and hierarchy [12]. Set X arises as a substructure in many other applications, including portfolio optimization [13], optimal control [21], image segmentation [26], signal denoising [9].

See also⁴

4.3.3. Control theory and others

4.3.4. Pure Mathematics

Theorem 2.2 ([10]). Let G be a graph on n vertices with adjacency matrix A and clique number ω . The optimal value of the quadratic program

$$\begin{aligned} & \max_{x \in \mathbb{R}^n} && x^\top A x \\ & \text{subject to} && x \geq 0, \\ & && \sum_{i=1}^n x_i = 1 \end{aligned}$$

is $1 - \frac{1}{\omega}$.⁵

³On the convex hull of convex quadratic optimization problems with indicators -

⁴An outer approximation method for solving mixed-integer convex quadratic programs with indicators

⁵<https://arxiv.org/pdf/2008.05558.pdf>

SOME REFERENCES Harjunkoski, Westerlund, Pörn, and Skrifvars 1998; Meyer and Floudas 2006; Misener and Floudas 2009; Quesada and Grossmann 1995; Rinaldi, Voigt, and Woeginger 2002; Karuppiah and Grossmann 2006; Bienstock 2007; Bussieck, Drud, and Meeraus 2003; Caprara and Monaci 2009; D'Ambrosio, Frangioni, Liberti, and Lodi 2010; Sherali and Alameddine 1992

4.4 Sparse Portfolio optimization

6

Portfolio selection problems. Given a universe of n risky assets, we denote the mean return vector as μ and the covariance matrix as Q . The mean-variance portfolio selection problem with sparsity, minimum investment, maximum investment, and minimum return constraints can be formulated as follows:

$$\begin{aligned} \min_{x,y} \quad & y^\top Qy \\ \text{s.t.} \quad & \sum_{i=1}^n y_i = 1 \\ & \mu^\top y \geq \rho \\ & \alpha_i x_i \leq y_i \leq u_i x_i, \quad \forall i \in [n] \\ & \sum_{i=1}^n x_i \leq k \\ & x \in \{0,1\}^n. \end{aligned}$$

4.5 Portfolio Optimization with Higher Order Moments

7

2.1. Higher-order-moment portfolio selection model

In this subsection, we first consider the problem of constructing a portfolio from n financial products, which will be selected by our two-clustering method introduced in Section 3. A portfolio $X = (x_1, x_2, \dots, x_n)^T$ is a vector of the proportions in each of these n assets with $x_1 + x_2 + \dots + x_n = 1$. The mean return of the i -th asset is $\bar{R}_i, i = 1, \dots, n$. \bar{R} is the vector of the expected return, while V is the covariance matrix. Denote σ_i^2 as the variance of asset i 's return, and σ_{ij} as the covariance between the returns of assets i and j . The expected return and its variance, skewness and kurtosis are thus computed as follows:

$$\begin{aligned} \mathbb{E}(X) &= X^T \bar{R} = \sum_{i=1}^n x_i \bar{R}_i \\ \text{Var}(X) &= X^T V X = \sum_{i=1}^n x_i^2 \sigma_i^2 + \sum_{i=1}^n \sum_{j=1}^n x_i x_j \sigma_{ij} (i \neq j) \end{aligned}$$

⁶AN OUTER APPROXIMATION METHOD FOR SOLVING MIXED-INTEGER CONVEX QUADRATIC PROGRAMS WITH INDICATORS LINCHUAN WEI, SIMGE KUCUKYAVUZ

⁷A hybrid approach for portfolio selection with higher-order moments: Empirical evidence from Shanghai Stock Exchange Bilian Chena,b , Jingdong Zhongc , Yuanyuan Chend,

$$\text{Skew}(X) = \mathbb{E} (X^T (R - \bar{R})^3),$$

$$\text{Kurt}(X) = \mathbb{E} (X^T (R - \bar{R})^4).$$

The classical mean-variance portfolio selection problem is thus expressed as the following multi-objective optimization problem. $\max \mathbb{E}(X) \min \text{Var}(X)$ s.t. $X^T 1 = 1, X \geq 0$. where 1 is a all-one vector. The constraint that $X \geq 0$ is the no shorting constraint. In other words, short selling is not allowed in our model. Because a portfolio with a larger skewness and a smaller kurtosis is more appealing to the investors, we improve the classical mean-variance model's performance by including skewness and kurtosis. The higher-order-moment model is thus described as the following multi-objective optimization problem (P_1).

$$(P_1) \left\{ \begin{array}{l} \max \mathbb{E}(X) \\ \min \text{Var}(X) \\ \max \text{Skew}(X) \\ \min \text{Kurt}(X) \\ \text{s.t. } X^T 1 = 1, X \geq 0. \end{array} \right.$$

In order to solve problem (P_1) , we can transform it into the following non-linear programming:

$$(P_2) \left\{ \begin{array}{l} \max \lambda_1 (X^T \bar{R}) - \lambda_2 (X^T V X) + \lambda_3 [\mathbb{E} (X^T (R - \bar{R})^3)] \\ \quad - \lambda_4 [\mathbb{E} (X^T (R - \bar{R})^4)] \\ \text{s.t. } X^T 1 = 1, X \geq 0, \end{array} \right.$$

where $\lambda_1, \lambda_2, \lambda_3$ and λ_4 are non-negative investment preference factors, corresponding to the objectives: mean, variance, skewness and kurtosis, respectively. In fact, the auxiliary parameters, $\lambda_i, i = 1, 2, 3, 4$, capture the risk preference of the investor. For example, an investor who doesn't care about the kurtosis can set the parameter λ_4 , which is corresponding to the kurtosis, to be zero. We focus on the solution scheme for model (P_2) with different sets of given auxiliary parameters $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ in this paper.

Can you solve this problem with sparsity enforced as well?

4.6 Generalizations of Knapsack Problem

⁸ The 0 – 1 cubic knapsack problem (CKP), an extension of the celebrated 0 – 1 quadratic knapsack problem (QKP), consists of maximizing a 0 – 1 cubic function subject to a single knapsack constraint:

$$\text{CKP : maximize } F(x) = \sum_{i=1}^n c_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n C_{ij} x_i x_j + \sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^n D_{ijk} x_i x_j x_k$$

subject to

$$\sum_{\substack{i=1 \\ x \text{ binary}}}^n a_i x_i \leq b$$

⁸ Strengthening a linear reformulation of the 0-1 cubic knapsack problem via variable reordering Published: 21 January 2022 Richard J. Forrester & Lucas A. Waddell

Within CKP, the $c_i, C_{ij}, D_{ijk}, a_i$, and b coefficients are all assumed to be nonnegative. For notational ease, we henceforth let the indices i, j , and k run from 1 to n unless otherwise stated. Furthermore, it is convenient to define the constraint set of CKP (with the binary restrictions relaxed) as

$$X \equiv \{x \in \mathbb{R}^n : (1), 0 \leq x_i \leq 1 \text{ for all } i\}$$

Many problems can be modeled by 0 – 1 cubic programs such as CKP, with applications including the satisfiability problem Max 3-SAT (Kofler et al. 2014), the selection problem (Gallo et al. 1989), the network alignment problem (Mohammadi et al. 2017), and screening for and treating sexually transmitted diseases (Zhao 2008). The quadratic and cubic terms in the objective function are often used to account for interactions of yes-no decisions, such as those in the area of capital budgeting (Fox et al. 1984; Weingartner 1966), where the binary variables indicate whether or not to invest in different projects.

4.7 Machine Learning

Safe Screening Rules for ℓ_0 -Regression from Perspective Relaxations - Alper Atamturk and Andres Gomez

In machine learning and optimization communities, there is an increasing interest in regression models with ℓ_0 and ℓ_2 regularization:

$$\begin{aligned} & \min_{x \in \mathbb{R}^n} \|y - Ax\|_2^2 + \frac{1}{\gamma} \|x\|_2^2 + \mu \|x\|_0, \text{ and} \\ & \min_{x \in \mathbb{R}^n} \|y - Ax\|_2^2 + \frac{1}{\gamma} \|x\|_2^2 \text{ s.t. } \|x\|_0 \leq k, \end{aligned}$$

(CARD) where $A \in \mathbb{R}^{m \times n}$ is the model matrix, $y \in \mathbb{R}^m$ is the vector of response variables, and $x \in \mathbb{R}^n$ is the vector of decision variables, i.e., regression coefficients to be estimated. Problem (CARD) has an explicit cardinality constraint on the number of non-zeros of x , whereas (REG) is the regularized version of it. In these models, the ℓ_0 terms impose sparsity (Miller, 2002), which is a necessity for large-dimensional model inference (Hastie et al., 2001; 2015), and the ℓ_2 (ridge) regularization (Hoerl & Kennard, 1970) imposes bias/shrinkage in the regression coefficients. The ℓ_2 regularization can be interpreted, from the robust optimization perspective, as a correction term to account for uncertainty in the model matrix A (El Ghaoui & Lebret, 1997; Xu et al., 2009), and has been shown to improve the performance of sparse regression models in high-noise regimes (Mazumder et al., 2017).

The popular ℓ_1 (lasso, Tibshirani, 1996) and $\ell_1 - \ell_2$ (elastic net, Zou & Hastie, 2005) regularizations perform shrinkage and model selection simultaneously and, as convex proxies for (REG), they are very fast. However, thanks to substantial progress in the field of mixed-integer optimization (MIO), there is an increasing interest in solving the non-convex problems (REG)-(CARD) directly. Indeed, several studies (Bertsimas et al., 2016; Cozad et al., 2014; Gómez & Prokopyev, 2018; Miyashiro & Takano, 2015; Park & Klabjan, 2017) have shown that problems (REG)-(CARD) with hundreds of variables can be solved to optimality simply by employing general purpose MIO solvers, and the resulting estimators outperform their ℓ_1 counterparts. Nonetheless, solving the ℓ_0 problems in this manner is orders-of-magnitude

slower than solving the ℓ_1 approximations and does not scale to problems with $n \geq 1,000$. Therefore, fast heuristics such as ℓ_1 approximations, thresholding, local (but combinatorial) search algorithms or greedy methods (Hastie et al., 2017; Hazimeh & Mazumder, 2018; Xie & Deng, 2020) may still be preferable in large-scale instances.

4.8 Airline scheduling

Aircraft Rescheduling with Cruise Speed Control M. Selim Aktürk, Alper Atamtürk, Sinan Gürel Airline operations are subject to frequent disruptions typically due to unexpected aircraft maintenance requirements and undesirable weather conditions. Recovery from a disruption often involves propagating delays in downstream flights and increasing cruise stage speed when possible in an effort to contain the delays. However, there is a critical trade-off between fuel consumption (and its adverse impact on air quality and greenhouse gas emissions) and cruise speed. Here we consider delays caused by such disruptions and propose a flight rescheduling model that includes adjusting cruise stage speed on a set of affected and unaffected flights as well as swapping aircraft optimally.

To the best of our knowledge, this is the first study in which the cruise speed is explicitly included as a decision variable into an airline recovery optimization model along with the environmental constraints and costs. The proposed model allows one to investigate the trade-off between flight delays and the cost of recovery. We show that the optimization approach leads to significant cost savings compared to the popular recovery method delay propagation.

Flight time controllability, nonlinear delay, fuel burn and CO₂ emission cost functions, and binary aircraft swapping decisions complicate the aircraft recovery problem significantly. In order to mitigate the computational difficulty we utilize the recent advances in conic mixed integer programming and propose a strengthened formulation so that the nonlinear mixed integer recovery optimization model can be solved efficiently. Our computational tests on realistic cases indicate that the proposed model may be used by operations controllers to manage disruptions in real time in an optimal manner instead of relying on ad-hoc heuristic approaches.

4.9 Inverse Optimization

Inverse Optimization for Imputing Constraints in Mathematical Programs Archis Ghate Clemson University

$$(P_c) \min \sum_{j=1}^n c_j x_j$$

$$\sum_{j=1}^n a_{ij} x_j \geq b_i, i = 1 : m.$$

Given a_{ij} and b_i , for $i = 1 : m$ and $j = 1 : n$. Given feasible $x^* \in \mathbb{R}^n$. Find a $c \in \mathbb{R}^n$ that makes x^* optimal to (P_c) . Markowitz portfolio problem (QP) Forward Find a portfolio $x = (x_1, \dots, x_n)$ that minimizes risk while meeting a return target γ .

$$\min_{x \in \mathbb{R}^n} \left\{ \frac{1}{2} x^T \Sigma x \mid \sum_{j=1}^n r_j x_j \geq \gamma, \sum_{j=1}^n x_j = 1, x \geq 0 \right\}$$

Inverse Find returns $r = (r_1, \dots, r_n)$ and a covariance matrix Σ for which a given portfolio x^* minimizes risk when the target return is γ .

Cancer radiotherapy problem (QCQP) Forward Find a nonnegative dosing plan $d = (d_1, \dots, d_n)$ to max tumor biological effect.

$$\begin{aligned} & \max \alpha_0 \sum_{t=1}^n d_t + \beta_0 \sum_{t=1}^n d_t^2 \\ & s_m \sum_{t=1}^n d_t + s_m^2 \rho_m \sum_{t=1}^n d_t^2 \leq T \delta_m (1 + \rho_m \delta_m), m \in \mathcal{M} \\ & d \geq 0 \end{aligned}$$

Inverse Find response parameters α_m, β_m , for $m \in \mathcal{M}$, that render a treatment protocol d^* optimal.

Air pollution control problem (SILP) Forward Find nonnegative fractional reductions $x = (x_1, \dots, x_n)$ in pollutant quantities emitted by sources $j = 1 : n$.

$$\begin{aligned} & \min \sum_{j=1}^n c_j x_j \\ & \sum_{j=1}^n (1 - x_j) a_j(\theta) \leq b(\theta), \theta \in [0, 1] \\ & 0 \leq x \leq 1 \end{aligned}$$

Inverse Which $(c_j, a_j(\theta), \forall \theta \in [0, 1])$ pairs, for sources $j = 1 : n$, would render an emission reduction plan x^* optimal.

4.10 Wireless Networks

Lee, Yu, and Li 2020

A. Resource Allocation in D2D Communications

As depicted in Fig. 1, we consider an uplink single-cell system with K CUs in a set $\mathcal{K} = \{1, \dots, K\}$ and L D2D pairs in a set $\mathcal{L} = \{1, \dots, L\}$. We assume that each uplink CU connects to the Evolved NodeB (eNB) with an orthogonal channel. Moreover, we assume that D2D pairs transmit data by reusing the uplink

channels of CUs. In the D2D networks, the number of D2D pairs is usually smaller than that of cellular users [22], [23]. Therefore, we assume $K \geq L$ in this paper. Nevertheless, our proposed method can also be utilized while $K \leq L$.

As in Fig. 1, we denote h_{kl}^{CD} as the instantaneous channel power gain of the interference link between CU k and the receiver of D2D pair l , h_k^{CB} as the channel power gain between CU k and the eNB, h_l^D as the channel power gain between D2D pair l , and h_l^{DB} as the channel power gain of the interference link between the transmitter of D2D pair l and the eNB.

We further introduce $\rho = [\rho_{kl}]$ as the indicator vector of the channel allocation. Specifically, $\rho_{kl} = 1$ if the channel of CU k is reused by D2D pair l , and $\rho_{kl} = 0$ otherwise. Denote $p^C = [p_k^C]$ and $p^D = [p_{kl}^D]$ as the transmit power vectors for K CUs and L D2D pairs, respectively, where p_k^C denotes the allocated transmit power for CU k and p_{kl}^D denotes the power of D2D pair l on the channel of CU k .

Since each CU channel is assumed to be reused by at most one D2D pair, the signal-to-interference-plus-noise ratio (SINR) of D2D pair l on the channel of CU k can be written as

$$SINR_{kl}^D(p^C, p^D, \rho) = \frac{\rho_{kl} p_{kl}^D h_l^D}{\sigma_N^2 + p_k^C h_{kl}^{CD}},$$

where σ_N^2 denotes the power of the additive white Gaussian noise (AWGN). Similarly, the SINR achieved by CU k can be expressed as

$$SINR_k^C(p^C, p^D, \rho) = \frac{p_k^C h_k^{CB}}{\sigma_N^2 + \sum_{l \in \mathcal{L}} \rho_{kl} p_{kl}^D h_l^{DB}}.$$

Accordingly, the data rates in bits per second per hertz (i.e., normalized by the channel bandwidth) of CU k and D2D pair l on all channels can be written as

$$\begin{aligned} R_k^C(p^C, p^D, \rho) &= \log(1 + SINR_k^C(p^C, p^D, \rho)), \\ R_l^D(p^C, p^D, \rho) &= \sum_{k \in \mathcal{K}} \rho_{kl} R_{kl}^D(p^C, p^D, \rho) \\ &= \sum_{k \in \mathcal{K}} \rho_{kl} \log(1 + SINR_{kl}^D(p^C, p^D, \rho)), \end{aligned}$$

respectively, where R_{kl}^D is the data rate of D2D pair l on the channel of CU k .

Resource allocation in D2D communications is to determine p^C, p^D , and ρ to optimize the overall network performance. B. Problem Formulation

To achieve fairness among different users, we specifically consider a resource allocation problem to maximize the minimum data rate of D2D pairs in this paper with the following constraints. First, the minimum data rate of each CU is required to be no less than given thresholds. Second, the power of individual links is also constrained. Finally, as mentioned above, each channel can be reused by at most one D2D pair to limit the interference between different D2D pairs. Therefore, the resource allocation problem can be mathematically formulated as

$$\max_{\{p^C, p^D, \rho\}} \min_{l \in \mathcal{L}} R_l^D(p^C, p^D, \rho),$$

subject to

$$\begin{aligned}\rho_{kl} &\in \{0, 1\}, \quad \forall k \in \mathcal{K}, l \in \mathcal{L}, \\ \sum_{l \in \mathcal{L}} \rho_{kl} &\leq 1, \quad \forall k \in \mathcal{K}, \\ \sum_{k \in \mathcal{K}} \rho_{kl} p_{kl}^D &\leq P_{\max}^D, \quad \forall l \in \mathcal{L}, \\ R_k^C(p^C, p^D, \rho) &\geq R_{\min}^C, \quad \forall k \in \mathcal{K}, \\ p_k^C &\leq P_{\max}^C, \quad \forall k \in \mathcal{K},\end{aligned}$$

where R_{\min}^C is the minimum rate of CUs, P_{\max}^C and P_{\max}^D are the maximum transmit power level of CUs and D2D pairs, respectively. Note that, the proposed algorithm can be extended to other resource allocation problems with different objective functions since the B&B algorithm is a general approach to solve MINLP problems.

4.11 Watershed Management

Boddiford, Kaufman, Skipper, and Uhan 2023

2.2. Linear multiplicative programming The problem that we study in this paper can be formulated as a generalized linear multiplicative program. A generalized linear multiplicative program is an optimization problem of the form

$$\underset{\mathbf{x}}{\text{minimize}} f(\mathbf{x}) = \sum_{i=1}^t \prod_{j=1}^p (\mathbf{c}_{ij}^\top \mathbf{x} + d_{ij})$$

subject to $\mathbf{Ax} \leq \mathbf{b}$, $1 \leq \mathbf{x} \leq \mathbf{u}$, where \mathbf{x} is a vector of decision variables, \mathbf{c}_{ij} ($i = 1, \dots, t$ and $j = 1, \dots, p$), \mathbf{b} , 1 , and \mathbf{u} are real-valued vectors; d_{ij} ($i = 1, \dots, t$ and $j = 1, \dots, p$) are real numbers; and \mathbf{A} is a real-valued matrix. Such optimization models have been applied to a variety of disciplines, including financial optimization (Konno, Shirakawa, & Yamazaki, 1993; Maranas, Androulakis, Floudas, Berger, & Mulvey, 1997), very large-scale integration (VLSI) chip design (Dorneich & Sahinidis, 1995; Maling, Heller, & Mueller, 1982), multi-objective optimization (Geoffrion, 1967), plant layout design (Quesada & Grossmann, 1996), robust optimization (Mulvey, Vanderbei, & Zenios, 1995), and data mining and pattern recognition (Bennett & Mangasarian, 1993).

3. Problem description and optimization models 3.1. Preliminaries and notation

The geographic area under consideration in our models is partitioned into parcels; we use P to denote the set of parcels. For each parcel $p \in P$, α_p represents the size of p in acres, ϕ_p represents the pollutant load contributed by p in pounds per acre, and B^P represents the set of BMPs that could be applied to parcel p . In the CAST watershed model, the set of BMPs that can be applied to a parcel depends on the location of the land, the land usage (farming, commercial, residential, etc.), and the government entity that manages it. For more details about the CAST watershed model, which captures more details about each parcel of land than our models require, see (Chesapeake Bay Program, 2020; Kaufman et al., 2021).

Our models include only efficiency BMPs, BMPs whose impacts have been quantified by efficiency ratings. The efficiency rating, or effectiveness, of BMP $b \in B^P$ when it is applied to parcel $p \in P$, denoted

$\eta_{p,b}$, represents the fraction of load that is removed, or filtered, by b . Thus, $0 \leq \eta_{p,b} \leq 1$, and if, for example, $\eta_{p,b} = 0.1$, the pollutant load on any acreage in p to which b is applied is reduced by 10%. The pass-through percentage of b when applied to p is $\theta_{p,b} = 1 - \eta_{p,b}$. In our example, $\theta_{p,b} = 0.9$, meaning 90% of the original load "passes through" to reach the water. We use the BMP effectiveness ratings from CAST, which are determined by panels of experts convened by the CBP (Devereux & Rigelman, 2014). In addition to a BMP's efficiency rating, we also need its cost, which we denote τ_b . BMP cost is measured in US\$ per acre and does not depend on the parcel to which b is applied.

The main complication in our models is "overlapping" BMPs. Some BMPs can be applied simultaneously (or overlapping) on the same acreage, while others cannot. We use BMP groups to indicate which BMPs are allowed to overlap. BMPs in the same group cannot overlap, while BMPs in different groups can overlap. Note that BMPs in the same group can be implemented on the same parcel (additive effect), as long as they are not overlapping (multiplicative effect). For example, consider a 10 -acre parcel. Given two BMPs from the same group, the sum of their implemented acres of the parcel. The impact of multiple overlapping BMPs is captured by multiplying their pass-through percentages. For example, if $\theta_{b_1,p} = \theta_{b_2,p} = 0.9$, collectively, they allow $0.9^2 = 0.81$, or 81%, of the load to pass-through when applied together on the same land.

We use $\mathcal{G}^p = \{G_1, G_2, \dots, G_{k_p}\}$ to denote the set of BMP groups that can be applied to parcel p (k_p represents the number of BMP groups that can be applied to parcel p). Every BMP that can be applied to p is in some BMP group: $B^p = \bigcup_{i=1}^{k_p} G_i$. Moreover, each BMP in B^p appears in exactly one group $G_i \in \mathcal{G}^p$ except the so-called null BMP, which appears in every BMP group. The null BMP, which we denote b_0 , represents selecting no treatment from a group. Thus, the null BMP has zero cost, an efficiency rating of zero, and a passthrough percentage of one (or 100%). Null BMPs are required in our linear model. For consistency, we use null BMPs in the nonlinear model, too.

We present our nonlinear model in Section 3.2 and our linear model in Section 3.3. The goal of both models is to find a management strategy (i.e., a portfolio of BMPs) that minimizes pollutant load subject to a given implementation budget, while respecting BMP implementation rules.

3.2. Nonlinear model

An optimal solution to (\mathcal{P}) (defined below) corresponds to a management strategy for a given budget that minimizes posttreatment pollutant load as calculated in CAST. For every $p \in P$ and $b \in B^p$, decision variable $x_{p,b}$ represents the number of acres of parcel p to which BMP b is applied; \mathbf{x} is the vector of decision variables. In order for the decision variables to be uniquely defined, each group G must have a uniquely indexed null BMP, say b_0^G . Using this notation along with the notation defined in the previous section, we have the following linear multiplicative program:

$$\begin{aligned} \text{minimize } & L(\mathbf{x}) := \sum_{p \in P} \alpha_p \phi_p \prod_{G \in \mathcal{G}^p} \left(1 - \sum_{b \in G} \frac{\eta_{p,b}}{\alpha_p} x_{p,b} \right) \\ \text{subject to } & \sum_{p \in P} \sum_{b \in B} \tau_b x_{p,b} \leq T; \\ & \sum_{b \in G} x_{p,b} = \alpha_p, \quad \forall p \in P, G \in \mathcal{G}^p; \\ & \mathbf{X} \geq 0. \end{aligned}$$

The objective function (\mathcal{P}_a) is the CAST load function, L , which represents the amount of pollutant load that all parcels collectively contribute, taking into account the application of efficiency BMPs. The

quantity $\alpha_p \phi_p$ represents the pre-treatment load on parcel p ; i.e., the load on parcel p before the efficiency BMPs have been implemented. The quantity $\frac{\eta_{p,b}}{\alpha_p} x_{p,b}$ represents the fraction of pollutant that is removed by BMP b when applied to parcel p , weighted by the fraction of acres of p to which b is applied; $1 - \sum_{b \in G} \frac{\eta_{p,b}}{\alpha_p} x_{p,b}$ is the percent of pollutant that is allowed to "pass through" the non-overlapping BMPs from group G that are applied to parcel p . The product $\prod_{G \in \mathcal{G}^p} \left(1 - \sum_{b \in G} \frac{\eta_{p,b}}{\alpha_p} x_{p,b}\right)$ captures the multiplicative effect of overlapping BMPs from different BMP groups on the same parcel. For example, if the BMPs applied from groups G_1 and G_2 each allow 90% of the pollutant to pass through, then when these groups are layered on the same parcel, together they allow 81% of the pollutant to pass through.

4.12 AC Optimal Power Flow Problem

THE ALTERNATING CURRENT OPTIMAL POWER FLOW (AC/OPF) PROBLEM The AC/OPF problem is a cornerstone in the field of power systems engineering, representing a critical application of mixed-integer nonlinear programming (MINLP). It involves the optimization of certain objectives, such as minimizing the cost of generation or maximizing system efficiency, subject to a set of physical and operational constraints.

DESCRIPTION OF THE PROBLEM The primary goal of the AC/OPF problem is to determine the optimal set points for the control variables of a power system, such as generator outputs, voltage magnitudes, and transformer tap settings, while satisfying a set of constraints. These constraints typically include power balance equations, transmission line limits, and voltage magnitude bounds, ensuring the secure and stable operation of the power system.

Understanding the AC/OPF Problem

Basic Concepts

- **Power System:** A network of electrical components used to supply, transmit, and use electric power, including power generation units, transformers, transmission lines, and consumers.
- **Bus:** A node in the power system where power is either generated, transformed, or consumed, acting as a junction for power lines.
- **Transmission Line:** Cables or wires through which electricity travels from one point to another.

Core Elements of AC/OPF

- **Active and Reactive Power:** Components of electricity; active power is used to run appliances, while reactive power maintains voltage levels.
- **Voltage and Phase Angle:** Voltage is the force pushing electric current, and phase angle is the timing difference in electricity waves across the system.
- **Generation and Demand:** Generation is the production of power, while demand is the power needed by consumers.

Problem Description

The AC/OPF problem involves optimizing the operation of a power system to meet consumer demands efficiently, minimize costs, and ensure system stability within safe operating limits.

Main Goals

1. **Minimizing Costs:** Finding cost-effective ways to produce the required electricity.
2. **Meeting Demand:** Ensuring the power generated meets the exact needs of consumers.
3. **Operating Within Limits:** Respecting maximum power capacities and safe voltage levels.
4. **Ensuring Stability:** Maintaining system reliability even with sudden changes in demand or other disturbances.

Mathematical Formulation

The AC/OPF problem is formulated mathematically with equations representing power flows, generation limits, and other constraints, aiming to optimize variables such as power generation and voltage levels while minimizing a cost function.

VARIABLES

- P_{G_i} : Active power generation at bus i .
- Q_{G_i} : Reactive power generation at bus i .
- V_i : Voltage magnitude at bus i .
- θ_i : Phase angle at bus i .

PARAMETERS

- P_{D_i} : Active power demand at bus i .
- Q_{D_i} : Reactive power demand at bus i .
- G_{ij} : Conductance of the transmission line between buses i and j .
- B_{ij} : Susceptance of the transmission line between buses i and j .
- S_{ij}^{\max} : Maximum apparent power flow limit of the transmission line between buses i and j .
- $P_{G_i}^{\min}, P_{G_i}^{\max}$: Minimum and maximum limits for active power generation at bus i .
- $Q_{G_i}^{\min}, Q_{G_i}^{\max}$: Minimum and maximum limits for reactive power generation at bus i .
- V_i^{\min}, V_i^{\max} : Minimum and maximum permissible voltage magnitudes at bus i .
- $\Delta\theta_{ij}^{\max}$: Maximum permissible phase angle difference between buses i and j .
- $C_i(\cdot)$: Cost function of generation at bus i .

MATHEMATICAL FORMULATION Consider a power system with a set of buses N and a set of transmission lines E . Let P_{G_i} and Q_{G_i} denote the active and reactive power generation at bus i , and P_{D_i} and Q_{D_i} denote the active and reactive power demand. The AC/OPF problem can be formulated as follows:

Objective Function:

$$\min \sum_{i \in N} C_i(P_{G_i}) \quad (4.1)$$

where $C_i(\cdot)$ represents the cost function of generation at bus i .

Subject to:

1. Power Balance Constraints:

$$P_{G_i} - P_{D_i} = \sum_{j \in N} (G_{ij}V_iV_j \cos(\theta_i - \theta_j) + B_{ij}V_iV_j \sin(\theta_i - \theta_j)), \forall i \in N \quad (4.2)$$

$$Q_{G_i} - Q_{D_i} = \sum_{j \in N} (G_{ij}V_iV_j \sin(\theta_i - \theta_j) - B_{ij}V_iV_j \cos(\theta_i - \theta_j)), \forall i \in N \quad (4.3)$$

where V_i and θ_i are the voltage magnitude and phase angle at bus i , and G_{ij} and B_{ij} are the conductance and susceptance of the line between buses i and j .

2. Line Flow Constraints:

$$(P_{ij}^2 + Q_{ij}^2) \leq (S_{ij}^{\max})^2, \forall (i, j) \in E \quad (4.4)$$

where P_{ij} and Q_{ij} are the active and reactive power flows on the line from bus i to bus j , and S_{ij}^{\max} is the maximum apparent power flow limit of the line.

3. Generator Limits:

$$P_{G_i}^{\min} \leq P_{G_i} \leq P_{G_i}^{\max}, \forall i \in N \quad (4.5)$$

$$Q_{G_i}^{\min} \leq Q_{G_i} \leq Q_{G_i}^{\max}, \forall i \in N \quad (4.6)$$

where $P_{G_i}^{\min}$, $P_{G_i}^{\max}$, $Q_{G_i}^{\min}$, and $Q_{G_i}^{\max}$ are the minimum and maximum limits for active and reactive power generation at bus i

1. Voltage Magnitude Limits:

$$V_i^{\min} \leq V_i \leq V_i^{\max}, \forall i \in N \quad (4.7)$$

where V_i^{\min} and V_i^{\max} are the minimum and maximum permissible voltage magnitudes at bus i .

2. Phase Angle Difference Constraints:

$$|\theta_i - \theta_j| \leq \Delta\theta_{ij}^{\max}, \forall (i, j) \in E \quad (4.8)$$

where $\Delta\theta_{ij}^{\max}$ is the maximum permissible phase angle difference between buses i and j .

CHALLENGES AND COMPLEXITY The AC/OPF problem is inherently nonlinear and nonconvex due to the trigonometric relationships in the power flow equations. This complexity makes it a challenging problem to solve, particularly for large-scale power systems. Advanced optimization techniques, such as interior-point methods, branch-and-bound algorithms, and heuristic approaches, are often employed to find feasible and near-optimal solutions.

APPLICATIONS The AC/OPF problem is critical in the operation and planning of power systems. It is used for determining the most economical generation dispatch, ensuring system reliability and stability, and planning for future expansions and upgrades. The solutions of the AC/OPF problem provide valuable insights into the efficient and secure operation of power systems.

Resources

[Dan Bienstock - Gurobi Talk Part 1](#)

[Dan Bienstock - Gurobi Talk Part 2](#)

[Electrical Transmission System Cascades and Vulnerability: An Operations Research Viewpoint, Dan Bienstock, ISBN 978-1-611974-15-7. SIAM-MOS Series on Optimization \(2015\).](#)

4.13 Other applications

Günlük and Linderoth 2012; Günlük and Linderoth 2010

5.1 Separable quadratic UFL

The Separable Quadratic Uncapacitated Facility Location Problem (SQUFL) was introduced by Günlük et al. [20]. In the SQUFL, there is a set of customers N , a set of facilities M and there is a fixed cost c_i for opening a facility $i \in M$. All customers have unit demand that can be satisfied using open facilities only. The shipping cost is proportional to the square of the quantity delivered. Letting z_i indicate if facility $i \in N$

is open, and x_{ij} denote the fraction of customer j 's demand met from facility i , SQUFL can be formulated as follows:

$$\begin{aligned} & \min \sum_{i \in M} c_i z_i + \sum_{i \in M} \sum_{j \in N} q_{ij} x_{ij}^2 \\ & \text{subject to } x_{ij} \leq z_i \quad \forall i \in M, \forall j \in N, \\ & \sum_{i \in M} x_{ij} = 1 \quad \forall j \in N, \\ & z_i \in \{0, 1\}, \quad x_{ij} \geq 0 \quad \forall i \in M, \forall j \in N. \end{aligned}$$

5.2 Network design with congestion constraints

The next application is a network design problem with requirements on queuing delay. Similar models appear in [10, 6, 11]. In the problem, there is a set of commodities K to be shipped over a capacitated directed network $G = (N, A)$. The capacity of arc $(i, j) \in A$ is u_{ij} , and each node $i \in N$ supplies or demands a specified amount b_i^k of commodity k . There is a fixed cost c_{ij} of opening each arc $(i, j) \in A$, and we introduce $\{0, 1\}$ -variables z_{ij} to indicate whether arc $(i, j) \in A$ is opened. The quantity of commodity k routed on arc (i, j) is measured by variable x_{ij}^k and $f_{ij} = \sum_{k \in K} x_{ij}^k$ denotes the total flow on the arc. A typical measure of the total weighted congestion (or queuing delay) is

$$\rho(f) \stackrel{\text{def}}{=} \sum_{(i, j) \in A} r_{ij} \frac{f_{ij}}{1 - f_{ij}/u_{ij}},$$

where $r_{ij} \geq 0$ is a user-defined weighting parameter for each arc. We use a decision variables y_{ij} to measure the contribution of the congestion on arc (i, j) to the total congestion $\rho(f)$. The network should be designed so as to keep the total queuing delay less than a given value β , and this is to be accomplished at minimum cost. The resulting optimization model (NDCC) can be written as

$$\begin{aligned} & \min \sum_{(i, j) \in A} c_{ij} z_{ij} \\ & \text{subject to } \sum_{(j, i) \in A} x_{ij}^k - \sum_{(i, j) \in A} x_{ij}^k = b_i^k \quad \forall i \in N, \forall k \in K, \\ & \sum_{k \in K} x_{ij}^k - f_{ij} = 0 \quad \forall (i, j) \in A, \\ & f_{ij} \leq u_{ij} z_{ij} \quad \forall (i, j) \in A, \\ & y_{ij} \geq \frac{r_{ij} f_{ij}}{1 - f_{ij}/u_{ij}} \quad \forall (i, j) \in A, \\ & \sum_{(i, j) \in A} y_{ij} \leq \beta, \\ & z_{ij} \in \{0, 1\} \quad \forall (i, j) \in A, \\ & x \in \mathbb{R}_+^{|A| \times |K|}, y \in \mathbb{R}_+^{|A|}, f \in \mathbb{R}_+^{|A|}. \end{aligned}$$

5.3 Mean-variance optimization

A canonical optimization problem in financial engineering is to find a minimum variance portfolio that meets a minimum return requirement [26]. In the problem, there is a set N of assets available for purchase. The expected return of asset $i \in N$ is given by α_i , and the covariance of the returns between every pair of assets is given in the form a positive-definite matrix $Q \in \mathbb{R}^{n \times n}$. The canonical problem is often augmented with a number of business rules that require the introduction of binary variables in

straightforward optimization models. For example, there may be minimum (ℓ_i) and maximum (u_i) buy-in thresholds for each asset $i \in N$, resulting in the following optimization problem (MVOBI):

$$\min \left\{ x^T Q x \mid e^T x = 1, \alpha^T x \geq \rho, \ell_i z_i \leq x_i \leq u_i z_i \forall i \in N \right\},$$

where the decision variable x_i is the percentage of the portfolio invested in asset i and z_i is a binary variable indicating the purchase of asset i . Imposing a cardinality constraint on the number of different assets purchased can be achieved by adding a constraint $\sum_{i \in N} z_i \leq K$. Unfortunately, direct application of the perspective reformulation to 17 is not possible, as the objective is not a separable function of the decision variables x .

4.14 Distance Geometry Problems

Leo Liberti - Distance Geometry Problems 2014

Distance Geometry Problem (DGP). Given an integer $K > 0$ and a simple undirected graph $G = (V, E)$ whose edges are weighted by a nonnegative function $d : E \rightarrow \mathbb{R}_+$, determine whether there is a function $x : V \rightarrow \mathbb{R}^K$ such that:

$$\forall \{u, v\} \in E \quad \|x(u) - x(v)\| = d(\{u, v\}).$$

4.14.1. Molecular Distance Geometry Problem

The Discretizable Molecular Distance Geometry Problem Carlile Lavor¹, Leo Liberti², Nelson Maculan³
It is well known that the role and function of a molecule is determined by both its chemical structure (the atoms that compose it and the way they bond) and its three-dimensional structure. Supposing the chemical structure is known, finding the conformation of the atoms in \mathbb{R}^3 is usually tackled by a mixture of chemical analysis and mathematical methods. Some insight as to the molecular spatial conformation can be gained by employing Nuclear Magnetic Resonance (NMR) techniques, which are able to give a measure of the distance between (but not of the positions of) pairs of atoms closer than around 5 Å. The problem of finding the atomic positions given a subset of atomic distances can be formalized as follows.

Molecular Distance Geometry Problem (MDGP): given a weighted undirected graph $G = (V, E, d)$, is there a function $x : G \rightarrow \mathbb{R}^3$ such that $\|x(u) - x(v)\| = d(u, v)$ for each $\{u, v\} \in E$?

The atoms are represented by the set of vertices V , the atomic positions by $x(v)$, for $v \in V$, and the interatomic distance between u and v is given by $d(u, v)$, for $\{u, v\} \in E$. This problem has been shown to be NP-complete via a reduction from SUBSET-SUM [19], although the problem is solvable in linear

time when all the inter-atomic distances are known [6]. The MDGP is usually formulated as a continuous nonconvex optimization problem:

$$\min_x g(x) = \sum_{\{u,v\} \in E} (\|x(u) - x(v)\|^2 - d(u,v)^2)^2.$$

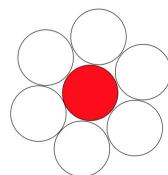
Obviously, x solves the problem if and only if $g(x) = 0$.

4.14.2. Kissing Number

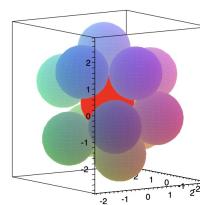
Leo Liberti - Lecture Slides

- Optimization version. Given $K \in \mathbb{N}$, determine the maximum number $\text{kn}(K)$ of unit spheres that can be placed adjacent to a central unit sphere so their interiors do not overlap
- Decision version. Given $n, K \in \mathbb{N}$, is $\text{kn}(K) \leq n$? in other words, determine whether n unit spheres can be placed adjacent to a central unit sphere so that their interiors do not overlap Funny story: Newton and Gregory went down the pub... Some examples

$n = 6, K = 2$



$n = 12, K = 3$

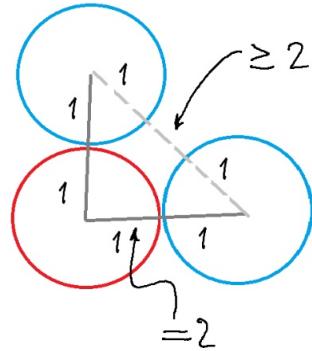


more dimensions

n	τ (lattice)	τ (nonlattice)
0	0	
1	2	
2	6	
3	12	
4	24	
5	40	
6	72	
7	126	
8	240	
9	272 (306)*	
10	338 (500)*	
11	438 (582)*	
12	756 (840)*	
13	918 (1130)*	
14	1422 (1582)*	
15	2340	
16	4320	
17	5346	
18	7398	
19	10668	
20	17400	
21	27720	
22	49896	

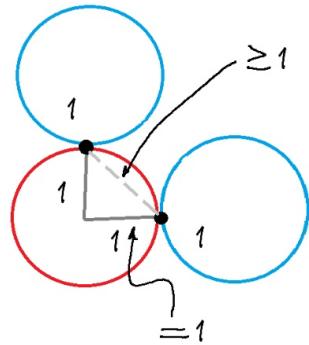
RADIUS FORMULATION Given $n, K \in \mathbb{N}$, determine whether there exist n vectors $x_1, \dots, x_n \in \mathbb{R}^K$ such that:

$$\begin{aligned}\forall i \leq n \quad & \|x_i\|_2^2 = 4 \\ \forall i < j \leq n \quad & \|x_i - x_j\|_2^2 \geq 4\end{aligned}$$



CONTACT POINT FORMULATION Given $n, K \in \mathbb{N}$, determine whether there exist n vectors $x_1, \dots, x_n \in \mathbb{R}^K$ such that:

$$\begin{aligned}\forall i \leq n \quad & \|x_i\|_2^2 = 1 \\ \forall i < j \leq n \quad & \|x_i - x_j\|_2^2 \geq 1\end{aligned}$$



SPHERICAL CODES

- $S^{K-1} \subset \mathbb{R}^K$ unit sphere centered at origin
- K-dimensional spherical z-code:
- (finite) subset $\mathcal{C} \subset S^{K-1}$
- $\forall x \neq y \in \mathcal{C} \quad x \cdot y \leq z$
- non-overlapping interiors:

$$\begin{aligned}
& \forall i < j \quad \|x_i - x_j\|_2^2 \geq 1 \\
\Leftrightarrow & \quad \|x_i\|_2^2 + \|x_j\|_2^2 - 2x_i \cdot x_j \geq 1 \\
\Leftrightarrow & \quad 1 + 1 - 2x_i \cdot x_j \geq 1 \\
\Leftrightarrow & \quad 2x_i \cdot x_j \leq 1 \\
\Leftrightarrow & \quad x_i \cdot x_j \leq \frac{1}{2} = \cos\left(\frac{\pi}{3}\right) = z
\end{aligned}$$

MINLP formulation Maculan, Michelon, Smith 1995 Parameters:

- K : space dimension
- n : upper bound to $\text{kn}(K)$ Variables:
- $x_i \in \mathbb{R}^K$: center of i -th vector
- $\alpha_i = 1$ iff vector i in configuration

$$\left. \begin{array}{lll}
\max & \sum_{i=1}^n \alpha_i & \\
\forall i \leq n & \|x_i\|_2^2 = \alpha_i & \\
\forall i < j \leq n & \|x_i - x_j\|_2^2 \geq \alpha_i \alpha_j & \\
\forall i \leq n & x_i \in [-1, 1]^K & \\
\forall i \leq n & \alpha_i \in \{0, 1\} &
\end{array} \right\}$$

4.15 Obnoxious Facility Location Problem

Extremely non-convex optimization problems: the case of the multiple obnoxious facilities location - Paweł Kalczynski & Zvi Drezner 2021

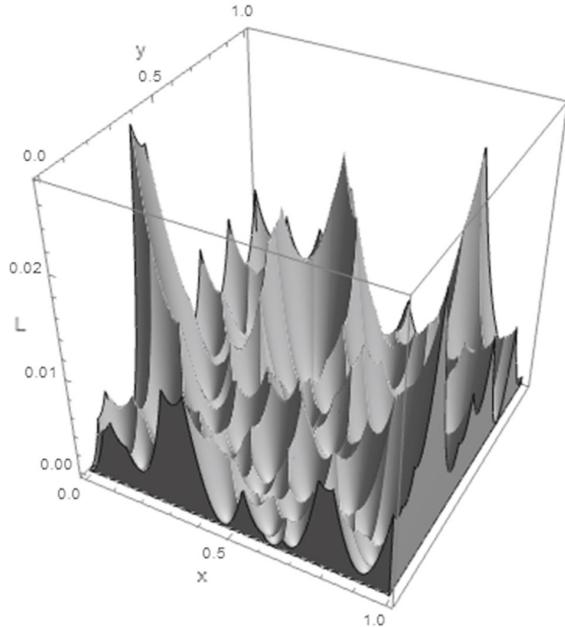
Let $A_i = (a_i, b_i)$ for $i = 1, \dots, n$ be the locations of communities, and $X_j = (x_j, y_j)$ for $j = 1, \dots, p$ be the

unknown locations of the p facilities. The non-linear programming formulation is:

$$\begin{aligned} \max \quad & L \\ \text{subject to:} \quad & (x_j - a_i)^2 + (y_j - b_i)^2 \geq L \quad \text{for } i = 1, \dots, n; j = 1, \dots, p \\ & (x_i - x_j)^2 + (y_i - y_j)^2 \geq D^2 \quad \text{for } 1 \leq i < j \leq p. \end{aligned}$$

In addition we need constraints that restrict the facilities' locations to a convex polygon or any finite region. Otherwise, the solution is to locate the facilities "at infinity". In the computational experiments we tested problems from Drezner et al. [11] that have a square feasible region.

This problem is extremely non-convex. The surface plot for the $n = 100$ instances (tested in Drezner et al. [11]) is plotted in Fig. 1. There are 202 "hilltops" (for $n = 1000$ there are 2002 hilltops) and the solution for the $p = 1$ instance is on the top of the highest hill. The hilltops are located at Voronoi points [1,19,21,22].



- The obnoxious facility location problem was initiated in the 1970's [2,3,7,13,17,18, 24], and intensively analyzed in the literature (for reviews see [6,11]). Suppose that n communities are located in an area. One needs to locate $p \geq 1$ obnoxious facilities in that area. Examples of obnoxious facilities include airports, industrial facilities, landfills, prisons, and others affecting residents living nearby.
- Most of the papers on the subject are modeled in discrete space, i.e., there is a limited number of potential locations for the facilities (for example in a network environment).
- However, in most of these applications, nuisance propagates "by air". Therefore, the use of planar Euclidean distances is recommended.
- Locations of facilities are usually far from the communities in open spaces, and thus not limited to a list of locations, so location anywhere in the area is useful as well.
- Alternative problem: multiple obnoxious facility location problem recently investigated in Drezner et al. [11]. The objective is to maximize the shortest distance between facilities and communities subject to a minimum distance D between facilities.

4.16 AC Optimal Power Flow Problem

THE ALTERNATING CURRENT OPTIMAL POWER FLOW (AC/OPF) PROBLEM The AC/OPF problem is a cornerstone in the field of power systems engineering, representing a critical application of mixed-integer nonlinear programming (MINLP). It involves the optimization of certain objectives, such as minimizing the cost of generation or maximizing system efficiency, subject to a set of physical and operational constraints.

DESCRIPTION OF THE PROBLEM The primary goal of the AC/OPF problem is to determine the optimal set points for the control variables of a power system, such as generator outputs, voltage magnitudes, and transformer tap settings, while satisfying a set of constraints. These constraints typically include power balance equations, transmission line limits, and voltage magnitude bounds, ensuring the secure and stable operation of the power system.

Understanding the AC/OPF Problem

Basic Concepts

- **Power System:** A network of electrical components used to supply, transmit, and use electric power, including power generation units, transformers, transmission lines, and consumers.
- **Bus:** A node in the power system where power is either generated, transformed, or consumed, acting as a junction for power lines.
- **Transmission Line:** Cables or wires through which electricity travels from one point to another.

Core Elements of AC/OPF

- **Active and Reactive Power:** Components of electricity; active power is used to run appliances, while reactive power maintains voltage levels.
- **Voltage and Phase Angle:** Voltage is the force pushing electric current, and phase angle is the timing difference in electricity waves across the system.
- **Generation and Demand:** Generation is the production of power, while demand is the power needed by consumers.

Problem Description

The AC/OPF problem involves optimizing the operation of a power system to meet consumer demands efficiently, minimize costs, and ensure system stability within safe operating limits.

Main Goals

1. **Minimizing Costs:** Finding cost-effective ways to produce the required electricity.
2. **Meeting Demand:** Ensuring the power generated meets the exact needs of consumers.
3. **Operating Within Limits:** Respecting maximum power capacities and safe voltage levels.
4. **Ensuring Stability:** Maintaining system reliability even with sudden changes in demand or other disturbances.

Mathematical Formulation

The AC/OPF problem is formulated mathematically with equations representing power flows, generation limits, and other constraints, aiming to optimize variables such as power generation and voltage levels while minimizing a cost function.

VARIABLES

- P_{G_i} : Active power generation at bus i .
- Q_{G_i} : Reactive power generation at bus i .
- V_i : Voltage magnitude at bus i .
- θ_i : Phase angle at bus i .

PARAMETERS

- P_{D_i} : Active power demand at bus i .
- Q_{D_i} : Reactive power demand at bus i .
- G_{ij} : Conductance of the transmission line between buses i and j .
- B_{ij} : Susceptance of the transmission line between buses i and j .
- S_{ij}^{\max} : Maximum apparent power flow limit of the transmission line between buses i and j .
- $P_{G_i}^{\min}, P_{G_i}^{\max}$: Minimum and maximum limits for active power generation at bus i .
- $Q_{G_i}^{\min}, Q_{G_i}^{\max}$: Minimum and maximum limits for reactive power generation at bus i .
- V_i^{\min}, V_i^{\max} : Minimum and maximum permissible voltage magnitudes at bus i .
- $\Delta\theta_{ij}^{\max}$: Maximum permissible phase angle difference between buses i and j .
- $C_i(\cdot)$: Cost function of generation at bus i .

MATHEMATICAL FORMULATION Consider a power system with a set of buses N and a set of transmission lines E . Let P_{G_i} and Q_{G_i} denote the active and reactive power generation at bus i , and P_{D_i} and Q_{D_i} denote the active and reactive power demand. The AC/OPF problem can be formulated as follows:

Objective Function:

$$\min \sum_{i \in N} C_i(P_{G_i}) \quad (4.1)$$

where $C_i(\cdot)$ represents the cost function of generation at bus i .

Subject to:

1. Power Balance Constraints:

$$P_{G_i} - P_{D_i} = \sum_{j \in N} (G_{ij}V_iV_j \cos(\theta_i - \theta_j) + B_{ij}V_iV_j \sin(\theta_i - \theta_j)), \forall i \in N \quad (4.2)$$

$$Q_{G_i} - Q_{D_i} = \sum_{j \in N} (G_{ij}V_iV_j \sin(\theta_i - \theta_j) - B_{ij}V_iV_j \cos(\theta_i - \theta_j)), \forall i \in N \quad (4.3)$$

where V_i and θ_i are the voltage magnitude and phase angle at bus i , and G_{ij} and B_{ij} are the conductance and susceptance of the line between buses i and j .

2. Line Flow Constraints:

$$(P_{ij}^2 + Q_{ij}^2) \leq (S_{ij}^{\max})^2, \forall (i, j) \in E \quad (4.4)$$

where P_{ij} and Q_{ij} are the active and reactive power flows on the line from bus i to bus j , and S_{ij}^{\max} is the maximum apparent power flow limit of the line.

3. Generator Limits:

$$P_{G_i}^{\min} \leq P_{G_i} \leq P_{G_i}^{\max}, \forall i \in N \quad (4.5)$$

$$Q_{G_i}^{\min} \leq Q_{G_i} \leq Q_{G_i}^{\max}, \forall i \in N \quad (4.6)$$

where $P_{G_i}^{\min}$, $P_{G_i}^{\max}$, $Q_{G_i}^{\min}$, and $Q_{G_i}^{\max}$ are the minimum and maximum limits for active and reactive power generation at bus i

1. Voltage Magnitude Limits:

$$V_i^{\min} \leq V_i \leq V_i^{\max}, \forall i \in N \quad (4.7)$$

where V_i^{\min} and V_i^{\max} are the minimum and maximum permissible voltage magnitudes at bus i .

2. Phase Angle Difference Constraints:

$$|\theta_i - \theta_j| \leq \Delta\theta_{ij}^{\max}, \forall (i, j) \in E \quad (4.8)$$

where $\Delta\theta_{ij}^{\max}$ is the maximum permissible phase angle difference between buses i and j .

CHALLENGES AND COMPLEXITY The AC/OPF problem is inherently nonlinear and nonconvex due to the trigonometric relationships in the power flow equations. This complexity makes it a challenging problem to solve, particularly for large-scale power systems. Advanced optimization techniques, such as interior-point methods, branch-and-bound algorithms, and heuristic approaches, are often employed to find feasible and near-optimal solutions.

APPLICATIONS The AC/OPF problem is critical in the operation and planning of power systems. It is used for determining the most economical generation dispatch, ensuring system reliability and stability, and planning for future expansions and upgrades. The solutions of the AC/OPF problem provide valuable insights into the efficient and secure operation of power systems.

Resources

Dan Bienstock - Gurobi Talk Part 1

Dan Bienstock - Gurobi Talk Part 2

Electrical Transmission System Cascades and Vulnerability: An Operations Research Viewpoint,
Dan Bienstock, ISBN 978-1-611974-15-7. SIAM-MOS Series on Optimization (2015).

4.17 Multi Player Nash Equilibrium

Multilinear Formulations for Computing a Nash Equilibrium of Multi-Player Games - Miriam Fischer, Akshay Gupte ORCID-Logo **fischer_et_al:LIPIcs.SEA.2023.12**

The multi-player multilinear formulation is an extension of a bilinear formulation for bimatrix games [9]. To motivate the multilinear formulation, we shortly recall the bilinear program that is equivalent to finding a Nash equilibrium in a bimatrix game. To do so, we introduce some notation. Let $A, B \in \mathbb{R}^{m \times n}$ be the payoff matrix of player 1 and player 2, with m pure strategies of player 1 and n pure strategies of player 2. Let $x \in \mathbb{R}^m$ with $x \geq 0$ and $\sum_{i=1}^m x_i = 1$ be a (possibly mixed) strategy of player 1, with x_s being the probability placed on pure strategy s . Let $y \in \mathbb{R}^n$ with $y \geq 0$ and $\sum_{j=1}^n y_j = 1$ be a (possibly mixed) strategy

of player 2. Let 1_n and 1_m denote vectors of all ones of dimension n and m . Any globally optimal solution (x, y, p, q) to the bilinear optimization problem in BLP is equivalent to a Nash equilibrium in a bimatrix game.

$$\begin{aligned} & \max_{x,y,p,q} x^\top A y + x^\top B y - p - q \\ \text{s.t. } & Ay \leq p 1_m, \quad B^\top x \leq q 1_n \\ & \sum_{i=1}^m x_i = 1, \quad \sum_{j=1}^n y_j = 1, \quad x, y \geq 0. \end{aligned}$$

Definition 4.1: Mixed Nash Equilibrium

Let $\Gamma = \langle \{1, \dots, n\}, (S_i), (A_i) \rangle$ be a game with $n, S_i, A_i, \mathbf{x}^i$ defined as above. Let $\mathbf{x}^i \geq 0$ with $\sum_{s \in S_i} x_s^i = 1$ be a mixed strategy of player i . Then, $\mathbf{x}^* = (\mathbf{x}^{*1}, \dots, \mathbf{x}^{*n})$ with $\mathbf{x}^{*i} \geq 0$ and $\sum_{s \in S_i} x_s^{*i} = 1$ for all players i is a (mixed) Nash equilibrium if for all players i and every mixed strategy \mathbf{x}^i , we have $\mathbb{E}[A_i[\mathbf{x}^*]] \geq \mathbb{E}[A_i[\mathbf{x}^i, \mathbf{x}^{*-i}]]$.

We now present the multilinear optimization formulation (MLP1)

$$\begin{aligned} & \max_{\mathbf{x}, \mathbf{p}} \sum_{i=1}^n \left(\sum_{\substack{(s, \hat{s}) \\ \in S_i \times S_{-i}}} A_i[s, \hat{s}] x_s^i \prod_{s_j \in \hat{s}} x_{s_j}^j \right) - \sum_{i=1}^n p^i \\ & \sum_{\substack{\hat{s} \in S_{-i} \\ s \in S_i}} A_i[s, \hat{s}] \prod_{s_j \in \hat{s}} x_{s_j}^j \leq p^i \quad \forall i \in [n], s \in S_i \\ & \sum_{s \in S_i} x_s^i = 1 \quad \forall i \in [n] \\ & 0 \leq x_s^i \leq 1 \quad \forall i \in [n], s \in S_i \end{aligned}$$

Theorem 4.2

A (mixed) strategy $(\mathbf{x}^1, \dots, \mathbf{x}^n)$ is a (mixed) Nash equilibrium of the n -player game (A_1, \dots, A_n) if and only if there exist numbers p^1, \dots, p^n such that $(\mathbf{x}^1, \dots, \mathbf{x}^n, p^1, \dots, p^n)$ is an optimal solution to the problem in MLP1.

4.18 Pricing Problem

Outer Approximation for Integer Nonlinear Programs via Decision Diagrams Danial Davarnia and Willem-Jan van Hoeve

In the context of marketing applications, this section focuses on a pricing problem where a firm aims to establish the prices of several new products entering a competitive market. A model is developed employing various strategies to address this problem.

Strategies and Model Formulation

Psychological Pricing Strategy

This strategy exploits the psychological impact of certain prices on consumer behavior. It is observed that a significant proportion of product prices in advertisements end with 0, 5, or 9. This leads to the discretization of price variables to integers within a certain range for precise adjustment. The price of product i is represented as x_i , within the integer range $[l_i, u_i] \cap \mathbb{Z}$.

Demand Function

The model incorporates a demand function reflective of the fairness effect, where customers are more sensitive to prices in a competitive market. The demand function is modeled as an exponential decay function $e^{-x_i^{k_i}}$, with k_i indicating the price sensitivity of product i .

Penetration Pricing Strategy

Aiming for market entry, this strategy involves setting initial low prices to attract customers and gain market share, followed by price adjustments for profit maximization. The objective function is to minimize a weighted sum of product prices, $\min \sum_{i=1}^n c_i x_i$, while ensuring a minimum profit margin through a profit satisfaction constraint.

Model Definition

The integrated strategies lead to the following pricing model:

$$\begin{aligned} & \min \sum_{i=1}^n c_i x_i \\ \text{s.t. } & \sum_{i=1}^n a_i^j x_i e^{-x_i^{k_i^j}} \geq b_j, \quad \forall j \in J \\ & x \in [l, u] \cap \mathbb{Z}^n \end{aligned}$$

Parameters and Variables

Parameters:

- l_i, u_i : Lower and upper bounds for the price of product i .
- c_i : Weight associated with the price of product i in the objective function.
- a_i^j : Nonnegative weight for product i in constraint j .
- k_i^j : Price sensitivity parameter for product i in constraint j .
- b_j : Profit margin for constraint j .
- J : Set of all constraints, representing different geographical districts or market segments.

Variables:

- x_i : Price of product i , which is an integer within the range $[l_i, u_i]$.

The model integrates psychological influences, market dynamics, and profit optimization in its approach to solving the pricing problem.

References

- Bienstock, D. (2007). "Histogram models for robust portfolio optimization". In: *Journal of Computational Finance* 11, pp. 1–64.
- Boddiford, Ashley N., Daniel E. Kaufman, Daphne E. Skipper, and Nelson A. Uhan (2023). "Approximating a linear multiplicative objective in watershed management optimization". In: *European Journal of Operational Research* 305.2, pp. 547–561. ISSN: 0377-2217. DOI: <https://doi.org/10.1016/j.ejor.2022.06.006>.

- Bussieck, M.R., A.S. Drud, and A. Meeraus (2003). “MINLPLib - A collection of test models for mixed-integer nonlinear programming”. In: *INFORMS Journal on Computing* 15, pp. 114–119.
- Caprara, A. and M. Monaci (2009). “Bidimensional packing by bilinear programming”. In: *Mathematical Programming* 118, pp. 75–108.
- D’Ambrosio, C., A. Frangioni, L. Liberti, and A. Lodi (2010). “Experiments with a feasibility pump approach for nonconvex MINLPs”. In: *Proceedings of the 9th Symposium on Experimental Algorithms (SEA 2010)*. Ed. by P. Festa. Vol. 6049. Lecture Notes in Computer Science. Springer, pp. 350–360.
- Günlük, Oktay and Jeff Linderoth (2010). “Perspective reformulations of mixed integer nonlinear programs with indicator variables”. In: *Mathematical Programming* 124.1, pp. 183–205. URL: <https://link.springer.com/article/10.1007/s10107-010-0360-z>.
- (2012). “Perspective Reformulation and Applications”. In: *Mixed Integer Nonlinear Programming*. Springer, pp. 61–89. URL: https://link.springer.com/chapter/10.1007/978-1-4614-1927-3_3.
- Harjunkoski, I., T. Westerlund, R. Pörn, and H. Skrifvars (1998). “Different transformations for solving non-convex trim loss problems by MINLP”. In: *European Journal of Operational Research* 105, pp. 594–603.
- Karuppiah, R. and I.E. Grossmann (2006). “Global optimization for the synthesis of integrated water systems in chemical processes”. In: *Computers and Chemical Engineering* 30, pp. 650–673.
- Lee, Mengyuan, Guanding Yu, and Geoffrey Ye Li (2020). “Learning to Branch: Accelerating Resource Allocation in Wireless Networks”. In: *IEEE Transactions on Vehicular Technology* 69.1, pp. 958–970. DOI: [10.1109/TVT.2019.2953724](https://doi.org/10.1109/TVT.2019.2953724).
- Meyer, C.A. and C.A. Floudas (2006). “Global optimization of a combinatorially complex generalized pooling problem”. In: *AICHE journal* 52, pp. 1027–1037.
- Misener, R. and C.A. Floudas (2009). “Advances for the pooling problem: Modeling, global optimization, and computational studies”. In: *Appl. Comput. Math* 8, pp. 3–22.
- Quesada, I. and I.E. Grossmann (1995). “Global optimization of bilinear process networks with multicomponent flows”. In: *Computers and Chemical Engineering* 19, pp. 1219–1242.
- Rinaldi, G., U. Voigt, and G.J. Woeginger (2002). “The mathematics of playing golf, or: a new class of difficult non-linear mixed integer programs”. In: *Mathematical Programming* 93, pp. 77–86.
- Sherali, H.D. and A. Alameddine (1992). “A new reformulation-linearization technique for bilinear programming problems”. In: *Journal of Global Optimization* 2, pp. 379–410.

5. Software

5.1 List of Leading MINLP Software

1. **BARON** - Branch and Reduce Optimization Navigator.
2. **BONMIN** - Basic Open-source Nonlinear Mixed INteger programming.
3. **COUENNE** - Convex Over and Under ENvelopes for Nonlinear Estimation.
4. **KNITRO** - An integrated package for nonlinear optimization.
5. **SCIP** - Solving Constraint Integer Programs.
6. **GAMS/CPLEX** - General Algebraic Modeling System with CPLEX solver.
7. **GAMS/GUROBI** - General Algebraic Modeling System with GUROBI solver.
8. **AIMMS** - Advanced Interactive Multidimensional Modeling Software.
9. **AMPL** - A Mathematical Programming Language.
10. **MINOTAUR** - Mixed-Integer Nonlinear Optimization Toolkit for Advanced Research.

Software	Types of Objectives	Types of Constraints
BARON	Nonlinear, Quadratic, Linear	Nonlinear, Quadratic, Linear
BONMIN	Linear, Quadratic, Non-linear	Linear, Quadratic, Non-linear
COUENNE	Nonlinear, Quadratic, Linear	Nonlinear, Quadratic, Linear
KNITRO	Linear, Nonlinear, Quadratic	Linear, Nonlinear, Quadratic
SCIP	Linear, Nonlinear, Quadratic, Pseudo-Boolean	Linear, Nonlinear, Quadratic, Logic
GAMS/CPLEX	Linear, Quadratic, Non-linear	Linear, Quadratic, Non-linear
GAMS/GUROBI	Linear, Quadratic	Linear, Quadratic

Software	Types of Objectives	Types of Constraints
AIMMS	Linear, Quadratic	Nonlinear, Quadratic
AMPL	Linear, Quadratic	Nonlinear, Quadratic
MINOTAUR	Nonlinear, Quadratic	Linear, Quadratic

List of Recent Julia Codes for MINLP and Optimization

1. **JuMP** - A modeling language for mathematical optimization.
2. **Pluto.jl** - A reactive notebook for Julia.
3. **Optim.jl** - Non-linear optimization tools.
4. **Convex.jl** - A package for convex optimization.
5. **JuliaOpt** - Julia for mathematical optimization.
6. **DiffOpt.jl** - Differentiable convex optimization.
7. **Juno.jl** - An integrated development environment (IDE) for Julia.
8. **Gurobi.jl** - Julia interface for Gurobi Optimizer.
9. **Ipopt.jl** - Interior point optimizer for large-scale nonlinear optimization.
10. **COBRA.jl** - COnstraint-Based Reconstruction and Analysis for systems biology.
11. **Juniper.jl** - An open-source solver for mixed-integer nonlinear programs.

MINDTpy

MindtPy Solver

The Mixed-Integer Nonlinear Decomposition Toolbox in Pyomo (MindtPy) solver allows users to solve Mixed-Integer Nonlinear Programs (MINLP) using decomposition algorithms. These decomposition algorithms usually rely on the solution of Mixed-Integer Linear Programs (MILP) and Nonlinear Programs (NLP).

The following algorithms are currently available in MindtPy:

- Outer-Approximation (OA) [Duran & Grossmann, 1986]
- LP/NLP based Branch-and-Bound (LP/NLP BB) [Quesada & Grossmann, 1992]
- Extended Cutting Plane (ECP) [Westerlund & Petterson, 1995]
- Global Outer-Approximation (GOA) [Kesavan & Allgor, 2004, MC++]
- Regularized Outer-Approximation (ROA) [Bernal & Peng, 2021, Kronqvist & Bernal, 2018]
- Feasibility Pump (FP) [Bernal & Vigerske, 2019, Bonami & Cornuéjols, 2009]

Usage and early implementation details for MindtPy can be found in the PSE 2018 paper Bernal et al., (ref, preprint). This solver implementation has been developed by David Bernal and Zedong Peng as part of research efforts at the Bernal Research Group and the Grossmann Research Group at Purdue University and Carnegie Mellon University.

5.2 List of (some) MINLP Researchers

This list is not comprehensive, but meant to give as a good starting point of names to look into for research in this area.

Contribution of many people

- Warren Adams
- Claire S. Adjiman
- Shabbir Ahmed
- Kurt Anstreicher
- Gennadiy Averkov
- Harold P. Benson
- Daniel Bienstock
- Natashia Boland
- Pierre Bonami
- Samuel Burer
- Kwanghun Chung
- Yves Crama
- Danial Davarnia
- Alberto Del Pia
- Marco Duran
- Hongbo Dong
- Christodoulos A. Floudas
- Ignacio Grossmann
- Oktay Günlük
- Akshay Gupte
- Thomas Kalinowski

- Fatma Kilinc-Karzan
- Aida Khajavirad
- Burak Kocuk
- Jan Kronqvist
- Jon Lee
- Adam Letchford
- Jeff Linderoth
- Leo Liberti
- Jim Luedtke
- Marco Locatelli
- Andrea Lodi
- Alex Martin
- Clifford A. Meyer
- Garth P. McCormick
- Ruth Misener
- Gonzalo Munoz
- Mahdi Namazifar
- Jean-Philippe P. Richard
- Fabian Rigterink
- Anatoliy D. Rikun
- Nick Sahinidis
- Hanif Sherali
- Lars Schewe
- Felipe Serrano
- Suvrajeet Sen
- Emily Speakman
- Fabio Tardella
- Mohit Tawarmalani
- Hoáng Tuy

- Juan Pablo Vielma
- Alex Wang

Part II

MINLP Theory and Algorithms

5.3 AC Optimal Power Flow Problem

THE ALTERNATING CURRENT OPTIMAL POWER FLOW (AC/OPF) PROBLEM The AC/OPF problem is a cornerstone in the field of power systems engineering, representing a critical application of mixed-integer nonlinear programming (MINLP). It involves the optimization of certain objectives, such as minimizing the cost of generation or maximizing system efficiency, subject to a set of physical and operational constraints.

DESCRIPTION OF THE PROBLEM The primary goal of the AC/OPF problem is to determine the optimal set points for the control variables of a power system, such as generator outputs, voltage magnitudes, and transformer tap settings, while satisfying a set of constraints. These constraints typically include power balance equations, transmission line limits, and voltage magnitude bounds, ensuring the secure and stable operation of the power system.

Understanding the AC/OPF Problem

Basic Concepts

- **Power System:** A network of electrical components used to supply, transmit, and use electric power, including power generation units, transformers, transmission lines, and consumers.
- **Bus:** A node in the power system where power is either generated, transformed, or consumed, acting as a junction for power lines.
- **Transmission Line:** Cables or wires through which electricity travels from one point to another.

Core Elements of AC/OPF

- **Active and Reactive Power:** Components of electricity; active power is used to run appliances, while reactive power maintains voltage levels.
- **Voltage and Phase Angle:** Voltage is the force pushing electric current, and phase angle is the timing difference in electricity waves across the system.
- **Generation and Demand:** Generation is the production of power, while demand is the power needed by consumers.

Problem Description

The AC/OPF problem involves optimizing the operation of a power system to meet consumer demands efficiently, minimize costs, and ensure system stability within safe operating limits.

Main Goals

1. **Minimizing Costs:** Finding cost-effective ways to produce the required electricity.
2. **Meeting Demand:** Ensuring the power generated meets the exact needs of consumers.
3. **Operating Within Limits:** Respecting maximum power capacities and safe voltage levels.
4. **Ensuring Stability:** Maintaining system reliability even with sudden changes in demand or other disturbances.

Mathematical Formulation

The AC/OPF problem is formulated mathematically with equations representing power flows, generation limits, and other constraints, aiming to optimize variables such as power generation and voltage levels while minimizing a cost function.

VARIABLES

- P_{G_i} : Active power generation at bus i .
- Q_{G_i} : Reactive power generation at bus i .
- V_i : Voltage magnitude at bus i .
- θ_i : Phase angle at bus i .

PARAMETERS

- P_{D_i} : Active power demand at bus i .
- Q_{D_i} : Reactive power demand at bus i .
- G_{ij} : Conductance of the transmission line between buses i and j .
- B_{ij} : Susceptance of the transmission line between buses i and j .
- S_{ij}^{\max} : Maximum apparent power flow limit of the transmission line between buses i and j .
- $P_{G_i}^{\min}, P_{G_i}^{\max}$: Minimum and maximum limits for active power generation at bus i .
- $Q_{G_i}^{\min}, Q_{G_i}^{\max}$: Minimum and maximum limits for reactive power generation at bus i .
- V_i^{\min}, V_i^{\max} : Minimum and maximum permissible voltage magnitudes at bus i .

- $\Delta\theta_{ij}^{\max}$: Maximum permissible phase angle difference between buses i and j .
- $C_i(\cdot)$: Cost function of generation at bus i .

MATHEMATICAL FORMULATION Consider a power system with a set of buses N and a set of transmission lines E . Let P_{G_i} and Q_{G_i} denote the active and reactive power generation at bus i , and P_{D_i} and Q_{D_i} denote the active and reactive power demand. The AC/OPF problem can be formulated as follows:

Objective Function:

$$\min \sum_{i \in N} C_i(P_{G_i}) \quad (5.1)$$

where $C_i(\cdot)$ represents the cost function of generation at bus i .

Subject to:

1. Power Balance Constraints:

$$P_{G_i} - P_{D_i} = \sum_{j \in N} (G_{ij}V_iV_j \cos(\theta_i - \theta_j) + B_{ij}V_iV_j \sin(\theta_i - \theta_j)), \forall i \in N \quad (5.2)$$

$$Q_{G_i} - Q_{D_i} = \sum_{j \in N} (G_{ij}V_iV_j \sin(\theta_i - \theta_j) - B_{ij}V_iV_j \cos(\theta_i - \theta_j)), \forall i \in N \quad (5.3)$$

where V_i and θ_i are the voltage magnitude and phase angle at bus i , and G_{ij} and B_{ij} are the conductance and susceptance of the line between buses i and j .

2. Line Flow Constraints:

$$(P_{ij}^2 + Q_{ij}^2) \leq (S_{ij}^{\max})^2, \forall (i, j) \in E \quad (5.4)$$

where P_{ij} and Q_{ij} are the active and reactive power flows on the line from bus i to bus j , and S_{ij}^{\max} is the maximum apparent power flow limit of the line.

3. Generator Limits:

$$P_{G_i}^{\min} \leq P_{G_i} \leq P_{G_i}^{\max}, \forall i \in N \quad (5.5)$$

$$Q_{G_i}^{\min} \leq Q_{G_i} \leq Q_{G_i}^{\max}, \forall i \in N \quad (5.6)$$

where $P_{G_i}^{\min}$, $P_{G_i}^{\max}$, $Q_{G_i}^{\min}$, and $Q_{G_i}^{\max}$ are the minimum and maximum limits for active and reactive power generation at bus i

1. Voltage Magnitude Limits:

$$V_i^{\min} \leq V_i \leq V_i^{\max}, \forall i \in N \quad (5.7)$$

where V_i^{\min} and V_i^{\max} are the minimum and maximum permissible voltage magnitudes at bus i .

2. Phase Angle Difference Constraints:

$$|\theta_i - \theta_j| \leq \Delta\theta_{ij}^{\max}, \forall (i, j) \in E \quad (5.8)$$

where $\Delta\theta_{ij}^{\max}$ is the maximum permissible phase angle difference between buses i and j .

CHALLENGES AND COMPLEXITY The AC/OPF problem is inherently nonlinear and nonconvex due to the trigonometric relationships in the power flow equations. This complexity makes it a challenging problem to solve, particularly for large-scale power systems. Advanced optimization techniques, such as interior-point methods, branch-and-bound algorithms, and heuristic approaches, are often employed to find feasible and near-optimal solutions.

APPLICATIONS The AC/OPF problem is critical in the operation and planning of power systems. It is used for determining the most economical generation dispatch, ensuring system reliability and stability, and planning for future expansions and upgrades. The solutions of the AC/OPF problem provide valuable insights into the efficient and secure operation of power systems.

Resources

[Dan Bienstock - Gurobi Talk Part 1](#)

[Dan Bienstock - Gurobi Talk Part 2](#)

Electrical Transmission System Cascades and Vulnerability: An Operations Research Viewpoint,
Dan Bienstock, ISBN 978- 1-611974-15-7. SIAM-MOS Series on Optimization (2015).

5.4 Distance Geometry Problems

Leo Liberti - Distance Geometry Problems 2014

Distance Geometry Problem (DGP). Given an integer $K > 0$ and a simple undirected graph $G = (V, E)$ whose edges are weighted by a nonnegative function $d : E \rightarrow \mathbb{R}_+$, determine whether there is a function $x : V \rightarrow \mathbb{R}^K$ such that:

$$\forall \{u, v\} \in E \quad \|x(u) - x(v)\| = d(\{u, v\}).$$

5.4.1. Molecular Distance Geometry Problem

The Discretizable Molecular Distance Geometry Problem Carlile Lavor¹, Leo Liberti², Nelson Maculan³ It is well known that the role and function of a molecule is determined by both its chemical structure (the atoms that compose it and the way they bond) and its three-dimensional structure. Supposing the chemical structure is known, finding the conformation of the atoms in \mathbb{R}^3 is usually tackled by a mixture of chemical analysis and mathematical methods. Some insight as to the molecular spatial conformation can be gained by employing Nuclear Magnetic Resonance (NMR) techniques, which are able to give a measure of the distance between (but not of the positions of) pairs of atoms closer than around 5 Å. The problem of finding the atomic positions given a subset of atomic distances can be formalized as follows.

Molecular Distance Geometry Problem (MDGP): given a weighted undirected graph $G = (V, E, d)$, is there a function $x : G \rightarrow \mathbb{R}^3$ such that $\|x(u) - x(v)\| = d(u, v)$ for each $\{u, v\} \in E$?

The atoms are represented by the set of vertices V , the atomic positions by $x(v)$, for $v \in V$, and the interatomic distance between u and v is given by $d(u, v)$, for $\{u, v\} \in E$. This problem has been shown to be NP-complete via a reduction from SUBSET-SUM [19], although the problem is solvable in linear

time when all the inter-atomic distances are known [6]. The MDGP is usually formulated as a continuous nonconvex optimization problem:

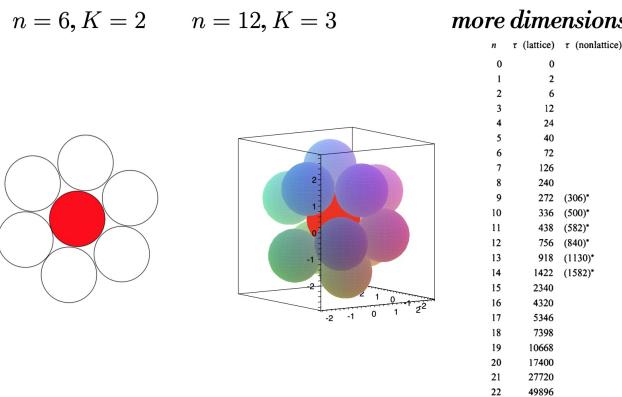
$$\min_x g(x) = \sum_{\{u, v\} \in E} (\|x(u) - x(v)\|^2 - d(u, v)^2)^2.$$

Obviously, x solves the problem if and only if $g(x) = 0$.

5.4.2. Kissing Number

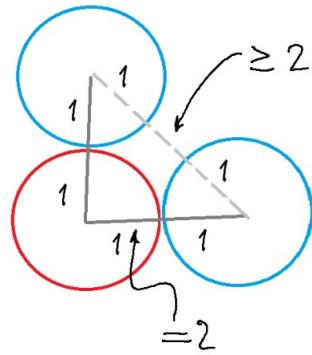
Leo Liberti - Lecture Slides

- Optimization version. Given $K \in \mathbb{N}$, determine the maximum number $kn(K)$ of unit spheres that can be placed adjacent to a central unit sphere so their interiors do not overlap
- Decision version. Given $n, K \in \mathbb{N}$, is $kn(K) \leq n$? in other words, determine whether n unit spheres can be placed adjacent to a central unit sphere so that their interiors do not overlap Funny story: Newton and Gregory went down the pub... Some examples



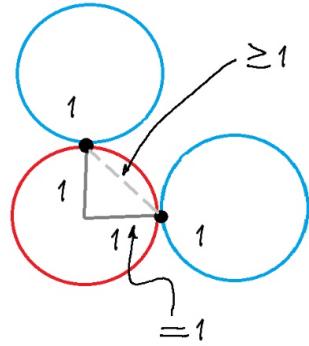
RADIUS FORMULATION Given $n, K \in \mathbb{N}$, determine whether there exist n vectors $x_1, \dots, x_n \in \mathbb{R}^K$ such that:

$$\begin{aligned} \forall i \leq n \quad \|x_i\|_2^2 &= 4 \\ \forall i < j \leq n \quad \|x_i - x_j\|_2^2 &\geq 4 \end{aligned}$$



CONTACT POINT FORMULATION Given $n, K \in \mathbb{N}$, determine whether there exist n vectors $x_1, \dots, x_n \in \mathbb{R}^K$ such that:

$$\begin{aligned}\forall i \leq n \quad \|x_i\|_2^2 &= 1 \\ \forall i < j \leq n \quad \|x_i - x_j\|_2^2 &\geq 1\end{aligned}$$



SPHERICAL CODES

- $S^{K-1} \subset \mathbb{R}^K$ unit sphere centered at origin
- K -dimensional spherical z-code:
- (finite) subset $\mathcal{C} \subset S^{K-1}$
- $\forall x \neq y \in \mathcal{C} \quad x \cdot y \leq z$
- non-overlapping interiors:

$$\begin{aligned}
& \forall i < j \|x_i - x_j\|_2^2 \geq 1 \\
\Leftrightarrow & \|x_i\|_2^2 + \|x_j\|_2^2 - 2x_i \cdot x_j \geq 1 \\
\Leftrightarrow & 1 + 1 - 2x_i \cdot x_j \geq 1 \\
\Leftrightarrow & 2x_i \cdot x_j \leq 1 \\
\Leftrightarrow & x_i \cdot x_j \leq \frac{1}{2} = \cos\left(\frac{\pi}{3}\right) = z
\end{aligned}$$

MINLP formulation Maculan, Michelon, Smith 1995 Parameters:

- K : space dimension
- n : upper bound to $\text{kn}(K)$ Variables:
- $x_i \in \mathbb{R}^K$: center of i -th vector
- $\alpha_i = 1$ iff vector i in configuration

$$\left. \begin{array}{lll}
\max & \sum_{i=1}^n \alpha_i & \\
\forall i \leq n & \|x_i\|_2^2 = \alpha_i & \\
\forall i < j \leq n & \|x_i - x_j\|_2^2 \geq \alpha_i \alpha_j & \\
\forall i \leq n & x_i \in [-1, 1]^K & \\
\forall i \leq n & \alpha_i \in \{0, 1\} &
\end{array} \right\}$$

5.5 Obnoxious Facility Location Problem

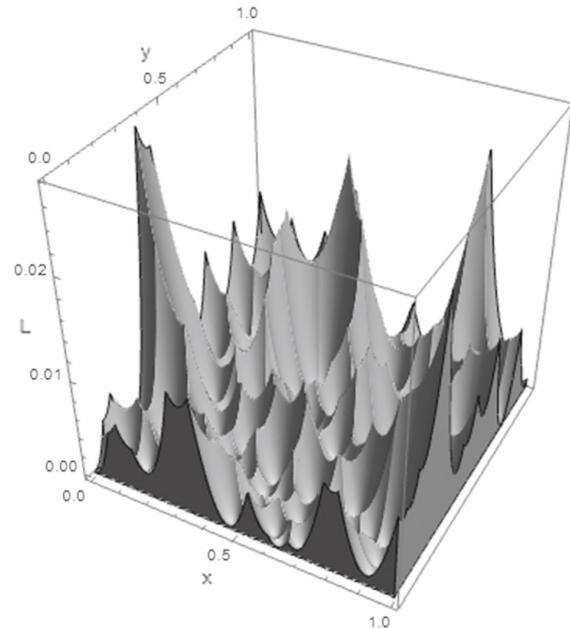
Extremely non-convex optimization problems: the case of the multiple obnoxious facilities location - Paweł Kalczynski & Zvi Drezner 2021

Let $A_i = (a_i, b_i)$ for $i = 1, \dots, n$ be the locations of communities, and $X_j = (x_j, y_j)$ for $j = 1, \dots, p$ be the unknown locations of the p facilities. The non-linear programming formulation is:

$$\begin{aligned}
\max & L \\
\text{subject to:} & \\
& (x_j - a_i)^2 + (y_j - b_i)^2 \geq L \quad \text{for } i = 1, \dots, n; j = 1, \dots, p \\
& (x_i - x_j)^2 + (y_i - y_j)^2 \geq D^2 \quad \text{for } 1 \leq i < j \leq p.
\end{aligned}$$

In addition we need constraints that restrict the facilities' locations to a convex polygon or any finite region. Otherwise, the solution is to locate the facilities "at infinity". In the computational experiments we tested problems from Drezner et al. [11] that have a square feasible region.

This problem is extremely non-convex. The surface plot for the $n = 100$ instances (tested in Drezner et al. [11]) is plotted in Fig. 1. There are 202 "hilltops" (for $n = 1000$ there are 2002 hilltops) and the solution for the $p = 1$ instance is on the top of the highest hill. The hilltops are located at Voronoi points [1,19,21,22].



- The obnoxious facility location problem was initiated in the 1970's [2,3,7,13,17,18, 24], and intensively analyzed in the literature (for reviews see [6,11]). Suppose that n communities are located in an area. One needs to locate $p \geq 1$ obnoxious facilities in that area. Examples of obnoxious facilities include airports, industrial facilities, landfills, prisons, and others affecting residents living nearby.
- Most of the papers on the subject are modeled in discrete space, i.e., there is a limited number of potential locations for the facilities (for example in a network environment).
- However, in most of these applications, nuisance propagates "by air". Therefore, the use of planar Euclidean distances is recommended.
- Locations of facilities are usually far from the communities in open spaces, and thus not limited to a list of locations, so location anywhere in the area is useful as well.
- Alternative problem: multiple obnoxious facility location problem recently investigated in Drezner et al. [11]. The objective is to maximize the shortest distance between facilities and communities subject to a minimum distance D between facilities.

5.6 Multi Player Nash Equilibrium

Multilinear Formulations for Computing a Nash Equilibrium of Multi-Player Games - Miriam Fischer, Akshay Gupte ORCID-Logo **fischer_et_al:LIPIcs.SEA.2023.12**

The multi-player multilinear formulation is an extension of a bilinear formulation for bimatrix games [9]. To motivate the multilinear formulation, we shortly recall the bilinear program that is equivalent to finding a Nash equilibrium in a bimatrix game. To do so, we introduce some notation. Let $A, B \in \mathbb{R}^{m \times n}$ be the payoff matrix of player 1 and player 2 , with m pure strategies of player 1 and n pure strategies of player 2 . Let $x \in \mathbb{R}^m$ with $x \geq 0$ and $\sum_{i=1}^m x_i = 1$ be a (possibly mixed) strategy of player 1 , with x_s being the probability placed on pure strategy s . Let $y \in \mathbb{R}^n$ with $y \geq 0$ and $\sum_{j=1}^n y_j = 1$ be a (possibly mixed) strategy of player 2 . Let 1_n and 1_m denote vectors of all ones of dimension n and m . Any globally optimal solution (x, y, p, q) to the bilinear optimization problem in BLP is equivalent to a Nash equilibrium in a bimatrix game.

$$\begin{aligned} & \max_{x,y,p,q} x^\top A y + x^\top B y - p - q \\ \text{s.t. } & Ay \leq p 1_m, \quad B^\top x \leq q 1_n \\ & \sum_{i=1}^m x_i = 1, \quad \sum_{j=1}^n y_j = 1, \quad x, y \geq 0. \end{aligned}$$

Definition 5.1: Mixed Nash Equilibrium

Let $\Gamma = \langle \{1, \dots, n\}, (S_i), (A_i) \rangle$ be a game with $n, S_i, A_i, \mathbf{x}^i$ defined as above. Let $\mathbf{x}^i \geq 0$ with $\sum_{s \in S_i} x_s^i = 1$ be a mixed strategy of player i . Then, $\mathbf{x}^* = (\mathbf{x}^{*1}, \dots, \mathbf{x}^{*n})$ with $\mathbf{x}^{*i} \geq 0$ and $\sum_{s \in S_i} x_s^{*i} = 1$ for all players i is a (mixed) Nash equilibrium if for all players i and every mixed strategy \mathbf{x}^i , we have $\mathbb{E}[A_i[\mathbf{x}^*]] \geq \mathbb{E}[A_i[\mathbf{x}^i, \mathbf{x}^{*-i}]]$.

We now present the multilinear optimization formulation (MLP1)

$$\begin{aligned} \max_{\mathbf{x}, \mathbf{p}} & \sum_{i=1}^n \left(\sum_{\substack{(s, \hat{s}) \\ \in S_i \times S_{-i}}} A_i[s, \hat{s}] x_s^i \prod_{s_j \in \hat{s}} x_{s_j}^j \right) - \sum_{i=1}^n p^i \\ & \sum_{\hat{s} \in S_{-i}} A_i[s, \hat{s}] \prod_{s_j \in \hat{s}} x_{s_j}^j \leq p^i \quad \forall i \in [n], s \in S_i \\ & \sum_{s \in S_i} x_s^i = 1 \quad \forall i \in [n] \\ & 0 \leq x_s^i \leq 1 \quad \forall i \in [n], s \in S_i \end{aligned}$$

Theorem 5.2

A (mixed) strategy $(\mathbf{x}^1, \dots, \mathbf{x}^n)$ is a (mixed) Nash equilibrium of the n -player game (A_1, \dots, A_n) if and only if there exist numbers p^1, \dots, p^n such that $(\mathbf{x}^1, \dots, \mathbf{x}^n, p^1, \dots, p^n)$ is an optimal solution to the problem in MLP1.

- Knapsack with S shaped Objectives
- Chemical Engineering
- Feasibility in combinatorial optimization

6. Optimization approaches

Resources

- *Gurobi 9.0 webinar on non-convex quadratic programming*
- *Gurobi Slides*
- *Jan Kronqvist - Building upon linear MIP techniques to solve convex MINLP*
- *Workshop in MINLP*
- *Santanu Dey - IPCO 2021 Summer School*
- *Convexification for Non-Convex Mixed-Integer Quadratic Programming - Sam Burer - INFORMS 2021*

6.1 The general global optimization paradigm

Much of this was borrowed from lecture slides by Santanu Dey and Laura Sanita.

General optimization problem

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in S \subseteq \mathbb{R}^n, \\ & x \in [l, u], \end{aligned}$$

where

1. f is not necessarily a convex function, S is not necessarily a convex set.
2. Ideal goal: Find a globally optimal solution: x^* , i.e. $x^* \in S \cap [l, u]$ such that $OPT := f(x^*) \leq f(x) \forall x \in S \cap [l, u]$.
3. What we will usually settle for: $x^* \in S \cap [l, u]$ (may be approximately feasible) and a lower bound: LB such that:

$$x^* \in S \cap [l, u] \text{ and } \text{gap} := \frac{f(x^*) - LB}{LB} \text{ is "small".}$$

6.1.1. Heuristic Approach

A heuristic approach that finds a solution with probability 1 in infinite time is described below. The Multistart method, being the easiest Global Optimization ("GO) method, is utilized.

Algorithm 1 Multistart Heuristic Approach

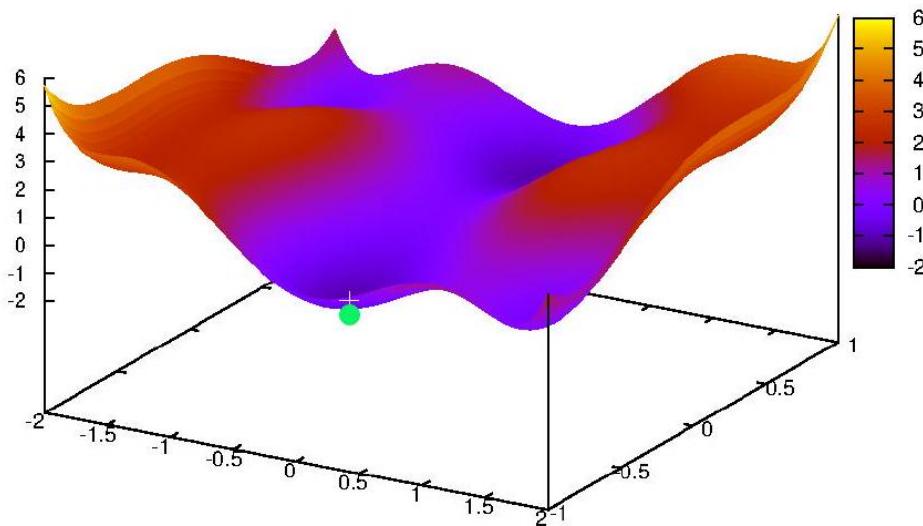
```

1:  $f^* \leftarrow \infty$ 
2:  $x^* \leftarrow (\infty, \dots, \infty)$ 
3: while not termination condition do
4:    $x' \leftarrow (\text{random}(), \dots, \text{random}())$ 
5:    $x \leftarrow \text{localSolve}(P, x')$ 
6:   if  $f_P(x) < f^*$  then
7:      $f^* \leftarrow f_P(x)$ 
8:      $x^* \leftarrow x$ 
```

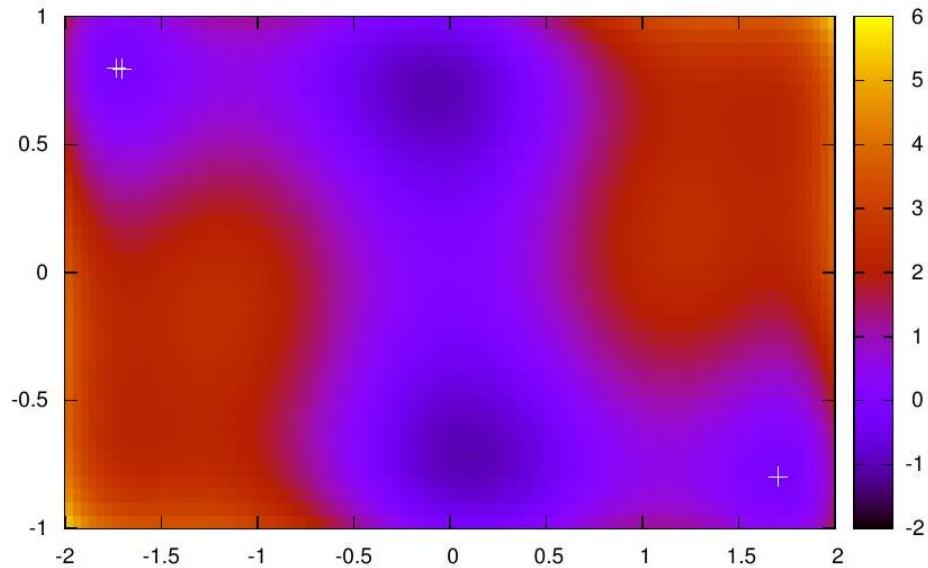
Termination condition can be, for example, repeating the process k times.

Six-hump camelback function

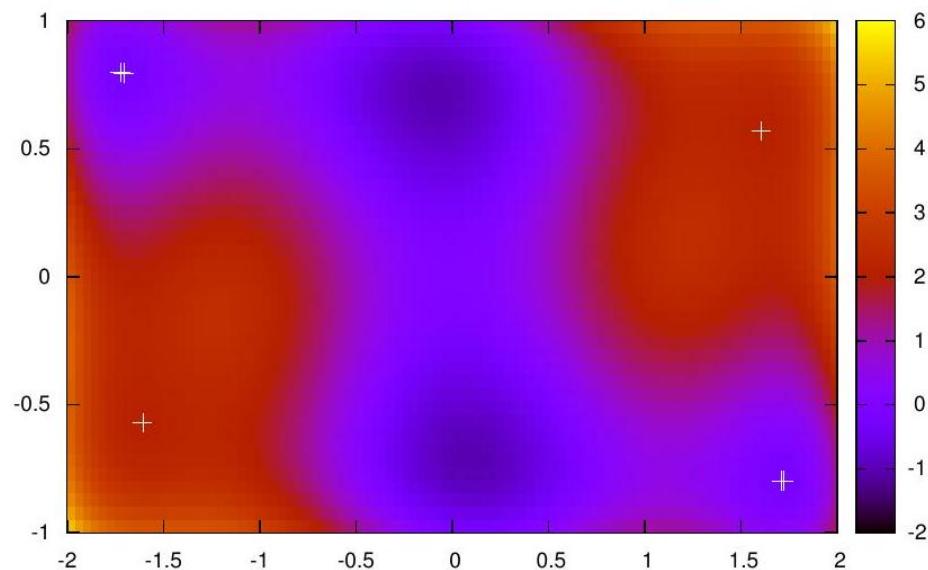
$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$



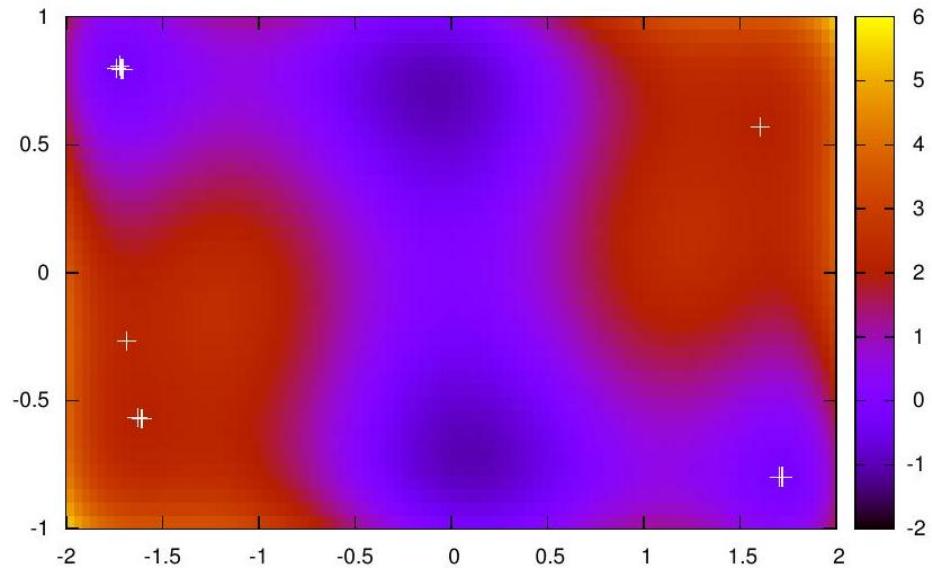
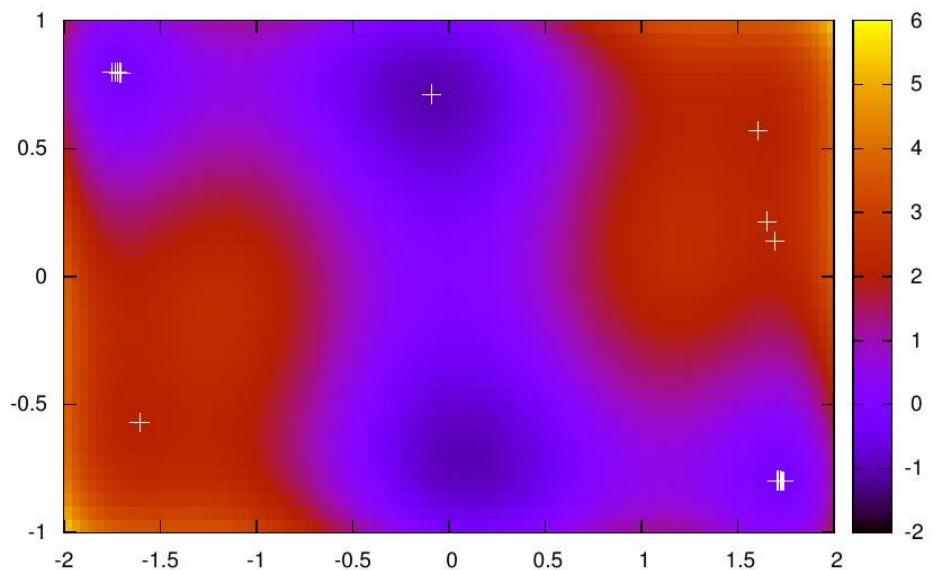
Global optimum (COUENNE)

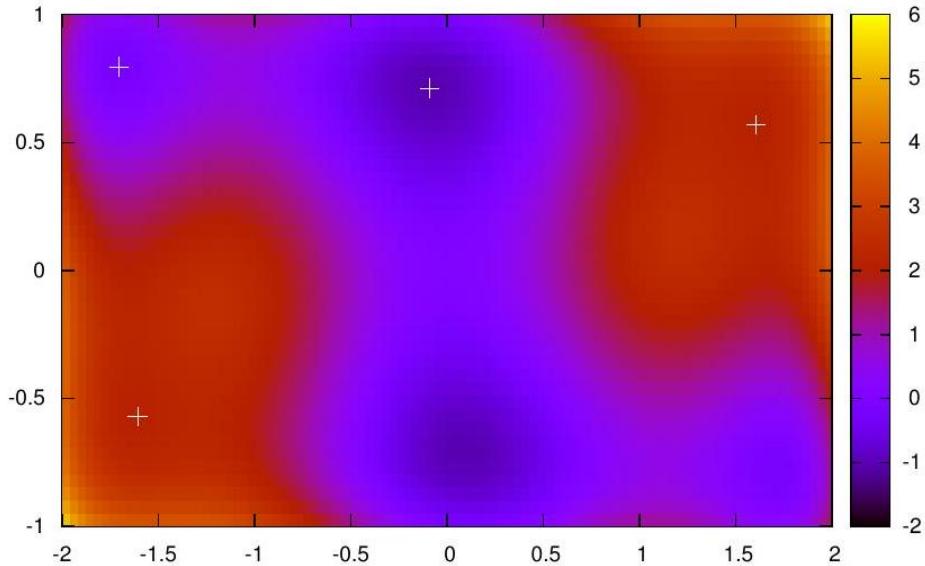


Multistart with IPOPT, $k = 5$



Multistart with IPOPT, $k = 10$

Multistart with IPOPT, $k = 20$ Multistart with IPOPT, $k = 50$



Multistart with SNOPT, $k = 20$

6.1.2. Relaxations and Convex Hulls

For a problem with a linear objective function

$$\min c^\top x \quad (6.1)$$

$$x \in \mathcal{X} \quad (6.2)$$

for some feasible set \mathcal{X} . Let $p_{\mathcal{X}}^*$ denote the optimal objective value.

Theorem 6.1: Fundamental Property

Let $\mathcal{F} \subseteq \mathbb{R}^n$ (not necessarily convex) be compact and let $\mathcal{R} \supseteq \text{conv}(\mathcal{F})$. Then

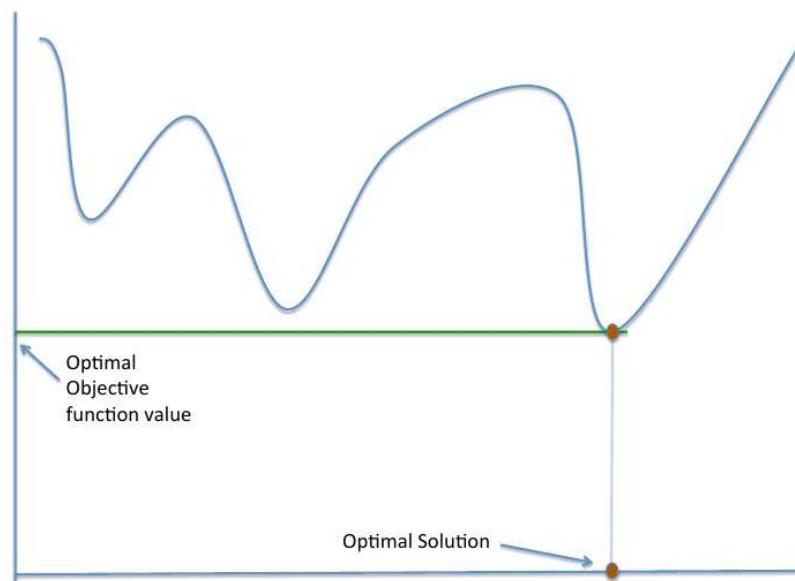
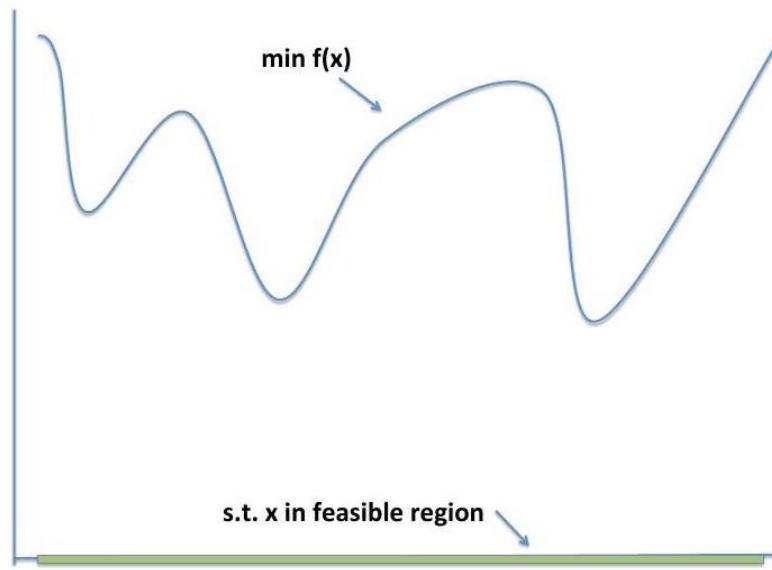
$$p_{\mathcal{F}}^* = p_{\text{conv}(\mathcal{F})}^* \geq p_{\mathcal{R}}^*,$$

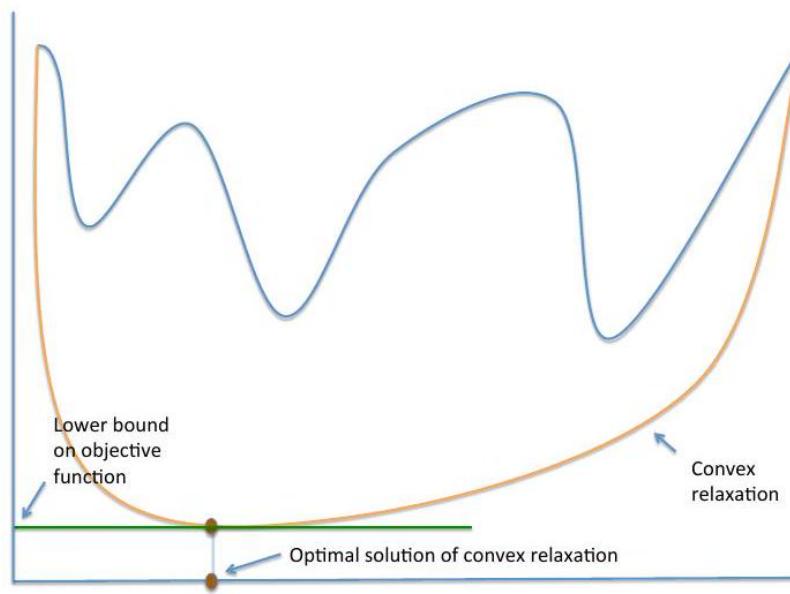
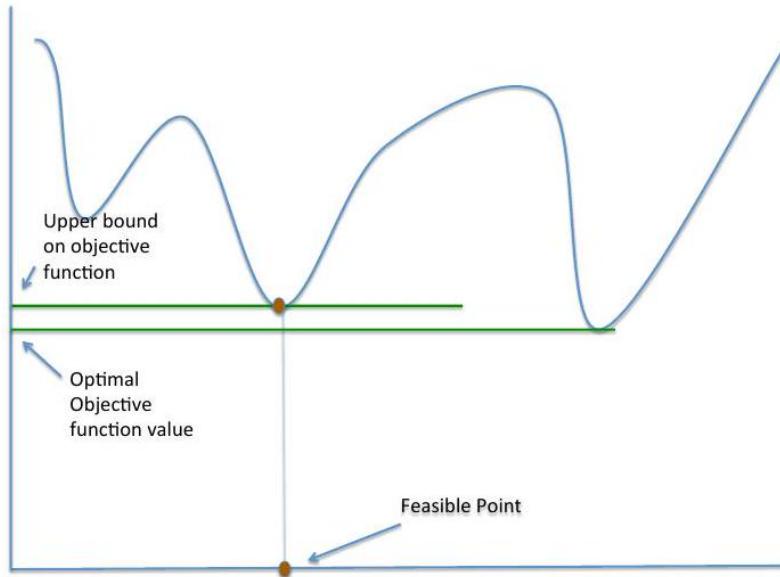
that is,

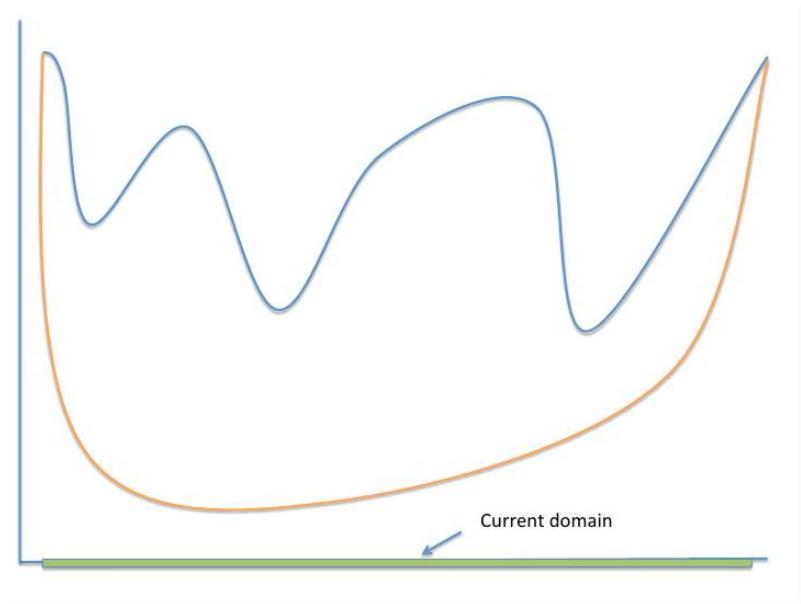
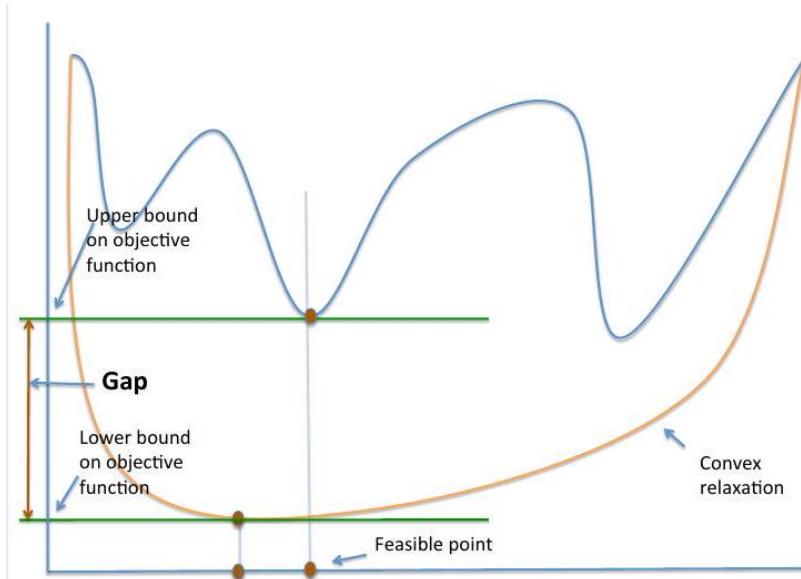
1. optimizing over the convex hull of feasible solutions returns the same objective value,
2. optimizing over a relaxation returns a lower bound.

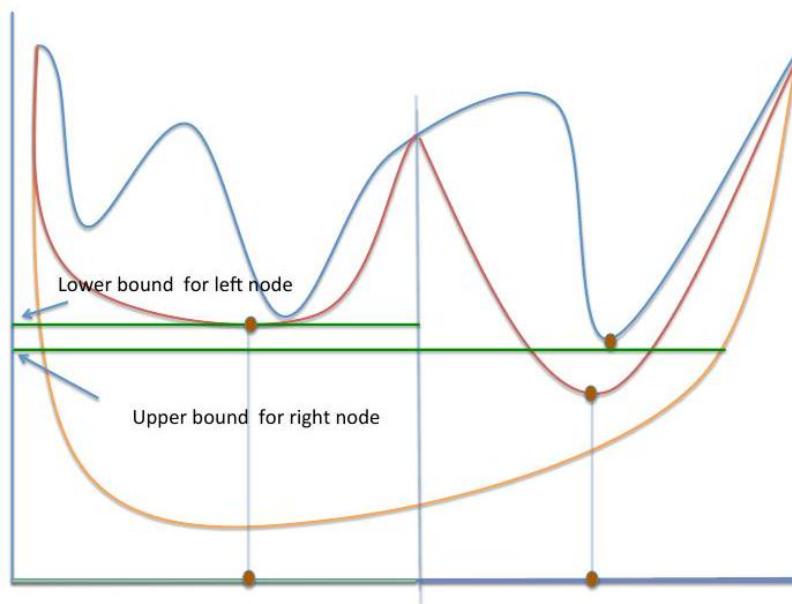
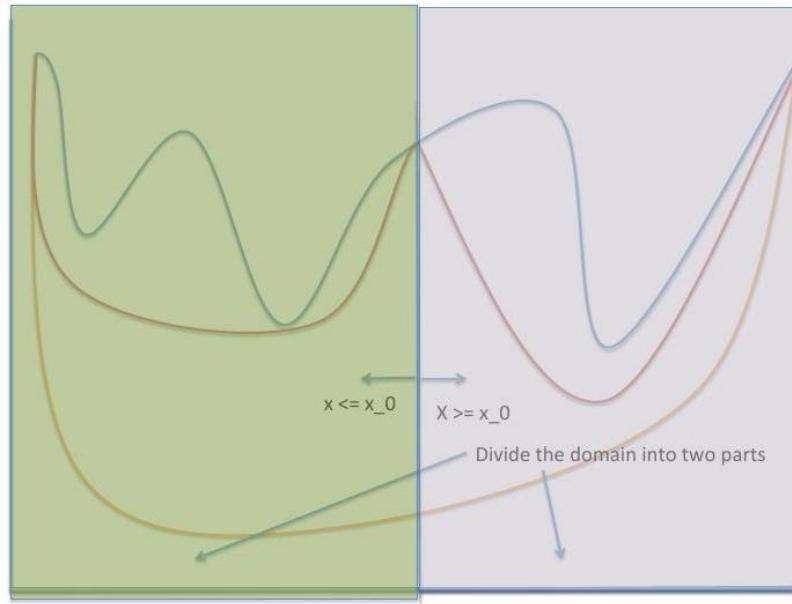
6.1.3. Solving using Branch-and Bound

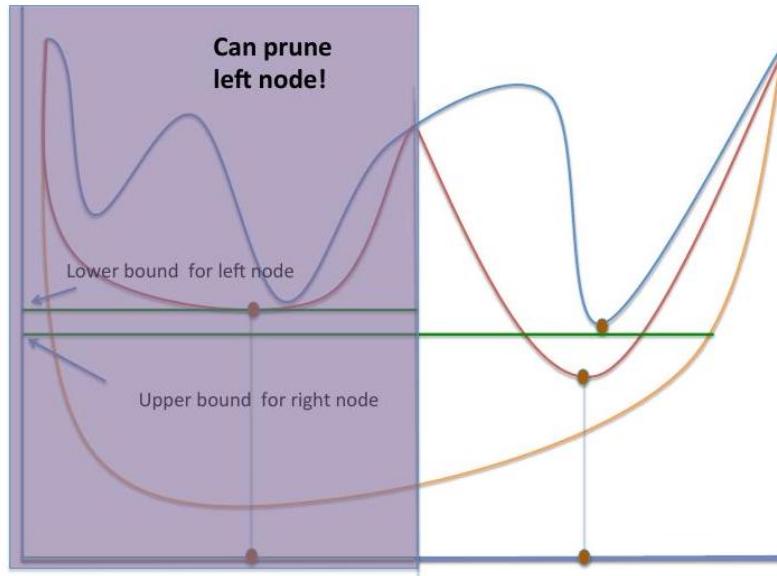
BRANCH-AND-BOUND











6.1.4. Discussion of Branch-and-bound algorithm

- The method works because: As the domain becomes "smaller" in the nodes, we are able to get a better (tighter) lower bound on $f(x)$.
- Usually S is not a convex set, then we need to obtain both: (1) a convex function that lower bounds $f(x)$ and (2) A convex relaxation of S .

Our task is to obtain:

- (1) Machinery for obtaining "Good" lower bounding function that are convex and satisfying (*)
- (2) "Good" convex relaxation of non-convex sets $S \cap [l, u]$.

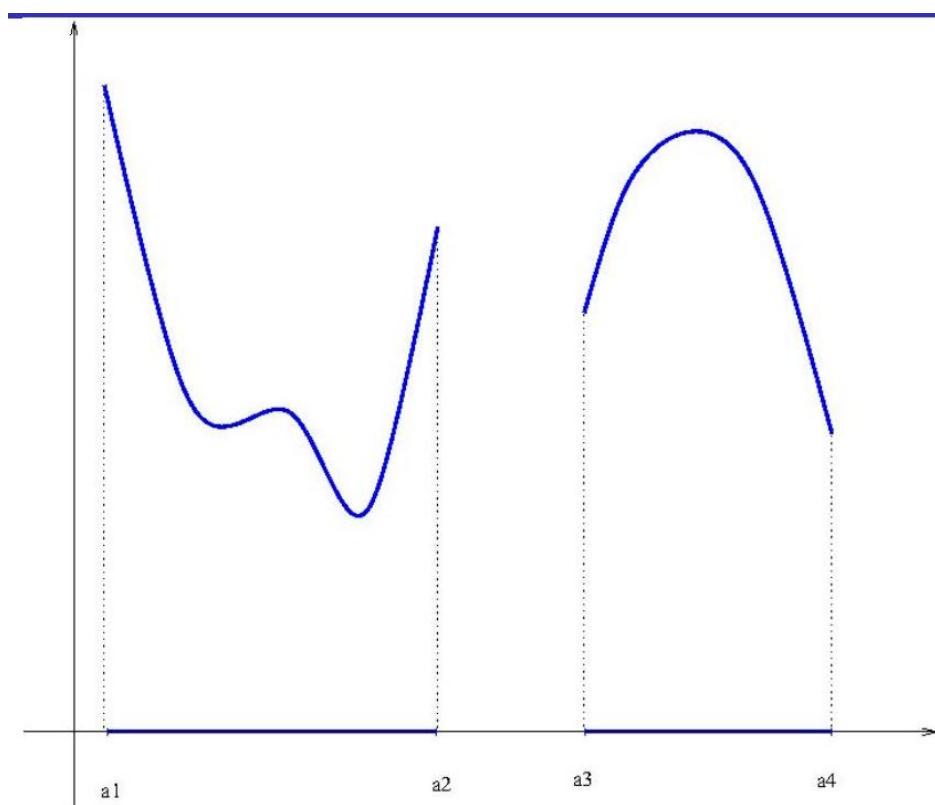
Other notes:

Falk and Soland (1969) "An algorithm for separable nonconvex programming problems".

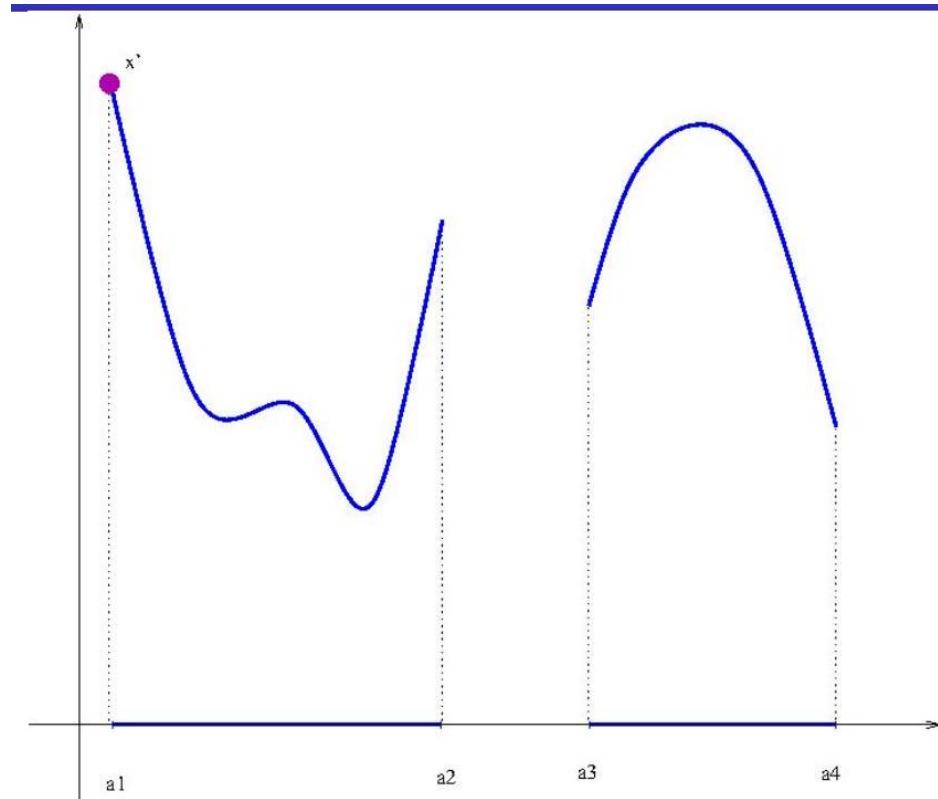
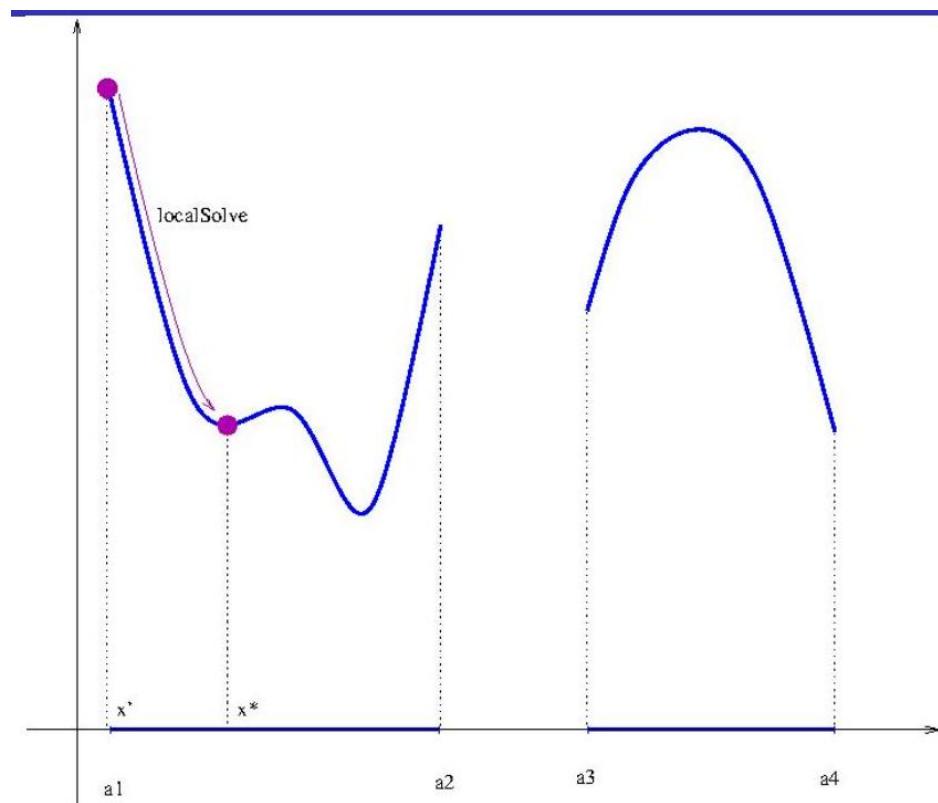
20 years ago: first general-purpose "exact" algorithms for nonconvex MINLP.

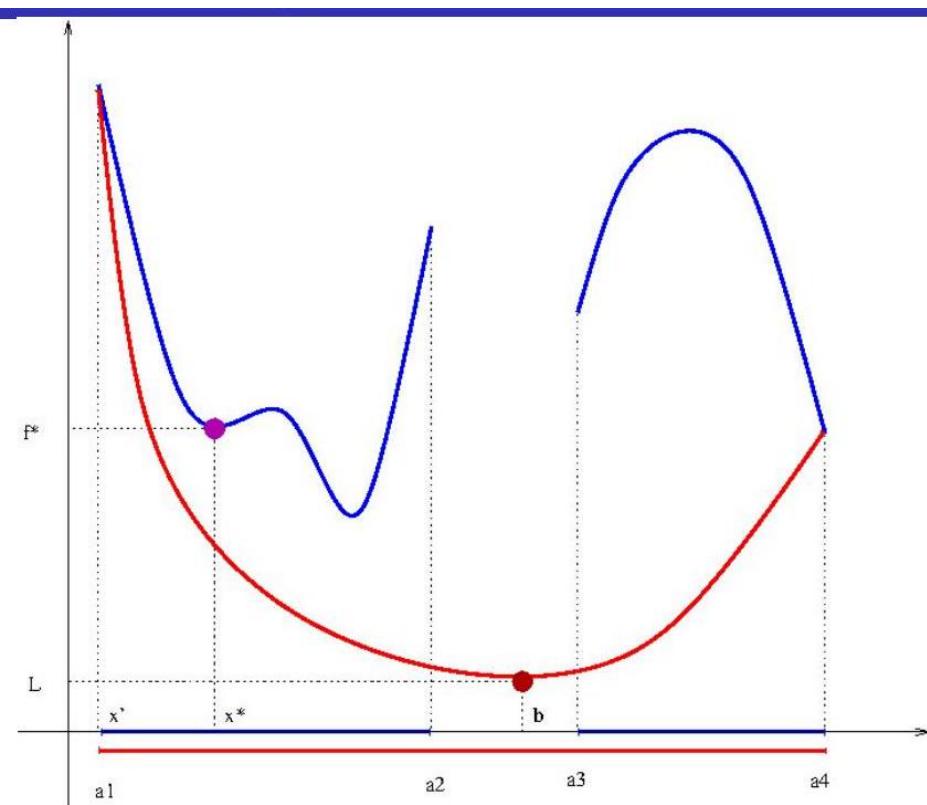
- Tree-like search
- Explores search space exhaustively but implicitly
- Builds a sequence of decreasing upper bounds and increasing lower bounds to the global optimum
- Exponential worst-case
- Only general-purpose "exact" algorithm for MINLP Since continuous vars are involved, should say " ϵ -approximate"
- Like BB for MILP, but may branch on continuous vars Done whenever one is involved in a nonconvex term

Another Spatial B&B: Example - Nonconvex Domain

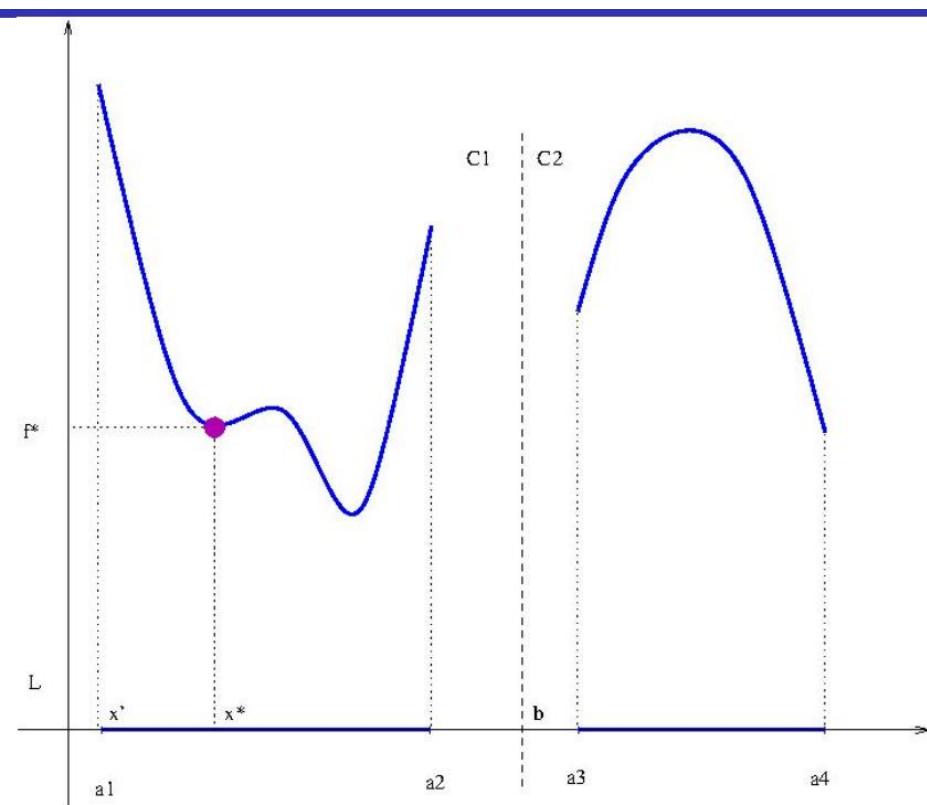


Original problem P

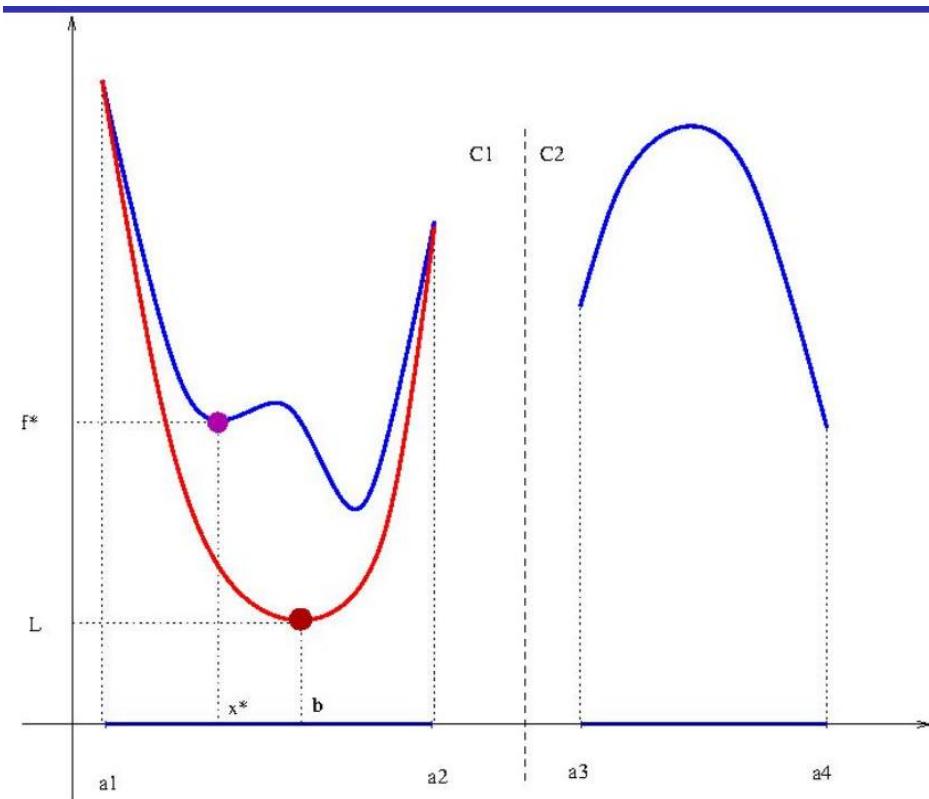
Starting point x' Local (upper bounding) solution x^*



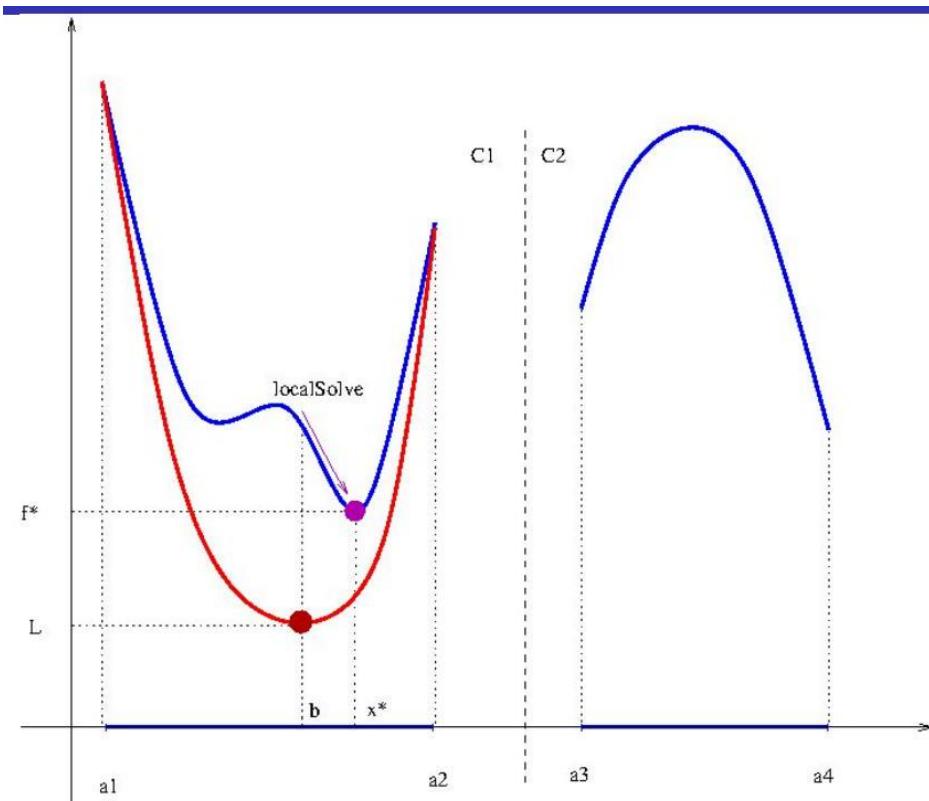
Convex relaxation (lower) bound \bar{f} with $|f^* - \bar{f}| > \varepsilon$



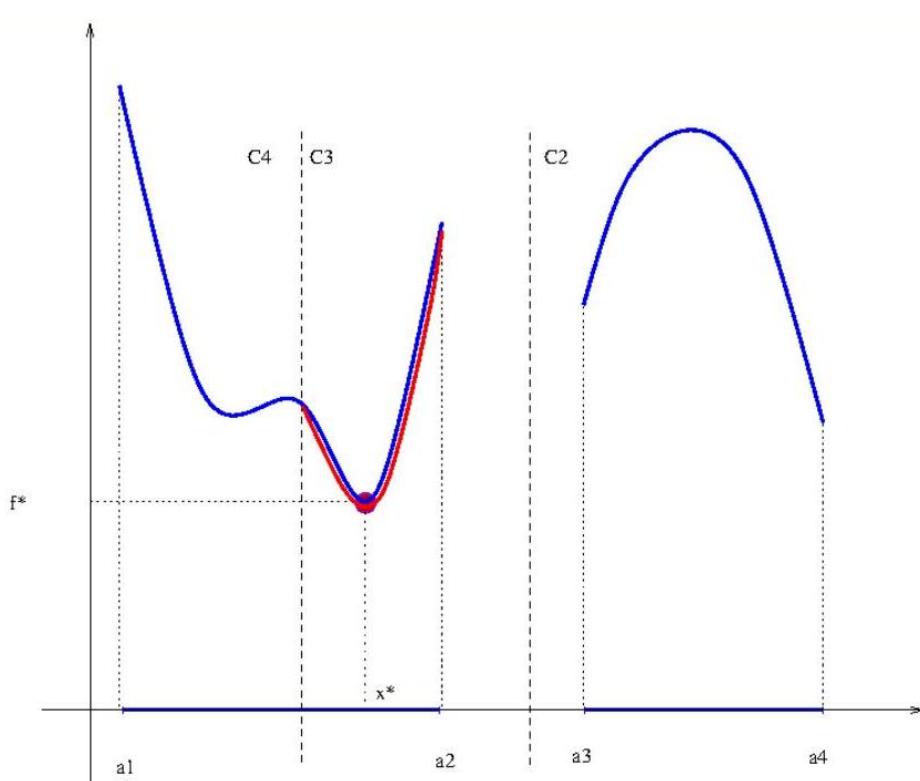
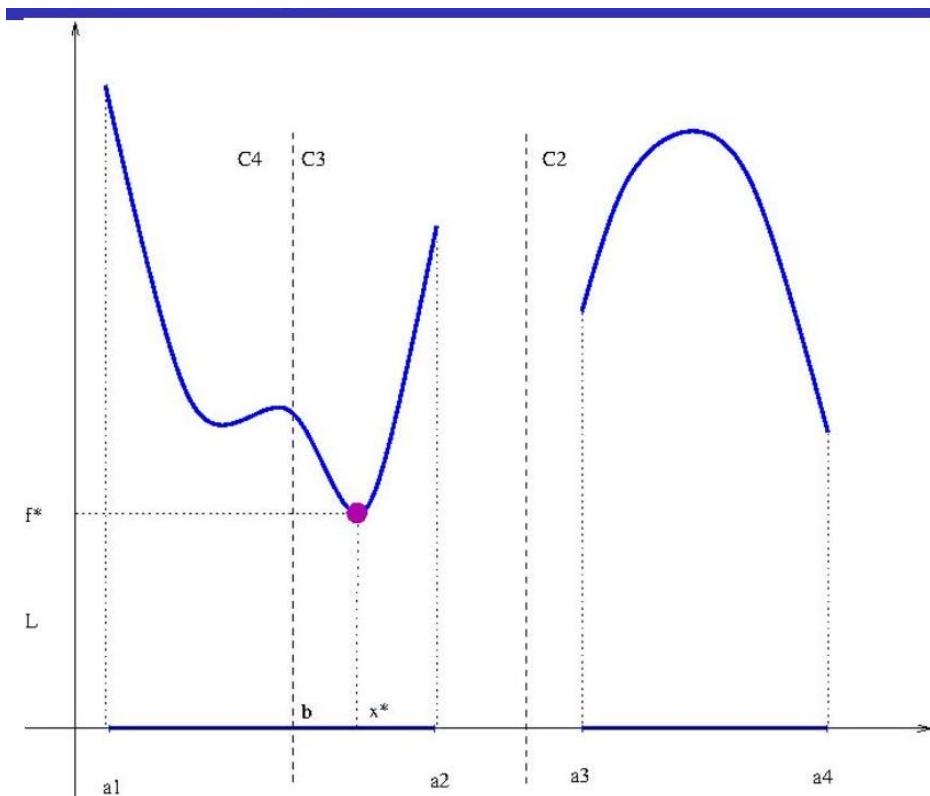
Branch at $x = \bar{x}$ into C_1, C_2

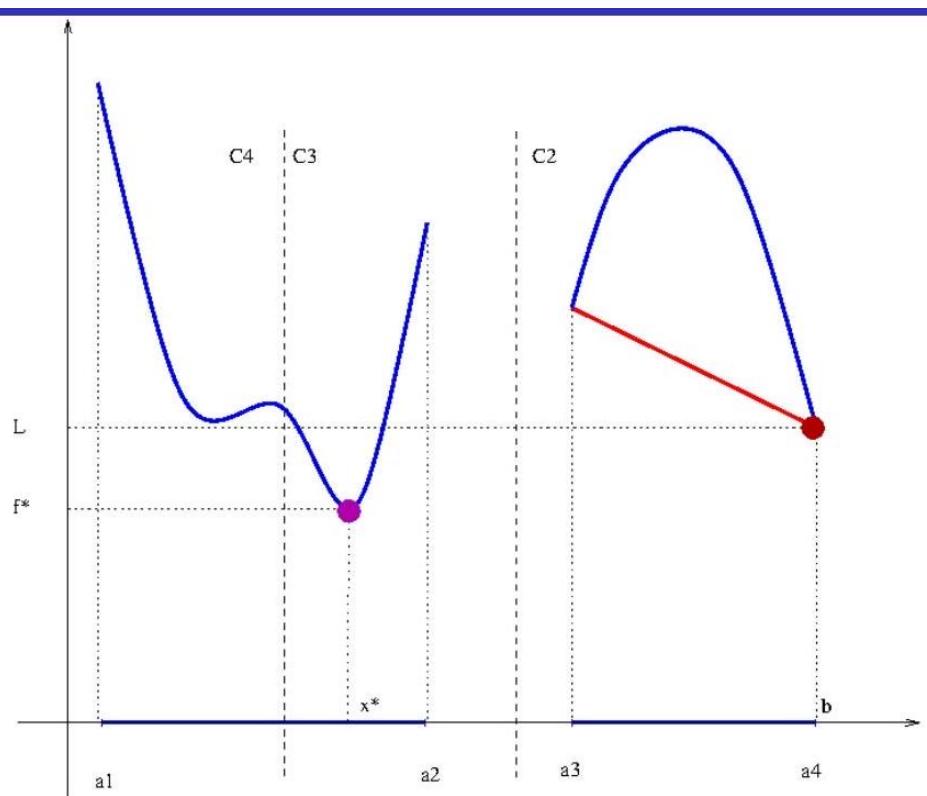


Convex relaxation on C_1 : lower bounding solution \bar{x}

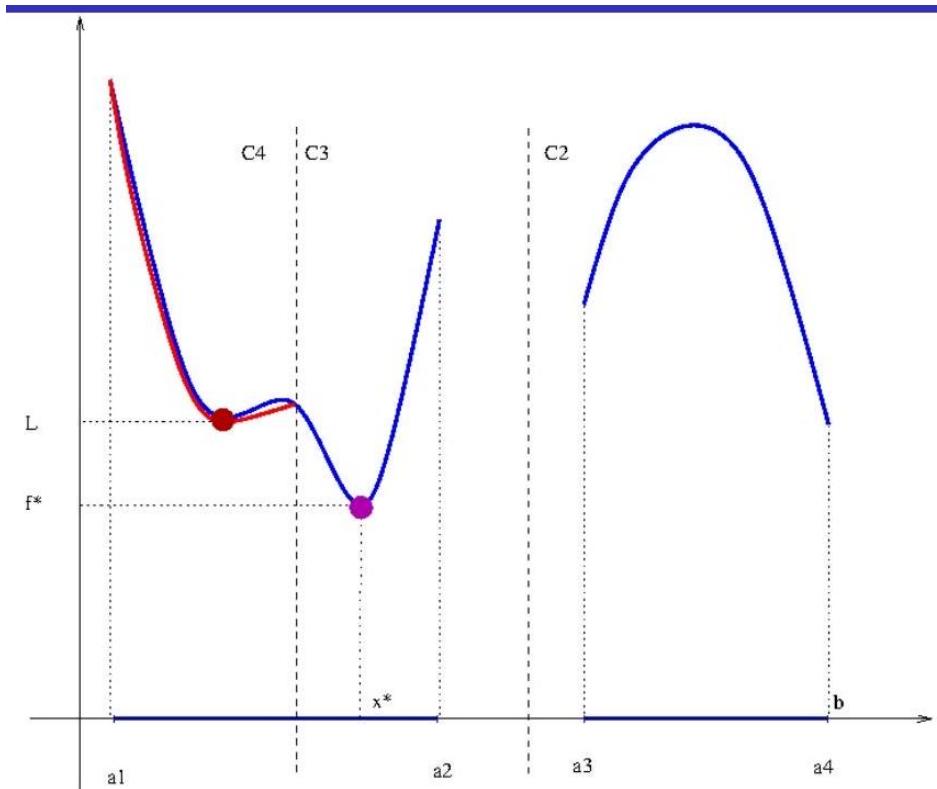


localSolve, from \bar{x} : new upper bounding solution x^*

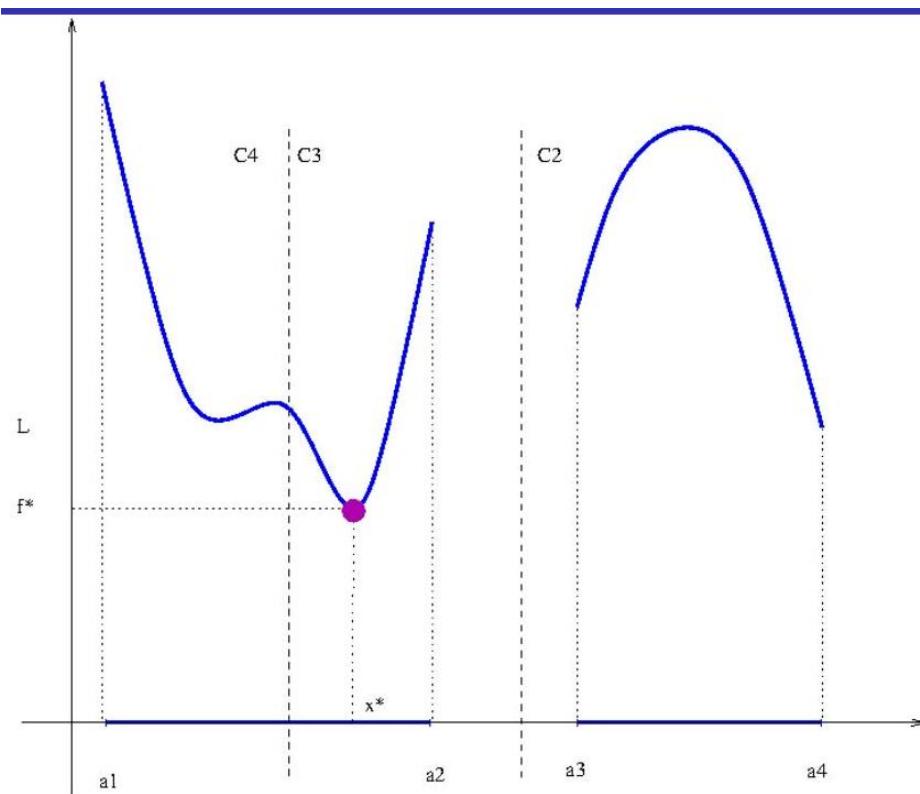




Repeat on $C_2 : \bar{f} > f^*$ (can't improve x^* in C_2)



Repeat on $C_4 : \bar{f} > f^*$ (can't improve x^* in C_4)



No more subproblems left, return x^* and terminate

6.1.4.1. Spatial B&B: Pruning

- P was branched into C_1, C_2
- (2) C_1 was branched into C_3, C_4
- (3) C_3 was pruned by optimality ($x^* \in \mathcal{G}(C_3)$ was found)
- (4) C_2, C_4 were pruned by bound (lower bound for C_2 worse than f^*)
- (6) No more nodes: whole space explored, $x^* \in \mathcal{G}(P)$
 - Search generates a tree
 - Subproblems are nodes
 - Nodes can be pruned by optimality, bound or infeasibility (when subproblem is infeasible)
 - Otherwise, they are branched

6.1.5. Spatial B&B: General idea

Aimed at solving "factorable functions", i.e., f and g of the form:

$$\sum_h \prod_k f_{hk}(x, y)$$

where $f_{hk}(x, y)$ are univariate functions $\forall h, k$.

- Exact reformulation of MINLP so as to have "isolated basic nonlinear functions" (additional variables and constraints).
- Relax (linear/convex) the basic nonlinear terms (library of envelopes/underestimators).
- Relaxation depends on variable bounds, thus branching potentially strengthen it.

6.1.6. Spatial B&B: Exact Reformulation to Standard Form

Consider a NLP for simplicity. Transform it in a standard form like:

$$\begin{aligned} & \min c^\top(x, w) \\ & A(x, w) \leq b \\ & w_{ij} = x_i \otimes x_j \quad \text{for suitable } i, j \\ & x \in X \\ & w \in W \end{aligned}$$

where, for example, $\otimes \in \{ \text{sum, product, quotient, power, exp, log, sin, cos, abs} \}$ (Couenne).

6.1.7. Spatial B&B: convexification

Relax $w_{ij} = x_i \otimes x_j \forall$ suitable i, j where $\otimes \in \{ \text{sum, product, quotient, power, exp, log, sin, cos, abs} \}$ such that:

$$\begin{aligned} w_{ij} &\leq \text{overestimator } (x_i \otimes x_j) \\ w_{ij} &\geq \text{underestimator } (x_i \otimes x_j) \end{aligned}$$

Convex relaxation is not the tightest possible, but built automatically .

- Underestimator/overestimator of convex/concave function: tangent cuts (OA)
- Odd powers or Trigonometric functions: separate intervals in which function is convex or concave and do as for convex/concave functions

- Product or Quotient: Mc Cormick relaxation

6.1.7.1. Spatial B&B: Examples of Convexifications

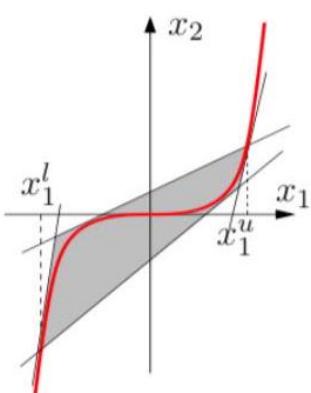


Figure 6.1: (a) $x_2 = x_1^3$

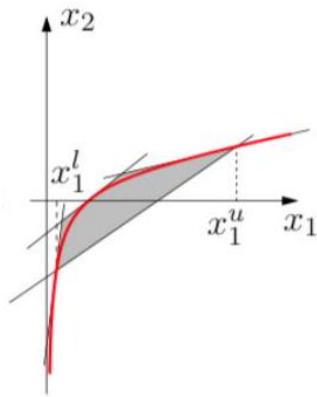


Figure 6.2: (b)
 $x_2 = \log x_1$

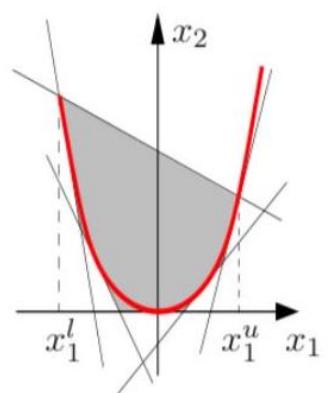


Figure 6.3: (c) $x_2 = x_1^2$

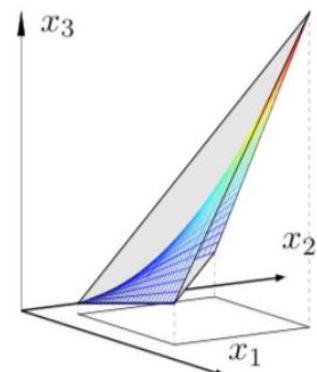


Figure 6.4: (d) $x_3 = x_1 x_2$

P. Belotti, J. Lee, L. Liberti, F. Margot, A. Wächter, "Branching and bounds tightening techniques for non-convex MINLP". Optimization Methods and Software 24(4-5): 597-634 (2009).

6.1.7.2. Example: Standard Form Reformulation

$$\begin{aligned} & \min x_1^2 + x_1 x_2 \\ & x_1 + x_2 \geq 1 \\ & x_1 \in [0, 1] \\ & x_2 \in [0, 1] \end{aligned}$$

becomes

$$\begin{aligned}
 & \min w_1 + w_2 \\
 & w_1 = x_1^2 \\
 & w_2 = x_1 x_2 \\
 & x_1 + x_2 \geq 1 \\
 & x_1 \in [0, 1] \\
 & x_2 \in [0, 1]
 \end{aligned}$$

Convex hull of pieces weaker than the whole convex hull

Consider the following feasible set:

$$\text{Convex hull: } x_1 + x_2 \geq 1$$

Convex hull of standard form

$$\begin{aligned}
 & x_1^2 + x_2^2 \geq 1 \\
 & x_1, x_2 \in [0, 2]
 \end{aligned}$$

$$\begin{aligned}
 & x_3 + x_4 \geq 1 \\
 & x_3 \leq x_1^2 \\
 & x_4 \leq x_2^2 \\
 & x_1, x_2 \in [0, 2]
 \end{aligned}$$

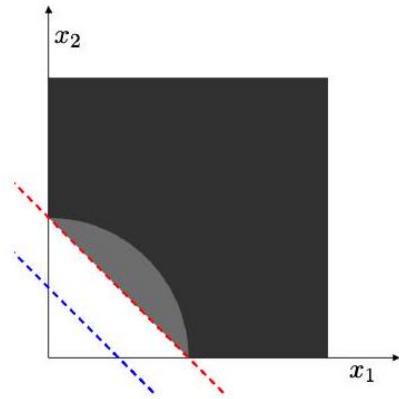


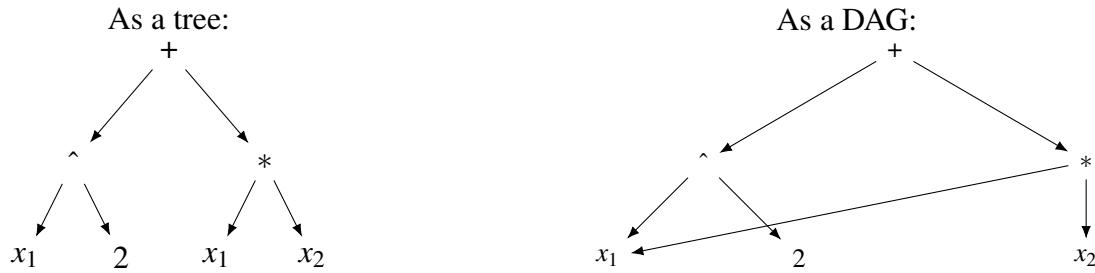
Figure: Source Belotti et al. (2013)

6.1.8. Expression trees

Representation of objective f and constraints g

Encode mathematical expressions in trees or DAGs

E.g. $x_1^2 + x_1 x_2$:



6.1.9. Variable ranges

- Crucial property for sBB convergence: convex relaxation tightens as variable range widths decrease
- convex/concave under/over-estimator constraints are (convex) functions of x^L, x^U
- it makes sense to tighten x^L, x^U at the sBB root node (trading off speed for efficiency) and at each other node (trading off efficiency for speed)

Bounds Tightening

- In sBB we need to tighten variable bounds at each node
- Two methods:
 - Optimization Based Bounds Tightening (OBBT)
 - Feasibility Based Bounds Tightening (FBBT)

OBBT:

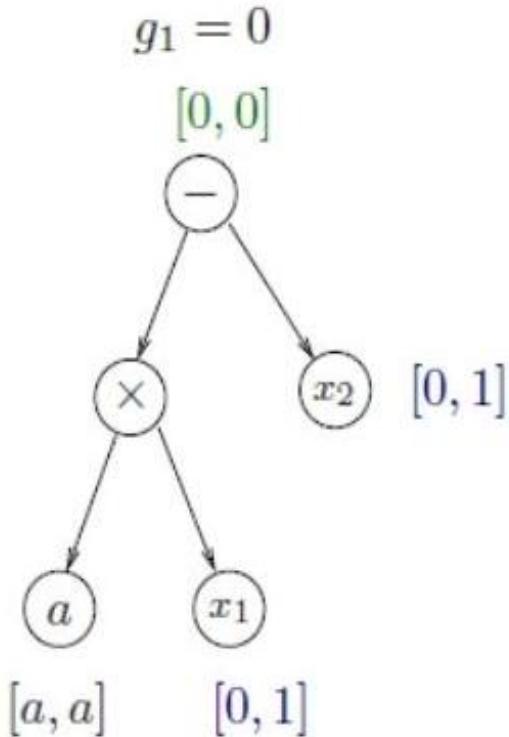
1. For each variable x in P compute

- $\underline{x} = \min\{x \mid \text{conv. rel. constr.}\}$
- $\bar{x} = \max\{x \mid \text{conv. rel. constr.}\}$

2. Set $\underline{x} \leq x \leq \bar{x}$

propagation of intervals up and down constraint expression trees, with tightening at the root node

Example: $5x_1 - x_2 = 0$.



FBBT: Propagation of intervals up and down constraint expression trees, with tightening at the root node

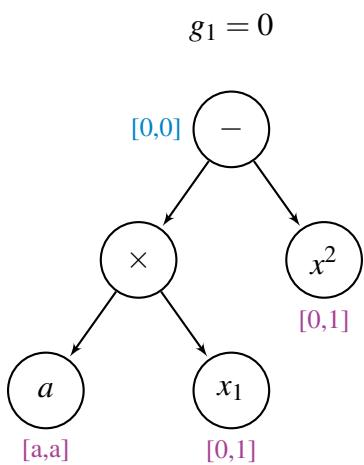
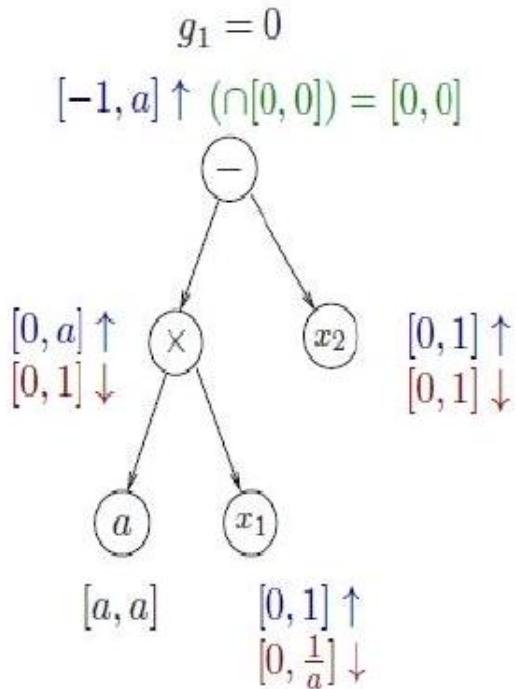
Example: $5x_1 - x_2 = 0$.

Up: $\otimes : [5, 5] \times [0, 1] = [0, 5]$; $\ominus : [0, 5] - [0, 1] = [-1, 5]$.

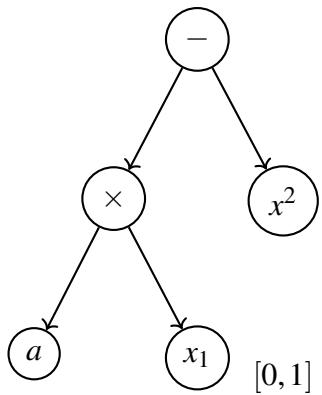
Root node tightening: $[-1, 5] \cap [0, 0] = [0, 0]$.

Downwards: $\otimes : [0, 0] + [0, 1] = [0, 1]$;

$$x_1 : [0, 1] / [5, 5] = [0, \frac{1}{5}]$$

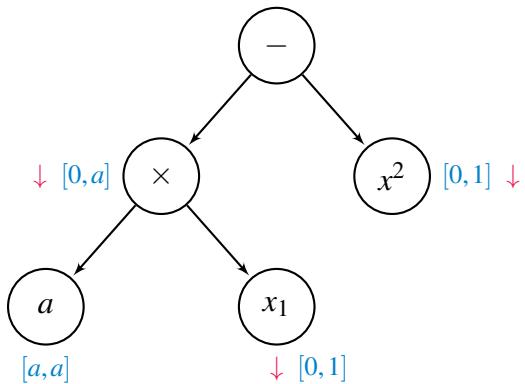


$$[0,0] \quad g_1 = 0$$



$$[a,a] \quad [0,1]$$

$$[-1,a] \uparrow \cap [0,0] = [0,0]$$

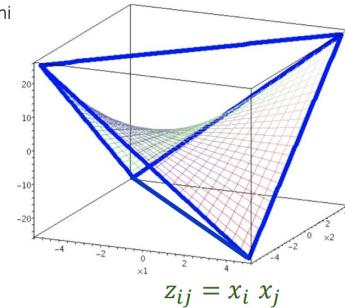


Spatial Branch and Bound with Gurobi on MIQCP

Spatial Branching



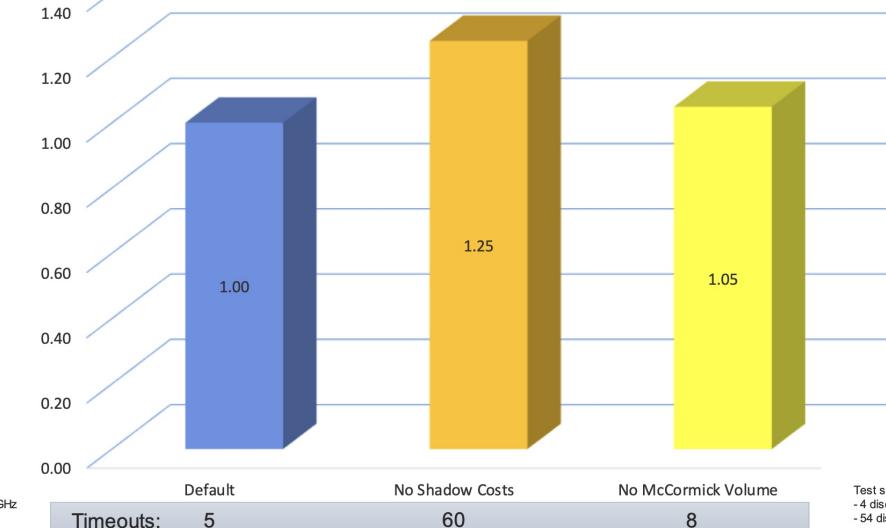
- Branching variable selection
 - What most solvers do: first branching on fractional integer variables as usual
 - If no fractional integer variable exists, select continuous variable in violated bilinear constraint
 - Our variable selection rule is a combination of:
 - sum of absolute bilinear constraint violations
 - reduce McCormick volume as much as possible
 - big McCormick polyhedron is turned into two smaller McCormick polyhedra after branching
 - sum of smaller volumes is smaller than big volume
 - shadow costs of variable for linear constraints
- Branching value selection
 - We use a standard way
 - a convex combination of LP value and mid point of current domain
 - Avoid numerical pitfalls
 - large branching values for unbounded variables
 - tiny child domains if LP value is very close to bound
 - very deep dives (node selection)



Performance Impact of Branching



Time limit: 10000 sec.
Intel Xeon CPU E3-1240 v3 @ 3.40GHz
4 cores, 8 hyper-threads
32 GB RAM



Test set has 444 models, using 5 random seeds:
 - 4 discarded due to inconsistent answers
 - 54 discarded that none of the versions can solve
 - speed-up measured on >10s bracket: 143 models

6.2 Convex Envelopes

Given $S \subseteq \mathbb{R}^n$ and $f : \mathbb{R}^n \rightarrow \mathbb{R}$, we want:

- A function $g : \mathbb{R}^n \rightarrow \mathbb{R}$ that is an under estimator of f over S and,
- g should be convex.

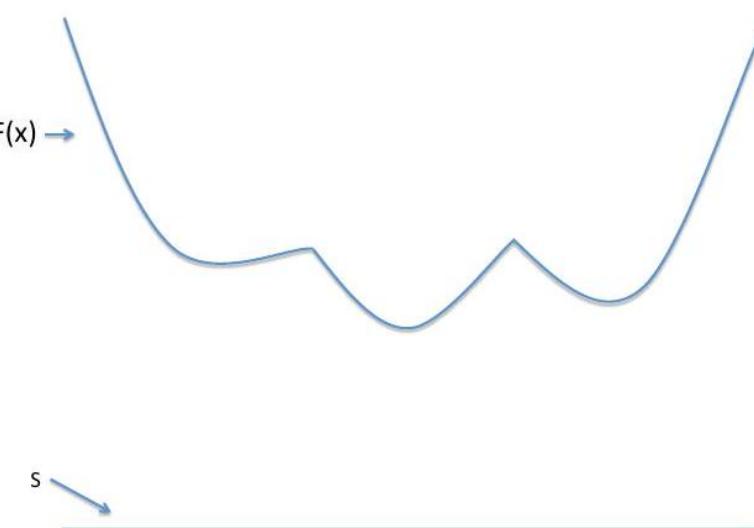
Because (pointwise) supremum of a collection of convex functions is a convex function, we can achieve "the best possible convex under estimator" as follows:

Definition 6.2: Convex envelope

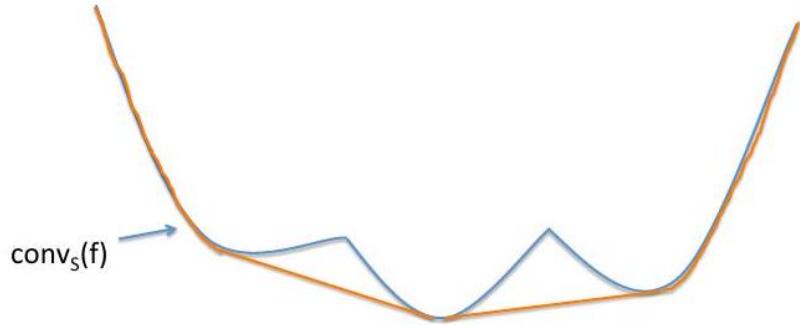
Given a set $S \subseteq \mathbb{R}^n$ and a function $f : S \rightarrow \mathbb{R}$, the convex envelope denoted as $\text{conv}_S(f)$ is:

$$\text{conv}_S(f)(x) = \sup\{g(x) \mid g \text{ is convex on } \text{conv}(S) \text{ and } g(y) \leq f(y) \forall y \in S\}.$$

Convex envelope example



Convex envelope example



6.2.0.1. Another way to think about convex envelope

Definition 6.3: Convex Envelope

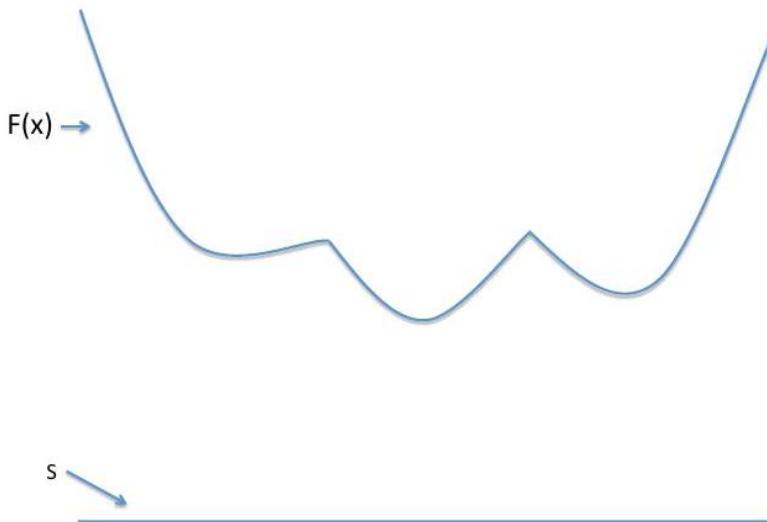
Given a set $S \subseteq \mathbb{R}^n$ and a function $f : S \rightarrow \mathbb{R}$, $\text{conv}_S(f)(x) = \sup\{g(x) \mid g \text{ is convex on } \text{conv}(S) \text{ and } g(y) \leq f(y) \forall y \in S\}$.

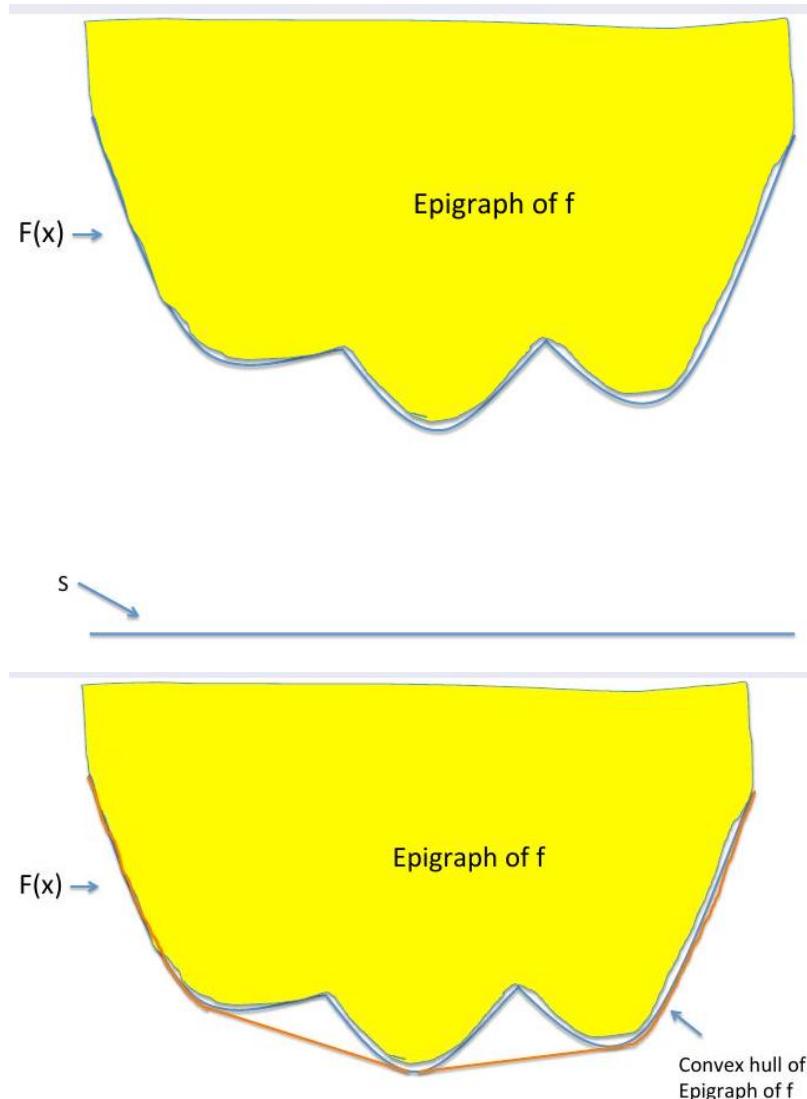
Proposition 6.4

Given a set $S \subseteq \mathbb{R}^n$ and a function $f : S \rightarrow \mathbb{R}$, let $\text{epi}_S(f) := \{(w, x) \mid w \geq f(x), x \in S\}$ denote the epigraph of f restricted to S . Then the convex envelope is:

$$\text{conv}_S(f)(x) = \inf \{y \mid (y, x) \in \text{conv}(\text{epi}_S(f))\}.$$

Example





s Epigraph of f

Proposition 6.5: A simple property of convex envelope

$$\text{conv}_S(f)(x) = \inf \{y \mid (y, x) \in \text{conv}(\text{epi}_S(f))\}.$$

Corollary 6.6

If x^0 is an extreme point of S , then $\text{conv}_S(f)(x^0) = f(x^0)$.

Proof. We verify the contrapositive:

- Consider any $\hat{x} \in S$. If $\text{conv}_S(f)(\hat{x}) < f(\hat{x})$, then (via Proposition (1)) there must be $\{x^i\}_{i=1}^{n+2} \in S$:

$$\hat{x} = \sum_{i=1}^{n+2} \lambda_i x^i, \quad f(\hat{x}) > \sum_{i=1}^{n+2} \lambda_i f(x^i),$$

where $\lambda \in \Delta$ (i.e. $\lambda_i \geq 0 \forall i \in [n+2], \sum_{i=1}^{n+2} \lambda_i = 1$).

- If $\hat{x} = x^i \forall i$, then $f(\hat{x}) \not> \sum_{i=1}^{n+2} \lambda_i f(x^i) \Rightarrow x \neq x^i \Rightarrow \hat{x}$ is not extreme.



When does extreme points of S describe the convex envelope of $f(x)$?

Let S be a polytope.

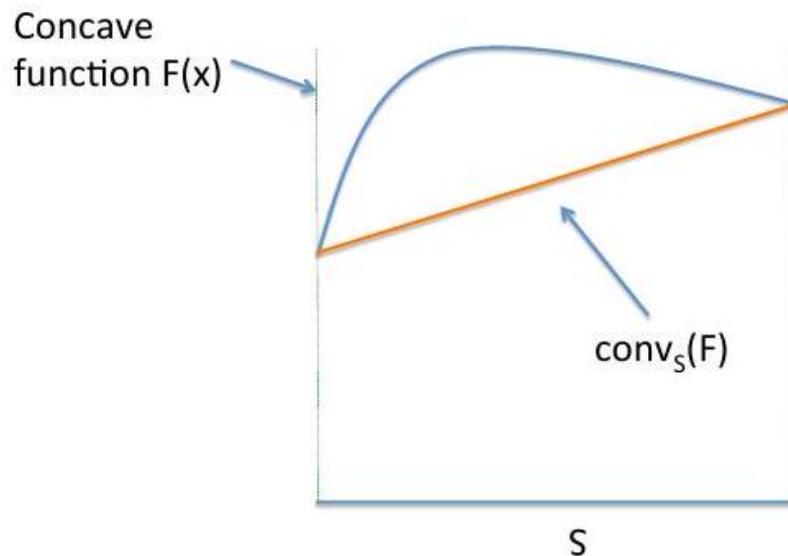
- We know now that $\text{conv}_S(f)(x^0) = f(x^0)$ for extreme points.
- For $x^0 \in S$ and $x^0 \notin \text{ext}(S)$, we know that

$$\text{conv}_S(f)(x^0) = \inf \left\{ y \mid y = \sum_i \lambda_i f(x^i), x^0 = \sum_i \lambda_i x^i, x^i \in S, \lambda \in \Delta \right\}.$$

- It would be nice (why?) if:

$$\text{conv}_S(f)(x^0) = \inf \left\{ y \mid y = \sum_i \lambda_i f(x^i), x^0 = \sum_i \lambda_i x^i, x^i \in \text{ext}(S), \lambda \in \Delta \right\}$$

Concave function work: proof by example



6.2.1. Sufficient condition for polyhedral convex envelope of $f(x)$

Definition 6.7: Edge concave function

Given a polytope $S \subseteq \mathbb{R}^n$. Let $S_D = \{d_1, \dots, d_k\}$ be a set of vectors such that for each edge E (one-dimensional face) of S , S_D contains a vector parallel to E . Let $f : S \rightarrow \mathbb{R}^n$ be a function. We say f is edge concave for S if it is concave on all line segments in S that are parallel to an edge of S , i.e., on all the sets of the form:

$$\{y \in S \mid y = x + \lambda d\}$$

for some $x \in S$ and $d \in S_D$.

Example of edge concave function: Bilinear function

- $S := \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x, y \leq 1\}$.
- $S_d = \{(0, 1), (1, 0)\}$.
- $f(x, y) = xy$ is linear for all segments in S that are parallel to an edge of S .
- Therefore f is a edge concave function over S .

Note: $f(x, y) = xy$ is not concave.

Theorem 6.8: Edge concavity gives polyhedral envelope [Tardella (1989)]

Let S be a polytope and $f : S \rightarrow \mathbb{R}^n$ is an edge concave function. Then $\text{conv}_S(f)(x) = \text{conv}_{\text{ext}(S)}(f)(x)$, where

$$\text{conv}_{\text{ext}(S)}(f)(x) := \min \left\{ y \mid y = \sum_i \lambda_i f(x^i), x = \sum_i \lambda_i x^i, x^i \in \text{ext}(S), \lambda \in \Delta \right\}$$

Corollary 6.9: [Rikun (1997)]

Let $f = \prod_i x_i$ and $S = [l, u]$. Then $\text{conv}_S(f)(x) = \text{conv}_{\text{ext}(S)}(f)(x)$.

Proof sketch

- Claim 1: Since f is edge concave, we obtain: $f(x) \geq \text{conv}_{\text{ext}(S)}(f)(x)$ for all $x \in S$.
- Claim 2: If $f(x) \geq \text{conv}_{\text{ext}(S)}(f)(x)$, then

$$\text{conv}_S(f)(x) = \text{conv}_{\text{ext}(S)}(f)(x)$$

Proof of Claim 1 To prove: $f(x) \geq \text{conv}_{\text{ext}(S)}(f)(x)$

Let $\hat{x} \in \text{relint}(F)$, F is a face of S . Proof by induction on the dimension of F .

- Base case: Consider \hat{x} which belongs to a one-dimensional face of S , i.e. \hat{x} belongs to an edge of f . Then since edge-concavity, we obtain that $f(\hat{x}) \geq \text{conv}_{\text{ext}}(S)(f)(\hat{x})$.
- Inductive step: Let F be a face of S where $\dim(F) \geq 2$. Consider $\hat{x} \in \text{relint}(F)$. If we show that there is x^1, x^2 belonging to proper faces of F , such that $\hat{x} = \lambda_1 x^1 + \lambda_2 x^2, \lambda_1 + \lambda_2 = 1, \lambda_1, \lambda_2 \geq 0$, and $f(\hat{x}) \geq \lambda_1 f(x^1) + \lambda_2 f(x^2)$. Then applying this argument recursively to $f(x^1)$ and $f(x^2)$ we obtain the result.
- Indeed, consider an edge of F and let d be the direction of this edge. Then there exists $\mu_1, \mu_2 > 0$ such that: $\hat{x} + \mu_1 d$ and $\hat{x} - \mu_2 d$ belong to lower dimensional faces of F . Now on this segment edge-concavity = concavity, so we are done .

Proof of Claim 2

$$\begin{aligned}\text{conv}_S(f)(x^0) &= \inf \left\{ y \mid y = \sum_i \lambda_i f(x^i), x^0 = \sum_i \lambda_i x^i, x^i \in S, \lambda \in \Delta \right\}. \\ \text{conv}_{\text{ext}}(S)(f)(x^0) &= \inf \left\{ y \mid y = \sum_i \lambda_i f(x^i), x^0 = \sum_i \lambda_i x^i, x^i \in \text{ext}(S), \lambda \in \Delta \right\}.\end{aligned}$$

To prove: $f(x) \geq \text{conv}_{\text{ext}}(S)(f)(x)$, implies $\text{conv}_S(f)(x) = \text{conv}_{\text{ext}}(S)(f)(x)$

- Note that $\text{conv}_S(f) \leq \text{conv}_{\text{ext}}(S)(f)$ (by definition), so it is sufficient to prove $\text{conv}_S(f) \geq \text{conv}_{\text{ext}}(S)(f)$.
- Indeed, observe that

$$\begin{aligned}\text{conv}_S(f) &\geq \text{conv}_S(\text{conv}_{\text{ext}}(S)(f)) \\ &= \text{conv}_{\text{ext}}(S)(f)\end{aligned}$$

where the first inequality because of Claim 1, $f(x) \geq \text{conv}_{\text{ext}}(S)(f)(x)$, and the second inequality because $\text{conv}_{\text{ext}}(S)(f)$ is a convex function.

6.3 MIQCQP

We want to study "convexification" for: Quadratically constrained quadratic program (QCQP)

$$\begin{array}{ll}\min & x^\top Q x + c^\top x \\ \text{s.t.} & x^\top Q^i x + (a^i)^\top x \leq b_i \forall i \in [m] \\ & x \in [l, u],\end{array}$$

Very general model:

- Bounded polynomial optimization (replace higher order terms by quadratic terms by introducing new variables). For example:

$$xyz \leq 3 \Leftrightarrow xy = w, wz \leq 3.$$

- Bounded integer programs (including 0-1 integer programs). For example:

$$x \in \{0, 1\} \Leftrightarrow x^2 - x = 0$$

- Beautiful theory of Lasserre hierarchy which gives convex hulls via a hierarchy of Semi-definite programs (SDPs). (Also called the sums-of-square approach).
- Instead we will consider simple functions and simple sets that are relaxations of general QCQPs are consider their "convexification": You can think of this as the MILP-approach. Even though there are nice hierarchies for obtaining convex hulls in IP, in practice, we construct linear programming relaxations within branch-and-bound algorithm, which are often strengthened by addition of constraints obtained from the convexification of simple substructures.
- There will be other connections with integer programming...
- Usually, we will stick to linear programming (LP) or second order cone representable (SOCr) convex functions and sets for our convex relaxations.

6.4 Convex hull of simple sets

6.4.1. McCormick envelope

McCormick envelope

$$P := \{(w, x, y) \mid w = xy, 0 \leq x, y \leq 1\}$$

We want to find $\text{conv}(P)$.

$$P = \{(w, x, y) \mid \underbrace{w = xy}_{f(x,y)=xy}, \underbrace{0 \leq x, y \leq 1}_{S}\}$$

- So we need to find the convex envelope (and similarly, concave envelope) of $f(x, y) = xy$ over $x, y \in [0, 1]$.
- By previous section result on edge-concavity, we only need to consider the extreme points of $S = [0, 1]^2$.
- $\text{conv}(P) = \text{conv}\{(0, 0, 0), (1, 0, 0), (0, 1, 0), (1, 1, 1)\}$

$$\text{conv}(P) = \{(w, x, y) \mid \underbrace{\begin{array}{l} w \geq 0, w \geq x + y - 1, w \leq x, w \leq y \end{array}}_{\text{McCormick Envelope}}\}.$$

Alternative proof of validity of McCormick envelope

-

$$\underbrace{(x-0)(y-0)}_{\text{product of 2 non-negative terms}} \geq 0 \Leftrightarrow xy \geq 0 \quad \underbrace{\Rightarrow}_{\text{replace } w=xy} \quad w \geq 0.$$

-

$$\underbrace{(1-x)(1-y)}_{\text{product of 2 non-negative terms}} \geq 0 \Leftrightarrow xy \geq x + y - 1 \Rightarrow w \geq x + y - 1.$$

- $(x-0)(1-y) \geq 0 \Rightarrow w \leq x$.
- $(1-x)(y-0) \geq 0 \Rightarrow w \leq y$.
- This is the Reformulation-linearization-technique (RLT) view point (Sherali-Adams).

(Generalized) McCormick Envelope

For two variables, $l_i \leq x_i \leq u_i$ and $l_j \leq x_j \leq u_j$, we first write these inequalities as

$$x_i - l_i \geq 0, \quad u_i - x_i \geq 0 \tag{6.1}$$

$$x_j - l_j \geq 0, \quad u_j - x_j \geq 0 \tag{6.2}$$

Multiplying any two inequalities yields a quadratic inequality, that can be linearized using the Y_{ij} variable. For example,

$$0 \leq (x_i - l_i)(x_j - l_j) = x_i x_j - l_i x_j - l_j x_i + l_i l_j = Y_{ij} - l_i x_j - l_j x_i + l_i l_j. \tag{6.3}$$

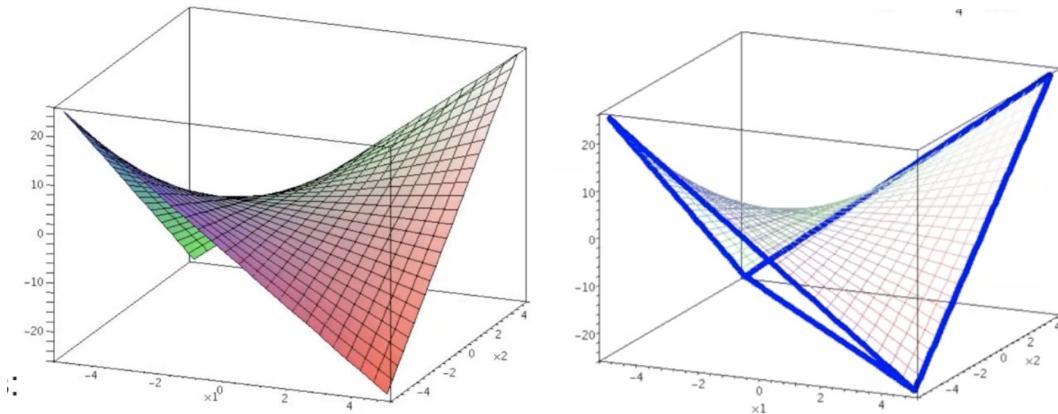
Enumerating all four pairs yields the McCormick Inequalities.

McCormick Inequalities:

$$l_i x_j + l_j x_i - l_i l_j \leq Y_{ij} \leq u_i x_j - l_j x_i + u_i l_j \tag{6.4}$$

$$u_i x_j + u_j x_i - u_i u_j \leq Y_{ij} \leq u_j x_i - l_i x_j + u_j l_i \tag{6.5}$$

McCormick Envelope



6.4.2. Our first convex relaxation of QCQP

$$\begin{aligned} & \text{minimize} && x^T A_0 x + a_0^T x \\ & \text{subject to} && x^T A_k x + a_k^T x \leq b_k, \quad k = 1, \dots, K \\ & && l \leq x \leq u \end{aligned} \tag{QCQP}$$

$$\begin{aligned} & \text{minimize} && \underbrace{A_0 \cdot X}_{\Sigma_{i,j} (A_0)_{ij} X_{ij}} + a_0^T x \\ & \text{subject to} && \underbrace{A_k \cdot X}_{\Sigma_{i,j} (A_k)_{ij} X_{ij}} + a_k^T x \leq b_k, \quad k = 1, \dots, K \\ & && l \leq x \leq u \\ & && X = xx^\top \quad (\text{Nonconvexity}) \end{aligned} \tag{Lifted QCQP}$$

(Note: X is the "outer product" of x , i.e. X is $n \times n$)

McCormick (LP) Relaxation: replace $X = xx^\top$ above by McCormick Inequalities.

6.4.3. Semi-definite programming (SDP) relaxation of QCQPs

$$\begin{aligned}
 & \text{minimize} && x^T A_0 x + a_0^T x \\
 & \text{subject to} && x^T A_k x + a_k^T x \leq b_k, \quad k = 1, \dots, K \\
 & && l \leq x \leq u
 \end{aligned} \tag{QCQP}$$

$$\begin{aligned}
 & \text{minimize} && \underbrace{A_0 \cdot X}_{\sum_{i,j} (A_0)_{ij} X_{ij}} + a_0^T x \\
 & \text{subject to} && \underbrace{A_k \cdot X}_{\sum_{i,j} (A_k)_{ij} X_{ij}} + a_k^T x \leq b_k, \quad k = 1, \dots, K \\
 & && l \leq x \leq u \\
 & && X = xx^\top \quad (\text{Nonconvexity})
 \end{aligned} \tag{Lifted QCQP}$$

SDP Relaxation: replace $X - xx^\top = 0$ above by:

$$\begin{aligned}
 & X - xx^\top \in \text{cone of positive-semi definite matrix} \\
 \Leftrightarrow & \begin{bmatrix} 1 & x^\top \\ x & X \end{bmatrix} \in \text{cone of positive-semi definite matrix}.
 \end{aligned}$$

Comments

- The SDP relaxation is the first level of the sum-of-square hierarchy. (We will not discuss this more here)
- The McCormick relaxation is first (basic) level of the RLT hierarchy.
- The McCormick relaxation and the SDP relaxation are incomparable. So many times if one is able to solve SDPs, both the relaxations are thrown in together.
- Note that the McCormick relaxation has the (*) property, i.e. as the bounds $[l, u]$ get tighter, the McCormick envelopes gets better. In particular, if $l = u$, then the McCormick envelope is exact. Therefore, we can obtain "asymptotic convergence of lower and upper bound" using a branch and bound tree with McCormick relaxation, as the size of the tree goes off to infinity.

6.4.4. Extending the McCormick envelope ideas

Extending the McCormick envelope argument: Using extreme points of S to construct convex hull

$$(\text{Lifted QCQP}) : \min A_0 \cdot X + a_0^T x$$

$$\begin{aligned} \text{s.t. } & A_k \cdot X + a_k^T x \leq b_k \quad k = 1, \dots, K \\ & 0 \leq x \leq 1 \\ & X = xx^T \end{aligned}$$

For now ignore the x_i^2 terms and consider the set:

$$Q := \left\{ (X, x) \in \mathbb{R}^{\frac{n(n-1)}{2}} \times \mathbb{R}^n \mid X_{ij} = x_i x_j \forall i, j \in [n], i \neq j, x \in [0, 1]^n \right\}$$

(Here $l = 0$ and $u = 1$ without loss of generality, by rescaling the variables.) extreme points of S to construct convex hull

Theorem 6.10: BurerLetchford2009

Consider the set

$$Q := \left\{ (X, x) \in \mathbb{R}^{\frac{n(n-1)}{2}} \times \mathbb{R}^n \mid X_{ij} = x_i x_j \forall i, j \in [n], i \neq j, x \in [0, 1]^n \right\}$$

Then,

$$\text{conv}(Q) := \underbrace{\text{conv}(\{(X, x) \in \mathbb{R}^{\frac{n(n-1)}{2}} \times \mathbb{R}^n \mid X_{ij} = x_i x_j \forall i, j \in [n], i \neq j, x \in \{0, 1\}^n\})}_{\text{Boolean quadric polytope}}$$

Krein - Milman theorem

Theorem 6.11: (Krein - Milman Theorem)

Let $S \subseteq \mathbb{R}^n$ be a compact set. Then $\text{conv}(S) = \text{conv}(\text{ext}(S))$.

Proof. By Krein - Milman Theorem, It is sufficient to prove that the extreme points of Q :

$$Q := \left\{ (X, x) \in \mathbb{R}^{\frac{n(n-1)}{2}} \times \mathbb{R}^n \mid X_{ij} = x_i x_j \forall i, j \in [n], i \neq j, x \in [0, 1]^n \right\}$$

satisfy $x \in \{0, 1\}^n$.

- Suppose $(\hat{X}, \hat{x}) \in Q$ is an extreme point of S . Assume by contradiction $\hat{x}_i \notin \{0,1\}$. Consider the following points:

$$\begin{aligned}x_j^{(1)} &= \begin{cases} \hat{x}_j & j \neq i \\ \hat{x}_i + \epsilon & j = i \end{cases} & x_j^{(2)} &= \begin{cases} \hat{x}_j & j \neq i \\ \hat{x}_i - \epsilon & j = i \end{cases} \\ X_{uv}^{(1)} &= \begin{cases} \hat{X}_{uv} & u, v \neq i \\ \hat{x}_u x_v^{(1)} & v = i \end{cases} & X_{uv}^{(2)} &= \begin{cases} \hat{X}_{uv} & u, v \neq i \\ \hat{x}_u x_v^{(2)} & v = i \end{cases}\end{aligned}$$

- Since there is no "square term", $X^{(\cdot)}$ perturbs linearly with perturbation of one component of $x^{(\cdot)}$.
- So $(\hat{X}, \hat{x}) = 0.5 \cdot (X^{(1)}, x^{(1)}) + 0.5 \cdot (X^{(2)}, x^{(2)})$, which is the required contradiction.



Consequence: Can use IP technology to obtain better convexification of QCQP!

$$\begin{aligned}(\text{Lifted QCQP}) : \min & A_0 \cdot X + a_0^T x \\ \text{s.t. } & A_k \cdot X + a_k^T x \leq b_k \quad k = 1, \dots, K \\ & 0 \leq x \leq 1 \\ & X = xx^T\end{aligned}$$

McCormick Inequalities

McCormick Inequalities:

$$x_i + x_j - 1 \leq X_{ij} \tag{6.6}$$

$$0 \leq X_{ij} \tag{6.7}$$

$$X_{ij} \leq x_i \tag{6.8}$$

$$X_{ij} \leq x_j \tag{6.9}$$

Apart from the McCormick inequalities we can find more inequalities that are valid for BQP.

Triangle Inequalities

Triangle Inequalities Padberg1989:

$$x_i + x_j + x_k - X_{ij} - X_{ik} - X_{jk} \leq 1 \quad (6.10)$$

$$-x_i + X_{ij} + X_{ik} - X_{jk} \leq 0 \quad (6.11)$$

$$-x_j + X_{ij} - X_{ik} + X_{jk} \leq 0 \quad (6.12)$$

$$-x_k - X_{ij} + X_{ik} + X_{jk} \leq 0 \quad (6.13)$$

To see how these might be useful as cutting planes,

- Observe that fractional points with $x_i = x_j = x_k = 1/2$ and $X_{ij} = X_{ik} = X_{jk} = 0$ satisfy McCormick Envelope, but violate (6.10).
- Similarly, fractional points with $x_i = x_j = x_k = X_{ij} = X_{ik} = 1/2$ and $X_{jk} = 0$ satisfy McCormick Envelope, but violate (??).

Exercise: Derive Triangle Inequalities

Hint: Combine McCormick Inequalities and use Chvatal-Gomory Cuts.

Clique Inequalities:

$$S \subseteq V, |S| \geq 3, 1 \leq \alpha \leq |S| - 2, \alpha \in \mathbb{Z}$$

$$\alpha x(S) - X(E(S)) \leq \frac{\alpha(\alpha+1)}{2}$$

The easiest way to derive them is to note that, given any integer s , we have $s(s+1) \geq 0$. Thus, for any $S \subseteq V_n$ and any integer s , we have

$$\left(\sum_{i \in S} x_i - s \right) \left(\sum_{i \in S} x_i - s - 1 \right) \geq 0.$$

Expanding this and re-arranging yields

$$(2s+1) \sum_{i \in S} x_i - \sum_{i \in S} x_i^2 \leq 2 \sum_{\{i,j\} \subseteq S} x_i x_j + s(s+1).$$

Linearising and dividing by two yields the clique inequalities:

$$s \sum_{i \in S} x_i \leq \sum_{\{i,j\} \subseteq S} y_{ij} + \binom{s+1}{2} \quad (S \subseteq V_n, s = 0, \dots, |S|-1).$$

Padberg showed that these induce facets when $|S| \geq 3$ and $1 \leq s \leq |S|-2$. Note that the clique inequalities (4.10) reduce to the triangle inequalities (4.8) when $|S|=3$ and $s=1$. Moreover, the inequalities (4.5) can be regarded as "degenerate" clique inequalities with $|S|=2$ and $s=1$. In a similar way, the nonnegativity inequalities $y_{ij} \geq 0$ can be regarded as "degenerate" clique inequalities with $|S|=2$ and $s=0$.

Cut Inequalities - padberg:

For any disjoint sets $S, T \subset V_n$, we have

$$\sum_{i \in S, j \in T} y_{ij} \leq \sum_{i \in T} x_i + \sum_{\{i,j\} \subseteq S} y_{ij} + \sum_{\{i,j\} \subseteq T} y_{ij} \quad (S, T \subseteq V_n, S \cap T = \emptyset).$$

Proof.

$$\left(\sum_{i \in S} x_i - \sum_{i \in T} x_i \right) \left(\sum_{i \in S} x_i - \sum_{i \in T} x_i - 1 \right) \geq 0.$$



They induce facets when $|S| \geq 1$ and $|T| \geq 2$. Note that the cut inequalities (4.11) reduce to the triangle inequalities (4.9) when $|S|=2$ and $|T|=1$. Moreover, the inequalities (4.3) and (4.4) can be regarded as "degenerate" cut inequalities with $|S|=|T|=1$.

Next, we observe that the arguments for proving the validity of the clique and cut inequalities can be easily generalised.

Other cuts:

Indeed, for any disjoint sets $S, T \subset V_n$ and any $s \in \mathbf{Z}$, we have

$$\left(\sum_{i \in S} x_i - \sum_{i \in T} x_i - s \right) \left(\sum_{i \in S} x_i - \sum_{i \in T} x_i - s - 1 \right) \geq 0.$$

Expanding and linearising yields

$$s \sum_{i \in S} x_i + \sum_{i \in S, j \in T} y_{ij} \leq (s+1) \sum_{i \in T} x_i + \sum_{\{i,j\} \subseteq S} y_{ij} + \sum_{\{i,j\} \subseteq T} y_{ij} + \binom{s+1}{2}.$$

These inequalities, which include all those mentioned so far, have been rediscovered many times (e.g., [9, 15, 21, 69]). They define facets when $|S| + |T| \geq 3$ and $1 - |T| \leq s \leq |S| - 2$. We remark that they can also be derived by taking the clique inequality (4.10), and switching on T .

An even larger family of valid inequalities was found by Boros and Hammer [9].

Boros and Hammer:

Take an arbitrary vector $\mathbf{v} \in \mathbb{Z}^n$ and integer s , and consider the quadratic inequality $(\mathbf{v}^T \mathbf{x} - s)(\mathbf{v}^T \mathbf{x} - s - 1) \geq 0$. Expanding and linearising yields:

$$\sum_{i \in V_n} v_i (2s + 1 - v_i) x_i \leq 2 \sum_{1 \leq i < j \leq n} v_i v_j y_{ij} + s(s+1)$$

Although the Boros-Hammer inequalities are infinite in number, it is known that they define a polytope [45]. That is, a finite number of them dominate all the others. At the time of writing, however, a necessary and sufficient condition for a BorosHammer inequality to define a facet of BQP_n is not known.

We remark that switching a Boros-Hammer inequality is remarkably easy. Indeed, to switch on a set $S \subset V_n$, it suffices to change the sign of v_i for all $i \in S$.

Still more valid inequalities for BQP_n can be derived from a connection between BQP_n and the cut polytope. This is explained in the next section.

The Cut Polytope

As before, let $K_n = (V_n, E_n)$ denote the complete graph on n nodes. Given any set $S \subseteq V_n$, we let $\delta(S)$ denote the set of edges in E_n that have exactly one end-node in S . The set $\delta(S)$ is called an edge-cutset or simply cut. Given an integer $n \geq 3$ and a weight $w_e \in \mathbf{Q}$ for all $e \in E_n$, the max-cut problem calls for a cut of maximum total weight.

It is well-known (e.g., [7, 14]) that any QUBO instance with n variables can be converted into a max-cut instance with $n + 1$ variables, and vice-versa. This result turns out to have a polyhedral counterpart. Before explaining this, we first present the standard 0-1 LP formulation of the max-cut problem.

For all $e \in E_n$, let z_e be a binary variable, taking the value 1 if and only if e belongs to the cut. The max-cut problem can be formulated as:

$$\max \quad \sum_{e \in E_n} w_e z_e \tag{6.14}$$

$$\text{s.t.} \quad z_{ij} + z_{ik} + z_{jk} \leq 2 (\{i, j, k\} \subseteq V_n) \tag{6.15}$$

$$z_{ij} - z_{ik} - z_{jk} \leq 0 (\{i, j\} \in E_n, k \in V_n \setminus \{i, j\}) \tag{6.16}$$

$$\mathbf{z} \in \{0, 1\}^{\binom{n}{2}}. \tag{6.17}$$

The constraints (6.15), (6.16) are (somewhat confusingly) also called triangle inequalities.

Theorem 6.12: BQP equivalence with Cut polytope

Let $\mathbf{x}^* \in \mathbf{R}^n$ and $\mathbf{y}^* \in \mathbf{R}^{\binom{n}{2}}$ as follows:

$$\begin{aligned} z_{i,n+1}^* &= x_i^* \quad (i \in V_n) \\ z_{ij}^* &= x_i^* + x_j^* - 2y_{ij}^* \quad (\{i,j\} \in E_n). \end{aligned}$$

Then $(\mathbf{x}^*, \mathbf{y}^*) \in \text{BQP}_n$ if and only if $\mathbf{z} \in \text{CUT}_{n+1}$.

The linear transformation in Theorem 6.12 has come to be known as the covariance map [21]. A consequence of Theorem 4.1 is that the inequality $\alpha^T z \leq \beta$ is valid for CUT_{n+1} if and only if the inequality

$$\sum_{i \in V_n} \left(\sum_{j \in V_{n+1} \setminus \{i\}} \alpha_{ij} \right) x_i - 2 \sum_{e \in E_n} \alpha_e y_e \leq \beta$$

is valid for BQP_n . This enables one to easily convert valid (or facet-defining) inequalities for the cut polytope into valid (or facet-defining) inequalities for the Boolean quadric polytope, and vice-versa.

6.4.5. Reformulation Linearization Technique (RLT)

The McCormick envelope inequalities are in fact a specific type of RLT inequality. In particular, for any two valid inequalities

$$a^1 x \leq b_1$$

$$a^2 x \leq b_2$$

the inequality

$$(a^1 x - b_1)(a^2 x - b_2) \geq 0$$

is also a valid inequality. Using the lifted variables this could be rewritten as

$$\Gamma \circ Y + \beta^\top x + \alpha \geq 0$$

which is a valid linear inequality for $\mathcal{F}_{\text{lift}}$. This inequality is referred to as an *RLT inequality*.

Resources

Sherali and Alameddine 1992 - RLT for bilinear problems

6.4.6. Boolean Quadric Polytope (BQP)

In this section, we assume that the upper and lower bounds are normalized, that is,

$$0 \leq x \leq 1.$$

This can be achieved easily by an affine transformation of the problem. Alternatively, one can extract generalizations of the results in this section to arbitrary lower and upper bounds via an affine transformation in the reverse direction.

Boolean Quadric Polytope (BQP):

$$\text{BQP} = \text{conv} \left((x, Y) \in \{0, 1\}^{n+E} \mid Y_{ij} = x_i x_j \quad \forall (i, j) \in E \right) \quad (6.18)$$

Theorem 6.13

$$\text{BQP} = \text{conv} \left((x, Y) \in [0, 1]^{n+E} \mid Y_{ij} = x_i x_j \quad \forall (i, j) \in E \right) \quad (6.19)$$

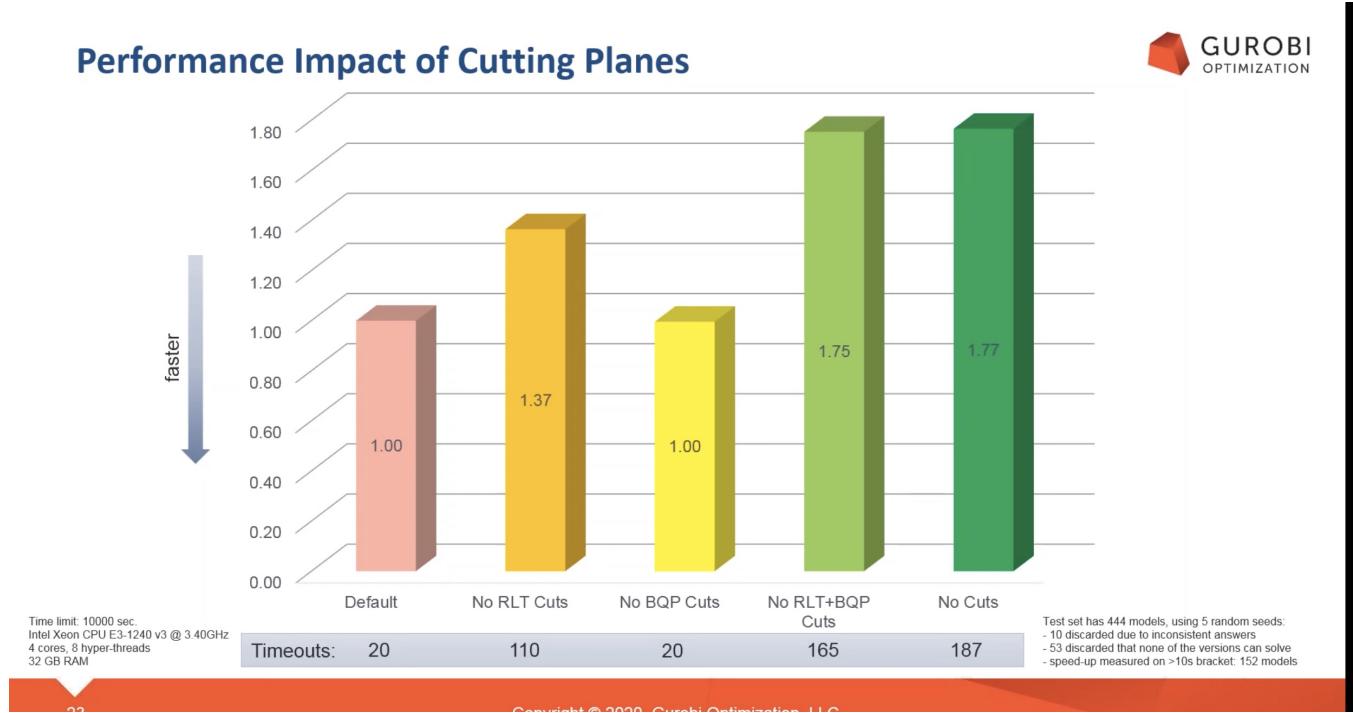
Cutting Planes for Mixed Bilinear Programs



All MILP cutting planes apply

Special cuts for bilinear constraints

- RLT Cuts
 - Reformulation Linearization Technique (Sherali and Adams, 1990)
 - multiply linear constraints with single variable, linearize resulting product terms
 - very powerful for bilinear programs, also helps a bit for convex MIQCPs and MILPs
- BQP Cuts
 - facets from Boolean Quadric Polytope (Padberg 1989)
 - equivalent to Cut Polytope
 - currently implemented: triangle inequalities (special case of Padberg's clique cuts for BQP)
- PSD Cuts
 - tangents of PSD cone defined by $Z = xx^T$ relationship: $Z - xx^T \succeq 0$ (Sherali and Fraticelli, 2002)
 - not yet implemented in Gurobi



Resources

<https://arxiv.org/pdf/2009.11674.pdf> - BQP for bilinear Problems
 Letchford 2022 - Fantastic Survey on BQP

Solving Box-Constrained Nonconvex Quadratic Programs - Pierre Bonami · Oktay Günlük · Jeff Linderoth

- $\{0, \frac{1}{2}\}$ Chvatal-Gomory cuts for BQP recently used successfully by [Bonami, Günlük, Linderoth (2018)]

$$BQP := \{(X, x) \mid X_{ij} \geq 0, X_{ij} \geq x_i + x_j - 1, X_{ij} \leq x_i, X_{ij} \leq j \forall (i, j) \in [n], x \in \{0, 1\}^n\}$$

We consider the non-convex quadratic programming problem with box constraints

$$\text{BoxQP: } \begin{aligned} z_{\text{BOXQP}} := \min \quad & 1/2x^T Qx + c^T x \\ \text{s.t.} \quad & u \geq x \geq \ell, \end{aligned}$$

where $Q \in \mathbb{R}^{n \times n}$ is a symmetric matrix. We assume that the bounds on x are finite, so we may without loss of generality assume that $\ell = 0, u = 1$.

A standard mechanism for obtaining a convex relaxation of BoxQP begins with the construction of an extended formulation. Let N denote the set $\{1, 2, \dots, n\}$ and let

$$E := \{(i, j) \in N \times N \mid i \neq j \text{ and } Q_{ij} \neq 0\}$$

consist of the indices of the non-zero entries in Q .

Consider the optimization problem

$$\begin{aligned} \text{EBoxQP: } & \min \sum_{\{i,j\} \in E} Q_{ij} X_{ij} + 1/2 \sum_{i \in N} Q_{ii} Y_i + \sum_{i \in N} c_i x_i \\ \text{s.t. } & Y_i = x_i^2 \quad \forall i \in N \\ & X_{ij} = x_i x_j \quad \forall \{i,j\} \in E \\ & 1 \geq x_i \geq 0 \quad \forall i \in N. \end{aligned}$$

$$\mathcal{Q} := \left\{ (x, X, Y) \in \mathbb{R}^n \times \mathbb{R}^{|E|} \times \mathbb{R}^n \mid (3) - (5) \right\}$$

the nonlinear inequalities (3)-(4) with the McCormick inequalities, resulting in the set

$$\begin{aligned} \mathcal{M} := \left\{ (x, X, Y) \in \mathbb{R}^n \times \mathbb{R}^{|E|} \times \mathbb{R}^n \mid & x_i \geq Y_i \geq 2x_i - 1, \quad Y_i \geq 0 \quad \forall i \in N \\ & x_i \geq X_{ij}, \quad x_j \geq X_{ij} \geq x_i + x_j - 1 \quad \forall \{i,j\} \in E \\ & X_{ij} \geq 0 \quad \forall \{i,j\} \in E, \quad 1 \geq x_i \geq 0 \quad \forall i \in N \right\} \end{aligned}$$

Clearly $\mathcal{Q} \subseteq \mathcal{M}$, and consequently, the linear program

$$z_{\mathcal{M}} := \min \left\{ \sum_{\{i,j\} \in E} Q_{ij} X_{ij} + 1/2 \sum_{i \in N} Q_{ii} Y_i + \sum_{i \in N} c_i x_i \mid (x, X, Y) \in \mathcal{M} \right\}$$

gives a relaxation of BoxQP, $z_{\text{BOXQP}} \geq z_{\mathcal{M}}$.

Chvátal-Gomory cuts

Consider the feasible set of solutions to a generic integer program $P^I = P^{LP} \cap \mathbb{Z}^n$ where

$$P^{LP} = \{x \in \mathbb{R}^n \mid Ax \geq b\}$$

and the matrix A has m rows. For any non-negative vector $\alpha \in \mathbb{R}_+^m$, the inequality $\alpha^T A x \geq \alpha^T b$ is satisfied by all feasible solutions of P^{LP} . Furthermore, if $\alpha^T A \in \mathbb{Z}^n$ then the strengthened inequality

$$\alpha^T A x \geq \lceil \alpha^T b \rceil$$

is also satisfied by all feasible solutions of P^I . This inequality is called a Gomory fractional cut [21], or, Chvátal-Gomory cut [17].

In the special case when $\alpha \in \{0, 1/2\}^m$, Inequality (12) is called a 0-1/2 cut [16].

If a Chvátal-Gomory (or, 0-1/2) cut can be expressed as a positive combination of a valid inequality for P^{LP} and another Chvátal-Gomory (or, 0-1/2) cut, then we say that the first cut is dominated by the second.

A cut is called non-dominated, if there is no cut dominating it. We next describe the well-known odd cycle inequalities for BQP and relate them to 0-1/2 cuts.

Odd Cycle Inequalities for BQP

Consider the McCormick formulation of $\text{BQP} = \text{conv}(\text{BQP}^{LP} \cap \mathbb{Z}^n \times \mathbb{Z}^{|E|})$ where

$$\text{BQP}^{LP} = \left\{ (x, X) \in \mathbb{R}^n \times \mathbb{R}^{|E|} \mid \min \{x_i, x_j\} \geq X_{ij} \geq \max \{0, x_i + x_j - 1\} \forall \{i, j\} \in E \right\}.$$

Notice that the McCormick inequalities in the description of BQP^{LP} imply that $x_j \geq x_i + x_j - 1$. Consequently, $1 \geq x_i$ for all $i \in N$, and all variables are bounded between zero and one. Also note that the following inequalities are implied by McCormick inequalities for any $\{i, j\} \in E$:

$$\begin{aligned} 2X_{ij} - x_i - x_j &\geq -1, \\ -2X_{ij} + x_i + x_j &\geq 0. \end{aligned}$$

Let $\{i, j\}, \{j, k\}, \{k, i\} \in E$ be given and consider adding up inequalities $(A_{ij}), (A_{jk}),$ and (A_{ki}) associated with these indices and dividing the resulting inequality by 2 :

$$X_{ij} + X_{jk} + X_{ki} - x_i - x_j - x_k \geq -\frac{3}{2}.$$

As all variables above are integral, taking the ceiling of the right hand side leads to the $0 - 1/2$ cut

$$X_{ij} + X_{jk} + X_{ki} - x_i - x_j - x_k \geq -1.$$

Similarly, combining inequalities $(A_{ij}), (B_{jk}),$ and (B_{ki}) with weights $1/2$ gives

$$X_{ij} - X_{jk} - X_{ki} + x_k \geq 0.$$

Odd Cycle Inequalities

Combining an odd number of inequalities of type (A_{ij}) with inequalities of type (B_{ij}) yields more general valid inequalities.

More precisely, given $E^A, E^B \subseteq E$ such that $|E^A|$ is odd and $E^A \cup E^B$ gives a simple cycle of the graph $G = (N, E)$, let $N^A \subseteq N$ denote the nodes that are incident to exactly two edges in E^A , and let $N^B \subseteq N$ denote the set of nodes that are incident to exactly two edges in E^B .

Combining inequalities (A_{ij}) for $\{i, j\} \in E^A$ with inequalities (B_{ij}) for $\{i, j\} \in E^B$ and rounding up the right hand side leads to the valid inequality

$$\sum_{i \in N^B} x_i - \sum_{i \in N^A} x_i - \sum_{\{i, j\} \in E^B} X_{ij} + \sum_{\{i, j\} \in E^A} X_{ij} \geq \left\lceil -\frac{|E^A|}{2} \right\rceil$$

which is called an odd-cycle inequality.

Exercise: Draw Odd Cycles

The odd cycle inequality is dominated by another odd cycle inequality unless $E^A \cup E^B$ gives a chordless simple cycle of the graph $G = (N, E)$.

Consequently, the only non-dominated odd cycle inequalities for BQP⁺ are the triangle inequalities as the underlying graph $G = (N, E^+)$ does not have any other chordless cycles.

Furthermore, Boros, Crama, and Hammer [8] have shown that for BQP⁺ the triangle cuts dominate, and therefore are equally as strong as, Chvátal-Gomory cuts. In the next subsection, we show that for BQP the odd cycle inequalities dominate, and therefore are equally as strong as, Chvátal-Gomory cuts.

Theorem 6.14

All non-dominated 0-1/2 cuts for the BQP are odd cycle inequalities.

Theorem 6.15

All non-dominated Chvátal-Gomory cuts for the BQP are 0 – 1/2 cuts.

Extended formulations

Tight compact extended relaxations for nonconvex quadratic programming problems with box constraints
- Sven de Vries & Bernd Perscheid

Extended Formulations

Extended formulations for convex hulls of some bilinear functions - Akshay Gupte, Thomas Kalinowski, Fabian Rigterink, Hamish Waterer

6.5 Exercises**Exercise 1 : Triangle Inequalities**

Derive the triangle inequalities of the BQP using the McCormick Inequalities and Gomory-Chvatal cuts.

Exercise 2 : MINLP Standard Form

Consider the optimization problem

$$\min \cos(e^{x+y+z^2} - z^3) \quad (6.1)$$

$$\text{s.t. } 0 \leq x, y, z \leq 1 \quad (6.2)$$

1. Write the problem in standard form.
2. Write a Directed Acyclic Graph to describe the decomposition into standard form.

Exercise 3 : Convex Envelopes

Consider a separable function $f : \mathbb{R}^{n_1} \times \dots \times \mathbb{R}^{n_t} \rightarrow \mathbb{R}$ defined as a sum $f := \sum_{i=1}^t f_i$, where each $f_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}$ operates on orthogonal subspaces.

1. Prove that the convex envelope of a separable function f over the domain $X := X_1 \times \dots \times X_t$ can be decomposed into a sum of convex envelopes, formally expressed as:

$$\text{conv}_X(f) = \sum_{i=1}^t \text{conv}_{X_i}(f_i).$$

2. Provide an example of a separable function f and its domain X , and calculate its convex envelope using the decomposition property.

Exercise 4 : RLT and Projection

Consider the following mixed-integer 0-1 constraint region:

$$X = \{(x, y) : x + y \leq 2, -x + y \leq 1, 2x - 2y \leq 1, x \in \{0, 1\}, y \geq 0\}$$

1. Lift this problem to a higher dimensional space by defining the new variable $w = xy$. Then add RLT inequalities by multiplying each constraint by x and $1 - x$ and linearizing it using the product $w = xy$ and the fact that $x = x^2$.
2. Use Fourier Motzkin Elimination to project out w so that you obtain a formulation in the original space of variables x and y .
3. Show that the projection is the convex hull of the mixed integer set in the original space.

7. Convex Hulls with Special Structures

7.1 Convex Hulls with Special Structures

(Lifted QCQP) : $\min A_0 \cdot X + a_0^T x$

$$\begin{aligned} \text{s.t. } & A_k \cdot X + a_k^T x \leq b_k \quad k = 1, \dots, K \\ & 0 \leq x \leq 1 \\ & X = xx^T \end{aligned}$$

- We have explored convex hull of set of the form:

$$Q := \left\{ (X, x) \in \mathbb{R}^{\frac{n(n-1)}{2}} \times \mathbb{R}^n \mid X_{ij} = x_i x_j \forall i, j \in [n], i \neq j, x \in [0, 1]^n \right\}$$

- Now we want to consider sets which includes the data, for example: A_k 's.

7.2 Convex hulls of structured sets

7.2.1. A packing-type bilinear knapsack set

Proposition 7.1: The convex-hull of packing-type bilinear set - Coppersmith, Günlük, Lee, Leung (1999))

Let

$$P := \left\{ (x, y) \in [0, 1]^n \times [0, 1]^n \mid \sum_{i=1}^n a_i x_i y_i \leq b \right\}$$

where $a_i \geq 0$ for all $i \in [n]$. Then

$$\text{conv}(P) := \left\{ (x, y) \mid \underbrace{\exists w, \sum_{i=1}^n a_i w_i \leq b,}_{\text{Relaxed McCormick envelope}} \underbrace{w_i, x_i, y_i \in [0, 1], w_i \geq x_i + y_i - 1,}_{\text{Relaxed McCormick envelope}} \forall i \in [n] \right\}.$$

- Convex hull is a polytope.
- Shows the power of McCormick envelopes.

Proof.

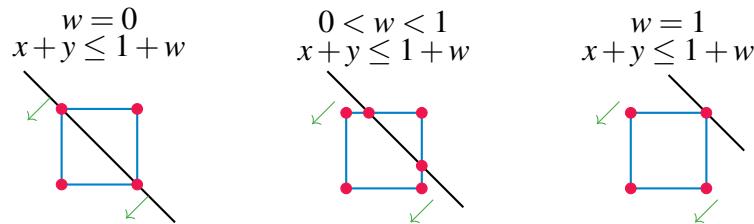
$$\text{conv}(P) := \underbrace{\text{Proj}_{x,y}(\left\{(x,y,w) \mid \begin{array}{l} \sum_{i=1}^n a_i w_i \leq b, \\ w_i, x_i, y_i \in [0,1], w_i \geq x_i + y_i - 1 \forall i \in [n] \end{array}\right\})}_R$$

Case \subseteq

- Observe $P \subseteq \text{Proj}_{x,y}(R) \Rightarrow \text{conv}(P) \subseteq \text{Proj}_{x,y}(R)$.

Case: $\text{conv}(P) \supseteq \text{Proj}_{x,y}(R)$:It is sufficient to prove that the (x,y) component of extreme points of R belong to P .Let $(\hat{w}, \hat{x}, \hat{y})$ be extreme point of R . For each i :

1. If $\hat{w}_i = 0$,
then $(\hat{x}_i, \hat{y}_i) \in \{(0,0), (0,1), (1,0)\}$, i.e. $\hat{x}_i \hat{y}_i = \hat{w}_i$.
2. If $0 < \hat{w}_i < 1$,
then $(\hat{x}_i, \hat{y}_i) \in \{(0,0), (0,1), (1,0), (1, \hat{w}_i), (\hat{w}_i, 1)\}$, i.e. $\hat{x}_i \hat{y}_i \leq \hat{w}_i$.
3. If $\hat{w} = 1$,
then $(\hat{x}_i, \hat{y}_i) \in \{(0,0), (1,0), (0,1), (1,1)\}$, i.e. $\hat{x}_i \hat{y}_i \leq \hat{w}_i$.

Thus, $\sum_{i=1}^n a_i \hat{x}_i \hat{y}_i \leq b$. ($\because a_i \geq 0 \forall i \in [n]$)**7.2.2. Product of a simplex and a polytope****A commonly occurring set**

$$S := \{(q, y, v) \in \mathbb{R}_+^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_1 n_2} \mid v_{ij} = q_i y_j \forall i \in [n_1], j \in [n_2], \underbrace{Ay \leq b}_{y \in P}, \underbrace{q \in \Delta}_{\sum_{i=1}^{n_1} q_i = 1}\}.$$

Some applications:

- Pooling problem ([Tawarmalani and Sahinidis (2002)])

- General substructure in "discretize NLPs" ([Gupte, Ahmed, Cheon, D. (2013)])
- Network interdiction ([Davarnia, Richard, Tawarmalani (2017)])

7.3 Dey recent results

BPP convex hulls and SOCP representability

7.4 To do - Add remaining Dey Slides

8. Perspective Reformulations

We will mainly look at these papers:

Resources

Perspective Reformulation and Applications - Oktay Günlük & Jeff Linderoth

Mixed-integer nonlinear programs featuring “on/off” constraints - Hassan Hijazi - Pierre Bonami - Gérard Cornuéjols - Adam Ouorou

a

b

Lecture notes on Fenchel Duality

On the convex hull of convex quadratic optimization problems with indicators - Linchuan Wei, Alper Atamtürk, Andrés Gómez & Simge Küçükyavuz

AN OUTER APPROXIMATION METHOD FOR SOLVING MIXED-INTEGER CONVEX QUADRATIC PROGRAMS WITH INDICATORS LINCHUAN WEI, SIMGE KÜCUKYAVU

Explicit convex hull description of bivariate quadratic sets with indicator variables - Antonio De Rosa and Aida Khajavirad

^aA note on quadratic constraints with indicator variables: Convex hull description and perspective relaxation - Andrés Gómez, Weijun Xie

^bOn the convex hull of convex quadratic optimization problems with indicators -

8.1 Perspective Reformulations

Convex optimization - Boyd and Vanderberghe

8.1.1. Perspective of a function

Definition 8.1: Perspective

If $f : \mathbf{R}^n \rightarrow \mathbf{R}$, then the perspective of f is the function $g : \mathbf{R}^{n+1} \rightarrow \mathbf{R}$ defined by

$$g(x, t) = tf(x/t)$$

with domain

$$\text{dom } g = \{(x, t) \mid x/t \in \text{dom } f, t > 0\}.$$

The perspective operation preserves convexity: If f is a convex function, then so is its perspective function g . Similarly, if f is concave, then so is g . This can be proved several ways, for example, direct verification of the defining inequality.

Exercise 8.2: Direct proof of perspective theorem

Give a direct proof that the perspective function g , of a convex function f is convex: Show that $\text{dom } g$ is a convex set, and that for $(x,t), (y,s) \in \text{dom } g$, and $0 \leq \theta \leq 1$, we have

$$g(\theta x + (1 - \theta)y, \theta t + (1 - \theta)s) \leq \theta g(x, t) + (1 - \theta)g(y, s).$$

Short proof using epigraphs:

Proof. For $t > 0$ we have

$$\begin{aligned} (x, t, s) \in \text{epig } g &\iff tf(x/t) \leq s \\ &\iff f(x/t) \leq s/t \\ &\iff (x/t, s/t) \in \text{epi } f. \end{aligned}$$

Therefore $\text{epi } g$ is the inverse image of $\text{epi } f$ under the perspective mapping that takes (u, v, w) to $(u, w)/v$. It follows (see §2.3.3 in Boyd – Vanderbergh) that $\text{epi } g$ is convex, so the function g is convex. ♠

Example 8.3: Euclidean norm squared

The perspective of the convex function $f(x) = x^T x$ on \mathbf{R}^n is

$$g(x, t) = t(x/t)^T (x/t) = \frac{x^T x}{t},$$

which is convex in (x, t) for $t > 0$.

We can deduce convexity of g using several other methods. First, we can express g as the sum of the quadratic-over-linear functions x_i^2/t , which were shown to be convex in §3.1.5. We can also express g as a special case of the matrix fractional function $x^T(tI)^{-1}x$ (see example 3.4).

Example 8.4: Negative logarithm

Consider the convex function $f(x) = -\log x$ on \mathbf{R}_{++} . Its perspective is

$$g(x, t) = -t \log(x/t) = t \log(t/x) = t \log t - t \log x,$$

and is convex on \mathbf{R}_{++}^2 . The function g is called the relative entropy of t and x . For $x = 1$, g reduces to the negative entropy function.

From convexity of g we can establish convexity or concavity of several interesting related functions. First, the relative entropy of two vectors $u, v \in \mathbf{R}_{++}^n$, defined as

$$\sum_{i=1}^n u_i \log(u_i/v_i)$$

is convex in (u, v) , since it is a sum of relative entropies of u_i, v_i .

Lemma 8.5

Suppose $f : \mathbf{R}^m \rightarrow \mathbf{R}$ is convex, and $A \in \mathbf{R}^{m \times n}, b \in \mathbf{R}^m, c \in \mathbf{R}^n$, and $d \in \mathbf{R}$. We define

$$g(x) = (c^T x + d) f((Ax + b)/(c^T x + d)),$$

with

$$\text{dom } g = \left\{ x \mid c^T x + d > 0, (Ax + b)/(c^T x + d) \in \text{dom } f \right\}.$$

Then g is convex.

8.2 Perspective Reformulation and Applications - Oktay Günlük & Jeff Linderoth

O. Günlük and J. Linderoth. Perspective reformulations of mixed integer nonlinear programs with indicator variables. Mathematical Programming, 124:183-205, 2010.

Definition 8.6: Perspective Function

Let $g : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^m$ be the perspective mapping of f defined as

$$g(\lambda, x) = \begin{cases} \lambda f(x/\lambda) & \text{if } \lambda > 0 \\ 0 & \text{if } \lambda = 0 \\ \infty & \text{otherwise} \end{cases}$$

Example 8.7: Separable quadratic UFL

The Separable Quadratic Uncapacitated Facility Location Problem (SQUFL) was introduced by

Günlük et al. [20].

$$\begin{aligned} & \min \sum_{i \in M} c_i z_i + \sum_{i \in M} \sum_{j \in N} q_{ij} x_{ij}^2 \\ & \text{subject to } x_{ij} \leq z_i \quad \forall i \in M, \forall j \in N, \\ & \sum_{i \in M} x_{ij} = 1 \quad \forall j \in N, \\ & z_i \in \{0, 1\}, \quad x_{ij} \geq 0 \quad \forall i \in M, \forall j \in N. \end{aligned}$$

To apply the perspective formulation, auxiliary variables y_{ij} are used to replace the terms x_{ij}^2 in the objective function. In addition the following constraints

$$\begin{aligned} x_{ij}^2 - y_{ij} &\leq 0 \quad \forall i \in M, j \in N \\ y_{ij} &\leq z_i \quad \forall i \in M, j \in N \end{aligned}$$

are added. In this reformulation, if $z_i = 0$, then $x_{ij} = y_{ij} = 0 \forall j \in N$, while if $z_i = 1$, the constraints 10 define the set of feasible points. Therefore, we can strengthen the formulation using the perspective counterparts of constraints 10

$$x_{ij}^2 - z_i y_{ij} \leq 0 \quad \forall i \in M, \forall j \in N$$

- A generic indicator-induced $\{0 - 1\}$ -MINLP is defined as:

$$z^* \stackrel{\text{def}}{=} \min_{(x,z) \in X \times (Z \cap \{0,1\}^{|I|})} \left\{ c^T x + d^T z \mid g_j(x, z) \leq 0 \forall j \in M, (x_{V_i}, z_i) \in S_i \forall i \in I \right\},$$

where:

- z are the indicator variables.
- x are the continuous variables.
- x_{V_i} denotes the collection of continuous variables $(x_j, j \in V_i)$ controlled by the indicator variable z_i .
- Sets may intersect, i.e., for some $i \neq j$, $V_i \cap V_j \neq \emptyset$.
- Sets $X \subseteq \mathbb{R}^n$ and $Z \subseteq \mathbb{R}^{|I|}$ are polyhedral sets of appropriate dimension.
- S_i is the set of points that satisfy all constraints associated with the indicator variable z_i :

$$S_i \stackrel{\text{def}}{=} \left\{ (x_{V_i}, z_i) \in \mathbb{R}^{|V_i|} \times \{0, 1\} \mid \begin{array}{ll} x_{V_i} = \hat{x}_{V_i} & \text{if } z_i = 0 \\ x_{V_i} \in \Gamma_i & \text{if } z_i = 1 \end{array} \right\}$$

where:

- $\Gamma_i \stackrel{\text{def}}{=} \left\{ x_{V_i} \in \mathbb{R}^{|V_i|} \mid f_j(x_{V_i}) \leq 0 \quad \forall j \in C_i, u_k \geq x_k \geq \ell_k \forall k \in V_i \right\}$ is bounded for all $i \in I$.
- $z_i \in \{0, 1\}$ for all $i \in I$.
- The objective function is assumed to be linear without loss of generality. If necessary, an additional variable can be used to move the nonlinearity from the objective function to the constraint set.

- This paper studies the convex hull description of the sets S_i when Γ_i is a convex set. An important observation is that Γ_i can be convex even when some functions f_j defining the set are non-convex.
- The convex hull of S_i , denoted as

$$S_i^c = \text{conv}(S_i)$$

, allows for a "tight" continuous relaxation of the original problem:

$$z^{PR} \stackrel{\text{def}}{=} \min_{(x,z) \in X \times Z} \left\{ c^T x + d^T z \mid g_j(x, z) \leq 0 \forall j \in M, (x_{V_i}, z_i) \in S_i^c \quad \forall i \in I \right\},$$

where S_i is replaced by its convex hull S_i^c . This formulation is referred to as the perspective relaxation of the original problem.

- A convex relaxation of S_i , when all f_j are convex and bounded for $j \in C_i$, can be obtained as follows:

$$\begin{aligned} S_i^R \stackrel{\text{def}}{=} \left\{ x_{V_i} \in \mathbb{R}^{|V_i|} \mid f_j(x_{V_i}) \leq (1 - z_i) f_j(\hat{x}_{V_i}) \quad \forall j \in C_i, \right. \\ \left. u_k z_i \geq x_k - (1 - z_k) \hat{x}_{V_i} \geq \ell_k z_i \quad \forall k \in V_i \right\} \end{aligned}$$

This leads to the natural continuous relaxation of the original problem:

$$z^{NR} \stackrel{\text{def}}{=} \min_{(x,z) \in X \times Z} \left\{ c^T x + d^T z \mid g_j(x, z) \leq 0 \forall j \in M, (x_{V_i}, z_i) \in S_i^R \forall i \in I \right\}$$

where S_i is replaced with S_i^R . Since S_i^R is convex and contains S_i , we have $S_i^c \subseteq S_i^R$ for all $i \in I$, thus:

$$z^* \geq z^{PR} \geq z^{NR}.$$

- The perspective relaxation is effective as it leads to an efficient computational approach and provides a good approximation of z^* , especially when it can be solved efficiently and S_i^c is the smallest convex set containing S_i .

8.2.1. A quadratic set with variable bounds

The purpose of this section is to present a convex hull description of the set:

$$Q = \left\{ (w, x, z) \in \mathbb{R}^{n+1} \times \{0, 1\}^n : w \geq \sum_{i=1}^n q_i x_i^2, \quad u_i z_i \geq x_i \geq l_i z_i, i \in I \right\},$$

where $I = \{1, \dots, n\}$ and $q, u, l \in \mathbb{R}_+^n$.

A low dimensional analogue

Let

$$S = \{(x, y, z) \in \mathbb{R}^2 \times \{0, 1\} : y \geq x^2, uz \geq x \geq lz, x \geq 0\},$$

where $u, l \in \mathbb{R}$. Define

$$S^c = \{(x, y, z) \in \mathbb{R}^3 : yz \geq x^2, uz \geq x \geq lz, 1 \geq z \geq 0, x, y \geq 0\}.$$

Lemma 8.8

$$\text{conv}(S) = S^c.$$

Proof.

- Begin with the decomposition of S into S^0 and S^1 :
 - $S^0 = \{(0, y, 0) \in \mathbb{R}^3 : y \geq 0\}$ represents points with $z = 0$.
 - $S^1 = \{(x, y, 1) \in \mathbb{R}^3 : y \geq x^2, u \geq x \geq l, x \geq 0\}$ represents points with $z = 1$ and additional constraints on x and y .

Since both S^0 and S^1 are subsets of S^c , and S^c is convex, it follows that $\text{conv}(S) \subseteq S^c$.

- Consider any point $\bar{p} = (\bar{x}, \bar{y}, \bar{z})$ in S^c and analyze it in two cases based on the value of \bar{z} :
 - **Case 1:** If $\bar{z} = 0$, then $\bar{p} = (0, \bar{y}, 0)$ where $\bar{y} \geq 0$. This implies $\bar{p} \in S^0$.
 - **Case 2:** If $\bar{z} \neq 0$, we decompose \bar{p} as $\bar{p} = p' + d$ where:

$$* p' = (\bar{x}, \bar{x}^2/\bar{z}, \bar{z}) \in S^c.$$

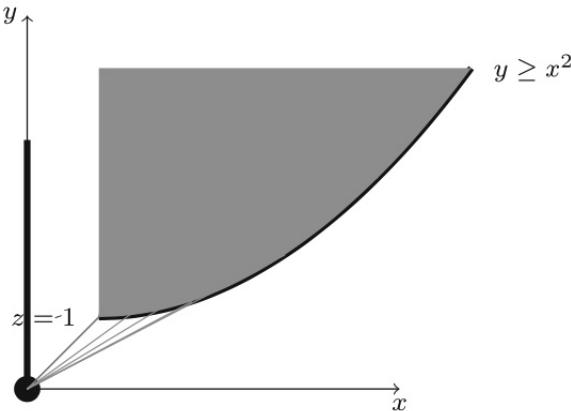
$$* d = (0, \bar{y} - \bar{x}^2/\bar{z}, 0) \geq 0.$$

* p' can be further expressed as a convex combination of $p_0 = (0, 0, 0) \in S^0$ and $p_1 = (\bar{x}/\bar{z}, \bar{x}^2/\bar{z}^2, 1) \in S^1$, i.e., $p' = (1 - \bar{z})p_0 + \bar{z}p_1$.

Given $1 \geq \bar{z} \geq 0$, $p' \in \text{conv}(S)$. Since $(0, 1, 0)$ is a direction of both S^0 and S^1 , it is also a direction of $\text{conv}(S)$, leading to $\bar{p} \in \text{conv}(S)$.

- Therefore, combining the two cases above, we conclude $S^c \subseteq \text{conv}(S)$, completing the proof that S^c and $\text{conv}(S)$ are equivalent.



Fig. 1 The set S^c 

8.2.2. An extended formulation for Q

Consider the following extended formulation of Q

$$\bar{Q} \stackrel{\text{def}}{=} \left\{ (w, x, y, z) \in \mathbb{R}^{3n+1} : w \geq \sum_i q_i y_i, (x_i, y_i, z_i) \in S_i, \quad i = 1, 2, \dots, n \right\}$$

where S_i has the same form as the set S discussed in the previous section except the bounds u and l are replaced with u_i and l_i .

Note that if $(w, x, y, z) \in \bar{Q}$ then $(w, x, z) \in Q$, and therefore $\text{proj}_{(w,x,z)}(\bar{Q}) \subseteq Q$. On the other hand, for any $(w, x, z) \in Q$, letting let $y'_i = x_i^2$ gives a point $(w, x, y', z) \in \bar{Q}$. Therefore, \bar{Q} is indeed an extended formulation of Q , or, in other words, $Q = \text{proj}_{(w,x,z)}(\bar{Q})$.

Definition 8.9

Let $P \subset \mathbb{R}^n$ be a closed set and let $p \in P$.

1. p is called an extreme point of P if it can not be represented as $p = 1/2p_1 + 1/2p_2$ for $p_1, p_2 \in P, p_1 \neq p_2$. Set P is called pointed if it has extreme points.
2. P is called integral with respect to a subset of the indices $I \subseteq \{1, \dots, n\}$ if for any extreme point $p \in P, p_i \in \mathbb{Z}$ for all $i \in I$.

Lemma 8.10

For $i = 1, 2$ let $P_i \subset \mathbb{R}^{n_i}$ be a closed and pointed set which is integral with respect to indices I_i . Let $P' = \{(x, y) \in \mathbb{R}^{n_1+n_2} : x \in P_1, y \in P_2\}$.

1. P' is integral with respect to $I_1 \cup I_2$.
2. $\text{conv}(P') = \{(x, y) \in \mathbb{R}^{n_1+n_2} : x \in \text{conv}(P_1), y \in \text{conv}(P_2)\}$.

Lemma 8.11

Let $P \subset \mathbb{R}^n$ be a given closed, pointed set and let $P' = \{(w, x) \in \mathbb{R}^{n+1} : w \geq ax, x \in P\}$ where $a \in \mathbb{R}^n$.

1. If P is integral with respect to I , then P' is integral with respect to I .
2. $\text{conv}(P') = P''$ where $P'' = \{(w, x) \in \mathbb{R}^{n+1} : w \geq ax, x \in \text{conv}(P)\}$.

We are now ready to present the convex hull of \bar{Q} . Let

$$\bar{Q}^c = \left\{ (w, x, y, z) \in \mathbb{R}^{3n+1} : w \geq \sum_i q_i y_i, (x_i, y_i, z_i) \in S_i^c, i = 1, 2, \dots, n \right\}.$$

Lemma 8.12

The set \bar{Q}^c is integral with respect to the indices of z variables. Furthermore,

$$\text{conv}(\bar{Q}) = \bar{Q}^c.$$

Proof.

- Define the set D as:

$$D = \{(x, y, z) \in \mathbb{R}^{3n} : (x_i, y_i, z_i) \in S_i, i = 1, 2, \dots, n\},$$

which represents a collection of points where each triplet (x_i, y_i, z_i) belongs to the set S_i for $i = 1, 2, \dots, n$.

- The set \bar{Q} is defined as:

$$\bar{Q} = \left\{ (w, x, y, z) \in \mathbb{R}^{3n+1} : w \geq \sum_{i=1}^n q_i y_i, (x, y, z) \in D \right\},$$

where w is bounded below by the weighted sum of y_i components, with weights q_i , for all elements in D .

- By Lemma 8.11, the convex hull of \bar{Q} , denoted as \bar{Q}^c , is determined by replacing D with its convex hull. This operation implies a direct application of the convex hull operation to the description of \bar{Q} without altering the condition on w .
- According to Lemma 8.10, replacing D with its convex hull is equivalent to individually taking the convex hulls of each S_i set, i.e., replacing each S_i with $\text{conv}(S_i)$ in the description of D .
- Finally, by applying Lemma 8.11 again, it concludes that \bar{Q}^c is integral, indicating that the convex hull of \bar{Q} is a set of points with integer coordinates, provided the conditions specified in Lemma 8.11 are met.



8.2.2.1. Convex hull description in the original space

$$Q^c = \left\{ (w, x, z) \in \mathbb{R}^{2n+1} : w \prod_{i \in S} z_i \geq \sum_{i \in S} q_i x_i^2 \prod_{l \in S \setminus \{i\}} z_l, S \subseteq \{1, 2, \dots, n\} \right. \\ \left. u_i z_i \geq x_i \geq l_i z_i, \quad x_i \geq 0, \quad i = 1, 2, \dots, n \right\}$$

Lemma 8.13

$$Q^c = \text{proj}_{(w, x, z)} (\bar{Q}^c).$$

Exercise 8.14

Prove this lemma.

8.2.3. More things....

The convex hull of the union of a point and a convex set

We next extend the observations presented in Sect. 2 to describe the convex hull of a point $\bar{x} \in \mathbb{R}^n$ and a bounded convex set defined by analytic functions. In other words, using an indicator variable $z \in \{0, 1\}$, define $W^0 = \{(x, z) \in \mathbb{R}^{n+1} : x = \bar{x}, z = 0\}$, and

$$W^1 = \{(x, z) \in \mathbb{R}^{n+1} : f_i(x) \leq 0 \text{ for } i \in I, u \geq x - \bar{x} \geq l, z = 1\}$$

where $u, l \in \mathbb{R}_+^n$, and $I = \{1, \dots, t\}$. We are interested in the convex hull of $W = W^1 \cup W^0$. Clearly, both W^0 and W^1 are bounded and W^0 is a convex set. Furthermore, if W^1 is also convex then

$$\text{conv}(W) = \{p \in \mathbb{R}^{n+1} : p = \alpha p^1 + (1 - \alpha)p^0, p^1 \in W^1, p^0 \in W^0, 1 \geq \alpha \geq 0\}.$$

We next present a description of $\text{conv}(W)$ in the space of original variables. To simplify notation we assume that $\bar{x} = 0$ in the remainder of this section. Note that there is no loss of generality as this is an affine transformation. We next write the description of $\text{conv}(W)$ in open form

$$\text{conv}(W) = \{(x, z) \in \mathbb{R}^{n+1} : 1 \geq \alpha \geq 0, x^0 = 0, z^0 = 0, z^1 = 1 \\ x = \alpha x^1 + (1 - \alpha)x^0, \quad z = \alpha z^1 + (1 - \alpha)z^0, \\ f_i(x^1) \leq 0 \text{ for } i \in I, \quad u \geq x^1 - \bar{x} \geq l\}.$$

The additional variables used in this description can be projected out to obtain a description in the space of the original variables.

Lemma 8.15

If W^1 is convex, then $\text{conv}(W) = W^- \cup W^0$, where

$$W^- = \{(x, z) \in \mathbb{R}^{n+1} : f_i(x/z) \leq 0 \text{ for } i \in I, \quad uz \geq x \geq lz, 1 \geq z > 0\}$$

Proof. As x^0, z^0 and z^1 are fixed in (XF), it is possible to substitute out these variables. In addition, as $z = \alpha$ after these substitutions, we can eliminate α . Furthermore, as $x = \alpha x^1 = zx^1$, we can eliminate x^1 by replacing it with x/z provided that $z > 0$. If, on the other hand, $z = 0$, clearly $(x, 0) \in \text{conv}(W)$ if and only if $(x, 0) \in W^0$. ♠

We next show that W^0 is contained in the closure of W^- .

Lemma 8.16

For $1 \geq z > 0$, let $Q^c(z) = \{x \in \mathbb{R}^n : f_i(x/z) \leq 0 \text{ for } i \in I, \quad uz \geq x \geq lz\}$. If all $f_i(x)$ are bounded in $[l, u]$, then,

$$\lim_{z \rightarrow 0^+} Q^c(z) = \{x \in \mathbb{R}^n : x = 0\}$$

Proof. Let $\{z_k\} \subset (0, 1)$ be a sequence converging to 0. As, by definition, $Q^c(z) \neq \emptyset$ for $z \in (0, 1)$, there exists a corresponding sequence $\{x_k\}$ such that $x_k \in Q^c(z_k)$. Clearly, $uz \geq x_k \geq lz$ and therefore $\{x_k\}$ converges to 0. ♠

Combining the previous lemmas, we obtain the following result.

Corollary 8.17

$$\text{conv}(W) = \text{closure}(W^-).$$

We would like to emphasize that even when $f(x)$ is a convex function $f_i(x/z)$ may not be convex. However, for $z > 0$ we have

$$f_i(x/z) \leq 0 \Leftrightarrow z^t f_i(x/z) \leq 0$$

for any $t \in \mathbb{R}$. In particular, taking $t = 1$ gives $zf_i(x/z)$ which is known to be convex provided that $f(x)$ is convex. We discuss this further in Sect. 4.1. We also note that if $f(x)$ is SOCP-representable, then $zf_i(x/z)$ is also SOCP-representable and in particular, if W^1 is defined by SOCP-representable functions, then so is $\text{conv}(W)$. We will show the benefits of employing SOC solvers for (non-quadratic) SOC-representable sets in Sect. 5.2.

We next show that when all $f_i(x)$ that define W^1 are polynomial functions, convex hull of W can be described explicitly.

Lemma 8.18

Let $f_i(x) = \sum_{t=1}^{p_i} c_{it} \prod_{j=1}^n x_j^{q_{itj}}$ for all $i \in I$. Let $q_{it} = \sum_{j=1}^n q_{itj}$, $q_i = \max_t \{q_{it}\}$ and $\bar{q}_{it} = q_i - q_{it}$. If all $f_i(x)$ are convex and bounded in $[l, u]$, then $\text{conv}(W) = W^c$, where

$$W^c = \left\{ (x, z) \in \mathbb{R}^{n+1} : \sum_{t=1}^{p_i} c_{it} z^{\bar{q}_{it}} \prod_{j=1}^n x_j^{q_{itj}} \leq 0 \text{ for } i \in I, \right. \\ \left. zu \geq x \geq lz, 1 \geq z \geq 0 \right\}$$

Proof. Note that $f_i(x/z) = \sum_{t=1}^{p_i} c_{it} z^{-q_{it}} \prod_{j=1}^n x_j^{q_{itj}}$. Therefore, multiplying $f_i(x/z) \leq 0$ by z^{q_i} , one obtains the expression above. Clearly, $W^c \cap \{z > 0\} = W^-$ and $W^c \cap \{z = 0\} = W^0$. ♠

8.2.3.1. Other examples**Example 8.19: Network design with congestion constraints**

There is a set of commodities K to be shipped over a capacitated directed network $G = (N, A)$.

The capacity of arc $(i, j) \in A$ is u_{ij} , and each node $i \in N$ supplies or demands a specified amount b_i^k of commodity k .

There is a fixed cost c_{ij} of opening each arc $(i, j) \in A$, and we introduce $\{0, 1\}$ -variables z_{ij} to indicate whether $(i, j) \in A$ is opened.

The quantity of commodity k routed on arc (i, j) is measured by variable x_{ij}^k and $f_{ij} = \sum_{k \in K} x_{ij}^k$ denotes the total flow on the arc. A typical measure of the total weighted congestion (or queuing delay) is

$$\rho(f) \stackrel{\text{def}}{=} \sum_{(i,j) \in A} r_{ij} \frac{f_{ij}}{1 - f_{ij}/u_{ij}},$$

where $r_{ij} \geq 0$ is a user-defined weighting parameter for each arc. We use a decision variables y_{ij} to measure the contribution of the congestion on arc (i, j) to the total congestion $\rho(f)$. The network should be designed so as to keep the total queuing delay less than a given value β , and this is to be accomplished at minimum cost. The resulting optimization model (NDCC) can be written as

$$\min \sum_{(i,j) \in A} c_{ij} z_{ij} \quad (8.1)$$

subject to

$$\sum_{(j,i) \in A} x_{ij}^k - \sum_{(i,j) \in A} x_{ij}^k = b_i^k \quad \forall i \in N, \forall k \in K, \quad (8.2)$$

$$\sum_{k \in K} x_{ij}^k - f_{ij} = 0 \quad \forall (i,j) \in A, \quad (8.3)$$

$$f_{ij} \leq u_{ij} z_{ij} \quad \forall (i,j) \in A, \quad (8.4)$$

$$y_{ij} \geq \frac{r_{ij} f_{ij}}{1 - f_{ij}/u_{ij}} \quad \forall (i,j) \in A, \quad (8.5)$$

$$\sum_{(i,j) \in A} y_{ij} \leq \beta, \quad (8.6)$$

$$z_{ij} \in \{0, 1\} \quad \forall (i,j) \in A, \quad (8.7)$$

$$x_{ij}^k \in \mathbb{R}_+ \quad \forall (i,j) \in A, \forall k \in K, \quad (8.8)$$

$$y_{ij}, f_{ij} \in \mathbb{R}_+ \quad \forall (i,j) \in A. \quad (8.9)$$

Proposition 8.20

The congestion inequalities (8.5) can be written as SOC constraints.

Proof. Multiplying both sides of (8.5) by $1 - f_{ij}/u_{ij} > 0$, adding $r_{ij} f_{ij}^2$ to both sides of the inequality, and factoring the left-hand-side gives an equivalent constraint

$$(y_{ij} - r_{ij} f_{ij}) (u_{ij} - f_{ij}) \geq r_{ij} f_{ij}^2. \quad (8.10)$$

Because the inequalities $y_{ij} \geq r_{ij} f_{ij}$ and $u_{ij} \geq f_{ij}$ most hold in any feasible solution, (8.10) is precisely a constraint in rotated second-order cone form. ♠

Cut-set inequalities (see, [25,8]) are known to strengthen the continuous relaxation of network design problems. In our computational experiments, we used the following most basic and effective cut-set inequalities: Let δ_i denote the total flow originating from node i and $\tau_i \in \mathbb{Z}_+$ be such that $\sum_{i,j \in A'} u_{ij} < \delta_i$ for all $A' \subset A$ such that $|A'| \leq \tau_i - 1$. The associated cut-set inequality for node i is

$$\sum_{(i,j) \in A} z_{ij} \geq \tau_i$$

We added these inequalities for both incoming and outgoing arcs for all $i \in N$. More elaborate inequalities could also be added (see [8]), but our goal is to examine the impact of the perspective reformulation, not strong linear inequalities.

In the formulation of NDCC, note that if $z_{ij} = 0$, then the constraints (8.4) force $f_{ij} = 0$, and the constraints are redundant for the arc (i, j) .

However, if $z_{ij} = 1$, then the definitional constraint 14 for the corresponding y_{ij} must hold. Therefore, each constraint can be replaced by its perspective counterpart:

$$z_{ij} \left(\frac{r_{ij}f_{ij}/z_{ij}}{1 - f_{ij}/(u_{ij}z_{ij})} - \frac{y_{ij}}{z_{ij}} \right) \leq 0 \quad (8.11)$$

Proposition 8.21

The constraints (8.11) can also be written as second order cone constraints in a similar fashion to the non-perspective version (8.5).

Proof. Specifically, simplifying the left-hand size of the inequality (8.11), adding $r_{ij}f_{ij}^2$ to both sides of the simplified inequality and factoring gives the equivalent constraints

$$(y_{ij} - r_{ij}f_{ij})(u_{ij}z_{ij} - f_{ij}) \geq r_{ij}f_{ij}^2$$

which is a rotated second-order cone constraint since $y_{ij} \geq r_{ij}f_{ij}$ and $u_{ij}z_{ij} \geq f_{ij}$ for any feasible solution.



The fact that the inequalities in the perspective reformulation of (8.5) are SOC-representable is no surprise. In fact, Ben-Tal and Nemirovski [5] (Page 96, Proposition 3.3.2) show that the perspective transformation of a function whose epigraph is a SOC-representable set is (under mild conditions) always SOC-representable.

Example 8.22: Mean-variance optimization

In the problem, there is a set N of assets available for purchase. The expected return of asset $i \in N$ is given by α_i , and the covariance of the returns between every pair of assets is given in the form a positive-definite matrix $Q \in \mathbb{R}^{n \times n}$. The canonical problem is often augmented with a number of business rules that require the introduction of binary variables in straightforward optimization models.

For example, there may be minimum (ℓ_i) and maximum (u_i) buy-in thresholds for each asset $i \in N$, resulting the the following optimization problem (MVOBI):

$$\min \left\{ x^T Q x \mid e^T x = 1, \alpha^T x \geq \rho, \ell_i z_i \leq x_i \leq u_i z_i \forall i \in N \right\}, \quad (8.12)$$

where the decision variable x_i is the percentage of the portfolio invested in asset i and z_i is a binary variable indicating the purchase of asset i . Imposing a cardinality constraint on the number of different assets purchased can be achieved by adding a constraint $\sum_{i \in N} z_i \leq K$.

Unfortunately, direct application of the perspective reformulation to (8.12) is not possible, as the objective is not a separable function of the decision variables x .

However, in many practical applications, the covariance matrix is obtained from a factor model and has the form

$$Q = B\Omega B^T + \Delta^2,$$

for a given exposure matrix, $B \in \mathbb{R}^{n \times f}$, positive-definite factor-covariance matrix $\Omega \in \mathbb{R}^{f \times f}$, and positive definite, diagonal specific-variance matrix $\Delta \in \mathbb{R}^{n \times n}$ [29]. If a factor model is given, a separable portion of the objective function is easily extracted by introducing variables y_i , changing the objective to

$$\min x^T (B\Omega B^T) x + \sum_{i \in N} \Delta_{ii} y_i$$

and enforcing the constraints $y_i \geq x_i^2 \forall i \in N$.

Even if the covariance matrix Q does not directly have embedded diagonal structure from a factor model, then, as suggested by Frangioni and Gentile [14], it is still possible to extract a separable component from Q .

Specifically, the matrix Q may be decomposed into $Q = R + D$, for some positive, diagonal matrix D such that $R = Q - D$ remains positive-definite.

The objective can be changed to $\min x^T Rx + x^T Dx$, and $x^T Dx$ is separable in x . Frangioni and Gentile [14] suggest using $D = \lambda_n I$, where $\lambda_n > 0$ is the smallest eigenvalue of Q . In our computational experiments, we follow their advice and use $D = (\lambda_n - \varepsilon) I$, where $\varepsilon = 0.001$ so that R is strictly positive definite. In subsequent work, Frangioni and Gentile [15] show how "more" of the separable structure of Q can be extracted into D through the solution of a semidefinite program.

In order to solve the instance entirely in a second-order cone programming framework, we use the well-known transformation [5] of a convex quadratic program into a second order cone program. To transform the instance, a Cholesky factorization of $R = MM^T$, is taken, the auxiliary variables $w = M^T x$ are introduced, so that $\|w\| = x^T Rx$.

$$\min v + \sum_{i \in N} D_{ii} y_i \tag{8.13}$$

$$\text{subject to } w - M^T x = 0 \tag{8.14}$$

$$v - \|w\| \geq 0 \tag{8.15}$$

$$y_i - x_i^2 \geq 0 \quad \forall i \in N \tag{8.16}$$

$$\sum_{i \in N} x_i = 1 \tag{8.17}$$

$$\sum_{i \in N} \alpha_i x_i \geq \rho \tag{8.18}$$

$$\ell_i z_i \leq x_i \leq u_i z_i \quad \forall i \in N \tag{8.19}$$

The inequalities (8.15) can easily be placed in rotated second order cone form. Since $z_i = 0$ implies that constraint (8.16) is redundant, and, while $z_i = 1$ implies that we would like the inequality to hold, the perspective reformulation may be applied, replacing the constraints (8.16) with inequalities

$$y_i z_i - x_i^2 \geq 0 \quad \forall i \in N \tag{8.20}$$

The inequalities (8.20) are precisely in the rotated second order cone form, so they can be effectively handled by software such as Mosek.

1

Resources

Applications and algorithms for mixed integer nonlinear programming Sven Leyffer¹, Jeff Linderoth², James Luedtke², Andrew Miller³ and Todd Munson¹

Advanced Math Programming by Leo Liberti

Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming Theory, Algorithms, Software, and Applications

Mixed-Integer Non-Linear Programming survey website

<https://www.minplib.org/applications.html>

8.3 Convexification On Off

Mixed-integer nonlinear programs featuring “on/off” constraints - Hassan Hijazi, Pierre Bonami, Gérard Cornuéjols & Adam Ouorou **Hijazi2011**

Given convex functions

- $g : \mathbb{R}^{n+K} \rightarrow (\mathbb{R} \cup \{\infty\})^m$,
- $h : \mathbb{R}^{n+K} \rightarrow \mathbb{R} \cup \{\infty\}$,
- $f^k : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}, \forall k \in \{1, 2, \dots, K\}$,
- and vectors \mathbf{l} and $\mathbf{u} \in \mathbb{R}^n$

we are interested in optimization problems of the form:

$$\begin{aligned} \min \quad & h(\mathbf{x}, \mathbf{z}) \\ \text{s.t.} \quad & g(\mathbf{x}, \mathbf{z}) \leq 0, \\ & f^k(\mathbf{x}) \leq 0 \quad \text{if } z_k = 1, \forall k \in \{1, 2, \dots, K\}, \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \\ & \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{z} \in \{0, 1\}^K. \end{aligned}$$

Our main object is to deal with the constraints $f^k(\mathbf{x}) \leq 0$ if $z_k = 1$.

¹A note on quadratic constraints with indicator variables: Convex hull description and perspective relaxation - Andrés Gómez, Weijun Xie

Three ways to handle it

Approach 1. Transform the "on/off" constraints into:

$$z_k f^k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \dots, K\}.$$

In this case, even the continuous relaxation of (??) becomes non-convex.

Approach 2. Since all variables are bounded, another alternative is to use a big- M formulation that leads to a compact convex continuous relaxation. Unfortunately, these models often provide weak lower bounds.

Approach 3. Use disjunctive programming (see [4, 8, 13, 21]). For $k = 1, \dots, K$, let us define

$$\Gamma_0^k = \{(\mathbf{x}, z_k) \in \mathbb{R}^n \times \mathbb{B} : z_k = 0, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$$

and $\Gamma_1^k = \{(\mathbf{x}, z_k) \in \mathbb{R}^n \times \mathbb{B} : z_k = 1, f^k(\mathbf{x}) \leq 0, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$ (\mathbb{B} denotes the set $\{0, 1\}$).

Problem (1) can be reformulated as a disjunctive program:

$$\begin{aligned} \min \quad & h(\mathbf{x}, \mathbf{z}) \\ \text{s.t.} \quad & g(\mathbf{x}, \mathbf{z}) \leq 0, \\ & (\mathbf{x}, z_k) \in \Gamma_0^k \cup \Gamma_1^k, \quad \forall k \in \{1, 2, \dots, K\}, \\ & \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{z} \in \{0, 1\}^K. \end{aligned}$$

Taking the convex hull of each union $\Gamma_0^k \cup \Gamma_1^k$ separately, (2) can be rewritten as:

$$\begin{aligned} \min \quad & h(\mathbf{x}, z) \\ \text{s.t.} \quad & g(\mathbf{x}, \mathbf{z}) \leq 0, \\ & (\mathbf{x}, z_k) \in \text{conv}(\Gamma_0^k \cup \Gamma_1^k), \quad \forall k \in \{1, 2, \dots, K\}, \\ & \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{z} \in \{0, 1\}^K. \end{aligned}$$

Convex hull of $\Gamma_0 \cup \Gamma_1$

We start by studying a simple example with one "on/off" constraint (i.e. $K = 1$) in \mathbb{R}^3 :

$$\begin{aligned} \min \quad & h(\mathbf{x}, z) \\ \text{s.t.} \quad & (\mathbf{x}, z) \in \text{conv}(\Gamma_0 \cup \Gamma_1), \\ & \Gamma_0 = \{(\mathbf{x}, z) \in \mathbb{R}^2 \times \mathbb{B} : z = 0, l_1 \leq x_1 \leq u_1, l_2 \leq x_2 \leq u_2\}, \\ & \Gamma_1 = \{(\mathbf{x}, z) \in \mathbb{R}^2 \times \mathbb{B} : z = 1, f(\mathbf{x}) \leq 0, l_1 \leq x_1 \leq u_1, l_2 \leq x_2 \leq u_2\}, \end{aligned}$$

where $f(\mathbf{x}) = \frac{1}{c_1 - x_1} + \frac{1}{c_2 - x_2} - d, u_1 \leq c_1$ and $u_2 \leq c_2$.

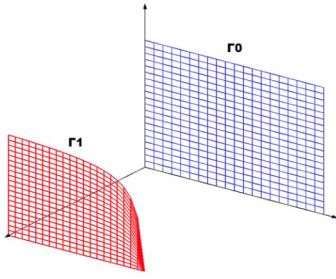
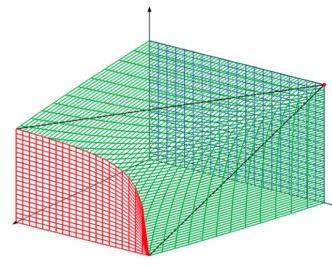
Fig. 1 The sets Γ_0 and Γ_1 **Fig. 2** $\text{conv}(\Gamma_0 \cup \Gamma_1)$ 

Figure 2 gives a representation of $\text{conv}(\Gamma_0 \cup \Gamma_1)$ in \mathbb{R}^3 . An interesting remark is that this convex hull can be derived using perspective functions. The perspective function $\tilde{f} : \mathbb{R}^{n+1} \rightarrow \mathbb{R} \cup \{+\infty\}$ of f is defined by:

$$\tilde{f}(\mathbf{x}, z) \equiv \begin{cases} zf(\mathbf{x}/z), & \text{if } z > 0, \\ +\infty, & \text{if } z \leq 0. \end{cases}$$

In our example, the nonlinear constraint needed to describe $\text{conv}(\Gamma_0 \cup \Gamma_1)$ is obtained by taking the perspective function of f from the point $\mathbf{x}^* = (u_1, u_2)$. For this specific example, $\text{conv}(\Gamma_0 \cup \Gamma_1)$ is written as follows:

$$\text{conv}(\Gamma_0 \cup \Gamma_1) = \text{cl} \left\{ \begin{array}{l} (\mathbf{x}, z) \in \mathbb{R}^2 \times [0, 1] : \\ \tilde{f}(\mathbf{x} - (1-z)\mathbf{x}^*, z) \leq 0 \\ l_1 \leq x_1 \leq u_1, l_2 \leq x_2 \leq u_2 \end{array} \right\} = \text{cl} \left\{ \begin{array}{l} (\mathbf{x}, z) \in \mathbb{R}^2 \times [0, 1] : \\ \frac{z}{zc_1 - x_1 + (1-z)u_1} + \frac{z}{zc_2 - x_2 + (1-z)u_2} \leq d \\ l_1 \leq x_1 \leq u_1, l_2 \leq x_2 \leq u_2. \end{array} \right\}. \quad (8.1)$$

Theorem 8.23: Theorem 1... [Ceria and Soares, 8]

Consider a closed convex set $P \subseteq \mathbb{R}^n$ defined by

$$P \equiv \text{cl conv}(K), K \equiv \bigcup_{i=1}^p K^i,$$

where every set K^i is a closed convex set having the representation

$$K^i \equiv \{\mathbf{x} \in \mathbb{R}^n : G^i(\mathbf{x}) \leq 0\},$$

and $G^i : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a vector mapping whose components are closed convex functions.

Let $I \equiv \{i : K^i \neq \emptyset\}$. If the set K is bounded below or above, then

$$P \equiv \text{cl} \left\{ x \in \mathbb{R}^n : \forall i \in I, \exists \lambda_i > 0 \text{ and } \mathbf{x}^i \in \mathbb{R}^n \right. \\ \left. \text{with } \mathbf{x} = \sum_{i \in I} \mathbf{x}^i, \sum_{i \in I} \lambda_i = 1, \tilde{G}^i(\mathbf{x}^i, \lambda_i) \leq 0, i \in I \right\}.$$

Let us emphasize that the above formulation introduces $(2n+4)$ variables to convexify one "on/off" constraint ($i = 2$ in our case), thus a total of $|K|(2n+4)$ variables must be added to convexify all the disjunctive constraints in (2). Therefore reducing the space dimension can have a very important impact when dealing with large optimization problems including many "on/off" constraints.

Using fewer variables

In the following lemma, for our case of interest, we show that the convex hull formulation corresponding to one "on/off" constraint can be obtained by adding only n new variables.

Lemma 8.24: [15]

Let $f : E \rightarrow \mathbb{R}$, $E \subseteq \mathbb{R}^n$, be a closed convex function,

$$\begin{aligned}\Gamma_0 &= \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 0, \mathbf{l}^0 \leq \mathbf{x} \leq \mathbf{u}^0\}, \\ \Gamma_1 &= \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 1, f(\mathbf{x}) \leq 0, \mathbf{l}^1 \leq \mathbf{x} \leq \mathbf{u}^1\} \neq \emptyset.\end{aligned}$$

Then $\text{conv}(\Gamma_0 \cup \Gamma_1) = \text{pr}_{(\mathbf{x}, z)} \text{cl}(\Gamma)$, with

$$\Gamma = \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{2n+1} : \\ z f(\mathbf{y}/z) \leq 0 \\ \mathbf{x} - (1-z)\mathbf{u}^0 \leq \mathbf{y} \leq \mathbf{x} - (1-z)\mathbf{l}^0 \\ z \mathbf{l}^1 \leq \mathbf{y} \leq z \mathbf{u}^1 \\ 0 < z \leq 1 \end{array} \right\}$$

Proof. Theorem 8.23, allows to write the exact formulation of $\text{conv}(\Gamma_0 \cup \Gamma_1)$ as follows: $\text{conv}(\Gamma_0 \cup \Gamma_1) = \text{pr}_{(\mathbf{x}, z)} \text{cl}(\Gamma)$, where

$$\Gamma = \left\{ \begin{array}{l} (\mathbf{x}, z, \lambda_0, \lambda_1, z_0, z_1, \mathbf{x}^0, \mathbf{x}^1) \in \mathbb{R}^{3n+5} : \\ \mathbf{x} = \mathbf{x}^0 + \mathbf{x}^1 \\ z = z_0 + z_1 \\ \lambda_0 + \lambda_1 = 1 \\ \tilde{f}(\mathbf{x}^1, \lambda_1) \leq 0 \\ \mathbf{l}^0 \lambda_0 \leq \mathbf{x}^0 \leq \mathbf{u}^0 \lambda_0 \\ \mathbf{l}^1 \lambda_1 \leq \mathbf{x}^1 \leq \mathbf{u}^1 \lambda_1 \\ z_0 = 0 \\ z_1 = \lambda_1 \\ 0 < \lambda^1, 0 \leq \lambda^0 \end{array} \right\} \equiv \left\{ \begin{array}{l} (\mathbf{x}, z, \lambda_0, \mathbf{x}^0, \mathbf{x}^1) \in \mathbb{R}^{3n+2} : \\ \mathbf{x} = \mathbf{x}^0 + \mathbf{x}^1 \\ \lambda_0 + z = 1 \\ \tilde{f}(\mathbf{x}^1, z) \leq 0 \\ \mathbf{l}^0 \lambda_0 \leq \mathbf{x}^0 \leq \mathbf{u}^0 \lambda_0 \\ \mathbf{l}^1 z \leq \mathbf{x}^1 \leq \mathbf{u}^1 z \\ 0 \leq \lambda_0 \\ 0 < z \end{array} \right\}.$$

Substituting $\mathbf{x}^0 = \mathbf{x} - \mathbf{x}^1$ and $\lambda_0 = 1 - z$, we obtain:

$$\Gamma \equiv \left\{ \begin{array}{l} (\mathbf{x}, z, \mathbf{x}^1) \in \mathbb{R}^{2n+1} : \\ \tilde{f}(\mathbf{x}^1, z) \leq 0 \\ \mathbf{l}^0 \lambda_0 \leq \mathbf{x} - \mathbf{x}^1 \leq \mathbf{u}^0 \lambda_0 \\ \mathbf{l}^1 z \leq \mathbf{x}^1 \leq \mathbf{u}^1 z \\ 0 \leq 1-z \\ 0 < z \end{array} \right\} \equiv \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{2n+1} : \\ \tilde{f}(\mathbf{y}, z) \leq 0 \\ \mathbf{x} - (1-z)\mathbf{u}^0 \leq \mathbf{y} \leq \mathbf{x} - (1-z)\mathbf{l}^0 \\ z\mathbf{l}^1 \leq \mathbf{y} \leq z\mathbf{u}^1 \\ 0 < z \leq 1 \end{array} \right\}$$



Lemma 8.24 shows that only n additional variables are needed to explicitly describe the convex hull.

One can notice that these variables appear in the perspective function of f and the constraints:

$$\mathbf{x} - (1-z)\mathbf{u}^0 \leq \mathbf{y} \leq \mathbf{x} - (1-z)\mathbf{l}^0$$

and

$$z\mathbf{l}^1 \leq \mathbf{y} \leq z\mathbf{u}^1.$$

We observe that, if we consider only the last two sets of constraints, Fourier-Motzkin elimination can be applied in a straightforward way to eliminate \mathbf{y} leading to the constraints $z\mathbf{l}^1 + (1-z)\mathbf{l}^0 \leq \mathbf{x} \leq z\mathbf{u}^1 + (1-z)\mathbf{u}^0$. Nevertheless, the projection becomes harder to describe when the nonlinear constraint in Γ is taken into account. Next, we show how the \mathbf{y} variables can be projected out in the case where the function f is isotone or order preserving (see definition below).

Definition 8.25: Independently Increasing and Independently Monotone

Let $f : E \rightarrow \mathbb{R}, E \subseteq \mathbb{R}^n$.

- f is independently increasing (resp. decreasing) on coordinate i if for all $\mathbf{x} \in \text{dom}(f)$ and $\lambda > 0$ such that $\mathbf{x} + \lambda e_i \in \text{dom}(f)$, we have $f(\mathbf{x} + \lambda e_i) \geq f(\mathbf{x})$ (resp. $f(\mathbf{x} + \lambda e_i) \leq f(\mathbf{x})$).
- f is independently monotone on coordinate i if it is independently increasing or independently decreasing on the i th coordinate.
- f is isotone if it is independently monotone on every coordinate.

Example 8.26

1. $f(x_1, x_2, x_3) = e^{(2x_1-x_2)} + x_3, \mathbf{x} \in \mathbb{R}^3, f$ is independently increasing on coordinate 1 and 3 , independently decreasing on coordinate 2 , therefore it is an isotone function.
2. $f(x, y) = x^4 + y^2, (x, y) \in \mathbb{R}^2$, the variation of f depends on the sign of the variables, f is not an isotone function.
3. $f(\mathbf{x}) = \sum_{i=1}^n \frac{1}{c_i - x_i}$, where $x_i \in]-\infty, c_i]$ for $i = 1, \dots, n$. Since f is a sum of one-variable increasing functions, it is an isotone function.

Additively separable functions which are sums of one-variable monotone functions are a commonly encountered case of isotone functions.

Theorem 8.27: Theorem 2 [15]

Let $f : E \rightarrow \mathbb{R}$, $E \subseteq \mathbb{R}^n$, be an isotone closed convex function, with J^1 (resp. J^2) the set of indices on which f is independently increasing (resp. decreasing),

$$\begin{aligned}\Gamma_0 &= \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 0, \mathbf{l}^0 \leq \mathbf{x} \leq \mathbf{u}^0\}, \\ \Gamma_1 &= \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 1, f(\mathbf{x}) \leq 0, \mathbf{l}^1 \leq \mathbf{x} \leq \mathbf{u}^1\} \neq \emptyset.\end{aligned}$$

Then $\text{conv}(\Gamma_0 \cup \Gamma_1) = \text{cl}(\Gamma')$, where

$$\Gamma' = \left\{ \begin{array}{l} (\mathbf{x}, z) \in \mathbb{R}^{n+1} : \\ z q_S(\mathbf{x}/z) \leq 0 \quad \forall S \subset \{1, 2, \dots, n\} \\ z \mathbf{l}^1 + (1-z) \mathbf{l}^0 \leq \mathbf{x} \leq z \mathbf{u}^1 + (1-z) \mathbf{u}^0 \\ 0 < z \leq 1 \end{array} \right\}$$

$q_S = (f \circ h_S)$, and $h_S(\mathbb{R}^n \times]0, 1]) \rightarrow \mathbb{R}^n$ is defined by

$$(h_S(\mathbf{x}, z))_i = \begin{cases} l_i^1 & \forall i \in S \cap J_1 \\ u_i^1 & \forall i \in S \cap J_2 \\ x_i - \frac{(1-z)u_i^0}{z} & \forall i \in J_1, i \notin S \\ x_i - \frac{(1-z)l_i^0}{z} & \forall i \in J_2, i \notin S \end{cases}$$

Proof. We prove that $\text{cl}(\Gamma')$ is the projection onto the (\mathbf{x}, z) space of $\text{cl}(\Gamma)$, the set defined in Lemma 8.24.

Step 1: We show that $(\mathbf{x}, z) \in \text{cl}(\Gamma') \Rightarrow \exists \mathbf{y} \in \mathbb{R}^n$ s.t. $(\mathbf{x}, \mathbf{y}, z) \in \text{cl}(\Gamma)$.

For any given point $(\mathbf{x}, z) \in \Gamma'$, $z \neq 0$, let $\mathbf{y} \in \mathbb{R}^n$ be defined as follows:

$$\begin{aligned}y_i &= \max \{z l_i^1, x_i - (1-z) u_i^0\}, \quad \forall i \in J^1 \quad \text{and} \\ y_i &= \min \{z u_i^1, x_i - (1-z) l_i^0\}, \quad \forall i \in J^2.\end{aligned}$$

- $\exists S \subseteq \{1, 2, \dots, n\}$ s.t. $h_S\left(\frac{\mathbf{x}}{z}\right) = \frac{\mathbf{y}}{z}$.
- Given $z q_S\left(\frac{\mathbf{x}}{z}\right) = z f\left(h_S\left(\frac{\mathbf{x}}{z}\right)\right) \leq 0$ in Γ' , implies $z f\left(\frac{\mathbf{y}}{z}\right) \leq 0$.
- Hence, $(\mathbf{x}, \mathbf{y}, z) \in \Gamma$.

Now suppose $z = 0$.

- For $(\mathbf{x}, 0) \in \text{cl}(\Gamma')$, $\exists \{(\mathbf{x}^k, z^k)\} \in \Gamma'$ with $\lim_{k \rightarrow \infty} (\mathbf{x}^k, z^k) = (\mathbf{x}, 0)$.
- With $\mathbf{y}^k = \mathbf{y} \forall k \in \mathbb{N}$, $\lim_{k \rightarrow \infty} (\mathbf{x}^k, \mathbf{y}^k, z^k) = (\mathbf{x}, \mathbf{y}, 0)$.
- Therefore, $(\mathbf{x}, \mathbf{y}, 0) \in \text{cl}(\Gamma)$.

Step 2. We show that $(\mathbf{x}, \mathbf{y}, z) \in \text{cl}(\Gamma) \Rightarrow (\mathbf{x}, z) \in \text{cl}(\Gamma')$. Let $(\mathbf{x}, \mathbf{y}, z)$ be a point in Γ ($z \neq 0$). By definition of Γ and functions $h_S(\mathbf{x})$, we have $\forall S \subset \{1, 2, \dots, n\}$

$$\frac{y_i}{z} \geq \max \left\{ l_i^1, \frac{x_i}{z} - \frac{(1-z)u_i^0}{z} \right\} \geq (h_S(\mathbf{x}/z))_i, \quad \forall i \in J^1$$

and

$$\frac{y_i}{z} \leq \min \left\{ u_i^1, \frac{x_i}{z} - \frac{(1-z)l_i^0}{z} \right\} \leq (h_S(\mathbf{x}/z))_i, \quad \forall i \in J^2.$$

- f being an isotone function we have $zf(h_S(\mathbf{x}/z)) \leq zf(\mathbf{y}/z) \leq 0, \forall S \subset \{1, 2, \dots, n\}$.
- Finally, we notice that the constraints $z\mathbf{l}^1 + (1-z)\mathbf{l}^0 \leq \mathbf{x} \leq z\mathbf{u}^1 + (1-z)\mathbf{u}^0$ are obtained by composing the last two set of constraints defining set Γ .

The extension to the closure is trivial. ♠

Now, we are able to describe $\text{conv}(\Gamma_0 \cup \Gamma_1)$ in (3) without having to deal with additional variables. Note that this formulation might include an exponential number of constraints. In the next corollary, we show that only one constraint for each disjunction is sufficient to build a valid formulation of our original MINLP defined in (2).

Corollary 8.28: 1

Let $f : E \rightarrow \mathbb{R}, E \subseteq \mathbb{R}^n$, be an isotone closed convex function with J^1 (resp., J^2) the set of indices on which f is independently increasing (resp. decreasing),

$$\begin{aligned} \Gamma_0 &= \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 0, \mathbf{l}^0 \leq \mathbf{x} \leq \mathbf{u}^0\}, \\ \Gamma_1 &= \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 1, h(\mathbf{x}) \leq 0, \mathbf{l}^1 \leq \mathbf{x} \leq \mathbf{u}^1\} \neq \emptyset, \\ \Gamma'' &= \left\{ \begin{array}{l} (\mathbf{x}, z) \in \mathbb{R}^{n+1} : \\ zq_\emptyset(\mathbf{x}, z) \leq 0 \\ z\mathbf{l}^1 + (1-z)\mathbf{l}^0 \leq \mathbf{x} \leq z\mathbf{u}^1 + (1-z)\mathbf{u}^0 \\ 0 < z \leq 1 \end{array} \right\} \end{aligned}$$

with $q_\emptyset = (f \circ h_\emptyset)$ and $h_\emptyset(\mathbb{R}^n \times]0, 1]) \rightarrow \mathbb{R}^n$ defined as

$$(h_\emptyset(\mathbf{x}, z))_i = \begin{cases} \frac{x_i - (1-z)u_i^0}{z} & \forall i \in J_1, \\ \frac{x_i - (1-z)l_i^0}{z} & \forall i \in J_2. \end{cases}$$

Then

1. $\text{cl}(\Gamma'')$ is a valid convex relaxation for $\text{conv}(\Gamma_0 \cup \Gamma_1)$;
2. $\text{cl}(\Gamma'') \cap \{z \in \{0, 1\}\} \equiv \Gamma_0 \cup \Gamma_1$.

Proof.

1. $\text{cl}(\Gamma'')$ is a valid convex relaxation of $\text{conv}(\Gamma_0 \cup \Gamma_1)$ since it only contains a subset of the constraints defining the convex hull in Theorem 2.

2. For $z = 1$, one can check that $\Gamma'' \cap \{(\mathbf{x}, z) : z = 1\} \equiv \Gamma_1$. For $z = 0$, we have $(\Gamma_0 \cup \Gamma_1) \subseteq \text{cl}(\Gamma'')$ since $\text{cl}(\Gamma'')$ is a valid convex relaxation of $\Gamma_0 \cup \Gamma_1$. Therefore intersecting with $\{(\mathbf{x}, z) : z = 0\}$ gives $(\Gamma_0 \cup \Gamma_1) \cap \{(\mathbf{x}, z) : z = 0\} = \Gamma_0 \subseteq \text{cl}(\Gamma'') \cap \{(\mathbf{x}, z) : z = 0\}$.

On the other hand, by definition of Γ'' , when $z = 0$, all the constraints of Γ_0 are satisfied in $\text{cl}(\Gamma'')$, that is $\text{cl}(\Gamma'') \cap \{(\mathbf{x}, z) : z = 0\} \subseteq \Gamma_0$.



8.3.1. Relationships to previous theoretical results

Balas [3,4] was the first to introduce the explicit algebraic formulation of the convex hull of a union of polyhedra in a higher dimensional space. Generalizations and extensions have been established for unions of nonlinear convex sets [8, 13, 21]. Günlük and Linderoth [14] have characterized the convex hull of the union of a point and a convex set in the space of original variables. We show that this characterization is a special case of Lemma 1 , obtained by fixing $\mathbf{l}^0 = \mathbf{u}^0 = 0$ in the definition of this lemma:

$$\text{conv}(\Gamma_0 \cup \Gamma_1) = \text{cl} \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{2n+1} : \\ zf(\mathbf{y}/z) \leq 0 \\ \mathbf{x} \leq \mathbf{y} \leq \mathbf{x} \\ z\mathbf{l}^1 \leq \mathbf{y} \leq z\mathbf{u}^1 \\ 0 < z \leq 1 \end{array} \right\} \equiv \text{cl} \left\{ \begin{array}{l} (\mathbf{x}, z) \in \mathbb{R}^{n+1} : \\ zf(\mathbf{x}/z) \leq 0 \\ z\mathbf{l}^1 \leq \mathbf{x} \leq z\mathbf{u}^1 \\ 0 < z \leq 1 \end{array} \right\}$$

which corresponds exactly to the result established in [14].

Technical issues

Let us note that some technical issues must be considered due to the fact that the formulations introduced previously involve topological closures of sets. In practice, when implementing perspective functions, one must avoid dividing by zero (for $z = 0$).

A first alternative is to use cutting plane methods. Instead of explicitly writing the convex hull formulas in the original model, valid cuts are generated to strengthen the formulation (see [8, 12, 21] for more details).

A second alternative proposed in [10, 13, 19], is to approximate the constraints by adding epsilon values to the corresponding functions.

We show that, for our application of interest, these difficulties can be avoided while still guaranteeing exact convex MINLP models.

8.3.2. Application: the delay-constrained routing problem

This problem in telecommunications was first introduced by Ben-Ameur and Ouorou [5] and revisited in [15].

The constant rise of traffic in telecommunication induce that networks (or certain parts of them) become congested. One of the main consequences of this congestion is a delay in the communications that go through the congested parts of the network. Parallel to this congestion is the introduction of more and more real time services (voice on IP, video on demand, gaming,...) that can only operate properly if the delay of the communications is controlled. Managing these delays is therefore an important question. A common approach is to study routing problems under average end-to-end delay constraints, see [6, 17, 20].

Unfortunately, this approach is not adequate with real time services since it ignores the heterogeneous nature of real world networks. These services rather require the consideration of individual source-to-destination delays (i.e., packet delay from the source to all destinations) and need to be differentiated from delay-tolerant services. The delay-constrained routing model we consider takes into account an end-to-end delay guarantee for each type of service. This application can be formulated as a mixed-integer nonlinear program including "on/off" constraints.

Mathematical models

Let G be a finite directed network,

1. Parameters:

- (a) V represents the set of vertices, E the set of arcs and K the set of commodities.
- (b) For each commodity $k \in K$, $v_k \in \mathbb{R}$ is the quantity of demand that needs to be routed and $\alpha_k \in \mathbb{R}$ the maximum delay.
- (c) Each commodity has a set of candidate paths denoted by $P(k) = \{P_k^1, P_k^2, \dots, P_k^{n_k}\}$, each one of these corresponding to a different routing for commodity k . N represents the maximum authorized number of activated paths per commodity.
- (d) For each arc e , $c_e \in \mathbb{R}$ represents its capacity and $w_e \in \mathbb{R}$ its cost.

2. Variables:

- (a) We call ϕ_k^i the fraction of the k th demand carried by its i th path, $\phi_k^i \in [0, 1]$.
- (b) z_k^i is a binary variable indicating if path P_k^i is activated.
- (c) x_e denotes the total amount of flow crossing over arc e , $x_e \in \mathbb{R}$.

Initial mathematical model (P) The objective function (5) is to minimize the total routing cost on all used links. For each commodity k , constraint (6) ensures that the total flow routed is greater than 1 in order to satisfy demand. Constraints (7) define the variables x_e on each arc as the sum of all the flows passing through e . In (8), x_e is bounded by the capacity installed on the link. Constraint (8.2e) represents the main

"on/off" delay constraints: the delay guarantee associated to a given commodity must be satisfied on its candidate path if the latter is activated. As mentioned above, this model allows to fix a maximum number of active paths per commodity. This is established in (10). In (11), we link the indicator variables z_k^i to the ϕ_k^i variables. Finally, bounds on all variables are introduced in (12)-(14). Let us point out that if $N = 1$, i.e. only one path can be activated per commodity (mono-routing), the variables ϕ_k^i become redundant and can be replaced by the z_k^i variables. Ben-Ameur and Ouorou showed in [5] that as soon as one considers two candidate paths per commodity, the underlying feasibility problem (ignoring the objective function) is NP-complete.

$$\min \sum_{e \in E} w_e x_e \quad (8.2a)$$

$$\text{s.t. } \sum_{i=1}^{n_k} \phi_k^i \geq 1, \quad \forall k \in K, \quad (8.2b)$$

$$\sum_{k \in K} \sum_{P_k^i \ni l} \phi_k^i v_k \leq x_e, \quad \forall e \in E, \quad (8.2c)$$

$$x_e \leq c_e, \quad \forall e \in E, \quad (8.2d)$$

$$\sum_{e \in P_k^i} \frac{1}{c_e - x_e} \leq \alpha_k, \quad \forall k \in K, \forall P_k^i \in P(k) \text{ if } z_k^i = 1, \quad (8.2e)$$

$$\sum_{P_k^i \in P(k)} z_k^i \leq N, \quad \forall k \in K, \quad (8.2f)$$

$$\phi_k^i \leq z_k^i, \quad \forall k \in K, \forall P_k^i \in P(k), \quad (8.2g)$$

$$z_k^i \in \{0, 1\}, \quad \forall k \in K, \forall P_k^i \in P(k), \quad (8.2h)$$

$$\phi_k^i \in [0, 1], \quad \forall k \in K, \forall P_k^i \in P(k), \quad (8.2i)$$

$$x_e \in \mathbb{R}, \quad \forall e \in E. \quad (8.2j)$$

The continuous relaxation of this model is obviously non-convex due to the presence of "on/off" constraints in (8.2e).

We will next introduce four different convex models equivalent to (8.2) and offering each a different continuous relaxation.

Big-M relaxation: (P_{big-M}) A classical convex relaxation of constraint (8.2e) is the big- M relaxation:

$$\min \sum_{e \in E} w_e x_e \quad (8.3a)$$

$$\text{s.t. } \text{constraints (6), (7), (8), (10), (11), (12) – (14)}, \quad (8.3b)$$

$$\sum_{e \in P_k^i} \frac{1}{c_e - x_e} \leq M - z_k^i(M - \alpha_k), \quad \forall k \in K, \forall P_k^i \in P(k). \quad (8.3c)$$

This formulation is exact if z_k^i is a binary variable, provided that the constant M is big enough. When $z_k^i = 0$, the constraint (8.3c) is redundant, due to the big- M quantity on its right-hand-side; when $z_k^i = 1$, the big- M disappears leading to the original delay constraint formula.

Since the validity of constraint (8.3c) and the quality of the bound given by this formulation depends on the constant M , one has to compute it accurately. Ben-Ameur and Ouorou [5] pointed out that the flow on a given arc e always admits an upper bound u_e verifying $u_e < c_e$. If a link e is used in an activated path P_k^i , then one can write $\frac{1}{c_e - x_e} \leq \alpha_k - \sum_{e' \neq e} \frac{1}{c_{e'}}$. Based on these observations, one can easily deduce an upper bound α_e for the delay on each arc and therefore obtain an upper bound on the total delay generated on any given path. In other words, the big M constant can be replaced by $\alpha_k^i = \sum_{e \in P_k^i} \alpha_e$.

Higher space convex hull relaxation model (P_{high})

First, we present a model based on the state of the art in disjunctive programming using Theorem 8.23.

$$\min \sum_{e \in E} w_e x_e \quad (8.4a)$$

$$\text{s.t. constraints (6), (7), (8), (10), (11), (12) – (14),} \quad (8.4b)$$

$$\sum_{e \in P_k^i} \left(\frac{z_k^i}{z_k^i c_e - \lambda_e^{(1,i,k)}} \right) - z_k^i \alpha_k \leq 0, \quad \forall k \in K, \forall P_k^i \in P(k), \quad (8.4c)$$

$$x_e = \lambda_e^{(0,i,k)} + \lambda_e^{(1,i,k)}, \quad \forall k \in K, \forall P_k^i \in P(k), \forall e \in P_k^i, \quad (8.4d)$$

$$0 \leq \lambda_e^{(0,i,k)} \leq (1 - z_k^i) u_e^0, \quad \forall k \in K, \forall P_k^i \in P(k), \forall e \in P_k^i, \quad (8.4e)$$

$$0 \leq \lambda_e^{(1,i,k)} \leq z_k^i u_e^1, \quad \forall k \in K, \forall P_k^i \in P(k), \forall e \in P_k^i. \quad (8.4f)$$

Next, we use the results to introduce new formulations. To every path P_k^i corresponds a delay constraint (8.2e) in (P) that is written: $f^{(i,k)}(\mathbf{x}) \leq 0$ if $z_k^i = 1$, with $f^{(i,k)} : \mathbb{R}^n \rightarrow \mathbb{R}$, $f^{(i,k)}(\mathbf{x}) = \sum_{e \in P_k^i} \frac{1}{c_e - x_e} - \alpha_k$. The functions $f^{(i,k)}$ being closed convex functions, Lemma 8.24 applies leading to the following corollary.

Corollary 8.29: 2

Let

$$f : \mathbb{R}_+^n \rightarrow \mathbb{R}, \quad f(\mathbf{x}) = \sum_{i=1}^n \left(\frac{1}{c_i - x_i} \right) - b, \quad b \geq 0,$$

$$\Gamma_0 = \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 0, 0 \leq \mathbf{x} \leq \mathbf{u}\},$$

$$\Gamma_1 = \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 1, f(\mathbf{x}) \leq 0, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\} \text{ non empty.}$$

Then $\text{conv}(\Gamma_0 \cup \Gamma_1) = \{(\mathbf{x}, z) \in \mathbb{R}^{n+1} \mid \exists \mathbf{y} \in \mathbb{R}^n \text{ with } (\mathbf{x}, \mathbf{y}, z) \in \text{cl}(\Gamma)\}$, where

$$\Gamma = \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{2n+1} : \\ \sum_{i=1}^n \left(\frac{z^2}{zc_i - y_i} \right) - zb \leq 0 \\ \mathbf{x} - (1-z)\mathbf{u} \leq \mathbf{y} \leq \mathbf{x} \\ z\mathbf{l} \leq \mathbf{y} \leq z\mathbf{u} \\ 0 < z \leq 1 \end{array} \right\}.$$

As discussed previously, the values of the functions in the nonlinear constraints are not well defined in

$z_k^i = 0$. In the following proposition, we suggest a new valid relaxation of the convex hull which overcomes this issue while still being exact for $z_k^i \in \{0, 1\}$.

Proposition 8.30: 1

Let

$$f : \mathbb{R}_+^n \rightarrow \mathbb{R}, \quad f(\mathbf{x}) = \sum_{i=1}^n \left(\frac{1}{c_i - x_i} \right) - b, \quad b \geq 0.$$

$$\Gamma_0 = \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 0, 0 \leq \mathbf{x} \leq \mathbf{u} < \mathbf{c}\},$$

$$\Gamma_1 = \{(\mathbf{x}, z) \in \mathbb{R}^n \times \mathbb{B} : z = 1, f(\mathbf{x}) \leq 0, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\} \text{ non empty.}$$

For $\varepsilon > 0$, let

$$\Gamma^\varepsilon = \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{2n+1} : \\ \sum_{i=1}^n \left(\frac{z^2}{zc_i - y_i + (1-z)\varepsilon} \right) - zb \leq 0 \\ \mathbf{x} - (1-z)\mathbf{u} \leq \mathbf{y} \leq \mathbf{x} \\ z\mathbf{l} \leq \mathbf{y} \leq z\mathbf{u} \\ 0 \leq z \leq 1 \end{array} \right\}$$

Then

1. $\text{proj}_{(\mathbf{x}, z)}(\Gamma^\varepsilon)$ is a valid convex relaxation of $\text{conv}(\Gamma_0 \cup \Gamma_1)$,
2. $\text{proj}_{(\mathbf{x}, z)}(\Gamma^\varepsilon) \cap \{z \in \{0, 1\}\} \equiv \Gamma_0 \cup \Gamma_1$.

Proof. First, we show that all the constraints of Γ^ε are convex. The only nonlinear constraint in Γ^ε is $g(\mathbf{y}, z) \leq 0$ with $g(\mathbf{y}, z) = \sum_{i=1}^n g_i(y_i, z) - zb$ and $g_i(y_i, z) = \frac{z^2}{zc_i - y_i + (1-z)\varepsilon}$. The Hessian matrix of g_i is

$$\mathcal{H}(g_i) = \begin{pmatrix} \frac{2(y_i - \varepsilon)^2}{(z(c_i - \varepsilon) - (y_i - \varepsilon))^3} & \frac{-2z(y_i - \varepsilon)}{(z(c_i - \varepsilon) - (y_i - \varepsilon))^3} \\ \frac{-2z(y_i - \varepsilon)}{(z(c_i - \varepsilon) - (y_i - \varepsilon))^3} & \frac{2z^2}{(z(c_i - \varepsilon) - (y_i - \varepsilon))^3} \end{pmatrix}$$

Having $y_i \leq zu_i \leq zc_i + (1-z)\varepsilon, \forall i \in \{1, \dots, n\}$, one can check that \mathcal{H} is positive semidefinite, that is the functions g_i are all convex, g being a sum of convex functions is convex. Note also that, since $\frac{z^2}{zc_i - y_i + (1-z)\varepsilon} \leq \frac{z^2}{zc_i - y_i}$, the validity of the constraint is preserved.

Next we prove that the projection of Γ^ε on the (\mathbf{x}, z) -space contains both Γ_0 and Γ_1 .

- For $z = 0$ we have

$$\Gamma^\varepsilon = \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, 0) \in \mathbb{R}^{2n+1} : \\ \mathbf{x} \leq \mathbf{u}, \mathbf{y} = 0 \\ \mathbf{x} \geq \mathbf{y} = 0 \end{array} \right\},$$

in this case $\text{proj}_{(\mathbf{x}, z)}(\Gamma^\varepsilon) = \Gamma_0$.

- For $z = 1$ we have

$$\Gamma^\varepsilon = \left\{ \begin{array}{l} (\mathbf{x}, \mathbf{y}, 1) \in \mathbb{R}^{2n+1} : \\ \sum_{i=1}^n \left(\frac{1}{c_i - y_i} \right) - b \leq 0 \\ \mathbf{x} = \mathbf{y}, \mathbf{l} \leq \mathbf{y} \leq \mathbf{u} \end{array} \right\}$$

in this case $\text{proj}_{(\mathbf{x}, z)}(\Gamma^\varepsilon) = \Gamma_1$.



Based on this proposition, we introduce a new convex MINLP equivalent to (P) which gives a tighter continuous relaxation than the big- M model.

Reduced convex hull relaxation model (P_{red})

We replace constraints (8.2e) by the convex relaxations defined in Proposition 1:

$$\min \quad \sum_{e \in E} w_e x_e \tag{8.5a}$$

$$\text{s.t. constraints (6), (7), (8), (10), (11), (12) – (14),} \tag{8.5b}$$

$$\sum_{e \in P_k^i} \left(\frac{z_k^{i^2}}{z_k^i c_e - y_e^{(i,k)} + (1 - z_k^i) \varepsilon} \right) - z_k^i \alpha_k \leq 0, \quad \forall k \in K, \forall P_k^i \in P(k), \tag{8.5c}$$

$$x_e - (1 - z_k^i) u_e \leq y_e^{(i,k)} \leq x_e, \quad \forall k \in K, \forall P_k^i \in P(k), \forall e \in P_k^i, \tag{8.5d}$$

$$z_k^i l_e \leq y_e^{(i,k)} \leq z_k^i u_e, \quad \forall k \in K, \forall P_k^i \in P(k), \forall e \in P_k^i. \tag{8.5e}$$

Table 3 Results with $N = 2$ (bi-routing)

	P_{big-M}		P_{proj}	
	cpu	nodes	cpu	nodes
2 candidate paths per commodity				
rdata1	0.8	0	0.4	0
rdata2	2.4	0	10.9	0
adata3	47.6	0	4.7	0
rdata4	5.6	0	3.4	0
rdata5	38.3	0	50.4	0
adata6	40.1	0	42.4	0
3 candidate paths per commodity				
rdata1	16.7	0	2.9	0
rdata2	129.2	18	59	0
adata3	291.5	760	171.3	615
rdata4	154.9	62	343.8	472
rdata5	[0.15%]	91430	609.1	5290
adata6	1579.8	8176	747.7	7056
10 candidate paths per commodity				
rdata1	1909	56788	399	7846
rdata2	28.8	0	288.8	666
adata3	[0.11%]	67705	[0.13%]	77129
rdata4	[1.2%]	23984	[0.6%]	32939
rdata5	[2%]	11772	[1.2%]	16285
adata6	[0.7%]	6480	[0.14%]	22364

Exercise 5 :

$$\begin{aligned} & \min f(\mathbf{x}, \mathbf{z}) \\ \text{s.t. } & \mathbf{h}(\mathbf{x}, \mathbf{z}) \leq 0 \end{aligned}$$

Consider optimization problems of the form

$$(\mathbf{x}, z_k) \in \Gamma_0^k \cup \Gamma_1^k, \forall k \in K$$

$$\Gamma_0^k = \{(\mathbf{x}, z_k) : z_k = 0, \mathbf{l}^0 \leq \mathbf{x} \leq \mathbf{u}^0\}$$

$$\Gamma_1^k = \{(\mathbf{x}, z_k) : z_k = 1, g_k(\mathbf{x}) \leq 0, \mathbf{l}^1 \leq \mathbf{x} \leq \mathbf{u}^1\}.$$

$$\mathbf{x} \in \mathbb{R}^n, \mathbf{z} \in \mathbb{Z}^m.$$

Let $K = \{1\}$ and let $g_1(\mathbf{x}) = \sum_{i=1}^n a_i x_i - b$. Derive the convex hull of $\Gamma_0 \cup \Gamma_1$. Specifically, these sets are defined as follows:

$$\Gamma_0 = \{(x, z) \in \mathbb{R}^{n+1} : z = 0, l_i^0 \leq x_i \leq u_i^0, \forall i \in \{1, \dots, n\}\}$$

$$\Gamma_1 = \left\{ (x, z) \in \mathbb{R}^{n+1} : z = 1, \sum_{i=1}^n a_i x_i \leq b, l_i^1 \leq x_i \leq u_i^1, \forall i \in \{1, \dots, n\} \right\}, \text{ non empty.}$$

9. Convex Quadratic Reformulation

Resources

- **HammerRubin1970:** Hammer, P.L., Rubin, A.A.: Some remarks on quadratic programming with 0–1 variables. *Revue Francaise d’Informatique Et de Recherche Operationnelle*. 4(3), 67-79 (1970).
- **Billionnet2007:** Billionnet, A., Elloumi, S.: Using a mixed integer quadratic programming solver for the unconstrained quadratic 0–1 problem. *Math. Program.* 109, 55-68 (2007).
- **Billionnet2009:** Billionnet, A., Elloumi, S., Plateau, M.-C.: Improving the performance of standard solvers for quadratic 0–1 programs by a tight convex reformulation: the QCR method. *Discrete Appl. Math.* 157(6), 1185-1197 (2009).
- **Billionnet2012:** Billionnet, A., Elloumi, S., Lambert, A.: Extending the QCR method to general mixedinteger programs. *Mathematical Programming* 131(1-2), 381-401 (2012).
- **Billionnet2016:** Billionnet, A., Elloumi, S., Lambert, A.: Exact quadratic convex reformulations of mixed-integer quadratically constrained problems. *Mathematical Programming* 158(1), 235-266 (2016).
- **Elloumi2019:** Elloumi, S., Lambert, A.: Global solution of non-convex quadratically constrained quadratic programs. *Optimization Methods and Software* 34(1), 98-114 (2019).
- **Galli2014:** Galli, L., Letchford, A.N.: A compact variant of the qcr method for quadratically constrained quadratic 0–1 programs. *Optimization Letters* 8(4), 1213–1224 (2014).
- **Wiese2021:** Wiese, S.: A computational practicability study of MIQCQP reformulations. (2021).
- **Saxena2011:** Saxena, A., Boman, P., Lee, J.: Convex relaxations of non-convex mixed integer quadratically constrained programs: Projected formulations. *Mathematical Programming* 130, 359-413 (2011)

9.1 Tools: Semidefinite Programming Duality

12.2 Semidefinite Programs and their Duals

Given this understanding of psd matrices, we can now look at semidefinite programs (SDPs), and define their duals. Let us describe two common forms of writing SDPs. Consider symmetric matrices

A_1, A_2, \dots, A_m, C , and reals b_1, b_2, \dots, b_m . The first form is the following one.

$$\begin{aligned} & \min C \bullet X \\ \text{s.t. } & A_i \cdot X = b_i \quad i = 1 \dots m \\ & X \succeq 0 \end{aligned}$$

Another common form for writing SDPs is the following.

$$\begin{aligned} \max & \sum_{i=1}^m b_i y_i = b^\top y \\ \text{s.t. } & \sum_{i=1}^m A_i y_i \preceq C \end{aligned}$$

This of course means that $C - \sum A_i y_i \succeq 0$. If we set $S = C - \sum A_i y_i$ and thus $S \succeq 0$ then it is clear that this constraint can be rewritten as $\sum y_i A_i + S = C$ for $S \succeq 0$.

$$\begin{aligned} & \max b^\top y \\ \text{s.t. } & \sum_{i=1}^m A_i y_i + S = C \\ & S \succeq 0 \end{aligned}$$

Given an SDP in the form (12.1), we can convert it into an SDP in the form (12.3), and vice versa - this requires about a page of basic linear algebra.

The primal form of an SDP can be expressed as:

$$\begin{aligned} & \underset{X}{\text{minimize}} \quad \langle C, X \rangle \\ \text{subject to } & \langle A_i, X \rangle = b_i, \quad i = 1, \dots, m, \\ & X \succeq 0, \end{aligned} \tag{9.1}$$

where X is the variable matrix, C is the cost matrix, A_i are the constraint matrices, b_i are scalars, and $X \succeq 0$ denotes that X is semidefinite. The inner product $\langle A, B \rangle$ is defined as $\text{Tr}(A^\top B)$.

The dual problem is

$$\begin{aligned} & \underset{y, Z}{\text{maximize}} \quad \sum_{i=1}^m y_i b_i \\ \text{subject to } & \sum_{i=1}^m y_i A_i + S = C, \\ & S \succeq 0. \end{aligned} \tag{9.2}$$

SDP Duality Derivation

Primal Problem

Minimize $\langle C, X \rangle$ subject to:
 $\langle A_i, X \rangle = b_i, \quad i = 1, \dots, m,$
 $X \succeq 0.$

Dual Problem

Maximize $\sum_{i=1}^m y_i b_i$ subject to:
 $\sum_{i=1}^m y_i A_i + S = C,$
 $S \succeq 0.$

We use Lagrangian duality. The Lagrangian for the primal problem can be written as:

$$\mathcal{L}(X, y, S) = \langle C, X \rangle + \sum_{i=1}^m y_i (b_i - \langle A_i, X \rangle) + \langle S, X \rangle, \quad (9.3)$$

where y_i are the dual variables associated with the equality constraints, and S is a semidefinite matrix introduced for the constraint $X \succeq 0$.

The dual function $q(y, S)$ is the infimum of the Lagrangian over $X \succeq 0$:

$$q(y, S) = \inf_{X \succeq 0} \mathcal{L}(X, y, S). \quad (9.4)$$

For the dual function to be bounded from below (i.e., not $-\infty$), the coefficient of X in the Lagrangian must equal zero. This yields the constraint for the dual problem:

$$C - \sum_{i=1}^m y_i A_i = S. \quad (9.5)$$

Substituting the condition back into the Lagrangian and focusing on the terms involving y and S , we get the dual objective:

$$\sum_{i=1}^m y_i b_i. \quad (9.6)$$

Thus, the dual SDP seeks to maximize this objective subject to $S \succeq 0$ and the derived constraint from the Lagrangian. This completes the derivation, showing how the dual formulation is directly related to the primal SDP through Lagrangian duality.

Example

Consider the following simple SDP:

$$\begin{aligned} & \underset{X}{\text{minimize}} && \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \bullet X \\ & \text{subject to} && \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \bullet X = 1, \\ & && X \succeq 0. \end{aligned}$$

To form the dual, we introduce a dual variable y for the constraint and a semidefinite matrix S for ensuring $X \succeq 0$. The dual problem, according to the corrected notation and understanding, is:

$$\begin{aligned} & \underset{y, S}{\text{maximize}} \quad y \\ & \text{subject to} \quad y \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + S = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\ & \quad S \succeq 0. \end{aligned}$$

9.2 Lagrangian Dual of QCQP

Consider a nonconvex QCQP of the form:

$$\begin{aligned} & \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x) = x^T Q_0 x + c_0^T x + d_0 \\ & \text{subject to} \quad g_i(x) = x^T Q_i x + c_i^T x + d_i \leq 0, \quad i = 1, \dots, m, \end{aligned}$$

where $Q_0, Q_i \in \mathbb{R}^{n \times n}$ are symmetric matrices that are not necessarily positive semidefinite (making the problem nonconvex), $c_0, c_i \in \mathbb{R}^n$ are vectors, and $d_0, d_i \in \mathbb{R}$ are scalars.

The Lagrangian $\mathcal{L}(x, \lambda)$ of the QCQP is given by:

$$\mathcal{L}(x, \lambda) = f(x) + \sum_{i=1}^m \lambda_i g_i(x),$$

where $\lambda = [\lambda_1, \dots, \lambda_m]^T$ are the Lagrange multipliers associated with the constraints, with $\lambda_i \geq 0$ for all i .

The Lagrangian dual function $d(\lambda)$ is the infimum of the Lagrangian over x , for a given λ :

$$d(\lambda) = \inf_{x \in \mathbb{R}^n} \mathcal{L}(x, \lambda).$$

The dual problem seeks to maximize this dual function over λ , subject to $\lambda \geq 0$.

Transformation to a Convex SDP

To show that the dual problem is a convex SDP, we first express the infimum of the Lagrangian in a form that reveals its convexity. The key observation is that for any fixed λ , the function $L(x, \lambda)$ is a quadratic function of x , and its infimum can be represented in terms of a semidefinite condition.

Specifically, $L(x, \lambda)$ can be rewritten as:

$$L(x, \lambda) = x^T \left(Q_0 + \sum_{i=1}^m \lambda_i Q_i \right) x + \left(c_0 + \sum_{i=1}^m \lambda_i c_i \right)^T x + d_0 + \sum_{i=1}^m \lambda_i d_i.$$

For $d(\lambda)$ to be bounded below (and not $-\infty$), the matrix $Q_0 + \sum_{i=1}^m \lambda_i Q_i$ must be positive semidefinite. This condition ensures that $L(x, \lambda)$ is convex in x for any fixed λ , and its infimum is well-defined.

Therefore, the dual problem:

$$\underset{\lambda \geq 0}{\text{maximize}} \quad d(\lambda)$$

is equivalent to a convex SDP because it maximizes a linear function of λ over the convex set defined by the semidefinite constraints on the matrices $Q_0 + \sum_{i=1}^m \lambda_i Q_i$.

$$\max_{\lambda, S} \left\{ - \sum_{j=1}^m \lambda_j c_j : \sum_{j=1}^m \lambda_j Q_j + S = Q_0, S \succcurlyeq 0, \sum_{j=1}^m \lambda_j b_j = b_0, \lambda_j \geq 0 \right\}.$$

Here, λ represents the dual variables associated with the linear equality constraints of the primal problem, and S is a semidefinite matrix ensuring the semidefiniteness of the combined Q_j matrices aligned with Q_0 .

The dual of the dual is:

$$(SDP) \quad \begin{cases} \inf_{X \in \mathcal{S}_n} \langle Q_0, X \rangle + b_0^T x, & X \in \mathcal{S}_n, x \in \mathbb{R}^n, \\ \langle Q_j, X \rangle + b_j^T x + c_j = 0, & j = 1, \dots, m, \\ \begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \succcurlyeq 0. \end{cases}$$

Exercise 6 :

Consider the primal quadratic programming problem with a single quadratic constraint as described below:

$$\begin{aligned} & \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2} x^T Q x \\ & \text{subject to} \quad \frac{1}{2} x^T x \leq \frac{1}{2}, \end{aligned}$$

where Q is an indefinite, symmetric, rational matrix of size n .

1. Derive the Lagrangian dual of this problem. Start by explicitly writing the Lagrangian, then work through to find the dual problem.
2. Solve the dual problem.
3. Find a corresponding solution to the primal problem.
4. Conclude that there is no duality gap for this problem.

9.3 Convex Quadratic Reformulations

We solve the quadratic optimization problem given by:

$$\min_{x \in X} f(x) = x^\top Qx + c^\top x + d, \quad (9.1)$$

where Q is a symmetric matrix that is not necessarily positive semidefinite, c is a vector, and d is a scalar, and the optimization is over $x \in X$.

9.3.1. Convex Quadratic Reformulation by Diagonal Perturbation

Convex quadratic reformulation by diagonal perturbation is a technique used to address the challenges posed by non-convex quadratic constraints in optimization problems.

This method involves decomposing a quadratic function into two parts: a convex part that retains the tractable properties of convex optimization and a concave part that is separable and can be linearized effectively. The key to this approach is the introduction of a diagonal perturbation that facilitates the decomposition, allowing for a more manageable treatment of the non-convexity inherent in the problem.

Decomposition into Convex and Concave Parts

Consider a general quadratic function

$$f(x) = x^\top Qx + c^\top x + d$$

The goal is to decompose $f(x)$ into a convex quadratic function $f_{\text{convex}}(x)$ and a concave quadratic function $f_{\text{concave}}(x)$ such that

$$f(x) = f_{\text{convex}}(x) + f_{\text{concave}}(x)$$

The decomposition is achieved by identifying a diagonal matrix D with non-negative entries such that $Q + D$ is positive semidefinite (and thus convex) and D is as small as possible to maintain the structure of

the original problem. The function then becomes:

$$f_{\text{convex}}(x) = x^\top(Q + D)x + c^\top x + d, \quad (9.2)$$

and

$$f_{\text{concave}}(x) = -x^\top Dx. \quad (9.3)$$

Linearization of the Concave Part

Given the separability of the concave part $f_{\text{concave}}(x)$, we can linearize it using auxiliary variables y_i for each x_i^2 term, reducing the non-convexity to a set of constraints of the form

$$y_i = x_i^2$$

. This transformation allows us to replace the quadratic terms with linear ones, subject to additional constraints that capture the relationship between x_i and y_i .

$$\min f(x) = x^\top(Q + D)x + c^\top x + d - \sum_{i=1}^n D_{ii}y_i \quad (9.4a)$$

$$\text{such that } x \in X \quad (9.4b)$$

$$y_i = x_i^2 \quad \forall i = 1, \dots, n \quad (9.4c)$$

After applying the diagonal perturbation and decomposing the quadratic function, we introduce auxiliary variables y_i to linearize the concave part, leading to a reformulated problem that is more amenable to convex optimization techniques.

9.3.2. Choosing the Diagonal Perturbation Matrix D

The effectiveness of the convex quadratic reformulation method hinges on the appropriate choice of the diagonal perturbation matrix D . This section explores two primary strategies for selecting D : utilizing the smallest eigenvalue of Q and solving a semidefinite programming (SDP) problem to determine an optimal D .

9.3.2.1. Using the Smallest Eigenvalue of Q

One straightforward approach to determine D is based on the eigenvalues of the matrix Q . Specifically, the matrix D can be chosen as $D = \lambda_{\min}I$, where λ_{\min} is the smallest eigenvalue of Q , and I is the identity matrix. This choice ensures that $Q + D$ is positive semidefinite, as adding $\lambda_{\min}I$ effectively shifts all eigenvalues of Q to be non-negative, thus rendering $Q + D$ convex.

MATHEMATICAL JUSTIFICATION Given that Q has an eigenvalue decomposition $Q = V\Lambda V^\top$, where Λ is a diagonal matrix of eigenvalues and V is the matrix of eigenvectors, the smallest eigenvalue λ_{\min} can be added to each eigenvalue to ensure non-negativity. The resulting matrix $D = \lambda_{\min}I$ ensures that:

$$Q + D = V\Lambda V^\top + \lambda_{\min}I = V(\Lambda + \lambda_{\min}I)V^\top, \quad (9.5)$$

which is positive semidefinite since all eigenvalues in $\Lambda + \lambda_{\min}I$ are non-negative.

9.3.2.2. Solving a Semidefinite Programming (SDP) Problem

While using the smallest eigenvalue provides a simple method to determine D , a more refined approach involves solving an SDP problem to find an optimal D . This method seeks to minimize the perturbation introduced to the original problem by finding the smallest D that makes $Q + D$ positive semidefinite.

FORMULATION The optimal D can be found by solving the following SDP:

$$\begin{aligned} & \underset{D}{\text{minimize}} \quad \text{trace}(D) \text{ or } \|D\|_0 \text{ or} \\ & \text{subject to } Q + D \succeq 0, \\ & \quad D \text{ diagonal,} \\ & \quad D \geq 0. \end{aligned} \quad (9.6)$$

This optimization problem seeks to find the diagonal matrix D with the smallest trace (sum of diagonal elements) that ensures $Q + D$ is positive semidefinite ($\succeq 0$). This approach directly targets the perturbation's impact, striving for minimal alteration of the original problem's structure while ensuring convexity.

9.3.3. Diagonalization of Q and Variable Rotation for Perturbation

A nuanced method for achieving convex quadratic reformulation involves diagonalizing the matrix Q of the quadratic function and then rotating the problem's variables according to this diagonalization. This process not only simplifies the representation of the quadratic function but also highlights a pathway to identifying an efficient perturbation that ensures convexity with minimal distortion of the original problem.

Eigendecomposition of Q

The process begins with the eigendecomposition of the matrix Q , which is expressed as $Q = V\Lambda V^\top$, where V is the matrix of eigenvectors and Λ is the diagonal matrix containing the eigenvalues of Q . This decomposition essentially transforms Q into a diagonal matrix Λ , with the transformation governed by the eigenvectors in V .

MATHEMATICAL BASIS Given a quadratic function $f(x) = x^\top Qx + c^\top x + d$, applying the eigendecomposition of Q allows the function to be rewritten in terms of the new variables $y = V^\top x$, leading to:

$$f(y) = y^\top \Lambda y + (V^\top c)^\top y + d. \quad (9.7)$$

This transformation results in a new quadratic function where the quadratic term is now represented by the diagonal matrix Λ , simplifying the analysis and manipulation of the function.

Variable Rotation and Identifying Perturbation

The rotation of variables $y = V^\top x$ essentially aligns the problem with the axes defined by the eigenvectors of Q . This alignment exposes the structure of Q in its simplest form, where the eigenvalues directly indicate the convexity or concavity along the corresponding directions.

STRATEGIES FOR PERTURBATION Once Q is diagonalized, the selection of a perturbation matrix D becomes more intuitive:

- For negative eigenvalues (indicating concavity), D can be chosen to offset these values and bring them to non-negative values, thus ensuring convexity.
- The perturbation should be minimal, affecting only the directions necessary to achieve convexity, which can be directly inferred from the diagonal elements of Λ .

Utilizing Linear Equality Constraints for Objective Perturbation

Consider the optimization problem given by:

$$\min f(x) = x^\top Qx + c^\top x, \quad (9.8a)$$

$$\text{such that } Ax = b \quad (9.8b)$$

$$x_i \in \{0, 1\} \quad \text{for } i \in I \quad (9.8c)$$

$$x \in X \quad (9.8d)$$

Let $\alpha \in \mathbb{R}^{m \times n}$, $u \in \mathbb{R}^I$. Then an equivalent problem is

$$\min g_{\alpha,u}(x) = x^\top Qx + c^\top x + \sum_{k=1}^m \left(\sum_{i=1}^n \alpha_{ki} x_i \right) \left(\sum_{j=1}^n a_{kj} x_j - b_k \right) + \sum_{i \in I} u_i (x_i^2 - x_i) \quad (9.9a)$$

$$\text{such that } Ax = b \quad (9.9b)$$

$$x_i \in \{0, 1\} \quad \text{for } i \in I \quad (9.9c)$$

$$x \in X \quad (9.9d)$$

We can rewrite this in matrix notation as

$$g_{\alpha,u}(x) = x^\top Qx + c^\top x + \sum_{k=1}^m \left(\sum_{i=1}^n \alpha_{ki} x_i \right) \left(\sum_{j=1}^n a_{kj} x_j - b_k \right) + \sum_{i \in I} u_i (x_i^2 - x_i)$$

$$g_{\alpha,u}(x) = x^\top \left(Q + \frac{1}{2}(\alpha^\top A + A^\top \alpha) + \text{diag}(u) \right) x + (c^\top - \alpha^\top b - u^\top) x$$

Observation 9.1

Every Boolean Quadratic Program (all variables are binary) is equivalent to one where the objective is convex.

Let

$$X = \{x : Ax = b, x \in \{0,1\}^n\}$$

be the set of feasible solutions of problem (QP) and

$$\bar{X} = \{x : Ax = b, x \in [0,1]^n\}$$

be the set of feasible solutions of the continuous relaxation of (QP) . The general term of matrix A is denoted by a_{ij} and the general term of Q by q_{ij} .

9.3.4. Optimizing choice of α and u

Define

$$(C(QP)) : \underset{\substack{\alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n \\ Q_{\alpha,u} \succeq 0}}{\text{Max}} \underset{x \in \bar{X}}{\text{Min}} g_{\alpha,u}(x).$$

and (SDQP)

$$\min \quad c^t x + \sum_{i=1}^n \sum_{j=1}^n q_{ij} X_{ij} \tag{9.10a}$$

$$\text{s.t.} \quad X_{ii} = x_i, \quad i = 1, \dots, n \tag{9.10b}$$

$$-b_k x_i + \sum_{j=1}^n a_{kj} X_{ij} = 0, \quad k = 1, \dots, m; i = 1, \dots, n \tag{9.10c}$$

$$Ax = b \tag{9.10d}$$

$$\begin{pmatrix} 1 & x^t \\ x & X \end{pmatrix} \succeq 0 \tag{9.10e}$$

$$x \in \mathbb{R}^n, \quad X \in \mathbf{S}_n \tag{9.10f}$$

Theorem 9.2

The optimum value of $(C(QP))$ is equal to the optimum value of the semidefinite program $(SDQP)$. For problem $(C(QP))$, optimal values $u_i^*(i = 1 \dots, n)$ are given by the optimal values of the dual variables associated with constraints (9.10b) and optimal values $\alpha_{ki}^*(k = 1, \dots, m; i = 1, \dots, n)$ are given by the optimal values of the dual variables associated with constraints (9.10c).

Proof. Consider the quadratic function $z_{\alpha,u,\beta}(x) = g_{\alpha,u}(x) + \beta^t(Ax - b)$ depending on the three multidimensional parameters $\alpha \in \mathbb{R}^{m \times n}$, $u \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^m$.

Let $z_{\alpha,u,\beta}(x) = x^t Q_{\alpha,u} x + c_{\alpha,u,\beta}^t x - \beta^t b$ with $c_{\alpha,u,\beta}^t = c_{\alpha,u}^t + \beta^t A$.

By observing that $x_i^2 \leq x_i$ is equivalent to $0 \leq x_i \leq 1$, our convexification problem $(C(QP))$ can be written as (D1):

$$(D1): \underset{\substack{\alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n \\ Q_{\alpha,u} \succeq 0}}{\text{Max}} \underset{x \in \mathbb{R}^n}{\text{Min}} \left\{ g_{\alpha,u}(x) : Ax = b, x_i^2 \leq x_i \quad (i = 1, \dots, n) \right\}$$

$g_{\alpha,u}(x)$ is a convex function; the constraints $Ax = b$ and $x_i^2 \leq x_i$ define a convex set.

Assuming the Slater's interiority condition is satisfied then, by Lagrangian duality, (D1) is equivalent to (D2):

$$(D2) : \underset{\substack{\alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n, \beta \in \mathbb{R}^m, \lambda \in \mathbb{R}_+^n \\ Q_{\alpha,u} \succeq 0}}{\text{Max}} \underset{x \in \mathbb{R}^n}{\text{Min}} \left\{ g_{\alpha,u}(x) + \sum_{i=1}^n \lambda_i (x_i^2 - x_i) + \beta^t(Ax - b) \right\}$$

Claim: (D2) is also equivalent to (D3):

$$\begin{aligned} (D3) : & \underset{\substack{\alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n, \beta \in \mathbb{R}^m \\ Q_{\alpha,u} \succeq 0}}{\text{Max}} \underset{x \in \mathbb{R}^n}{\text{Min}} \left\{ g_{\alpha,u}(x) + \beta^t(Ax - b) \right\} \\ & = \underset{\substack{\alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n, \beta \in \mathbb{R}^m \\ Q_{\alpha,u} \succeq 0}}{\text{Max}} \underset{x \in \mathbb{R}^n}{\text{Min}} z_{\alpha,u,\beta}(x). \end{aligned}$$

Indeed, if $(\alpha^*, u^*, \beta^*, \lambda^*)$ is an optimal solution of (D2), $(\alpha^*, u^* = u^* + \lambda^*, \beta^*)$ is a feasible solution of (D3) with the same value. Moreover, the optimal value of (D2) is obviously greater than or equal to the optimal value of (D3).

It is well known that a necessary condition for the quadratic function $z_{\alpha,u,\beta}(x)$ to have a minimum not equal to $-\infty$ is that matrix $Q_{\alpha,u}$ is positive semidefinite. Hence, (D3) is equivalent to (D4):

$$(D4) : \underset{\alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n, \beta \in \mathbb{R}^m}{\text{Max}} \underset{x \in \mathbb{R}^n}{\text{Min}} z_{\alpha,u,\beta}(x).$$

This last problem is the Lagrangian dual obtained from (QP') by relaxing all the constraints:

$$(QP') : \underset{}{\text{Min}} \left\{ g(x) : Ax = b, x_i^2 = x_i \quad (i = 1, \dots, n), x_i \left(\sum_{j=1}^n a_{kj} x_j - b_k \right) = 0 \quad (i = 1, \dots, n; k = 1, \dots, m) \right\}.$$

Observe that (QP') is equivalent to the initial problem (QP) .

Following Lemaréchal and Oustry ([32], Corollary 4.2), the dual of (QP') is equivalent to the SDP problem (D5):

$$(D5) \quad \begin{cases} \text{Max} & r \\ \text{s.t.} & \begin{pmatrix} -\beta^t b - r & \frac{1}{2} c_{\alpha,u,\beta}^t \\ \frac{1}{2} c_{\alpha,u,\beta} & Q_{\alpha,u} \end{pmatrix} \succeq 0 \\ & r \in \mathbb{R}, \alpha \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n, \beta \in \mathbb{R}^m. \end{cases}$$

Following again Lemaréchal and Oustry ([32], Theorem 4.4), if we apply SDP duality to problem (D5), we get $(SDQP)$.

Note that there is no duality gap since

- (i) the feasible domain of $(SDQP)$ is nonempty (as (QP) admits a feasible solution) and then (D5) is bounded,
- (ii) (D5) satisfies Slater's condition.

For example, by setting $\alpha = \beta = 0$ and taking the u_i components positive and large, and r negative with $|r|$ large, it is possible to build a feasible matrix of (D5) that is positive definite. ♠

10. Discretization Approaches

10.1 Complete discretization - all binary variables

Let $x \in [l, u]$. Then we can replace x by an approximation in binary variables. Most common is to replace

$$x = l + \frac{1}{2^n} \sum_{i=1}^m 2^i \alpha_i \quad (10.1)$$

where $m = \lceil \log_2(u-l) \rceil$. More generally, any integer base $b > 2$ can be used as

$$x = l + \varepsilon \sum_{i=1}^n b^i \left(\sum_{k=0}^b k \alpha_{ik} \right) \quad \sum_{k=0}^b \alpha_{ik} = 1. \quad (10.2)$$

Note that for $b = 2$, we have variables $\alpha_{i0} + \alpha_{i1} = 1$, which reduces to the above case by eliminating the variable α_{i0} .

Under this transformation, the main problem is now a binary problem. Next, for any product appearing in the formulation,

$$\prod_{i \in I} \alpha_i$$

we can create a new variable $\beta_I = \prod_{i \in I} \alpha_i$. Since $\alpha_i \in \{0, 1\}$, $\beta_I \in \{0, 1\}$, and we can model their equivalence as

$$\beta_I \leq \alpha_i \quad \text{for all } i \in I \quad \text{and } 1 + \sum_{i \in I} (\alpha_i - 1) \leq \beta_I. \quad (10.3)$$

Hence, the resulting formulation is a linear binary program in variables α_i and β_I .

10.2 Products of continuous variables - NMDT

We want to linearize the product of two variables using discretization. That is, given

$$\begin{aligned} x &= s \cdot y \\ s_{\min} &\leq s \leq s_{\max} \\ y_{\min} &\leq y \leq y_{\max} \end{aligned} \quad (10.1)$$

we want to convert this to a linear model by using binary variables. There are two main steps to this procedure. First, we discretize s into binary variables and then we model the product of each binary variable with the continuous variable y .

10.2.1. Discretizing a continuous variable + small error

Discretizing a continuous variable + small error:

Suppose that we have a variable z with bounds

$$z_{\min} \leq z \leq z_{\max}. \quad (10.2)$$

We can transform z via a variable w to the unit interval as

$$\begin{aligned} z &= (z_{\max} - z_{\min})w + z_{\min} \\ 0 \leq w &\leq 1 \end{aligned} \quad (10.3)$$

Next, we can convert w into a discrete variable $w_d \in \left\{ \frac{i}{2^n} : i = 0, 1, \dots, 2^n \right\}$ and an error $\Delta w \in [0, \frac{1}{2^n}]$. This gives us

$$\begin{aligned} z &= (z_{\max} - z_{\min})(w_d + \Delta w) + z_{\min} \\ w_d &\in \left\{ \frac{i}{2^n} : i = 0, 1, \dots, 2^n \right\} \\ \Delta w &\in [0, \frac{1}{2^n}] \end{aligned} \quad (10.4)$$

Lastly, we convert the discrete w_d into binary variables as $w_d = \sum_{i=1}^n 2^{-i} \alpha_i$ with $\alpha_i \in \{0, 1\}$. Thus we have

$$\begin{aligned} z &= (z_{\max} - z_{\min}) \left(\sum_{i=1}^n 2^{-i} \alpha_i + \Delta w \right) + z_{\min} \\ \alpha_i &\in \{0, 1\} \text{ for } i = 1, \dots, n \\ \Delta w &\in [0, \frac{1}{2^n}] \end{aligned} \quad (10.5)$$

10.2.2. Binary and continuous variables

Modeling: product of binary and continuous variable:

$$\begin{aligned} x &= \delta \cdot y \\ y_{\min} \leq y &\leq y_{\max} \\ \delta &\in \{0, 1\} \end{aligned} \quad (10.6)$$

Thus, we want to have the following happen

$$x = \begin{cases} y & \text{if } \delta = 1 \\ 0 & \text{if } \delta = 0 \end{cases} \quad (10.7)$$

This can be accomplished by

Modeling: product of binary and continuous variable - Reformulation:

$$\begin{aligned} \delta y_{\min} &\leq x \leq \delta y_{\max} \\ (1 - \delta)y_{\min} &\leq \bar{x} \leq (1 - \delta)y_{\max} \\ y &= x + \bar{x} \\ \delta &\in \{0, 1\} \end{aligned} \quad (10.8)$$

Alternatively, we can eliminate the variable \bar{x} by $\bar{x} = y - x$, leaving us with

Modeling: product of binary and continuous variable - Equivalent reformulation:

$$\begin{aligned} \delta y_{\min} &\leq x \leq \delta y_{\max} \\ (1 - \delta)y_{\min} &\leq y - x \leq (1 - \delta)y_{\max} \\ \delta &\in \{0, 1\} \end{aligned} \quad (10.9)$$

10.2.3. Application to Pooling Problems

Resources

- Castro
- Akshay
- Beach

...

10.3 QCR + Discretization

Resources

- *Dong, Luo*
- *Beach, Hildebrand, Huchette*

10.4 Discretizations for QCQP

Resources

- *Beach, Hager, Hildebrand, Burlacu*

10.5 Adaptive Refinement Methods

Resources

- *An Adaptive Refinement Algorithm for Discretizations of Nonconvex QCQP - Akshay Gupta, Arie M. C. A. Koster, Sascha Kuhnke*

10.5.1. References

doi:10.1080/10556788.2016.1264397

11. Advanced Piecewise Linear Formulations

Resources

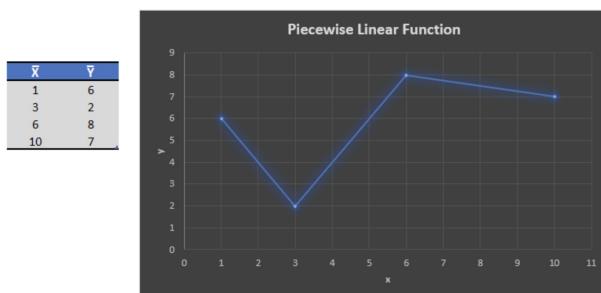
Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools = Joey Huchette and Juan Pablo Vielma Building Formulations for Piecewise Linear Relaxations of Non-linear Functions B Lyu, IV Hicks, J Huchette arXiv preprint arXiv:2304.14542

Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions Authors Juan Pablo Vielma, Shabbir Ahmed, George Nemhauser

The potential applications for this class of optimization problems are legion. Piecewise linear functions arise naturally throughout operations (Croxton et al. 2003, 2007, Liu and Wang 2015) and engineering (Fügenschuh et al. 2014, Graf et al. 1990, Silva et al. 2012). They are a natural choice for approximating nonlinear functions, as they often lead to optimization problems that are easier to solve than the original problem (Bergamini et al. 2005, 2008, Castro and Teles 2013, Geißler et al. 2012, Kolodziej et al. 2013, Misener et al. 2011, Misener and Floudas 2012). For example, there has been recently been significant interest in using piecewise linear functions to approximate complex nonlinearities arising in gas network optimization (Codas and Camponogara 2012, Codas et al. 2012, Martin et al. 2006, Mahlke et al. 2010, Misener et al. 2009, Silva and Camponogara 2014); see Koch et al. (2015) for a recent book on the subject.

Koch, T., B. Hiller, M. E. Pfetsch, L. Schewe, eds. 2015. Evaluating Gas Network Capacities. MOS-SIAM Series on Optimization, SIAM.

11.1 Standard Approaches



<https://math.stackexchange.com/questions/4065112/mip-modelling-of-piecewise-linear-function>

Suppose we have the following piecewise linear function $f(x)$:

$$\begin{aligned} & 3x + 8, x \in [0, 5] \\ & 33 - 2x, x \in [5, 10] \\ & 3 + x, x \in [10, 20] \end{aligned}$$

How to model the relations between $f(x)$ and x using integer variables and linear constraints. How to specify the values of M if we model it using big-M method.

Introduce a binary variable z_i for each segment and impose the following linear constraints:

$$\begin{aligned} z_1 + z_2 + z_3 &= 1 \\ 0z_1 + 5z_2 + 10z_3 &\leq x \leq 5z_1 + 10z_2 + 20z_3 \\ L_1(1 - z_1) &\leq y - (3x + 8) \leq U_1(1 - z_1) \\ L_2(1 - z_2) &\leq y - (33 - 2x) \leq U_2(1 - z_2) \\ L_3(1 - z_3) &\leq y - (3 + x) \leq U_3(1 - z_3) \end{aligned}$$

I'll leave the computations of the big-M values L_i and U_i to you.

<https://hochbaum.ier.berkeley.edu/files/ier269-2010.pdf>

Consider the following problem:

$$\min \sum_{i=1}^n f_i(x_i)$$

where $n \in \mathbf{N}$, and for $i \in \{1, \dots, n\}$, f_i is a piecewise linear function on k_i successive intervals. Interval j for function f_i has length w_i^j , and f_i has slope c_i^j on this interval, $(i, j) \in \{1, \dots, n\} \times \{1, \dots, k_i\}$. For $i \in \{1, \dots, n\}$, the intervals for function f_i are successive in the sense that the supremum of interval j for function f_i is the infimum of interval $j+1$ for function f_i , $j \in \{1, \dots, k_i - 1\}$. The infimum of interval 1 is 0, and the value of f_i at 0 is f_i^0 , for $i \in \{1, \dots, n\}$.

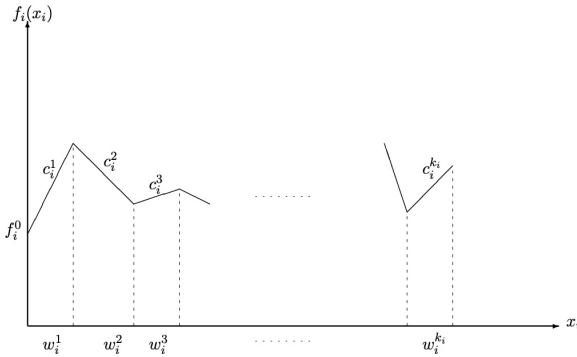


Figure 2: Piecewise linear objective function

We define the variable $\delta_i^j, (i, j) \in \{1, \dots, n\} \times \{1, \dots, k_i\}$, to be the length of interval j at which f_i is estimated. We have $x_i = \sum_{j=1}^{k_i} \delta_i^j$.

The objective function of this problem can be rewritten $\sum_{i=1}^n \left(f_i^0 + \sum_{j=1}^{k_i} \delta_i^j c_i^j \right)$. To guarantee that the set of δ_i^j define an interval, we introduce the binary variable:

$$y_i^j = \begin{cases} 1 & \text{if } \delta_i^j > 0 \\ 0 & \text{otherwise} \end{cases}$$

and we formulate the problem as:

$$\begin{aligned} & \min \sum_{i=1}^n \left(f_i^0 + \sum_{j=1}^{k_i} \delta_i^j c_i^j \right) \\ & \text{subject to } w_i^j y_i^{j+1} \leq \delta_i^j \leq w_i^j y_i^j \text{ for } (i, j) \in \{1, \dots, n\} \times \{1, \dots, k_i\} \\ & \quad y_i^j \in \{0, 1\}, \text{ for } (i, j) \in \{1, \dots, n\} \times \{1, \dots, k_i\} \end{aligned}$$

Written differently:

Method 7: INC - Incremental Formulation

In the incremental or delta method we add up all contributions of "earlier" segments, to find our x and y . Let s' be the segment that contains our current x . We define a binary variable δ_s as

$$\delta_s = \begin{cases} 1 & \text{for } s < s' \\ 0 & \text{for } s \geq s' \end{cases}$$

In addition we use a continuous variable $\lambda_s \in [0, 1]$ indicating how much of each segment we "use up". The contribution for earlier segments is 1 and for the current segment we have a fractional value. So we have:

$$\begin{cases} \lambda_s = 1 & \text{for } s < s' \\ \lambda_s \in [0, 1] & \text{for } s = s' \\ \lambda_s = 0 & \text{for } s > s' \end{cases}$$

With these definitions we can write:

$$\begin{aligned} x &= \bar{x}_1 + \sum_s \lambda_s (\bar{x}_{s+1} - \bar{x}_s) \\ y &= \bar{y}_1 + \sum_s \lambda_s (\bar{y}_{s+1} - \bar{y}_s) \end{aligned}$$

Now we need to formulate a structure that enforce these rules on δ and λ . The following will do that:

$$\lambda_{s+1} \leq \delta_s \leq \lambda_s$$

Typically you will need to implement this as two separate constraints.

The complete model looks like:

INC - Incremental Formulation

$$\begin{aligned} x &= \bar{x}_1 + \sum_s \lambda_s (\bar{x}_{s+1} - \bar{x}_s) \\ y &= \bar{y}_1 + \sum_s \lambda_s (\bar{y}_{s+1} - \bar{y}_s) \\ \lambda_{s+1} &\leq \delta_s \leq \lambda_s \\ \delta_s &\in \{0, 1\} \\ \lambda_s &\in [0, 1] \end{aligned}$$

<https://coral.ise.lehigh.edu/~ted/files/ie418/lectures/Lecture2.pdf>

- We can use binary variables to formulate arbitrary piecewise linear cost functions.
- The function is specified by ordered pairs $(a_i, f(a_i))$ and we wish to evaluate it at a point x .
- We have a binary variable y_i , which indicates whether $a_i \leq x \leq a_{i+1}$.
- To evaluate f , we take linear combinations $\sum_{i=1}^k \lambda_i f(a_i)$ of the given functions values.
- This only works if the only two nonzero λ'_i s are the ones corresponding to the endpoints of the interval in which x lies.

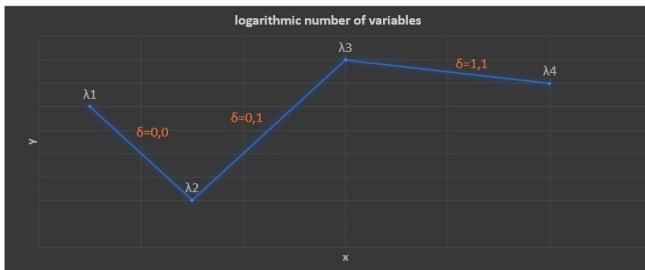
$$\begin{aligned}
 \min \quad & \sum_{i=1}^k \lambda_i f(a_i) \\
 \text{s.t.} \quad & \sum_{i=1}^k \lambda_i = 1, \\
 & \lambda_1 \leq y_1, \\
 & \lambda_i \leq y_{i-1} + y_i, \quad i \in [2..k-1], \\
 & \lambda_k \leq y_{k-1}, \\
 & \sum_{i=1}^{k-1} y_i = 1, \\
 & \lambda_i \geq 0, \\
 & y_i \in \{0, 1\}.
 \end{aligned}$$

- The key is that if $y_j = 1$, then $\lambda_i = 0, \forall i \neq j, j+1$.

11.2 Compact Forumulations

<https://yetanothermathprogrammingconsultant.blogspot.com/search?q=Piecewise+linear+functions+and+formulation>

Gray encoding of segments



The variables have the following role:

1. The δ variables are binary variables used to indicate which segment is selected.
2. The λ variables are continuous variables used to perform interpolation along the current segment.

In the picture above, we can see that if the first digit is 0, we can exclude the last segment, and breakpoint λ_4 . If we do this a bit systematically, we can formulate the implications:

$$\begin{aligned}
 \delta_1 = 0 &\Rightarrow \lambda_4 = 0 \\
 \delta_1 = 1 &\Rightarrow \lambda_1 = \lambda_2 = 0 \\
 \delta_2 = 0 &\Rightarrow \lambda_3 = \lambda_4 = 0 \\
 \delta_2 = 1 &\Rightarrow \lambda_1 = 0
 \end{aligned}$$

or

$$\begin{aligned}
 \lambda_4 &\leq \delta_1 \\
 \lambda_1 + \lambda_2 &\leq 1 - \delta_1 \\
 \lambda_3 + \lambda_4 &\leq \delta_2 \\
 \lambda_1 &\leq 1 - \delta_2
 \end{aligned}$$

Let's enumerate the results of these constraints. In the table below, we show all possible combinations for $\delta_b, b = 1, 2$. For each combination we mark with a 0 which λ_k 's will be fixed to zero.

Gray code implications					
δ_1	δ_2	λ_1	λ_2	λ_3	λ_4
0	0			0	0
0	1	0			0
1	1	0	0	0	0
1	0	0	0	0	

Why a Gray code?

The previous table explains the usefulness of the Gray code. If we would have used the standard binary encoding, we would not see the same table. Using the encoding: segment 1 = 00, segment 2 = 01, segment 3 = 10, we see:

Standard binary code implications					
δ_1	δ_2	λ_1	λ_2	λ_3	λ_4
0	0				0
0	1	0			0
1	0	0	0		0

This is not useful for our purposes. We want to allow two adjacent λ 's to be nonzero.

Developing a model

To make a model out of this, we define a boolean incidence matrix, where $v \in \{0, 1\}$.

$$I_{b,v,k} = \begin{cases} \text{true} & \text{if breakpoint } k \text{ is part of a segment } s \text{ that has binary digit } b \text{ equal to } v \\ \text{false} & \text{otherwise} \end{cases}$$

LOG - Logarithmic model

$$\begin{aligned} x &= \sum_k \bar{x}_k \lambda_k \\ y &= \sum_k \bar{y}_k \lambda_k \\ \sum_k \lambda_k &= 1 \\ \sum_{k \mid \text{not } I(b,0,k)} \lambda_k &\leq \delta_b \\ \sum_{k \mid \text{not } I(b,1,k)} \lambda_k &\leq 1 - \delta_b \quad \forall b \\ \lambda_k &\geq 0 \\ \delta_b &\in \{0, 1\} \\ k &\in \{1, \dots, K\} \\ s &\in \{1, \dots, K-1\} \\ b &\in \{1, \dots, \lceil \log_2(K-1) \rceil\} \end{aligned}$$

DCC - Disaggregated Convex Combination Formulation

$$\begin{aligned}
x &= \sum_{s,k|SK(s,k)} \bar{x}_k \lambda_{s,k} \\
y &= \sum_{s,k|SK(s,k)} \bar{y}_k \lambda_{s,k} \\
\delta_s &= \sum_{k|SK(s,k)} \lambda_{s,k} \\
\sum_s \delta_s &= 1 \\
\lambda_{s,k} &\geq 0 \\
\delta_s &\in \{0, 1\}
\end{aligned}$$

This model can be adapted to use the logarithmic encoding trick discussed before. We use a similar incidence table as before, but now it refers to $\lambda_{s,k}$. I.e.:

$$I_{b,v,s,k} = \begin{cases} \text{true} & \text{if breakpoint } k \text{ is part of a segment } s \text{ that has binary digit } b \text{ equal to } v \\ \text{false} & \text{otherwise} \end{cases}$$

where $v \in \{0, 1\}$.

This yields a model like:

DLOG - DCC Formulation with logarithmic number of binary variables.

$$\begin{aligned}
x &= \sum_{s,k|SK(s,k)} \bar{x}_k \lambda_{s,k} \\
y &= \sum_{s,k|SK(s,k)} \bar{y}_k \lambda_{s,k} \\
\sum_{s,k|SK(s,k)} \lambda_{s,k} &= 1 \\
\sum_{s,k|\text{not } I(b,0,s,k)} \lambda_{s,k} &\leq \delta_b \\
\sum_{s,k|\text{not } I(b,1,s,k)} \lambda_{s,k} &\leq 1 - \delta_b \quad \forall b \\
\lambda_{s,k} &\geq 0 \\
\delta_b &\in \{0, 1\} \\
k &\in \{1, \dots, K\} \\
s &\in \{1, \dots, K-1\} \\
b &\in \{1, \dots, \lceil \log_2(K-1) \rceil\}
\end{aligned}$$

1. Piecewise linear functions and formulations for interpolation (part 1), <http://yetanothermathprogrammingconsultant.blogspot.com/2019/02/piecewise-linear-functions-and.html>
2. Piecewise linear functions and formulations for interpolation (part 2), http://yetanothermathprogrammingconsultant.blogspot.com/2019/02/piecewise-linear-functions-and_22.html
3. L. J. Watters, Reduction of integer polynomial programming problems to zero-one linear programming problems, *Operations Research* 15, 11711174, 1967
4. Jaya Singhal, Roy E. Marsten, Thomas L. Morin, Fixed Order Branch-and-Bound Methods for Mixed-Integer Programming: The zoom System, *ORSA Journal on Computing*, 1989, vol. 1, issue 1, 44-51.
5. Juan Pablo Vielma, Shabbir Ahmed and George Nemhauser, Mixed-Integer Models for Nonseparable Piecewise Linear Optimization: Unifying Framework and Extensions, *Operations Research* 58(2):303-315, 2010
6. Juan Pablo Vielma and George L. Nemhauser, Modelling Disjunctive Constraints with a Logarithmic Number of Binary Variables and Constraints, *Mathematical Programming* 128 (2011), pp. 49-72.
7. Gray Code, https://en.wikipedia.org/wiki/Gray_code

Posted by Erwin Kalvelagen

11.3 Modern Formulations

Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools: Joey Huchette and Juan Pablo Vielma

Consider the function $f : D \rightarrow \mathbb{R}$, defined as follows:

- **Function Type:** f is a continuous piecewise linear function.
- **Domain:** $D \subset \mathbb{R}^n$ is bounded.

The function f is characterized by:

1. **Domain Pieces:** $\{C^i \subseteq D\}_{i=1}^d$.
2. **Affine Functions:** $\{f^i\}_{i=1}^d$.

We make the following assumptions:

- The domain pieces $\{C^i\}$ cover the domain D .
- The interiors of these domain pieces do not overlap.

- The function f is non-separable and cannot be decomposed into lower-dimensional piecewise linear functions.

Focus is primarily on:

- Low-dimensional cases: univariate ($n = 1$) or bivariate ($n = 2$) functions.
- Applications: Broadly applicable, especially under non-separability assumption.
- Example and notation: See Figure 1 for examples and Table 1 for notation.

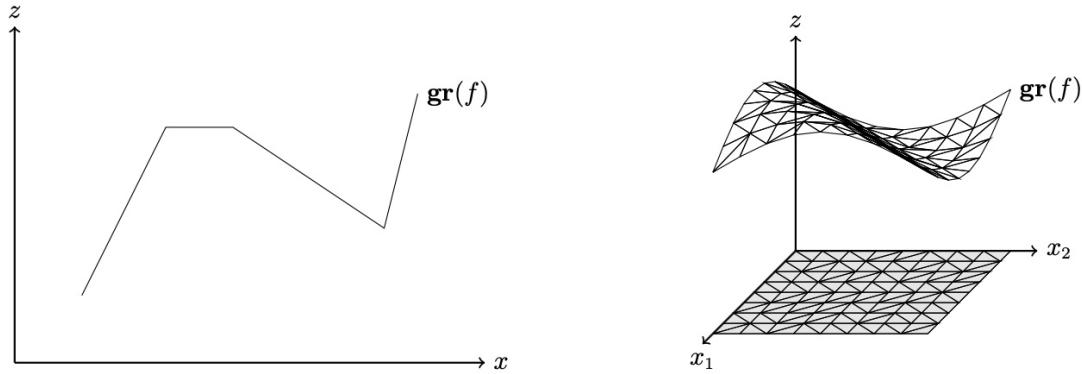


Figure 1 (Left) A univariate piecewise linear function, and (Right) a bivariate piecewise linear function with a grid triangulated domain.

To address an optimization problem involving f , we approach the construction of its graph.

Definition 11.1: Graph

The graph $\text{gr}(f)$ is defined as:

$$\text{gr}(f) \stackrel{\text{def}}{=} \{(x, f(x)) \mid x \in D\}$$

This formulation couples the argument x with the function output $f(x)$. The graph can be interpreted disjunctively, through the following structure:

- **Graph Union:** $\text{gr}(f) = \bigcup_{i=1}^d S^i$.
- Each $S^i = \{(x, f^i(x)) \mid x \in C^i\}$ represents a segment of the graph, indicating the relationship between x and $f^i(x)$ for each domain piece C^i .

This approach facilitates solving the optimization problem by providing a clear framework for analyzing the graph of f .

Example 11.2: Example 1.

Consider the univariate piecewise linear function $f : [1, 5] \rightarrow \mathbb{R}$ with the domain pieces $C^1 = [1, 2], C^2 = [2, 3], C^3 = [3, 4]$, and $C^4 = [4, 5]$, where

$$\begin{aligned} x \in C^1 &\implies f(x) = 4x - 4, & x \in C^2 &\implies f(x) = 3x - 2, \\ x \in C^3 &\implies f(x) = 2x + 1, & x \in C^4 &\implies f(x) = x + 5. \end{aligned}$$

The graph of the piecewise linear function is then

$$\text{gr}(f) = \{(x, 4x - 4) \mid x \in C^1\} \cup \{(x, 3x - 2) \mid x \in C^2\} \cup \{(x, 2x + 1) \mid x \in C^3\} \cup \{(x, x + 5) \mid x \in C^4\}.$$

Similarly, we take the bivariate piecewise linear function $g : [0, 1]^2 \rightarrow \mathbb{R}$ with the domain partition $C^1 = \{x \in [0, 1]^2 \mid x_1 \leq x_2\}$ and $C^2 = \{x \in [0, 1]^2 \mid x_1 \geq x_2\}$, and

$$x \in C^1 \implies g(x) = -x_1 + 3x_2 + 1, \quad x \in C^2 \implies g(x) = x_1 + x_2 + 1.$$

The corresponding graph is

$$\text{gr}(g) = \{(x_1, x_2, -x_1 + 3x_2 + 1) \mid x \in C^1\} \cup \{(x_1, x_2, x_1 + x_2 + 1) \mid x \in C^2\}.$$

Notation	Formal Definition	Description
$[d]$	$\{1, \dots, d\}$	All integers from 1 to d
$\mathbb{R}_{\geq 0}^n$	$\{x \in \mathbb{R}^n \mid x \geq 0\}$	Nonnegative orthant in n -dimensional space
Δ^V	$\{\lambda \in \mathbb{R}^V \mid \sum_{v \in V} \lambda_v = 1\}$	Unit simplex on ground set V
$\text{supp}(\lambda)$	$\{v \in V \mid \lambda_v \neq 0\}$	Nonzero values (support) of λ
$P(T)$	$\{\lambda \in \Delta^V \mid \text{supp}(\lambda) \subseteq T\}$	Face of the unit simplex given by components T
$\text{ext}(P)$	-	Extreme points of polyhedra P
$\text{gr}(f)$	$\{(x, f(x)) \mid x \in \text{dom}(f)\}$	Graph of the function f
$[V]^2$	$\{\{u, v\} \in V \times V \mid u \neq v\}$	All unordered pairs of elements in V
$\text{Em}(\mathcal{T}, H)$	$\bigcup_{i=1}^d P(T^i) \times \{H_i\}$	Embedding of disjunctive constraint (where H_i is the i -th row of H)
$\text{Conv}(S)$	-	Convex hull of S
$\mathcal{Q}(\mathcal{T}, H)$	$\text{Conv}(\text{Em}(\mathcal{T}, H))$	Convex hull of embedding
$\text{aff}(H)$	-	Affine hull of the rows of H
$L(H)$	$\{y - H_1 \mid y \in \text{aff}(H)\}$	Linear space parallel to the affine hull $\text{aff}(H)$ (where H_1 is first row of H)
$M(b)$	$\{y \in L(H) \mid b \cdot y = 0\}$	The hyperplane in $L(H)$ normal to b
$\text{Vol}(D)$	-	Volume of set D
$A * B$	$\{\{u, v\} \mid u \in A, v \in B\}$	Unordered pairs of elements in A and B

Table 1 Notation used throughout the paper.

Given a piecewise linear function, we analyze its combinatorial structure through the following:

- **Family of Sets \mathcal{T} :** Corresponds to the extreme points of each domain piece, $\mathcal{T} = (T^i = \text{ext}(C^i))_{i=1}^d$.
- **Shared Breakpoints:** Induces a combinatorial structure among graph segments over $V = \bigcup_{i=1}^d T^i$.

Define the key components as:

- **Standard Simplex Δ^V :** $\Delta^V \stackrel{\text{def}}{=} \{\lambda \in \mathbb{R}_{\geq 0}^V \mid \sum_{v \in V} \lambda_v = 1\}$.
- **Support $\text{supp}(\lambda)$:** Nonzero values of λ , $\text{supp}(\lambda) \stackrel{\text{def}}{=} \{v \in V \mid \lambda_v \neq 0\}$.
- **Face of the Simplex $P(T)$:** With support restricted to T , $P(T) \stackrel{\text{def}}{=} \{\lambda \in \Delta^V \mid \text{supp}(\lambda) \subseteq T\}$.

The graph $\text{gr}(f)$ can thus be expressed as:

$$\text{gr}(f) = \left\{ \sum_{v \in V} \lambda_v (v, f(v)) \mid \lambda \in \bigcup_{i=1}^d P(T^i) \right\}$$

This leads to a formulation for f through the combinatorial disjunctive constraint on convex multipliers λ , where each alternative $P(T^i)$ represents a face of the unit simplex Δ^V .

Reference: Huchette and Vielma (2019a)

Graphs of Functions f and g

Function f :

- The graph has $d = 4$ segments, with breakpoints defined by $V = [d + 1]$.
- A point (x, z) belongs to $\text{gr}(f)$ iff:

$$(x, z) = \sum_{v \in V} (v, f(v)) \lambda_v \text{ for some } \lambda \in \bigcup_{i=1}^4 P(\{i, i+1\})$$

Function g :

- For g , take $V = \{0, 1\}^2$.
- A point (x, z) belongs to $\text{gr}(g)$ iff:

$$(x, z) = \sum_{v \in V} (v, f(v)) \lambda_v \text{ for some } \lambda \in P(\{(0,0), (1,0), (1,1)\}) \cup P(\{(0,0), (0,1), (1,1)\})$$

Observations:

- Both functions illustrate how the graph of a piecewise linear function can be constructed from the sum of its breakpoints multiplied by some convex multipliers λ .
- The specific sets $P(T)$ for λ highlight the relationship between the graph segments and the combinatorial structure of the function.

11.4 (Section 3.) Formulations for univariate piecewise linear functions

In this section we will adapt a geometric formulation construction method to build novel strong logarithmic formulations for univariate piecewise linear functions.

11.4.0.1. (subsection 3.1) The embedding approach

Vielma's Embedding Approach (2018):

- **Objective:** Construct strong formulations for disjunctive constraints.
- **Method:** Assign to each alternative $P(T^i)$ a unique integer code $H_i \in \mathbb{Z}^r$.
- **Encoding:** The collection of all codes is represented as rows in a matrix $H \in \mathbb{Z}^{d \times r}$.
- **Embedding:** The disjunctive set is embedded in a higher-dimensional space as:

$$\text{Em}(\mathcal{T}, H) \stackrel{\text{def}}{=} \bigcup_{i=1}^d (P(T^i) \times \{H_i\})$$

Binary vs. General Integer Encodings:

- **Binary Encoding:** For $H \in \{0, 1\}^{d \times r}$, leads to a straightforward Mixed Integer Programming (MIP) formulation.
- **General Integer Encoding:** Requires careful construction to ensure the validity of the formulation.

Note: Vielma's embedding technique provides a framework for handling disjunctive constraints by utilizing unique integer codes to represent each alternative, thereby facilitating the construction of strong MIP formulations.

Definition 11.3: Definition 1.

Take the matrix $H \in \mathbb{Z}^{d \times r}$, and the collection of its rows as $\Lambda = \{H_i\}_{i=1}^d$.

- H is in convex position if $\text{ext}(\text{Conv}(\Lambda)) = \Lambda$.
- H is hole-free if $\text{Conv}(\Lambda) \cap \mathbb{Z}^r = \Lambda$.

Take $\mathcal{H}_r(d) \stackrel{\text{def}}{=} \{H \in \mathbb{Z}^{d \times r} \mid H \text{ is hole-free and in convex position, and each } H_i \text{ is distinct}\}$.

The following straightforward extension of Proposition 1 and Corollary 1 in Vielma (2018) shows that encodings in $\mathcal{H}_r(d)$ always lead to valid formulations.

Proposition 11.4: Proposition 1.

Take the family of sets $\mathcal{T} = (T^i \subseteq V)_{i=1}^d$, along with $r \geq \lceil \log_2(d) \rceil$ and $H \in \mathcal{H}_r(d)$. Then $Q(\mathcal{T}, H) \stackrel{\text{def}}{=} \text{Conv}(\text{Em}(\mathcal{T}, H))$ is a rational polyhedron, and an ideal formulation for $\bigcup_{i=1}^d P(T^i)$ is $\{(\lambda, y) \in Q(\mathcal{T}, H) \mid y \in \mathbb{Z}^r\}$. We call this the embedding formulation of \mathcal{T} associated to H .

Proposition 11.5: Proposition 2.

Take $H \in \mathcal{H}_r(d)$, along with $H_0 \equiv H_1$ and $H_{d+1} \equiv H_d$ for notational convenience.

Let $\mathcal{B} \subset L(H) \setminus \{0^r\}$ be normal directions such that $\{M(b)\}_{b \in \mathcal{B}}$ is the set of hyperplanes spanned by $\{H_{i+1} - H_i\}_{i=1}^{d-1}$ in $L(H)$. If $\mathcal{T} = (\{i, i+1\})_{i=1}^d$ is the family of sets defining the SOS2 constraint on $d+1$ breakpoints, then $Q(\mathcal{T}, H)$ is equal to all $(\lambda, y) \in \Delta^{d+1} \times \text{aff}(H)$ such that

$$\sum_{v=1}^{d+1} \min \{b \cdot H_{v-1}, b \cdot H_v\} \lambda_v \leq b \cdot y \leq \sum_{v=1}^{d+1} \max \{b \cdot H_{v-1}, b \cdot H_v\} \lambda_v \quad \forall b \in \mathcal{B}.$$

Gray Codes and Logarithmic Embedding Formulation**Gray Codes:**

- **Definition:** A sequence of codes where adjacent codes differ by exactly one component.
- **Notation:** $K^r \in \mathcal{H}_r(d)$ for $r = \lceil \log_2(d) \rceil$.
- **Property:** $\|K_{j+1}^r - K_j^r\|_1 = 1$ for all $j \in [d-1]$.
- **Advantage:** Facilitates parsimonious and simple descriptions for spanning hyperplanes.

Binary Reflected Gray Code (BRGC):

- A specific Gray code variant known for its simplicity and efficacy in encoding.
- Used for constructing the Logarithmic Embedding (LogE) formulation for the SOS2 constraint.

Definition 11.6: Formula for gray code

The formula to generate the r -th term of the binary reflected Gray code from a binary number b (written with r digits) is given by:

$$K_b^r = b \oplus (\lfloor b/2 \rfloor)$$

where \oplus denotes the bitwise XOR operation, and $\lfloor \cdot/2 \rfloor$ represents the right-shift operation by one position (equivalent to integer division by 2).

To clarify with an example, if you start with a binary number $b = 0101$ (which is 5 in decimal), the right-shifted version would be 0010 (which is 2 in decimal), and the XOR of 0101 \oplus 0010 would result in 0111, which is the Gray code for 5.

This process can be applied to each number in a sequence to generate a series of Gray codes.

Logarithmic Embedding (LogE) Formulation:

- **Application:** Applies Proposition 2 with the BRGC for SOS2 constraints.
- **Characteristics:** Ideal formulation with size scaling logarithmically with the number of segments d .
- **Benefit:** Provides an efficient and scalable approach to handle SOS2 constraints in optimization problems.

Reference: Vielma (2018) and Savage (1997) for the foundational concepts of Gray codes and their applications in optimization.

Example 3. The LogE formulation for the SOS2 constraint with $d = 4$ (arising in (1)) is:

$$\lambda_3 \leq y_1, \quad \lambda_1 + \lambda_5 \leq 1 - y_1, \quad \lambda_4 + \lambda_5 \leq y_2, \quad \lambda_1 + \lambda_2 \leq 1 - y_2, \quad (\lambda, y) \in \Delta^V \times \{0, 1\}^2.$$

11.4.1. (Subsection 3.3.) New zig-zag formulations for the SOS2 constraint

Addressing Degenerate Branching:

- Objective: Retain the size and strength of the Logarithmic Embedding (LogE) formulation while improving branching behavior.
- Assumption: d is a power-of-two for simplicity. Otherwise, extend to $\bar{d} = 2^{\lceil \log_2(d) \rceil}$ and set excess λ_v variables to zero.

New Encoding Techniques:

1. **Transformation to C^r :** From $K^r \in \{0, 1\}^{d \times n}$ to $C^r \in \mathbb{Z}^{d \times r}$, capturing the cumulative changes in the BRGC sequence.

$$C_{i,k}^r = \sum_{j=2}^i |K_{j,k}^r - K_{j-1,k}^r|$$

2. **Definition of Z^r :** A binary encoding derived from C^r using a linear map \mathcal{A} , where $\mathcal{A} : \mathbb{R}^r \rightarrow \mathbb{R}^r$ is the linear map given by $\mathcal{A}(y)_k = y_k - \sum_{\ell=k+1}^r y_\ell$ for each component $k \in [r]$.

Adjusting for cumulative changes.

$$Z_i^r = \mathcal{A}(C_i^r)$$

Application and Benefits:

- These encodings offer small, strong formulations for the SOS2 constraint, improving upon the LogE formulation's branching behavior.
- Demonstrated for $r = 3$ (see Figure 3), with formal definitions and proofs in Appendix A.

Note: The new encodings C^r and Z^r within $\mathcal{H}_r(d)$ framework ensure that the SOS2 constraint formulations are both compact and robust, addressing previous limitations in branching behavior.

Proposition 11.7: Proposition 3.

Take $r = \lceil \log_2(d) \rceil$, along with $C_0^r \equiv C_1^r$ and $C_{d+1}^r \equiv C_d^r$ for notational simplicity.

Then two ideal formulations for the SOS2 constraint with d segments are given by

$$\sum_{v=1}^{d+1} C_{v-1,k}^r \lambda_v \leq y_k \leq \sum_{v=1}^{d+1} C_{v,k}^r \lambda_v \quad \forall k \in [r], \quad (\lambda, y) \in \Delta^{d+1} \times \mathbb{Z}^r$$

and

$$\sum_{v=1}^{d+1} C_{v-1,k}^r \lambda_v \leq y_k + \sum_{\ell=k+1}^r 2^{\ell-k-1} y_\ell \leq \sum_{v=1}^{d+1} C_{v,k}^r \lambda_v \quad \forall k \in [r], \quad (\lambda, y) \in \Delta^{d+1} \times \{0, 1\}^r.$$

We dub (5) the binary zig-zag (ZZB) formulation for the SOS2 constraint, as its associated binary encoding Z^r "zig-zags" through the interior of the unit hypercube (See Figure 3). We will refer to formulation (4) as the general integer zig-zag (ZZI) formulation because of its use of general integer encoding $C^r \in \mathbb{Z}^{d \times r}$. We emphasize that ZZI and ZZB are logarithmically-sized in d and ideal: the same size and strength as the existing LogE formulation.

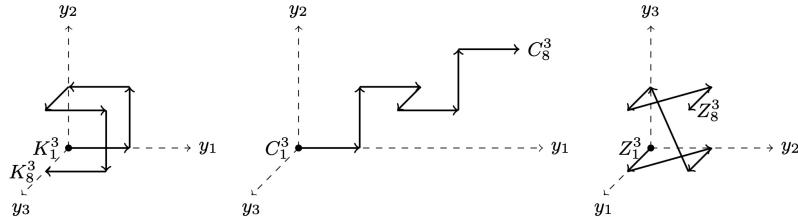


Figure 3 Depiction of K^3 (Left), C^3 (Center), and Z^3 (Right). The first row of each is marked with a dot, and the subsequent rows follow along the arrows. The axis orientation is different for Z^3 for visual clarity.

To study the branching behavior of the ZZI formulation, we return to the SOS2 constraint with $d = 4$ from Example 3. The formulation consists of all $(\lambda, y) \in \Delta^5 \times \mathbb{Z}^2$ such that

$$\lambda_3 + \lambda_4 + 2\lambda_5 \leq y_1 \leq \lambda_2 + \lambda_3 + 2\lambda_4 + 2\lambda_5, \quad \lambda_4 + \lambda_5 \leq y_2 \leq \lambda_3 + \lambda_4 + \lambda_5.$$

We have two possibilities for branching on y_1 , depicted in Figure 4: down on $y_1 \leq 0$ and up on $y_1 \geq 1$, or down on $y_1 \leq 1$ and up on $y_1 \geq 2$. We note that after imposing either $y_1 \leq 0$ or $y_1 \geq 2$, the relaxation is then exact, i.e. the relaxation is equal to exactly one of the segments of the graph

Part III

Working Material to be added

12. Other things...

12.1 Extended Space and Spatial Branch and Bound

We will discuss spatial branch and bound in the context of liftings of nonconvex problems to an extended space. This means that we will reformulate the problem by adding new variables. Specifically, consider a quadratic optimization problem with linear constraints (where Q is symmetric)

$$\min \quad x^\top Qx + c^\top x \tag{12.1}$$

$$Ax \leq b \tag{12.2}$$

$$x \in \mathbb{R}^n \tag{12.3}$$

Let $\mathcal{F} = \{x \in \mathbb{R}^n : Ax \leq b\}$ be the feasible region of the problem.

12.1.0.1. Extended Space

For any product terms $x_i x_j$ in the formulation, construct a new variable Y_{ij} as

$$Y_{ij} = x_i x_j. \tag{12.4}$$

If all product terms are present, this can be written as

$$Y = xx^\top. \tag{12.5}$$

We will use this notation, although it is not always necessary to constructed a lifted variable when the corresponding products are not present.

Now the reformulation of the quadratic minimization problem can be written as

$$\min \quad \tilde{Q} \circ Y + c^\top x \tag{12.6}$$

$$Ax \leq b \tag{12.7}$$

$$Y = xx^\top \tag{12.8}$$

$$x \in \mathbb{R}^n \tag{12.9}$$

$$Y \in \mathbb{R}^{n \times n} \tag{12.10}$$

The objective function is now linear, which is very nice!

Now, let

$$\mathcal{F}_{\text{lift}} = \{(x, Y) \in \mathbb{R}^n \times \mathbb{R}^{n \times n} : Ax \leq b, Y = xx^\top\}.$$

The set \mathcal{F} is non-convex, which is hard to deal with. So instead, we consider some convex relaxation \mathcal{R} of $\text{conv}(\mathcal{F})$. We restrict that

$$\text{conv}(\mathcal{F}) \subseteq \mathcal{R} \subseteq \{(x, Y) : Ax \leq b\},$$

that is, we enforce the original constraints from \mathcal{F} on the x variables. Thus, for any solution $(\bar{x}, \bar{Y}) \in \mathcal{R}$, although it may be that $(\bar{x}, \bar{Y}) \notin \mathcal{F}$, we do still have that \bar{x} is feasible for the original problem.

$$\min \quad \tilde{Q} \circ Y + c^\top x \tag{12.11}$$

$$Ax \leq b \tag{12.12}$$

$$(x, Y) \in \mathcal{R} \tag{12.13}$$

12.1.1. Spatial Branch and Bound

The goal of spatial branch and bound is to sequentially identify subregions of the feasible set with lower bounds that are worse than the best known feasible solution, and hence you can *prune* (remove) that region from the space of places to look for an optimal solution.

In this algorithm, we build a *branch and bound tree* that starts with a root node and then branches on nodes as we go.

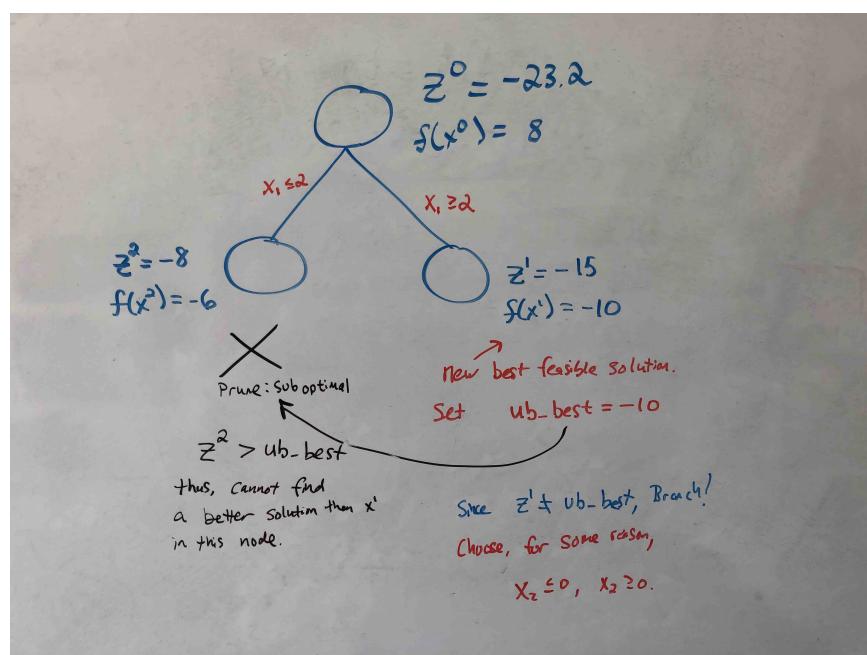
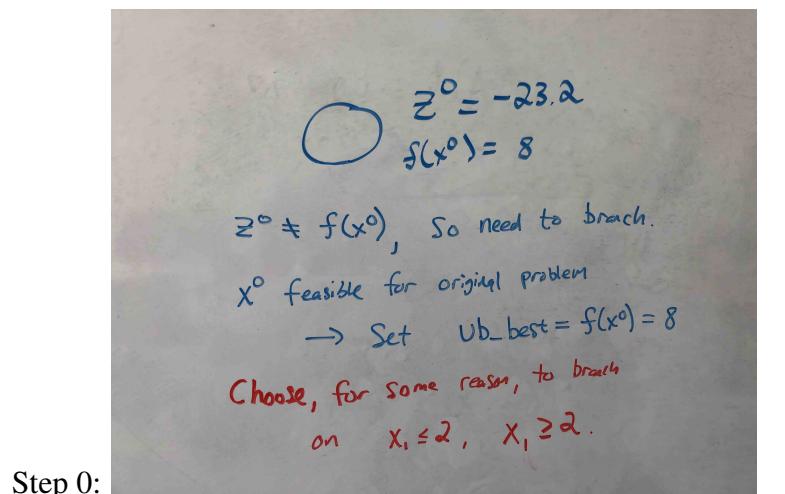
Process at a node:

Input: A feasible region $\mathcal{F}^i \subseteq \mathcal{F}$, a upper bound ubbest on $p_{\mathcal{F}}^*$

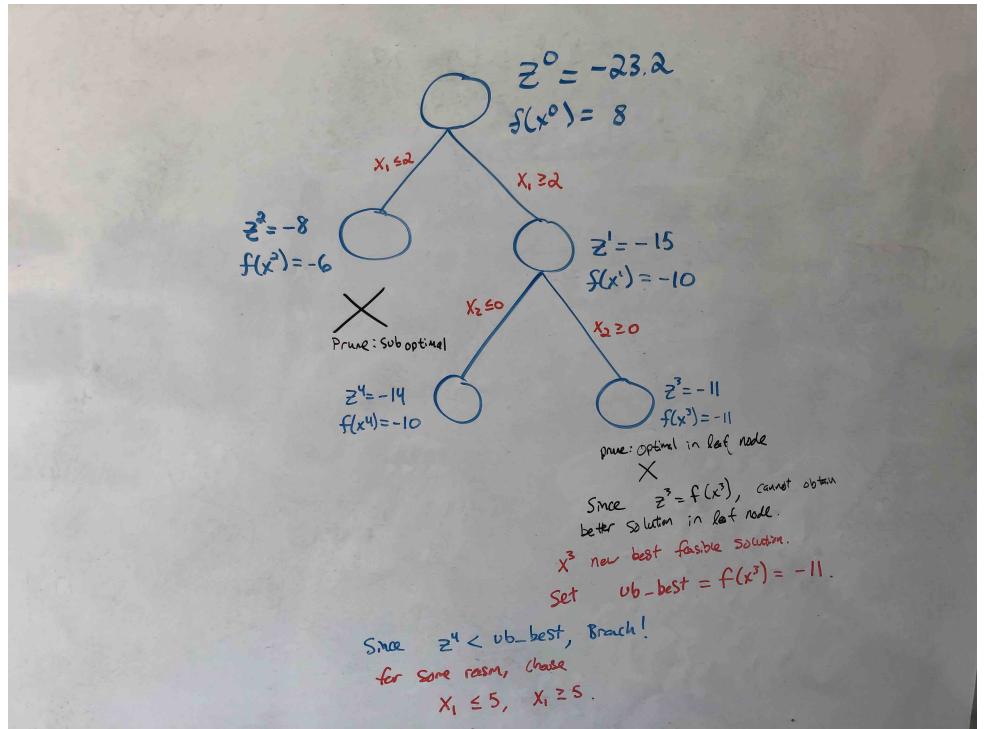
1. Write down the lifted problem (12.6).
2. Construct a relaxation \mathcal{R} of $\mathcal{F}_{\text{lift}}^i$.
3. Optimize the relaxed lifted problem (12.11) and return an objective value z^i and a point $(x^i, Y^i) \in \mathcal{R}$. Note that
 - $x^i \in \mathcal{F}^i \subseteq \mathcal{F}$
 - $p_{\mathcal{F}}^* \leq p_{\mathcal{F}^i}^* \leq f(x^i)$
 - $p_{\mathcal{F}^i}^* \geq z^i$
4. If $f(x^i) < \text{ubbest}$, set $\text{ubbest} \leftarrow f(x^i)$.
 - (a) If $\mathcal{F}^i = \emptyset$, i.e., is infeasible, **prune this node**.
 - (b) If $z^i > \text{ubbest}$, **prune this node**. No optimal solution for \mathcal{F} can be found in \mathcal{F}^i .

- (c) If $f(x^i) = z^i$, then $p_{\mathcal{F}_i}^* = f(x^i)$. Return x^i and **prune this node: found an optimal solution in \mathcal{F}^i** (but this may not be optimal for \mathcal{F}).
- (d) Else, if $z^i < f(x^i)$ and $z_i < \text{ubbest}$, then **Branch!**. Create two subproblems \mathcal{F}^{i+1} and \mathcal{F}^{i+2} such that $\mathcal{F}^i = \mathcal{F}^{i+1} \cup \mathcal{F}^{i+2}$, and process each node using this algorithm.

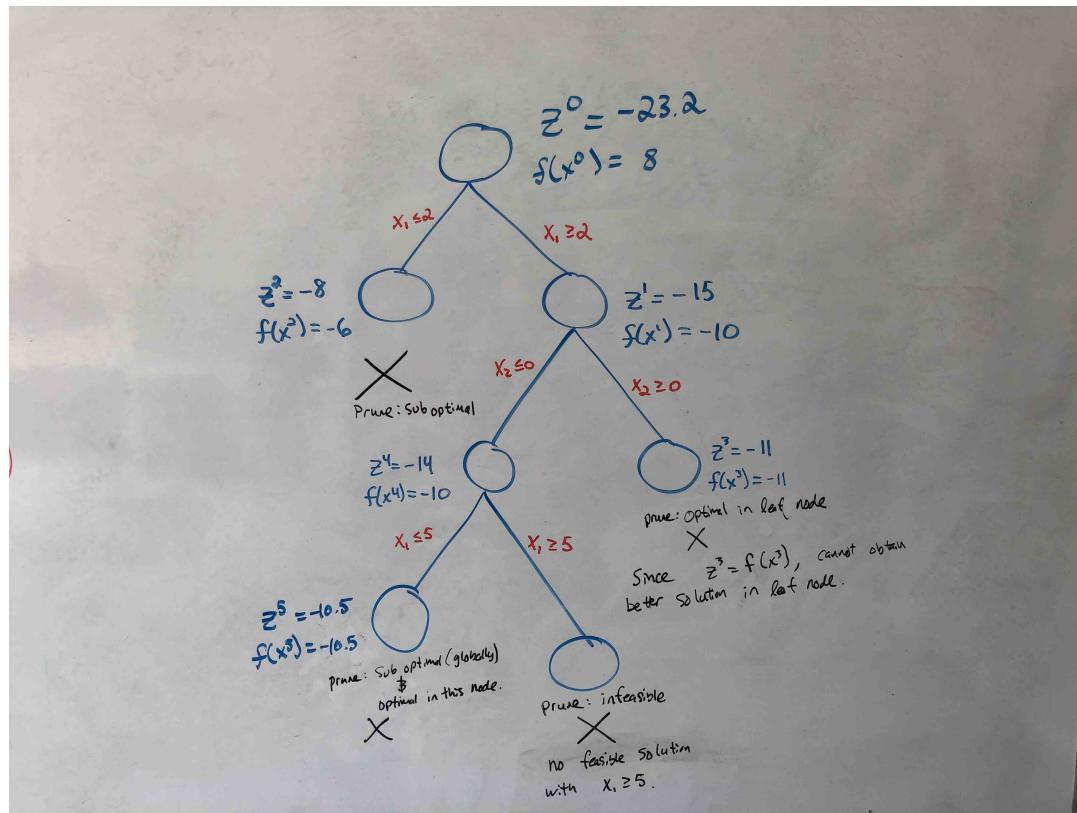
Here is an example of a branch and bound tree. It is important that each node ends in either infeasible, suboptimal, or optimal for the subproblem \mathcal{F}^i . Subproblems are created by changing lower or upper bounds on the variables x_i . The numbers in the example are fictitious and do not come from any particular function.

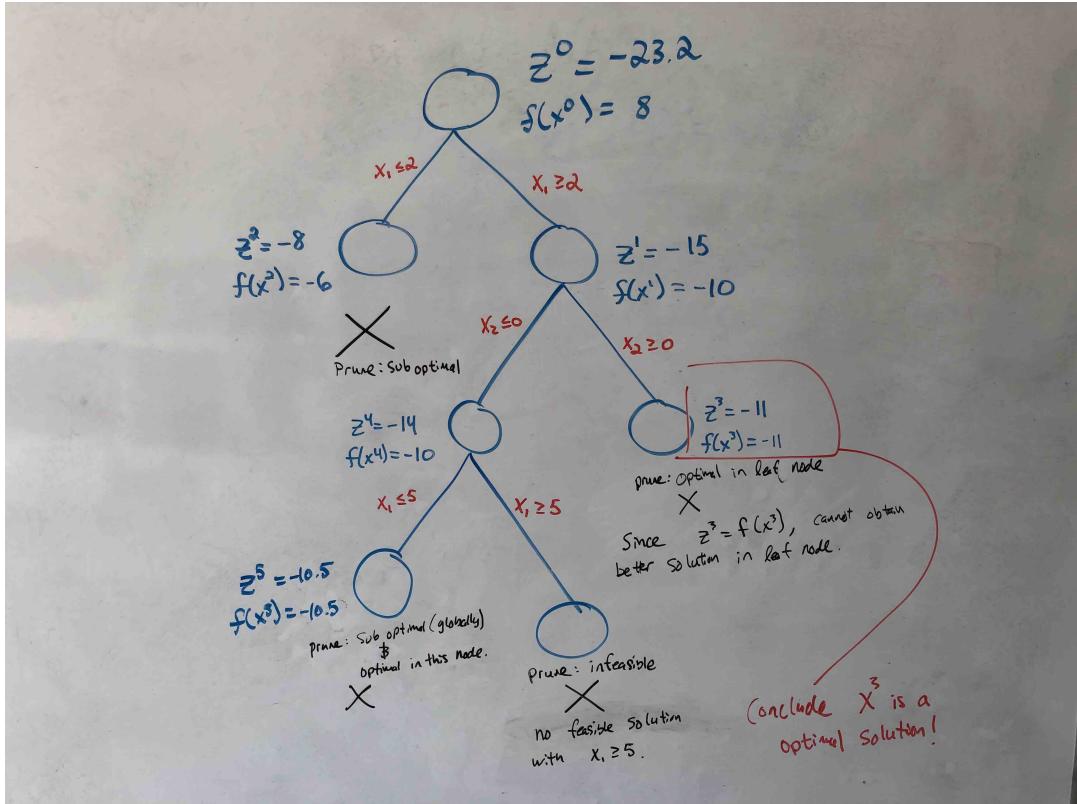


Step 2:



Step 3:





There are several methods of construction relaxations \mathbb{R} of $\mathcal{F}_{\text{lift}}$ including McCormick envelopes, using more inequalities from the Boolean Quadric Polytope, or SDP relaxations. These will be explored in the next few subsections.

12.2 McCormick Envelope

For two variables, $l_i \leq x_i \leq u_i$ and $l_j \leq x_j \leq u_j$, we first write these inequalities as

$$x_i - l_i \geq 0, \quad u_i - x_i \geq 0 \quad (12.1)$$

$$x_j - l_j \geq 0, \quad u_j - x_j \geq 0 \quad (12.2)$$

Multiplying any two inequalities yields a quadratic inequality, that can be linearized using the Y_{ij} variable. For example,

$$0 \leq (x_i - l_i)(x_j - l_j) = x_i x_j - l_i x_j - l_j x_i + l_i l_j = Y_{ij} - l_i x_j - l_j x_i + l_i l_j. \quad (12.3)$$

Enumerating all four pairs yields the McCormick Inequalities.

McCormick Inequalities:

$$l_i x_j + l_j x_i - l_i l_j \leq Y_{ij} \leq u_i x_j - l_j x_i + u_i l_j \quad (12.4)$$

$$u_i x_j + u_j x_i - u_i u_j \leq Y_{ij} \leq u_j x_i - l_i x_j + u_j l_i \quad (12.5)$$

12.3 Relaxation Linearization Technique (RLT)

The McCormick envelope inequalities are in fact a specific type of RLT inequality. In particular, for any two valid inequalities

$$a^1 x \leq b_1$$

$$a^2 x \leq b_2$$

the inequality

$$(a^1 x - b_1)(a^2 x - b_2) \geq 0$$

is also a valid inequality. Using the lifted variables this could be rewritten as

$$\Gamma \circ Y + \beta^\top x + \alpha \geq 0$$

which is a valid linear inequality for $\mathcal{F}_{\text{lift}}$. This inequality is referred to as an *RLT inequality*.

Resources

- [Sherali1992 - RLT for bilinear problems](#)
- [Sherali2013 - RLT for Discrete problems and Hierarchy to higher degrees](#)

12.3.1. Boolean Quadric Polytope (BQP)

In this section, we assume that the upper and lower bounds are normalized, that is,

$$0 \leq x \leq 1.$$

This can be achieved easily by an affine transformation of the problem. Alternatively, one can extract generalizations of the results in this section to arbitrary lower and upper bounds via an affine transformation in the reverse direction.

Boolean Quadric Polytope (BQP):

$$\text{BQP} = \text{conv} \left((x, Y) \in \{0, 1\}^{n+E} \mid Y_{ij} = x_i x_j \quad \forall (i, j) \in E \right) \quad (12.1)$$

Theorem 12.1

$$\text{BQP} = \text{conv} \left((x, Y) \in [0, 1]^{n+E} \mid Y_{ij} = x_i x_j \quad \forall (i, j) \in E \right) \quad (12.2)$$

Akshay-Gupte2019

Triangle Inequalities:

$$\begin{aligned} x_i + x_j + x_k - y_{ij} - y_{ik} - y_{jk} &\leq 1 \\ -x_i + y_{ij} + y_{ik} - y_{jk} &\leq 0 \\ -x_j + y_{ij} - y_{ik} + y_{jk} &\leq 0 \\ -x_k - y_{ij} + y_{ik} + y_{jk} &\leq 0 \end{aligned}$$

Clique Inequalities:

$$S \subseteq V, |S| \geq 3, 1 \leq \alpha \leq |S| - 2, \alpha \in \mathbb{Z}$$

$$\alpha x(S) - y(E(S)) \leq \frac{\alpha(\alpha+1)}{2}$$

? Inequalities:

Resources

<https://arxiv.org/pdf/2009.11674.pdf> - BQP for bilinear Problems

Letchford2022 - Fantastic Survey on BQP

12.3.2. SDP Relaxation

12.4 Lagrangian Relaxation

Consider a polynomial optimization problem

12.5 New Results

Theorem 12.2: Santana-Dey 2018

Let $S = \{x \in \mathbb{R}^n : x^\top Qx + c^\top T^\top x = g, x \in P\}$ for $P = \{x : Ax \leq b\}$ and Q symmetric. Then $\text{conv}(S)$ is SOCr. Proof is constructive.

Theorem 12.3: Gupte et. al. 2019

If G has nice structure, then $\text{BQP}(G)$ has a polynomial size extended formulation.

13. Approximations of SOCP and SDP

Some notes on applications for SDP: <http://www.seas.ucla.edu/~vandenbe/publications/sdp-apps.pdf>

13.1 Outer linear approximation of SDP

We describe a cutting plane technique for solving an SDP using Linear Programming and an oracle for finding eigenvectors of a matrix. This cutting plane technique could be applied to many other types of problems.

SDP:

$$\begin{aligned} & \min \quad c^\top x \\ \text{such that } & Ax \leq b \\ & \sum_{i=1}^n F_i x_i + G \preceq 0 \\ & x \in \mathbb{R}^n \end{aligned} \tag{*}$$

Let x^{LP} be the solution to the linear constraints to this problem, that is, we solve without the SDP constraint (*).

- If x^{LP} satisfies (*), then it is optimal.
- Otherwise, we find a new inequality $d^\top x \leq f$ that is valid for the problem but cuts off x^{LP} .

Let $A = \sum_{i=1}^n F_i x_i^{\text{LP}} + G$. If $A \not\preceq 0$, then we compute an eigenvector v of A with associated eigenvalue $\lambda > 0$.

Then,

$$v^\top \left(\sum_{i=1}^n F_i x_i + G \right) v \leq 0 \tag{13.1}$$

or equivalently

$$\sum_{i=1}^n v^\top F_i v x_i + v^\top G v \leq 0 \tag{13.2}$$

is a valid inequality for the problem that cuts off x^{LP} . Add this inequality to the LP and recompute x^{LP} . Continue this process until x^{LP} is feasible or close to feasible.

14. S-Lemma

Resources

Amir Ali Notes

15. Fractional MINLP

Convexification Techniques for Fractional Programs Taotao He, Siyue Liu, Mohit Tawarmalani

16. Binary Polynomial Optimization

Binary Polynomial Optimization: Theory, Algorithms, and Applications - Aida Khajavirad

Problem definition - Let $V = \{1, \dots, n\}$, let E be a set of subsets of cardinality at least two of V , and let V_1, V_2 be a partition of V . Consider the optimization problem:

$$\max \left\{ \sum_{e \in E} c_e \prod_{v \in e} z_v : z_v \in [0, 1] \forall v \in V_1, z_v \in \{0, 1\} \forall v \in V_2 \right\}.$$

- $V_1 = \emptyset$: Pseudo-Boolean optimization, unconstrained binary polynomial optimization, unconstrained binary nonlinear optimization - $V_2 = \emptyset$: maximizing a multilinear function over a box - Define $z_e := \prod_{v \in e} z_v$ for all $e \in E$:

$$\begin{aligned} \max \quad & \sum_{e \in E} c_e z_e \\ \text{s.t.} \quad & z_e = \prod_{v \in e} z_v, \forall e \in E \\ & z_v \in \{0, 1\}, \forall v \in V. \end{aligned}$$

The multilinear polytope - We define the multilinear set as:

$$\mathcal{S} = \left\{ z \in \{0, 1\}^{|V|+|E|} : z_e = \prod_{v \in e} z_v, \forall e \in E \right\}.$$

- Example:

$$\mathcal{S} = \left\{ z \in \{0, 1\}^8 : z_{12} = z_1 z_2, z_{24} = z_2 z_4, z_{123} = z_1 z_2 z_3, z_{134} = z_1 z_3 z_4 \right\}.$$

- We define the multilinear polytope as the convex hull of the multilinear set:

$$\text{MP} = \text{conv}(\mathcal{S})$$

- If $|e| = 2$ for all $e \in E$, then MP is the Boolean quadric polytope QP (Padberg, 89) and hence the cut polytope under a bijective linear transformation.

References - A. Del Pia and A. Khajavirad. A polyhedral study of binary polynomial programs. Mathematics of Operations Research, 2017. - A. Del Pia and A. Khajavirad. On decomposability of multilinear sets. Mathematical Programming, 2018. - A. Del Pia and A. Khajavirad. The multilinear polytope for acyclic hypergraphs, SIAM Journal on Optimization, 2018. - A. Del Pia and A. Khajavirad. The running intersection relaxation of the multilinear polytope, Mathematics of Operations Research, 2021. - A. Del Pia, A. Khajavirad, and N. V. Sahinidis. On the impact of running intersection inequalities for globally solving polynomial optimization problems, Mathematical Programming Computation, 2020. - A. Del Pia and A. Khajavirad. The multilinear polytope of beta-acyclic hypergraphs has polynomial extension complexity, arXiv:2212.11239, 2022.

17. Hierarchies

17.1 Lovasz-Schrijver, Sherali-Adams, Lasserre Hierarchies

Resources

- <https://web.stanford.edu/class/cs369h/lectures/lec1.pdf>
- <https://web.stanford.edu/class/cs369h/lectures/lec2.pdf>
- <https://web.stanford.edu/class/cs369h/lectures/lec3.pdf>

18. Generalized Benders Decomposition

Accelerating Generalized Benders Decomposition for Wireless Resource Allocation Mengyuan Lee, Ning Ma, Guanding Yu, and Huaiyu Dai

A. Problem Formulation for Wireless Resource Allocation

Resource allocation in wireless networks generally involves two kinds of variables: the continuous ones such as data rate and transmit power, and the discrete ones such as user selection and subcarrier assignment. Thus, many wireless resource allocation problems can be formulated as the following generic form of mixed integer optimization 1

$$\min_{\{x,y\}} f(x,y)$$

subject to

$$G(x,y) \leq 0,$$

$$x \in \mathcal{X}$$

$$y \in \mathcal{Y}$$

where $f(\cdot, \cdot)$ is the objective function such as energy consumption and communication delay, x is the vector of n_1 continuous variables, y is the vector of n_2 discrete variables, and $G(\cdot, \cdot)$ represents the vector of constraints defined on $\mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^{n_1} \times \mathbb{Z}^{n_2}$, such as power constraints and data rate requirements.

Problem (1) can be widely found in the study of wireless networks, such as joint user association and power allocation in carrier aggregation systems [2], optimal mode selection in D2D networks [3], resource allocation in multi-user mobileedge computation offloading [4], trajectory planning and recourse allocation for unmanned aerial vehicles [5], as well as joint power allocation and relay selection in amplify-andforward cooperative communication system [6].

Generally, Problem (1) is NP hard and it is computationally challenging to obtain its optimal solution. According to the linearity of $f(\cdot, \cdot)$ and $G(\cdot, \cdot)$ with fixing y , Problem (1) can be further categorized into two kinds: mixed integer linear programming (MILP) problems and MINLP problems. For MILP problems, both $f(\cdot, \cdot)$ and $G(\cdot, \cdot)$ with fixing y are supposed to be linear functions. Unfortunately, this condition is not met by most wireless resource allocation problems. Therefore, MILP problems are less often encountered than MINLP problems in wireless networks. The optimal algorithms for solving the MINLP problems include the B&B algorithm and the GBD algorithm. Generally, GBD has a simpler algorithm structure than the B&B algorithm. In the following, we introduce the basic idea of the GBD algorithm for MINLP problems.

B. Brief Introduction of GBD

The main idea of GBD is decomposing an MINLP problem into two sub-problems: a primal problem and a master problem. These two problems are iteratively solved until their solutions converge. To guarantee the convergence and the global optimality of the GBD algorithm, the MINLP problem in (1) should satisfy two conditions [8]: - Convexity: the problem should be convex on x given the discrete variables y ; -

Linear separability: the problem should be linear on y given the continuous variables x . Otherwise, GBD algorithm may fail to converge or converge to a local optimum. Note that linear separability is satisfied by many wireless resource allocation problems, such as Problem (6) in Section IV.A. In this paper, we only focus on the MINLP problems that satisfy the above two conditions, for which the GBD algorithm always converges to the global optimum. As for the non-convex MINLP problems, many variations of GBD have been proposed [27]-[29]. How to accelerate those modified GBD algorithms is interesting for future work.

For a convex and linear separable MINLP problem, the primal problem is also convex and can be solved by various convex optimization techniques. Specifically, the primal problem at the i -th iteration finds the optimal x by fixing the discrete variables y on a specific value $y^{(i-1)}$ (or a given initial $y^{(0)}$) and can be written as

$$\min_{x \in \mathcal{X}} f(x, y^{(i-1)})$$

subject to

$$G(x, y^{(i-1)}) \leq 0.$$

Note that some specific value $y^{(i-1)}$ may lead to an infeasible primal problem (2). If Problem (2) is feasible, the corresponding solution is denoted as $x^{(i)}$ and we add the iteration index i into the feasible primal problem index set, \mathcal{F} , as a new element. Clearly, the optimal value $f(x^{(i)}, y^{(i-1)})$ provides an upper bound to the original Problem (1). Given that Problem (2) is convex, solving the problem also provides the optimal Lagrange multipliers vector, i.e., the optimal dual variables vector, $\mu^{(i)}$, for constraint (2a). Then the Lagrange function can be written as

$$\mathcal{L}(x^{(i)}, y, \mu^{(i)}) = f(x^{(i)}, y) + \mu^{(i)T} G(x^{(i)}, y).$$

On the other hand, if Problem (2) is infeasible, we add the iteration index i into the infeasible primal problem index set, \mathcal{I} , as a new element and turn to an l_1 -minimization feasibilitycheck problem as follows [8]

$$\min_{\{x, \alpha\}} \alpha$$

subject to

$$\begin{aligned} G(x, y^{(i-1)}) &\leq \alpha, \\ x &\in \mathcal{X}, \\ \alpha &\geq 0. \end{aligned}$$

Problem (3) is named as feasibility-check problem because its result reflects the feasibility of Problem (2). Specifically, $\alpha = 0$ means a feasible point of Problem (2) has been found; otherwise, Problem (2) is infeasible. As mentioned above, we only turn to Problem (3) when Problem (2) is infeasible. Therefore, $\alpha = 0$ will not happen in the GBD algorithm. By solving the feasibility-check problem, we can get the Lagrange multiplier vector $\lambda^{(i)}$ for constraints (3a). The Lagrange function resulting from the infeasible primal problem is defined as

$$\overline{\mathcal{L}}(x^{(i)}, y, \lambda^{(i)}) = \lambda^{(i)T} (G(x^{(i)}, y) - \alpha).$$

Note that an upper bound is only obtained from a feasible primal problem.

As for the master problem, it is obtained based on nonlinear convex duality theory. Considering the Variant 2 of GBD (V2GBD) [8] without loss of generality, a relaxed master problem is given by

$$\min_{\{y, \eta\}} \eta$$

subject to

$$\begin{aligned} & y \in \mathcal{Y}, \\ & \eta \geq \mathcal{L}(x^{(j)}, y, \mu^{(j)}), \forall j \in \mathcal{F}, \mu^{(j)} \succeq 0, \\ & 0 \geq \overline{\mathcal{L}}(x^{(j)}, y, \lambda^{(j)}), \forall j \in \mathcal{I}, \end{aligned}$$

where constraints (4b) and (4c) are referred to as the optimality and feasibility cuts, respectively. Problem (4) is an MILP problem and can be solved by various MILP solvers, such as CPLEX². Note that CPLEX can be also used for MINLP problems. However, it can only solve the mixed integer second-order cone programming and mixed integer quadratic or quadratically constrained programming problems, which only cover a small part of the resource allocation problems in wireless networks. The relaxed master problem provides a lower bound, $\eta^{*(i)}$, to the original Problem (1) and its solution, $y^{(i)}$, is used to generate the primal problem in the next iteration. The whole procedure of GBD is summarized in ²<https://www.ibm.com/analytics/cplex-optimizer>

Algorithm 2 Generalized Benders Decomposition

- 1: **initialization**
 - 2: Set iteration index, $i = 0$.
 - 3: Set maximum iteration number, M .
 - 4: Set tolerance, Δ .
 - 5: Select an initial value for $y^{(0)}$.
 - 6: $UBD^{(0)} = \infty, LBD^{(0)} = -\infty$.
 - 7: **while** $\left| \left(UBD^{(i)} - LBD^{(i)} \right) / LBD^{(i)} \right| > \Delta$ and $i < M$ **do**
 - 8: $i = i + 1$.
 - 9: Solve the primal problem (2) by fixing y as $y^{(i-1)}$.
 - 10: **if** the primal problem is feasible **then**
 - 11: Obtain the optimal solution $x^{(i)}$.
 - 12: Obtain $\mathcal{L}(x^{(i)}, y, \mu^{(i)})$ and an optimality cut $C^{(i)}$.
 - 13: Set $UBD^{(i)} = \min \left(UBD^{(i-1)}, f(x^{(i)}, y^{(i-1)}) \right)$.
 - 14: **else**
 - 15: Solve the feasibility-check problem (3).
 - 16: Obtain $\overline{\mathcal{L}}(x^{(i)}, y, \lambda^{(i)})$ and a feasibility cut $C^{(i)}$.
 - 17: Add $C^{(i)}$ into the relaxed master problem (4).
 - 18: Solve the relaxed master problem (4) to obtain η^* and $y^{(i)}$.
 - 19: Set $LBD^{(i)} = \eta^*$.
-

A direct implementation of the classical GBD may require excessive computing time and memory. Many works have been dedicated to exploring ways to improve the convergence speed of the algorithm by reducing the required time for each iteration or the number of required iterations. The former goal can be achieved by improving the procedure used to solve the primal and relaxed master problems at each iteration, which needs to be designed according to specific optimization problems. On the other hand, the latter goal can be achieved by improving the quality of the generated cuts, which is applicable for all optimization problems. In the following, we introduce an acceleration method of this kind named multi-cut GBD [22], which is the basis of our ML-based acceleration method to be introduced in Section III.

The key idea of multi-cut GBD is generating a number of cuts at each iteration to reduce the number of required iterations. The detailed procedure of the multi-cut GBD is summarized in Table II. The main difference between Algorithms I and Π is that a set \mathcal{S} of discrete solutions, instead of only one solution, is obtained while solving the relaxed master problem (4). The set \mathcal{S} includes both optimal and suboptimal solutions for Problem (4). Generally, the size of \mathcal{S} is set as a given constant, S . To get \mathcal{S} , we first use MILP solvers to get all feasible solutions for Problem (4). Then we add the solution with the smallest gap between its objective value, η , and the optimum value, η^* , one by one into \mathcal{S} until $|\mathcal{S}| = S$. Note that S should not be set large. Otherwise, there may not exist enough feasible solutions. If the number of existing solutions is less than S , we will collect all existing solutions as the set \mathcal{S} . In this way, we can get $|\mathcal{S}|$ primal problems by fixing the discrete variables, y , in the Problem (2) as the solutions in set \mathcal{S} . By solving the $|\mathcal{S}|$ primal problems, $|\mathcal{S}|$ cuts can be obtained and added to the relaxed master problem (4). Then Problem (4) is solved and a new set \mathcal{S} is obtained,

TABLE II Multi-CUt GENERALIZED BENDERS DECOMPOSITION Algorithm 2 Multi-cut Generalized Benders Decomposition initialization Set iteration index, $i = 0$. Set maximum iteration number, M . Set tolerance, Δ . Select the initial solution set as $\mathcal{S} = \{y^{(0)}\}$. $UBD^{(0)} = \infty, LBD^{(0)} = -\infty$ while $\left| \left(UBD^{(i)} - LBD^{(i)} \right) / LBD^{(i)} \right| > \Delta$ and $i < M$ do $i = i + 1, s = 0$. while $|\mathcal{S}| \neq 0$ do $s = s + 1$. $y_s \leftarrow$ pop out the best solution from \mathcal{S} . Solve the primal problem (2) by fixing y as y_s . if the primal problem is feasible then Obtain the optimal solution $x^{(i)}$. Obtain $\mathcal{L}(x^{(i)}, y, \mu^{(i)})$ and an optimality cut $C_s^{(i)}$. Set $UBD^{(i)} = \min(UBD^{(i-1)}, f(x^{(i)}, y_s))$. else Solve the feasibility-check problem (3). Obtain $\overline{\mathcal{L}}(x^{(i)}, y, \lambda^{(i)})$ and a feasibility cut $C_s^{(i)}$. end if Add $C_s^{(i)}$ into the relaxed master problem (4). end while Solve problem (4) to obtain the optimal η^* and a set \mathcal{S} of multiple solutions for y . Set $LBD^{(i)} = \eta^*$. end while

which is used to generate new $|\mathcal{S}|$ primal problems in the next iteration.

According to [22], the multi-cut GBD does not change the optimal solution of the GBD algorithm for convex MINLP problems. The multiple cuts generated in the multi-cut GBD can improve the obtained lower bounds when solving the relaxed master problem. Therefore, the total number of required iterations decreases and fewer relaxed master problems need to be solved as well. On the other hand, more primal problems need to be solved in multi-cut GBD than in classical GBD. Fortunately, the $|\mathcal{S}|$ primal problems are independent and can be solved in parallel while implementing multi-cut GBD, which will not bring additional overhead in terms of computational time if parallel computing is available. Overall, the total

computational time can be expected to decrease when the multi-cut GBD is deployed, according to [22].

However, more cuts are added to the relaxed master problem at each iteration in the multi-cut GBD compared with the classical GBD. Given the fact that the relaxed master problem will be more time-consuming as more cuts are added, the main overhead of multi-cut GBD is the average time consumed by solving the relaxed master problem. Note that, among all the cuts generated at each iteration in multi-cut GBD, some cuts do not contribute much change on the lower bound but would increase the complexity of the relaxed master problem. We name these cuts as useless cuts and the others as useful cuts. Actually, only useful cuts need to be added into the relaxed master problem at each iteration. If we can distinguish between useful and useless cuts and only add useful cuts to the relaxed master problem at each iteration, the average complexity of the relaxed master problems in multi-cut GBD will decrease, which further accelerates the whole algorithm. According to [23], there is no need to optimally solve Problem (4) at each iteration to produce cuts for the convex MINLP problems. Optimality can be guaranteed even though we use the cuts generated from sub-optimal solutions of Problem (4). Therefore, our proposed methods can still achieve optimality for the convex MINLP problems by only adding useful cuts to the relaxed master problem. In the following, we propose to use ML techniques to achieve this goal.

19. Convexification of Nonconvex Compositions with Norms

Monday, November 6, 2023 4:06 PM - 4:24 PM Bayhill 24 (Lobby Level, Hyatt Regency Orlando) Abstract Nonconvex optimization problems involving compositions of functions with norms arise in diverse settings such as facility location [1], molecular energy minimization [2], and object packing [3]. These problems present a challenge for global optimization algorithms based on spatial branch-and-bound since they exhibit a high degree of symmetry, and factorable relaxations of reverse convex constraints and compositions of functions with norms are generally weak [4,5]. Global minima have been found for small instances [6], but many problems remain open. In this work, we introduce novel convex envelopes for common functions composed with norms based on the characterization of their generating sets. The theoretical development exploits the fact that the generating set of these functions is a finite collection of compact convex sets [7]. We implement our envelopes in the global optimization solver BARON and demonstrate their impact using open problems from the literature as a benchmark.

- [1] P. Kalczynski and Z. Drezner. Extremely non-convex optimization problems: the case of the multiple obnoxious facilities location. *Optimization Letters*, 16:1153–1166, 2022.
- [2] M. R. Hoare and P. Pal. Physical cluster mechanics: statics and energy surfaces for monatomic systems. *Advances in Physics*, 20:161–196, 1971.
- [3] A. Wang, C. L. Hanselman, and C. E. Gounaris. A customized branch-and-bound approach for irregular shape nesting. *Journal of Global Optimization*, 71:935–955, 2018.
- [4] A. Khajavirad. Packing circles in a square: a theoretical comparison of various convexification techniques. Technical report, Optimization Online, 2017
- [5] A. Wang and C. E. Gounaris. On tackling reverse convex constraints for non-overlapping of unequal circles. *Journal of Global Optimization*, 80:357–385, 2021.
- [6] M. C. Markót and T. Csendes. A New Verified Optimization Technique for the "Packing Circles in a Unit Square" Problems. *SIAM Journal on Optimization*, 16, 193– 219.
- [7] A. Khajavirad and N. V. Sahinidis. Convex envelopes generated from finitely many compact convex sets. *Mathematical Programming*, 137:371–408, 2013.

Presenting Author K Anatoliy Kuznetsov Georgia Institute of Technology Author S Nikolaos Sahinidis Georgia Institute of Technology

Nonconvex Optimization Problems Involving the Euclidean Distance Anatoliy Kuznetsov, School of Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA and Nikolaos

Sahinidis, H. Milton Stewart School of Industrial & Systems Engineering and School of Chemical & Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA

Recent Advances in Pyros: The Pyomo Solver for Two-Stage Nonconvex Robust Optimization Jason Sherman¹, Natalie Isenberg^{1,2}, John Siirola³ and Chrysanthos Gounaris¹, (1)Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA, (2)Brookhaven National Laboratory, Upton, NY, (3)Sandia National Laboratories, Albuquerque, NM

Advances in Constructing Tight Quadratic Underestimators for Global Optimization William Strahl, Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA, Arvind Raghunathan, Mitsubishi Electric Research Laboratories, Cambridge, MA, Nikolaos Sahinidis, H. Milton Stewart School of Industrial & Systems Engineering and School of Chemical & Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA and Chrysanthos Gounaris, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA

Improving the Efficiency of Branch-and-Bound for Global Dynamic Optimization Jason Ye, Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA and Joseph K. Scott, School of Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA

Mindtpy: The Mixed-Integer Nonlinear Decomposition Toolbox in Pyomo Zedong Peng, Davidson School of Chemical Engineering, Purdue University, West Lafayette, IN, David E. Bernal Neira, Universities Space Research Association – Research Institute of Advanced Computer Science, Mountain View, CA and Ignacio Grossmann, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA

20. Polynomial Optimization via Convex Optimization

20.1 Sums of Squares (SOS) Programming

Resources

Amir Ali Notes on SOS Techniques

20.1.1. Putinar

20.2 Sums of nonnegative circuit polynomials (SONC)

Nonnegative Polynomials and Circuit Polynomials

21. Polynomial Optimization via Algebraic Techniques

21.1 Polynomial Optimization - Algebraic Techniques

21.2 Cylindrical Algebraic Decomposition

21.3 Gröbner Bases

21.4 Quantifier Elimination and Semi-algebraic Sets

Theorem 21.1

Any alternating quantifier set can be recast as a semi-algebraic set.

21.5 Polynomial Optimization

Polynomial Optimization

$$\max h(x)$$

$$\text{s.t. } f_1(x) = 0, \dots, f_m(x) = 0,$$

$$g_1(x) \geq 0, \dots, g_n(x) \geq 0$$

$$x \in \mathbb{R}^n$$

Almost every combinatorial optimization can be modelled like this

Vertex Cover

Find a minimum size vertex set that covers every edge.

$$\bullet \min \sum x_i$$

$$\text{s.t. } x_i + x_j \geq 1 \quad (i, j) \in E$$

$$x_i^2 = x_i \quad (x_i \in \{0, 1\})$$

$$x_i = \begin{cases} 1 & \text{if } i \in V \\ 0 & \text{if } i \notin V \end{cases}$$

OR.

$$\bullet \max \sum x_i$$

$$\text{s.t. } x_i \cdot x_j = 0$$

$$x_i = \begin{cases} 0 & \text{if } i \in V \\ 1 & \text{if } i \notin V \end{cases}$$

Binary Search

We instead solve the feasibility problem

$$h(x) - \gamma \geq 0$$

$$f_i(x) = 0 \quad i=1, \dots, m$$

$$g_i(x) \geq 0 \quad i=1, \dots, k$$

When you know a range on values for objective function,
we can do binary search with γ .

Goal: To understand the feasibility question with polynomial constraints.

Problem Setup: $K = \text{field}$, $K[x_1, \dots, x_n] = K[x] = R$

(ring of polynomials with coefficients in K)

$$x^r = x_1^{a_1} \cdots x_n^{a_n} \quad (\text{monomial})$$

Solve $\begin{cases} f_i(x) = 0 & i=1, \dots, m \\ g_i(x) \geq 0 & i=1, \dots, k \end{cases}$

$$p(x) = \sum_{\alpha} \max x^{\alpha}, \quad \text{degree}(p(x)) = \max_{q: M_q \neq 0} \left(\sum_{i=1}^n a_i \right)$$

$$F := \{f_1, \dots, f_m\}$$

$$\langle F \rangle_R = \left\{ \sum_{i=1}^m \beta_i f_i \mid \beta_i \in R \right\} \quad \text{"Ideal generated by } F\text{"}$$

Defn: A polynomial $s(x)$ is a sum of squares (SOS) if
$$s(x) = \sum_{i \in I} [g_i(x)]^2$$

Defn: Let $g_1, \dots, g_k \in R$

$$\text{Cone}(G) = \left\{ s_0 + \sum_i s_i g_i + \sum_{i,j} s_{ij} g_i g_j + \dots + s_{ijk} g_i g_j g_k \right\}$$

s_α is a SOS polynomial }

Lemma: Fredholm's Alternative

$$\nexists x \text{ such that } Ax + b = 0 \Leftrightarrow \exists u \text{ such that } u^T A = 0, u^T b = 1$$

Theorem Hilbert's (weak) Nullstellensatz

$$\nexists x \text{ such that } f_i(x) = 0, i=1, \dots, m$$

$$\Leftrightarrow \exists \beta_1, \dots, \beta_m \in R \text{ s.t. } \sum \beta_i f_i = 1$$

$$\Leftrightarrow 1 \in \langle F \rangle_R$$

(Proof by Cox, Little, & O'shea)

General Farkas' Lemma:

$\exists x \text{ s.t. } Ax+b=0, Cx+d \geq 0$

$\Leftrightarrow \exists \mu, \lambda \text{ with } \lambda \geq 0 \text{ s.t.}$

$$\mu^T A + \lambda^T C = 0$$

$$\mu^T b + \lambda^T d = -1$$

Proof uses: Hahn Banach Theorem = Separation Hyperplane Theorem

Theorem: Positive Stellensatz (Stengle 1973)

$\exists x \text{ such that } f_i(x)=0, i=1, \dots, m$

$\nexists g_i(x) \geq 0, i=1, \dots, k$

\Leftrightarrow

$\exists f \in \langle F \rangle_R, g \in \text{Cone}(G) \text{ such that } f+g=1$

Example: $f_1(x) = x_1^2 - 1, f_2(x) = 2x_1 x_2 + x_3, f_3(x) = x_1 + x_2, f_4(x) = x_1 + x_3$
 $\{x \mid f_i(x) = 0\} ??$

Algorithm to find infeasibility certificate:

$$\mu_1(x^2 - 1) + \mu_2(2x_1 x_2 + x_3) + \mu_3(x_1 + x_2) + \mu_4(x_1 + x_3) = 1$$

• Assuming μ_i 's are constant

$$\Rightarrow -\mu_1 = 1$$

$$\mu_2 + \mu_4 = 0$$

$$\mu_2 + \mu_4 = 0$$

$$\mu_3 = 0$$

$$\mu_1 = 0$$

$\left. \begin{array}{l} \mu_2 + \mu_4 = 0 \\ \mu_2 + \mu_4 = 0 \\ \mu_3 = 0 \end{array} \right\} \text{infeasible} \Rightarrow \text{try with } \mu_i \text{'s as linear functions, then quadratics, etc.}$

Theorem: \exists an exponential bound on the degrees of β_i 's in the Hilbert infeasibility certificate.

Algorithm: NullA (uses linear algebra to solve system of polynomial equations)

for $d=1, \dots, D$ (exponentially bounded)

1. try to find β_i of degree d s.t. $\sum \beta_i f_i = 0$

2. If yes, stop \rightarrow report infeasible

3. If no, continue with $d=d+1$.

If END \rightarrow report feasible.

SOS

$$p(x) = \sum_{i \in I} [\beta_i(x)]^2$$

Theorem 1: $p(x) \in R$ is SOS $\Leftrightarrow p(x) = z^T Q z$ where

Q is PSD, z is vector of monomials

Example: $p(x_1, x_2) = x_1^2 - x_1 x_2^2 + x_2^4 + 1$

$$= \frac{3}{4}(x_1 - x_2)^2 + \frac{1}{4}(x_1 + x_2^2)^2 + 1$$

$$= \frac{1}{6} \begin{bmatrix} 1 & x_2 & x_2^2 & x_1 \end{bmatrix} \begin{bmatrix} 6 & 0 & -2 & 0 \\ 0 & 4 & 0 & 0 \\ -2 & 0 & 6 & -3 \\ 0 & 0 & -3 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ x_2 \\ x_2^2 \\ x_1 \end{bmatrix}$$

Defn: A symmetric matrix is PSD if $x^T A x \geq 0 \quad \forall x \in R^n$

Theorem : Let A be a symmetric matrix. ~~PSD~~

A is PSD \Leftrightarrow all eigenvalues of A are real & ≥ 0

$\Leftrightarrow A = C^T C$ where C is a $l \times n$ matrix.

Proof Theorem 1: (\Leftarrow) $p(x) = z^T Q z = z^T C^T C z = \sum_i (c_i z)^2 = \text{SOS}$

(\Rightarrow) $p(x) = \sum_i (\beta_i(x))^2 = x^T Q^T Q x$ where $Q = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_l \end{bmatrix}$ coefficients, $x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \end{bmatrix}$ all monomials.

□

21.5.1. Sums of Squares

Semi-Definite Programming (SDP)

$$\max_{\mathbf{X}} \sum_{ij} C_{ij} X_{ij} = \text{Trace}(C \cdot \mathbf{X})$$

$$\text{s.t. } \text{Trace}(A_i \cdot \mathbf{X}) = b_i \quad i \in I$$

$$\mathbf{X} \succeq 0 \quad (\mathbf{X} \text{ is a } n \times n \text{ matrix})$$

Thm: Grötschel-Lovasz-Schrijver (GLS)

Optimization / test feasibility over a convex set $S \subseteq \mathbb{R}^d$

is equivalent to separation over S .

(i.e. we need a separation oracle).

$$\text{let } S = \{\mathbf{L} \in \mathbb{R}^{n \times n} \mid \mathbf{L} \succeq 0\} \subseteq \mathbb{R}^{n \times n}$$

Separation oracle for S

- given a matrix \mathbf{Q} , decide if \mathbf{Q} belongs to S

OR Find a separating hyperplane.

→ find all eigenvalues of \mathbf{Q}

→ if $\lambda_i \geq 0 \forall i$, then $\mathbf{Q} \in S$

→ otherwise $\exists \mathbf{y} \in \mathbb{R}^n$ s.t.

$$\mathbf{Q}\mathbf{y} = \lambda \mathbf{y}, \quad \lambda < 0$$

$$\Rightarrow \mathbf{y}^T \mathbf{Q} \mathbf{y} = \mathbf{y}^T \lambda \mathbf{y} = \lambda \|\mathbf{y}\|_2^2 < 0$$

$$\rightarrow \text{Separating hyperplane } \mathbf{y}^T \mathbf{L} \mathbf{y} = 0$$

i.e. $\mathbf{y}^T \mathbf{L} \mathbf{y} \geq 0$ is valid for S .

Therefore, GLS \Rightarrow SDP can be solved (By Ellipsoid Method)

(but we can also do this with interior point methods)

Application in Combinatorial Optimization

Stable Set Problem (*NP-Complete*)

Find a maximum size stable set of vertices

where no two vertices are connected by an edge

Prop: ~~S~~ S is a stable set \Leftrightarrow V/S is a vertex cover

$$\text{Max } \sum x_i$$

$$\text{s.t. } x_i + x_j \leq 1 \quad \forall (i,j) \in E$$

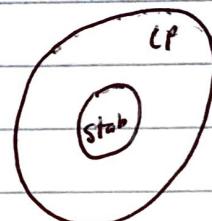
$$x_i \in \{0,1\}$$

$$LP(G) = \left\{ x \mid x_i + x_j \leq 1 \quad \forall (i,j) \in E, 0 \leq x_i \leq 1 \right\}$$

$$Stab(G) = \text{Conv}(\{x \mid x_i + x_j \leq 1, x_i \in \{0,1\}\})$$

Defn: A clique is a set of vertices s.t.

$i, j \in C$ for all $i, j \in C$ (the clique)



Observation: For every clique C, at most 1 vertex from C belongs to a stable set.

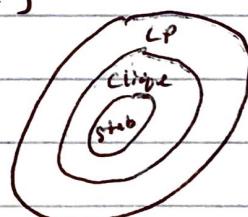
$$Clique(G) = \left\{ x \mid \sum_{i \in C} x_i \leq 1, 0 \leq x_i \leq 1 \right\}$$

Recall that we can write binary variables using quadratic constraints

• One formulation is

$$x_i^2 - x_i = 0$$

$$x_i \cdot x_j = 0 \quad \forall (i,j) \in E$$



Application in Combinatorial Optimization

Stable Set Problem (*NP-Complete*)

Find a maximum size stable set of vertices

where no two vertices are connected by an edge

Prop: ~~S~~ S is a stable set \Leftrightarrow V/S is a vertex cover

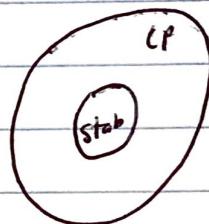
$$\begin{aligned} \text{Max } & \sum x_i \\ \text{s.t. } & x_i + x_j \leq 1 \quad \forall (i,j) \in E \\ & x_i \in \{0,1\} \end{aligned}$$

$$LP(G) = \left\{ x \mid x_i + x_j \leq 1 \quad \forall (i,j) \in E, 0 \leq x_i \leq 1 \right\}$$

$$Stab(G) = \text{Conv}(\{x \mid x_i + x_j \leq 1, x_i \in \{0,1\}\})$$

Defn: A clique is a set of vertices s.t.

$i, j \in C$ for all $i, j \in C$ (the clique)



Observation: For every clique C, at most 1 vertex from C belongs to a stable set.

$$Clique(G) = \left\{ x \mid \sum_{i \in C} x_i \leq 1, 0 \leq x_i \leq 1 \right\}$$

Recall that we can write binary variables using quadratic constraints

* One formulation is

$$x_i^2 - x_i = 0$$

$$x_i \cdot x_j = 0 \quad \forall (i,j) \in E$$



* Introduce "linearizing variables" y_{ij}

$$y_{ij} = x_i x_j, \quad y_{jj} = x_j^2 = x_j, \quad y_{0j} = y_{j0} = x_j, \quad y_{00} = 1$$

$$\Rightarrow Y = \begin{pmatrix} 1 & \mathbf{x} \\ \mathbf{x}^\top & 0 \end{pmatrix}$$

* Y is a positive semidefinite matrix

\Rightarrow Relaxation: $Y \succeq 0$

i.e. Need not be that $Y = (\)^{()}$ for rank 4 matrices.

\Rightarrow cannot recover x from the constraint.

$$\text{TH}(G) = \left\{ X \mid Y \succeq 0, \quad y_{ij} = 0 \quad \forall (i,j) \in E, \quad \begin{array}{l} \text{"Theta body"} \\ \text{from Lasserre} \end{array} \right. \\ \left. y_{0j} = y_{j0} = y_{jj} = x_j, \quad y_{00} = 1, \quad 0 \leq x_i \leq 1 \right\}$$

Theorem

$$\text{Stab}(G) \subseteq \text{TH}(G) \subseteq \text{Clique}(G)$$

Proof: Clearly $\text{Stab} \subseteq \text{TH}(G)$ by construction.

WTS. $\text{TH}(G) \subseteq \text{Clique}(G)$.

Assume $X \in \text{TH}(G) \Rightarrow \exists Y \succeq 0$ in $\text{TH}(G)$ definition.

$$\Rightarrow v^T Y v \geq 0 \quad \forall v \in \mathbb{R}^{n+1}$$

Take any clique C

$$\text{set } v_i = \begin{cases} 1 & \text{if } i=0 \\ -1 & \text{if } i \in C \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Then } 0 \leq v^T Y v = \sum_{i,j} y_{ij} v_i v_j$$

$$= \sum_{(i,j) \in E} y_{ij} v_i v_j + \sum_{i=1}^n y_{0i} v_0 v_0 + \sum_{j=1}^n y_{0j} v_0 v_j + y_{00} v_0^2$$

$$\star (y_{ij} = 0 \text{ if } (i,j) \notin E)$$

$$= 0 + \sum_{j=1}^n y_{0j} - \cancel{\sum_{i \in C} x_i} - \cancel{\sum_{i \in C} x_i} + 1$$

$$\Rightarrow \sum_{i \in C} x_i \leq 1 \quad \text{∴ i.e. } X \in \text{Clique}(G) \quad \square$$

* Introduce "linearizing variables" y_{ij}

$$y_{ij} = x_i x_j, \quad y_{jj} = x_j^2 = x_j, \quad y_{0j} = y_{j0} = x_j, \quad y_{00} = 1$$

$$\Rightarrow Y = \begin{pmatrix} 1 & x \\ x^T & \end{pmatrix}$$

* Y is a positive semidefinite matrix

\Rightarrow Relaxation: $Y \succeq 0$

i.e. need not be that $Y = \begin{pmatrix} 1 & x \\ x^T & \end{pmatrix}$ for rank 1 matrices.

\Rightarrow cannot recover x from the constraint.

$$TH(G) = \{x \mid Y \succeq 0, y_{ij} = 0 \text{ if } (i,j) \notin E, \dots\}$$

$$y_{0j} = y_{j0} = y_{jj} = x_j,$$

$$y_{00} = 1, \quad 0 \leq x_i \leq 1\}$$

"Theta body"

from Lovasz

Theorem

$$Stab(G) \subseteq TH(G) \subseteq \text{Clique}(G)$$

Proof: Clearly $Stab \subseteq TH(G)$ by construction.

WTS. $TH(G) \subseteq \text{Clique}(G)$.

Assume $x \in TH(G) \Rightarrow \exists Y \succeq 0$ in $TH(G)$ definition.

$$\Rightarrow v^T Y v \geq 0 \quad \forall v \in \mathbb{R}^{n+1}$$

Take any clique C

$$\text{set } v_i = \begin{cases} 1 & \text{if } i=0 \\ -1 & \text{if } i \in C \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Then } 0 \leq v^T Y v = \sum_{i,j} y_{ij} v_i v_j$$

$$= \sum_{(i,j) \in E} y_{ij} v_i v_j + \sum_{i=1}^n y_{0i} v_0 v_0 + \sum_{j=1}^n y_{0j} v_0 v_j + y_{00} v_0^2$$

* $(y_{ij} = 0 \text{ if } (i,j) \notin E)$

$$= 0 + \sum_{i=1}^n y_{0i} - \cancel{\sum_{i \in C} x_i} - \cancel{\sum_{i \in C} x_i} + 1$$

$$\Rightarrow \sum_{i \in C} x_i \leq 1 \quad (\because \text{i.e. } x \in \text{Clique}(G)) \quad \square$$

Max-Cut

Given a graph $G = (V, E)$ and weights on the edges,
find a partition of $V = S \sqcup V/S$
Such that the total weight of the edges crossing
the partition is maximized.

Goemans - Williamson Approximate Algorithm (SDP)

There is an efficient (polynomial) algorithm which gets
within 82% of the optimal max cut.

- \nexists any algorithm better than this unless $P=NP$

Formulation: $X_i = \begin{cases} 1 & \text{if } i \in S \\ -1 & \text{if } i \notin S \end{cases}, i \in V.$

$$X_i \in \{-1, 1\} \Leftrightarrow X_i^2 = 1$$

$$X_i \neq X_j \quad (\Leftrightarrow) \quad \frac{1 - X_i X_j}{2} = 1$$

$$\Rightarrow \max_{S.T.} W_{ij} \left(\frac{1 - X_i X_j}{2} \right)$$

$$X_i^2 = 1$$

Max-Cut

Given a graph $G = (V, E)$ and weights on the edges,
find a partition of $V = S \sqcup V/S$
such that the total weight of the edges crossing
the partition is maximized.

Goemans - Williamson Approximate Algorithm (SDP)

There is an efficient (polynomial) algorithm which gets
within 82% of the optimal max cut.

- $\#$ any algorithm better than this unless $P=NP$

Formulation: $X_i = \begin{cases} 1 & \text{if } i \in S \\ -1 & \text{if } i \notin S \end{cases}, i \in V.$

$$X_i \in \{-1\} \Leftrightarrow X_i^2 = 1$$

$$X_i \neq X_j \quad (\Rightarrow) \quad \frac{1 - X_i X_j}{2} = 1$$

$$\Rightarrow \max w_{ij} \left(\frac{1 - X_i X_j}{2} \right)$$

$$\text{s.t.} \quad X_i^2 = 1$$

SOS Theory

Defn: f is convex if $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$
 $\forall x, y \in \mathbb{R}^n, \lambda \in [0, 1]$

Prop: if f is twice-differentiable, then

$$f \text{ is convex} \Leftrightarrow H_f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_i \partial x_j} \end{bmatrix} \geq 0 \quad \forall x \in \mathbb{R}^n$$

"the Hessian is PSD for all $x \in \mathbb{R}^n"$

Polynomial Matrices $P(x) = \begin{bmatrix} P_{ij} \end{bmatrix}_{m \times m}$
 \downarrow
 Polynomial in x

Defn: $P(x)$ is a PSD matrix if $P(x)$ is PSD $\forall x \in \mathbb{R}^n$

Fact: $P(x)$ is PSD $\Leftrightarrow y^T P(x) y \geq 0$ as an element of $\mathbb{R}[x, y]$
 $y \in \mathbb{R}^m$

Defn: $P(x)$ is an SOS matrix if \exists a polynomial matrix $M(x)$
 such that $P(x) = M^T(x) M(x)$

Fact: $P(x)$ is SOS $\Rightarrow P(x)$ is PSD

Fact: $P(x)$ is an SOS matrix $\Leftrightarrow y^T P(x) y$ is a sum of squares
 polynomial.

Example PSD-Matrix not SOS-matrix

$$A(x) = \begin{bmatrix} x_1^2 + 2x_2^2 & -x_1x_2 & -x_1x_3 \\ -x_1x_2 & x_2^2 + 2x_3^2 & -x_2x_3 \\ -x_1x_3 & -x_2x_3 & x_3^2 + 2x_1^2 \end{bmatrix}$$

$A(x)$ is PSD, but $A(x)$ is not SOS

* $A(x)$ is not a Hessian Matrix

since $\frac{\partial}{\partial x_3} \left(\frac{\partial^2 p}{\partial x_1^2} \right) \neq \frac{\partial}{\partial x_1} \left(\frac{\partial p^2}{\partial x_1 \partial x_3} \right)$

Note: if $n=1$, then PSD = SOS

Deciding Complexity of a polynomial function:

Defn: $P(x)$ is SOS-convex if $H(x)$ is an SOS-matrix

Clearly SOS-convex \Rightarrow convex

Theorem (Amir Ali & Pablo Parrilo)

\exists a polynomial that is convex, but not SOS-convex.

Note: Deciding $H(x) = \text{SOS}$ is easy since we just show that $y^T H(x) y$ is a SOS polynomial
(use SDP to solve)

Time Complexity $n = \text{size of input}$

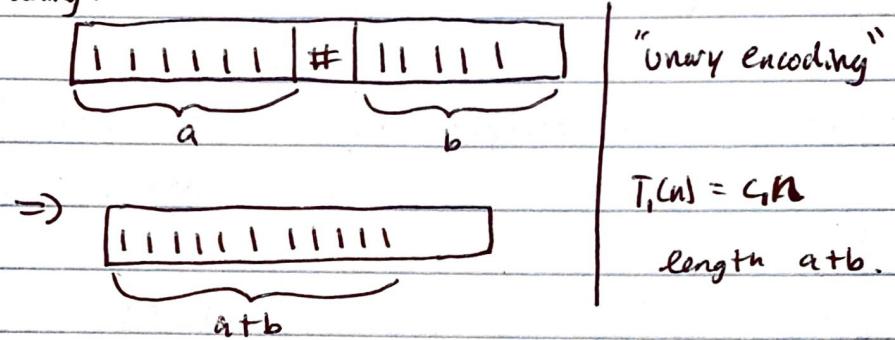
$T(n) = \# \text{ of steps/transitions the turing machine makes}$

Space complexity

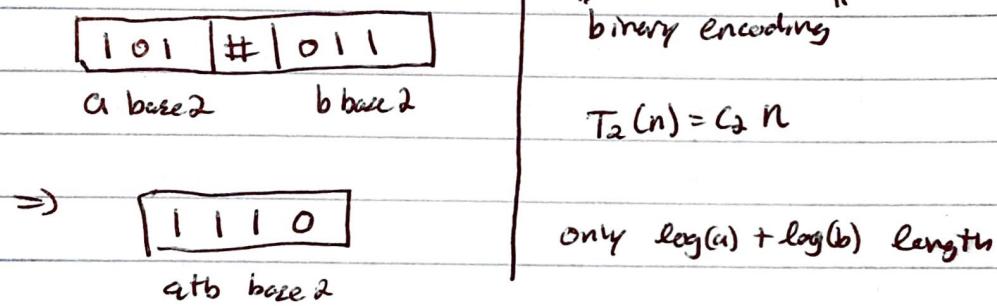
$S(n) = \text{amount of storage required}$

Example: important to consider your input!!!

Adding: $a+b$



Versus:



* both linear time algorithms, but T_2 is exponentially better than T_1 .

Defn: A problem is a set of strings on Σ

An instance is a particular string from the problem

Defn: The class of polynomial time problems P is the set of problems for which there exists a turing machine with polynomial time complexity.

Reduction: Problem B reduces to problem A if for

all instances of B there exists a map

$$R: B \rightarrow A, R(b) \in A, \text{ s.t. } R \text{ is polytime.}$$

Defn: A and B are polynomially equivalent if A reduces to B and B reduces to A.

Example: Vertex Cover

- Optimization asks for the minimum vertex cover

- Feasibility asks for a vertex cover of size $\leq k$

Stable Set

- Optimization asks for the maximum stable set

- feasibility asks for stable set of size $\geq n-k$

All of these are polynomially equivalent

Defn: A non-deterministic polynomial time (NP)

turing machine has a map

$$f: Q \times \Sigma \rightarrow 2^Q \times \Sigma^* \cup \{\epsilon\}$$

it can carry out parallel computations.

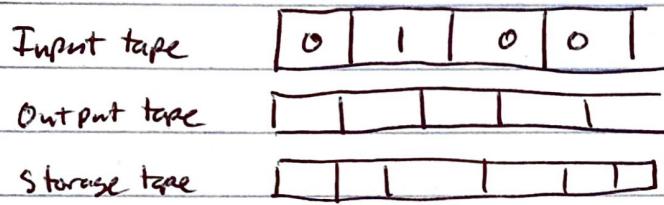
Alternative Defn: A problem is NP if there exists a polynomial time algorithm to check a given certificate.

Clearly $P \subseteq NP$. Probably $P \neq NP$.

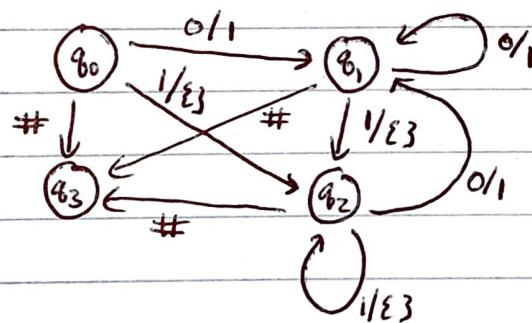
Complexity Theory

Defn: A Turing Machine includes the following

1. Q : a finite set of states $Q = \{q_0, \dots, q_m\}$
↑
initial ↑
final
2. Σ "alphabet" - a set of symbols
3. f "transition function" $f: Q \times \Sigma \rightarrow Q \times \Sigma \cup \{\epsilon\}$
↑
empty symbol
4. Tape : Input, Storage, & Output



Example: Suppose $\Sigma = \{0, 1, \#\}$



Storage tape ...

Output tape

1	1	1
---	---	---

#1's = #0's before first #

"Any reasonable model of computation is equivalent to a turing machine"

Time Complexity $n = \text{size of input}$

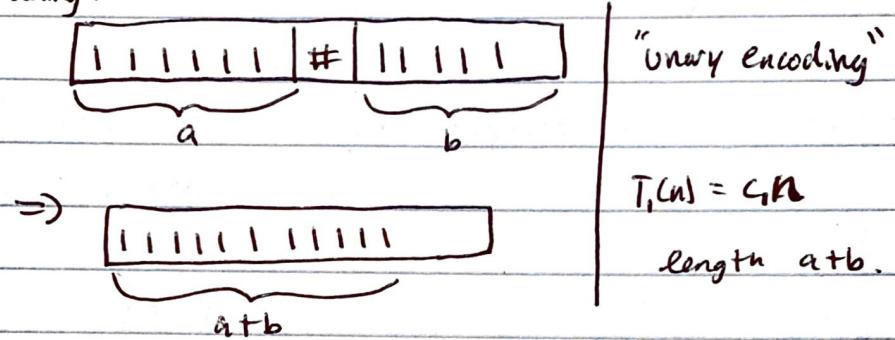
$T(n) = \# \text{ of steps/transitions the turing machine makes}$

Space complexity

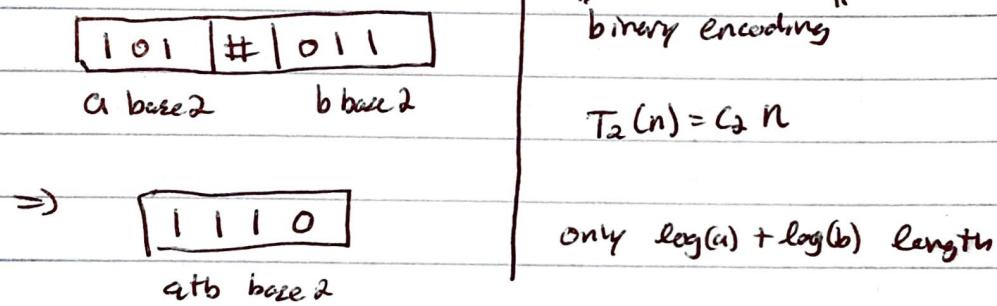
$S(n) = \text{amount of storage required}$

Example: important to consider your input!!!

Adding: $a+b$



Versus:



* both linear time algorithms, but T_2 is exponentially better than T_1 .

Time Complexity $n = \text{size of input}$

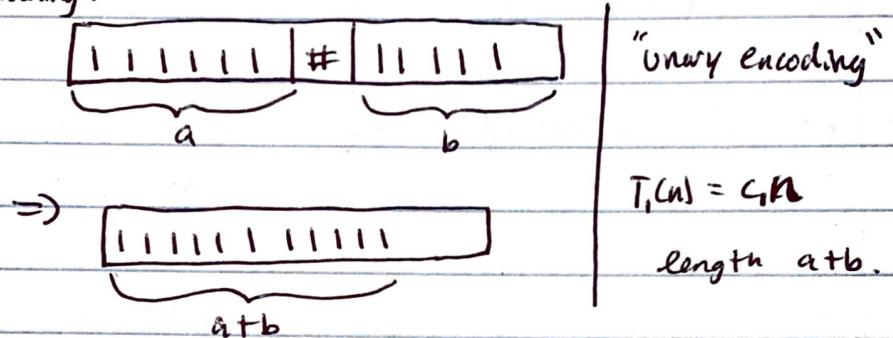
$T(n) = \# \text{ of steps/transitions the turing machine makes}$

Space complexity

$S(n) = \text{amount of storage required}$

Example: important to consider your input!!!

Adding: $a+b$

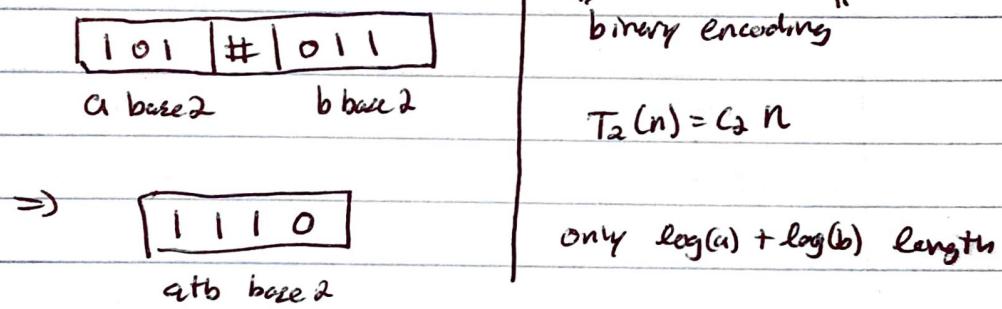


"unary encoding"

$$T_1(n) = c_1 n$$

length $a+b$.

Versus:



"binary encoding"

$$T_2(n) = c_2 n$$

only $\log(a) + \log(b)$ length

* both linear time algorithms, but T_2 is exponentially better than T_1 .

Defn: A problem is a set of strings on Σ

An instance is a particular string from the problem

Defn: The class of polynomial time problems P is the set of problems for which there exists a turing machine with polynomial time complexity.

Reduction: Problem B reduces to problem A if for

all instances of B there exists a map

$$R: B \rightarrow A, R(b) \in A, \text{ s.t. } R \text{ is polytime.}$$

Defn: A and B are polynomially equivalent if A reduces to B and B reduces to A.

Example: Vertex Cover

- Optimization asks for the minimum vertex cover
- Feasibility asks for a vertex cover of size $\leq k$

Stable Set

- Optimization asks for the maximum stable set
- feasibility asks for stable set of size $\geq n-k$

All of these are polynomially equivalent

Defn: A non-deterministic polynomial time (NP)

turing machine has a map

$$f: Q \times \Sigma \rightarrow 2^Q \times \Sigma^* \cup \{\epsilon\}^3$$

it can carry out parallel computations.

Alternative Defn: A problem is NP if there exists a polynomial time algorithm to check a given certificate.

Clearly $P \subseteq NP$. Probably $P \neq NP$.

Defn: A problem is a set of strings on Σ

An instance is a particular string from the problem

Defn: The class of polynomial time problems P is the set of problems for which there exists a turing machine with polynomial time complexity.

Reduction: Problem B reduces to problem A if for all instances of B there exists a map $R: B \rightarrow A$, $R(b) \in A$, s.t. R is polytime.

Defn: A and B are polynomially equivalent if A reduces to B and B reduces to A.

Example: Vertex Cover

- Optimization asks for the minimum vertex cover
- Feasibility asks for a vertex cover of size $\leq k$

Stable Set

- Optimization asks for the maximum stable set
- Feasibility asks for stable set of size $\geq n-k$

All of these are polynomially equivalent

Defn: A non-deterministic polynomial time (NP)

turing machine has a map

$$f: Q \times \Sigma \rightarrow 2^Q \times \Sigma^* \cup \{\epsilon\}$$

it can carry out parallel computations.

Alternative Defn: A problem is NP if there exists a polynomial time algorithm to check a given certificate.

Clearly $P \subseteq NP$. Probably $P \neq NP$.

Linear optimization

$$(P) \max \{c^T x \mid Ax = b, x \geq 0\} \quad A \in \mathbb{R}^{m \times n}$$

$$(D) \min \{b^T y \mid A^T y \geq c\} = \min \{b^T y \mid A^T y - s = c, s \geq 0\}$$

Theorem (Weak Duality)

If x is feasible for P, y feasible for D, then

$$b^T y \geq c^T x$$

Theorem (Strong Duality)

For each pair of linear programs, one of the following is true:

1. (P), (D) are both infeasible
2. (P) unbounded, (D) infeasible
3. (P) infeasible, (D) unbounded
4. (P) feasible, (D) feasible

$$\Rightarrow b^T y^* = c^T x^* \quad (\textcircled{1})$$

Theorem (Complementary Slackness)

When x^*, y^* are optimal solutions of P, D, then

$$x_i^* s_i^* = 0, \quad i=1, \dots, n$$

Moral: Solving LP's results to solving the following system
of equations and inequalities

$$Ax = b, \quad A^T y - s = c, \quad x^T s = 0$$

$$x \geq 0, \quad s \geq 0$$

Simplex Method (1947 George Dantzig)

Satisfy $Ax = b$, $A^T y - s = c$, $x \geq 0$

Defn: X is a Basic Feasible Solution if \exists
a submatrix A_B of A with rank m (invertible)
such that

$$x_j = 0 \text{ if } j \notin B$$

$$(\text{therefore } A_B x_B = b, \quad x_B = A_B^{-1} b)$$

Lemma: If there exists an optimal solution of (P) ,
then there exists an optimal BFS of (P) .
(Assume bounded solution)

Proof: Need 2 things:

1. Every BFS = an extreme point of (P)

and vice versa

2. Every point in P can be written as a convex
combination of extreme points + direction of unboundedness.

$$\text{hence } x^* = d + \sum \alpha_i v_i$$

• note that $c^T d = 0$ since otherwise we could scale in
that direction

$$\text{then } c^T x^* = \sum \alpha_i c^T v_i \leq \max_i c^T v_i \Rightarrow c^T x^* = c^T v_i$$

□

Representation

$$b = Ax = A_B x_B + A_N x_N \Rightarrow x_B = A_B^{-1} b - A_B^{-1} A_N x_N$$

$$\begin{aligned} z = c^T x &= c_B^T x_B + c_N^T x_N \\ &= c_B^T A_B^{-1} b + (c_N^T - \underbrace{c_B^T A_B^{-1} A_N}_{\parallel}) x_N \\ c_j - y \alpha_j &= s_j \quad y \end{aligned}$$

Pivot Rule: Select index from those non-basic variables that correspond to

$$s_j < 0$$

i.e. $c_j - \bar{y}^T a_j = -s_j > 0$.

Many choices for pivot rules!!!

Entering Variable: Call a_j the column vector of the entering variable $x_j = 0$.

Increase x_j slowly by $t > 0$

$$x_B^{\text{new}} = x_B^{\text{old}} - t(A_B^{-1} a_j)$$

\Rightarrow eventually one other variable will vanish

\rightarrow this variable leaves the basis

\rightarrow OR no variable vanishes \Rightarrow unbounded direction.

How to find first BFS? $Ax=b, x \geq 0, b \geq 0$ (otherwise multiply by -1)

Set $\bar{A} = (A, I_m)$, $\bar{x} = (x, x_{n+1}, \dots, x_{n+m})$

Solve $\bar{A}\bar{x} = b, \bar{x} \geq 0$, using $\bar{x}_0 = (0, b)$

i.e.

$$\max -(x_{n+1} + \dots + x_{n+m})$$

s.t. $\bar{A}\bar{x} = b$

if $\max = 0 \Rightarrow$ we found an initial BFS

if $\max < 0 \Rightarrow$ original problem is infeasible.

Open Problem: Can you find a plot rule s.t. the number of plot rules necessary to get an optimal solution is polynomial in $n \& m$?

Can you bound the diameter of a graph in $n \& m$?

Interior Point Methods (1984 Karmarkar)

Theorem: For all $\lambda > 0$, ~~the problem~~ has a unique real solution $(x^*(\lambda), y^*(\lambda), s^*(\lambda))$ s.t.

1.) $x^*(\lambda)$ is an optimal solution to

$$\max(c^T x + \sum_{i=1}^n \lambda \log(x_i) \mid Ax=b, x \geq 0)$$

2.) In the limit, the point $(x^*(0), y^*(0), s^*(0))$ is an optimal solution to (P).

Defn: The set of all real solutions of * as $\lambda \rightarrow 0$ is the Central Path

$$* \quad Ax=b, x_i s_i = \lambda, A^T y - s = c, x \geq 0, s \geq 0$$

Steps of IPM: 1. Set $\lambda = 1$, solve *, find initial point (x, y, s)
(or close enough point)

Given $\epsilon > 0$,

while $\lambda > \epsilon$ REPEAT

$$\lambda \rightarrow \left(1 - \frac{1}{2\sqrt{n}}\right)\lambda$$

$$(x, y, s) \rightarrow (x + \Delta x, y + \Delta y, s + \Delta s)$$

$$A(x + \Delta x) = b \quad A^T(y + \Delta y) - (s + \Delta s) = c$$

$$(x_i + \Delta x_i)(s_i + \Delta s_i) = \lambda$$

$$s_i + \Delta s_i \geq 0$$

$$x_i + \Delta x_i \geq 0$$

Cheat: $A(\Delta x) = 0, A^T(\Delta y) - \Delta s = 0$

$$s_i \Delta x_i + x_i \Delta s_i = \lambda$$

THREE FORMS

LP Applications

- Game theory
- Combinatorial optimization
 - maxflow / min cut
 - help Integer Programs
 - Approximation Algorithms

Game theory:

2-person, 0-Sum game

win for one person = loss for other person

		1	2	3	Player 1 "cops"
		1	-10	-10	pay off for player 1 = - (pay off for player 2)
Robbers		1	4	-8	
Player 2		2	-8	5	-3
3		3	-12	-12	9

$\Delta X = \{x \mid \sum x_i = 1\}$ $\Delta y = \{y \mid \sum y_i = 1\}$

Mixed strategy: Police patrol site i with probability x_i

Robbers attack site i with probability y_i

$$\sum x_i = 1 \quad \sum y_i = 1$$

Expected Pay off: $\sum_{i,j} a_{ij} x_i y_j = y^T A x$ Police want to maximize this

Police try to find $\bar{x} \in \Delta X$ s.t.

$$\bar{x} = \arg \max_{x \in \Delta X} \left(\min_{y \in \Delta Y} y^T A x \right)$$

\Rightarrow Police can guarantee a payoff of at least

$$\text{Payoff} \geq \max_x \left(\min_y y^T A x \right)$$

Similarly, let $\bar{y} = \arg \min_{y \in \Delta Y} \left(\max_{x \in \Delta X} (y^T A x) \right)$

$$\text{Then } \max_x \min_y y^T A x \leq \bar{y}^T A \bar{x} \leq \min_y \max_x y^T A x$$

Thm (von-Neuman)

$$\max_x \min_y y^T A x = \min_y \max_x y^T A x$$

Corollary: \bar{x}, \bar{y} is an equilibrium point
i.e. $\bar{y}^T A \bar{x} \leq \bar{y}^T A \bar{x}$

and $y^T A \bar{x} \geq \bar{y}^T A \bar{x}$

\Rightarrow no one has incentive to move.

Pseudo-Proof of Theorem:

Consider $\max_{x \in \mathbb{R}^n} \min_{y \in \mathbb{R}^m} y^T A x$

If we fix a particular x^* , then $\min_y y^T A x^* = \min_i (A x^*)_i$

(Put all chips in one bucket")

LP formulation:

$$\begin{aligned} v &= \max z \\ \text{(Primal)} \quad \text{s.t.} \quad z &\leq q_i x & q_i = i^{\text{th}} \text{ row of } A, i \text{ rows of } A. \\ &\sum x_i = 1 \\ &x_i \geq 0 \end{aligned}$$

\Rightarrow (Dual)

$$\begin{aligned} &\min \gamma \\ \text{s.t.} \quad \gamma &\geq y^T a^i \\ &\sum y_i = 1 \\ &y \geq 0 \end{aligned}$$

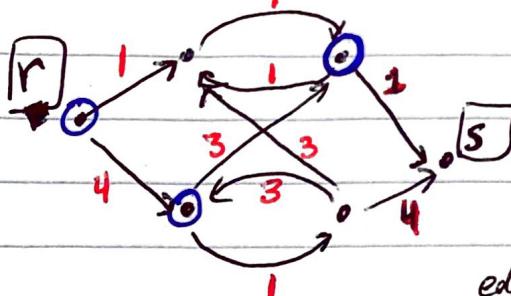
Dual equivalent to

$$\min_y \max_x y^T A x.$$

Since $\text{opt}(\text{Primal}) = \text{opt}(\text{dual}) \Rightarrow \text{Result}$



Max-flow / min-cut Problem



Consider max flow from left to right
 $\max \text{flow} \geq \min \text{cut}$.

edges leaving $\textcircled{1}$ is 4

Theorem: Given a network/directed graph with vertices $r, s,$

the maximum flow from r to s is exactly equal to the min cut between $r \& s$.

Defn: A $r-s$ cut is a partition of the vertices

$$V = R \sqcup S$$

$$\text{s.t. } r \in R, s \in S, \quad S \cap R = \emptyset$$

Defn: For a vertex v , $\delta^+(v) = \text{set of all edges leaving } v.$

$\delta^-(v) = \text{set of all edges entering } v.$

LP formulation: • let x_e for all $e \in E$ be the amount of flow along the edge.

$$\bullet \text{ for a vertex } v, \quad x(\delta^+(v)) = \sum_{e \in \delta^+(v)} x_e$$

$$\text{and } x(\delta^-(v)) = \sum_{e \in \delta^-(v)} x_e$$

Max flow $\max x(\delta^+(r)) - x(\delta^-(r))$

s.t.

$$x(\delta^+(v)) = x(\delta^-(v)) \quad \forall v \neq r, s$$

$$x_e \geq 0, \quad x_e \leq c_e$$

Min Cut

$$\min \sum_{e \in E} c_e y_e$$

$$\text{s.t. } z_r = 1, \quad z_s = 0$$

$$z_w - z_v + y_{vw} \geq 0 \quad \forall e \in E$$

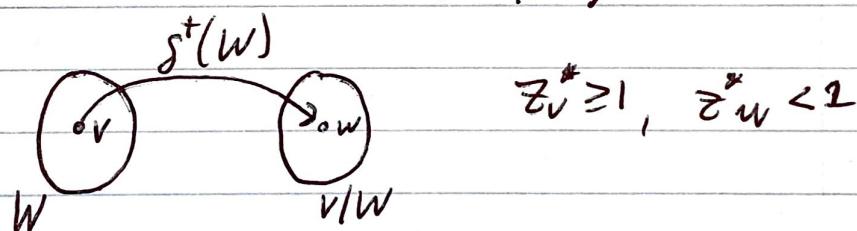
$$y_e \geq 0, \quad z_v \in \mathbb{R}$$

Complementary Slackness

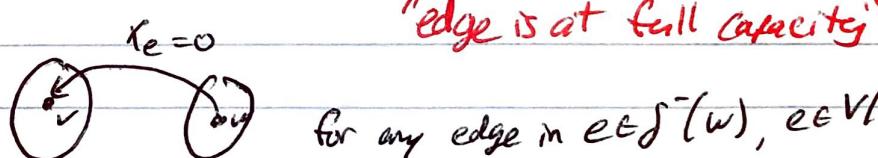
Let x^*, y^*, z^* be optimal solutions

$$\text{then } y_e^*(c_e - x_e^*) = 0, \quad (z_w - z_v + y_{vw})^* x_{vw} = 0$$

? $w = \{v \mid z_v^* \geq 1\}$; any edge in $\delta^+(w)$.



$$(z_w^* - z_v^* + y_{vw}^*) \geq 0 \Rightarrow y_{vw}^* > 0 \Rightarrow x_{vw}^* = c_{vw}$$



new case

for any edge in $e \in \delta^-(w), e \in V/W$

$$z_w^* - z_v^* + y_{vw}^* \geq z_w^* - z_v^* > 0$$

$$z_v^* < 1, \quad z_w^* \geq 1 \Rightarrow x_{vw}^* = 0$$

\Rightarrow Capacity of the cut is equal to the flow value.



LP Solves IP

Theory of Totally Unimodular Matrices

Defn: A is called Totally Unimodular if every square submatrix has determinant $\pm 1, 0$.

Thm: $A = T \cdot U$. then

$\min \{c^T x \mid Ax = b, x \geq 0\}$
always has integral optimal solutions for every
integral vector b .

Proof: Let x^* be the opt solution.

Then $\exists A_B$ ($m \times m$) s.t. $x_B^* = \tilde{A}_B^{-1} b$, $x_N = 0$.

\tilde{A}_B^{-1} is integral.

\Rightarrow by Cramers rule, x_B^* is integral.

Thm: Let A be a $\pm 1, 0$ matrix. If every column
of A has at most one $+1$ and one -1 ,
then A is T.U.

Proof: ~~if~~ Let B be a submatrix of A .

if B has a 0 column (or row) then $\det(B) = 0$.

if B has a column w/ exactly 1, or -1, then

expand about this column (induction)

if B has all $1 \neq -1$ columns, then sum of rows is 0.

\Rightarrow linearly dependent $\Rightarrow \det(B) = 0$.

D

Prop: $A = \text{LT.U.} \Leftrightarrow \begin{bmatrix} A \\ I \end{bmatrix} \text{ is T.U.} \Leftrightarrow \begin{bmatrix} A \\ I \end{bmatrix} \text{ is unimodular.}$

Vertex Cover:

- Find a subset of vertices which touch all edges.

Approximate Solution:

given weights c_v , $x_v \in \{0,1\}$

$$\min \sum_{v \in V} c_v x_v$$

$$\text{s.t. } x_v + x_w \geq 1 \quad \forall (v,w) \in E$$

$$0 \leq x_v \leq 1 \quad \forall v \in V.$$

Solve using LP to get x^* .

• If $x^* \in \mathbb{Z}^n$, done!

• If not, let

$$S = \{v \mid x_v^* \geq \frac{1}{2}\}$$

Claim 1: S is a vertex cover

Claim 2: $\boxed{\text{weight}(S) \leq 2 \text{ weight}(S^*)}$

$$= 2 \sum_{v \in V} c_v x_v^*$$

□

Ellipsoid Method (1979) Fachian

Input: a convex set $S \subseteq \mathbb{R}^n$

given to you by a separation oracle
 $\exists E.S.t. \text{Vol}(S) > \epsilon.$

Output: A point $x \in S$ or " $S = \emptyset$ ".

Idea: Assume \exists an ellipsoid

$$E_{M, z_0} = \left\{ x \in \mathbb{R}^n \mid (x - z_0)^T M (x - z_0) \leq 1 \right\}$$

$M = \text{positive definite}$

$$S \subseteq E_{M, z_0}, z_0 = \text{center of } E$$

* Check if $z_0 \in S$.

→ if $z_0 \notin S$, find new ellipsoid that encloses $H_k^+ \cap E_k$.

Construct $E_{M_{k+1}, z_{k+1}}$ & Repeat while

$$\text{Vol}(E_{M_k, z_k}) > \text{vol}(S).$$

Lemma: one can explicitly construct from E_{M_k, z_k} & H_k ,

the exact equations for $E_{M_{k+1}, z_{k+1}}$

with minimum volume &

$$= \frac{1}{2(n+1)}$$

$$\text{Vol}(E_{M_{k+1}, z_{k+1}}) < \text{Vol}(E_{M_k, z_k}) \cdot e$$

After K Steps

$$\text{Vol}(E_{M_k, z_k}) < \text{Vol}(E_{M_0, z_0}) e^{-\frac{k}{2(n+1)}}$$

Note: $\text{vol}(S) < \text{vol}(E_{M_0, z_0}) e^{-k/2(n+1)}$

$$\Rightarrow k \leq 2(n+1) \ln \left(\frac{\text{vol}(E_{M_0, z_0})}{\text{vol}(S)} \right)$$

Also: If $P = \{y \mid A^T y \leq b\}$, $A = n \times n$ matrix,

if $P \neq \emptyset$, then it must contain a ball of radius

$$-\gamma \phi^3$$

$$2 \quad \text{where } \phi = O(m \cdot n \cdot \max(\log_2 A_{ij}, \log_2 b_i))$$

History: 1989 (Renegar, Gonzaga, Roos \& Vial)

find an optimal ~~so~~ Interior Point Method in $O(n^3 L)$

2000 (Deza, Terlaky) \nearrow best possible ever.

Simplex Counter-example (exponential)

Klee-Minty Cube

$$0 \leq x_1 \leq 1, \quad \epsilon x_{k-1} \leq x_k \leq 1 - \epsilon x_{k-1}, \quad k=2, \dots, n$$



Other algorithms for LP:

- 1.) Criss - Cross Method
- 2.) Perceptron Algorithm
- 3.) Motzkin Relaxation Algorithm
- 4.) Vem pala Style Algorithms

Open: Prove any of those are polynomial

Curvature

Total curvature = length of the Gauss map

Curve $\gamma \rightarrow \frac{\dot{\gamma}}{\|\dot{\gamma}\|}$ tangent vector

- Gauss map takes point on curve to tangent vector pointing on the Sphere

Open: $2\pi n \geq$ total curvature

Strongly Polynomial = $O(n^k)$ k constant
* Holy grail for LP *

Linear optimization

$$(P) \max \{c^T x \mid Ax = b, x \geq 0\} \quad A \in \mathbb{R}^{m \times n}$$

$$(D) \min \{b^T y \mid A^T y \geq c\} = \min \{b^T y \mid A^T y - s = c, s \geq 0\}$$

Theorem (Weak Duality)

If x is feasible for P, y feasible for D, then

$$b^T y \geq c^T x$$

Theorem (Strong Duality)

For each pair of linear programs, one of the following is true:

1. (P), (D) are both infeasible
2. (P) unbounded, (D) infeasible
3. (P) infeasible, (D) unbounded
4. (P) feasible, (D) feasible

$$\Rightarrow b^T y^* = c^T x^* \quad (\textcircled{1})$$

Theorem (Complementary Slackness)

When x^*, y^* are optimal solutions of P, D, then

$$x_i^* s_i^* = 0, \quad i=1, \dots, n$$

Moral: Solving LP's results to solving the following system
of equations and inequalities

$$Ax = b, \quad A^T y - s = c, \quad x^T s = 0$$

$$x \geq 0, \quad s \geq 0$$

Simplex Method (1947 George Dantzig)

Satisfy $Ax = b$, $A^T y - s = c$, $x \geq 0$

Defn: X is a Basic Feasible Solution if \exists
a submatrix A_B of A with rank m (invertible)
such that

$$x_j = 0 \text{ if } j \notin B$$

$$(\text{therefore } A_B x_B = b, \quad x_B = A_B^{-1} b)$$

Lemma: If there exists an optimal solution of (P) ,
then there exists an optimal BFS of (P) .
(Assume bounded solution)

Proof: Need 2 things:

1. Every BFS = an extreme point of (P)

and vice versa

2. Every point in P can be written as a convex
combination of extreme points + direction of unboundedness.

$$\text{hence } x^* = d + \sum \alpha_i v_i$$

• note that $c^T d = 0$ since otherwise we could scale in
that direction

$$\text{then } c^T x^* = \sum \alpha_i c^T v_i \leq \max_i c^T v_i \Rightarrow c^T x^* = c^T v_i$$

□

Representation

$$b = Ax = A_B x_B + A_N x_N \Rightarrow x_B = A_B^{-1} b - A_B^{-1} A_N x_N$$

$$\begin{aligned} z = c^T x &= c_B^T x_B + c_N^T x_N \\ &= c_B^T A_B^{-1} b + (c_N^T - \underbrace{c_B^T A_B^{-1} A_N}_{\parallel}) x_N \\ c_j - y \alpha_j &= s_j \quad y \end{aligned}$$

Pivot Rule: Select index from those non-basic variables that correspond to

$$s_j < 0$$

i.e. $c_j - \bar{y}^T a_j = -s_j > 0$.

Many choices for pivot rules!!!

Entering Variable: Call a_j the column vector of the entering variable $x_j = 0$.

Increase x_j slowly by $t > 0$

$$x_B^{\text{new}} = x_B^{\text{old}} - t(A_B^{-1} a_j)$$

\Rightarrow eventually one other variable will vanish

\rightarrow this variable leaves the basis

\rightarrow OR no variable vanishes \Rightarrow unbounded direction.

How to find first BFS? $Ax=b, x \geq 0, b \geq 0$ (otherwise multiply by -1)

Set $\bar{A} = (A, I_m)$, $\bar{x} = (x, x_{n+1}, \dots, x_{n+m})$

Solve $\bar{A}\bar{x} = b, \bar{x} \geq 0$, using $\bar{x}_0 = (0, b)$

i.e.

$$\max -(x_{n+1} + \dots + x_{n+m})$$

s.t. $\bar{A}\bar{x} = b$

if $\max = 0 \Rightarrow$ we found an initial BFS

if $\max < 0 \Rightarrow$ original problem is infeasible.

Open Problem: Can you find a plot rule s.t. the number of plot rules necessary to get an optimal solution is polynomial in $n \& m$?

Can you bound the diameter of a graph in $n \& m$?

Interior Point Methods (1984 Karmarkar)

Theorem: For all $\lambda > 0$, ~~the problem~~ has a unique real solution $(x^*(\lambda), y^*(\lambda), s^*(\lambda))$ s.t.

1.) $x^*(\lambda)$ is an optimal solution to

$$\max(c^T x + \sum_{i=1}^n \lambda \log(x_i) \mid Ax=b, x \geq 0)$$

2.) In the limit, the point $(x^*(0), y^*(0), s^*(0))$ is an optimal solution to (P).

Defn: The set of all real solutions of * as $\lambda \rightarrow 0$ is the Central Path

$$* \quad Ax=b, x_i s_i = \lambda, A^T y - s = c, x \geq 0, s \geq 0$$

Intro NLP, Defn Concavity, Thm localmax \Rightarrow globalmax, $f''(x) \geq 0$ or ≤ 0

Non-Linear Programming

solutions not necessarily at a corner point	<u>Applications</u>		
can have finitely many optimal solutions	* portfolio Allocation		
optimal solutions in the interior	* Profit Maximization		
local optima are not necessarily global optima	* Volume Maximization		

Suppose you have n investments. The return on investment i in the next period is R_i , a random variable. Let x_i be the percentage of money to invest in i .

$$\text{Max. } Z = E \left[\sum_{i=1}^n x_i R_i \right] - \mu \text{Var} \left[\sum_{i=1}^n x_i R_i \right]$$

return

indicates level of risk willingness

non-linear!

$$\text{Constraints: } \sum_{i=1}^n x_i = 1, x_i \geq 0 \forall i \Rightarrow Z = \sum_{i=1}^n x_i E(R_i) - \mu \sum_{i \neq j} x_i x_j \text{Cov}(R_i, R_j)$$

Solve $Z'(x) = 0$. Is this sufficient and necessary for x to be a max? NO!

→ not sufficient: min, plateau, local max, local min

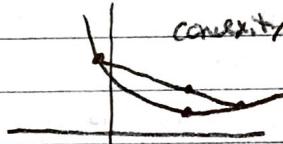
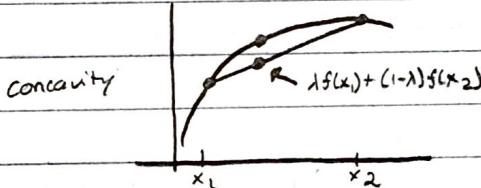
→ not necessary! not differentiable, boundary max or min, increasing function

Want f continuous, differentiable, domain of f is unbounded, concave!

$\Rightarrow f'(x_0) = 0$ iff x_0 is a global max.

Defn: A function is concave if for any x_1, x_2 in the domain of f and $\lambda \in [0,1]$ we have

$$f(\lambda x_1 + (1-\lambda)x_2) \geq \lambda f(x_1) + (1-\lambda) f(x_2)$$



Defn: A function is convex if "

"

Theorem: If f is concave on \mathbb{R} , then any local maximum of f is a global max of f .

Proof: let \bar{x} be a local max of f . Suppose \bar{x} is not a global max. Then there exists an $\bar{x}' \neq \bar{x}$ s.t. \bar{x}' is a global max of f and $f(\bar{x}') > f(\bar{x})$. Thus, for $\lambda \in [0,1]$

$$f(\lambda \bar{x} + (1-\lambda) \bar{x}') \geq \lambda f(\bar{x}) + (1-\lambda) f(\bar{x}') > \lambda f(\bar{x}) + (1-\lambda) f(\bar{x}) = f(\bar{x})$$

Since we can choose λ arbitrarily close to 1, we can find points with larger function values which violates that \bar{x} is a local max. \Rightarrow Thus \bar{x} is a global max

Thm: Concavity \Rightarrow continuity. No proof given.

Thm: If f'' exists everywhere then $f''(x) \geq 0$ everywhere (\Rightarrow f is convex)

$f''(x) \leq 0$ everywhere (\Rightarrow f is concave)

Solution $f'(x) = 0$. Use numerical approximating because solving analytically can be a pain.

→ Bisection Method: (f is concave) looking for global max x^* know lower & upper bound \underline{x} and \bar{x} .
Let $\hat{x} = \frac{\underline{x} + \bar{x}}{2}$. If $f'(\hat{x}) > 0$ then $\underline{x} = \hat{x}$, If $f'(\hat{x}) < 0$ then $\bar{x} = \hat{x}$.

Stop when \hat{x} is within ϵ of the optimal solution (pick ϵ in advance) $\Leftrightarrow \bar{x} - \underline{x} < 2\epsilon$

* See handout for example with $f(x) = e^x \cos(x) + x$, Max on $[0, \pi]$.

Bisection method takes K iterations where $\frac{\Delta \bar{x}}{2^K} \leq 2\epsilon \Rightarrow K = \log_2\left(\frac{\Delta \bar{x}}{\epsilon}\right) - 1$

→ Newton's Method: root finding method. Converges very quickly to optimal solutions.



Given x_i , the equation of the tangent line $(x - x_i)m = (y - y_i)$

$$\Rightarrow y = m(x - x_i) + y_i, \quad y_i = f(x_i), \quad y = 0, \quad m = f'(x_i)$$

$$\Rightarrow 0 = f'(x_i)(x - x_i) + f(x_i) \Rightarrow x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

But, we want $f'(x) = 0$, so $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$

→ Screwing with Newton's Method



Multivariate NLP

① All partial derivatives = 0 OR ② on a boundary OR ③ where at least one partial derivative DNE

$\nabla f \equiv \text{grad } f$ is the vector of partial derivatives $\nabla f(x, y) = f_x(x, y)\hat{i} + f_y(x, y)\hat{j}$

① $\Leftrightarrow \nabla f(x, y) = \vec{0}$

∇f points in the direction of maximum increase for f .

Directional Derivative: $f_{\vec{u}}(x, y) = \nabla f(x, y) \cdot \vec{u}$ (\vec{u} is a unit vector)

$\vec{u} \cdot \vec{v} = \|\vec{u}\| \|\vec{v}\| \cos \theta$ where θ is the angle between \vec{u} and \vec{v} .

$f_{\vec{u}}(x, y) = \|\nabla f(x, y)\| \|\vec{u}\| \cos \theta = \|\nabla f(x, y)\| \cos \theta$ what value of θ maximizes this? $\theta = 0!$

Hessian matrix $H(f) = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix}$ look at $\det(H)$

If $\nabla f(a, b) = \vec{0}$, $\nabla^2 f(a, b) = 0$ Then second partials test fails

\$\$ \det(H(a, b)) < 0 \text{ you have a saddle point at } (a, b)

\$\$ \det(H(a, b)) > 0 \text{ if } f_{xx} \text{ or } f_{yy} > 0 \Rightarrow \text{local min at } (a, b)

\$\$ \text{if } f_{xx} \text{ or } f_{yy} < 0 \Rightarrow \text{local max at } (a, b)

$$d(a, b) = f_{xx}(a, b)f_{yy}(a, b) - [f_{xy}(a, b)]^2$$

If f_{xx} & f_{yy} are both positive why don't we necessarily have a min at (a, b) ? Because of twisting

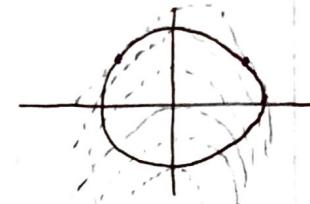
$$f(x, y) = x^4 + y^4 - 4xy + 1 : \nabla f(x, y) = (4x^3 - 4y)\hat{i} + (4y^3 - 4x)\hat{j} \Rightarrow \nabla f(x, y) = \vec{0} \Leftrightarrow \begin{cases} 4x^3 - 4y = 0 \\ 4y^3 - 4x = 0 \end{cases} \Leftrightarrow \begin{cases} x^3 = y \\ y^3 = x \end{cases}$$

$x = 0, \pm 1$ if $x = 0$ $y = 0$, if $x = 1, y = 1$, if $x = -1, y = -1$ next look at Hessian.

$\det(H)(0, 0) < 0 \Rightarrow$ Saddle

$(1, 1) > 0, f_{xx} > 0$ local min

$(-1, -1) > 0, f_{xx} > 0$ local min



Lagrange Multipliers: Multivariate constrained optimization

$$\text{Maximize } f(x,y) = x^2 + y \quad \text{s.t. } x^2 + y^2 = 1$$

∇f is perpendicular to circle at optimal point. $g(x,y) = x^2 + y^2$ when $g=1$. Think of circle as a level curve.

∇g at optimal points perpendicular to the circle $\Rightarrow \nabla f(x,y) = \lambda \nabla g(x,y)$ λ = Lagrange multiplier

$$\nabla f(x,y) = (2x, 1) \quad \nabla g(x,y) = (2x, 2y)$$

$$2x = \lambda 2x \quad 1 = \lambda 2y \quad x^2 + y^2 = 1 \Rightarrow (0,1), (0,-1), (\pm\frac{\sqrt{3}}{2}, \frac{1}{2})$$

First method to produce max & min on interior use Lagrange multipliers to find max/min on boundaries.

What does λ mean? Let (x_0, y_0) be the optimal solution to maximize $f(x,y)$ s.t. $g(x,y) = c$.

express as $(x_0(c), y_0(c))$. How does $f(x_0(c), y_0(c))$ change as c changes?

$$\frac{df}{dc} \Big|_{(x_0, y_0)} = \frac{df}{dx_0} \frac{x_0}{dc} + \frac{df}{dy_0} \frac{y_0}{dc} \quad \left(\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial y_0} \right) = \lambda \left(\frac{\partial g}{\partial x_0}, \frac{\partial g}{\partial y_0} \right)$$

at a point of optimal value $\Rightarrow \frac{df}{dc} \Big|_{(x_0, y_0)} = \lambda \frac{\partial g}{\partial x_0} \frac{\partial x_0}{dc} + \lambda \frac{\partial g}{\partial y_0} \frac{\partial y_0}{dc} = \lambda \frac{\partial g}{\partial c} \Big|_{(x_0, y_0)} = \lambda \rightarrow$ shadow price

λ is the shadow price of f with respect to the constraint $g(x,y) = c$ [note: only instantaneous shadow price]

$$f''(x) \leq 0 \Leftrightarrow f \text{ is concave} \quad (\text{if } f'' \text{ exists } \forall x \in \mathbb{R})$$

Characterization of Concavity

Defn: A function $f(\vec{x})$ is concave if $\forall \vec{x}_1, \vec{x}_2 \in \mathbb{R}^n$

$$f(\lambda \vec{x}_1 + (1-\lambda) \vec{x}_2) \leq \lambda f(\vec{x}_1) + (1-\lambda) f(\vec{x}_2)$$

Alternate characterization: Suppose f has continuous first partials in \mathbb{R}^n , then

$$f \text{ is concave} \Leftrightarrow \nabla f(\vec{x}) \leq \nabla f(\vec{x}_0) + \nabla^2 f(\vec{x}_0)(\vec{x} - \vec{x}_0) \quad \forall \vec{x}, \vec{x}_0 \in \mathbb{R}^n$$

$$\rightarrow \text{Single variable } y = f(x_0) + f'(x_0)(x - x_0)$$

Taylor's Theorem in 2 variables (up to first order)

If f is twice-differentiable on \mathbb{R}^2 then for some (a,b) on the line segment between (x_0, y_0) and (x_1, y_1)

$$f(x,y) = f(x_0, y_0) + f_x(x_0, y_0)(x-x_0) + f_y(x_0, y_0)(y-y_0) + \frac{1}{2} [f_{xx}(a,b)(x-x_0)^2 + 2f_{xy}(a,b)(x-x_0)(y-y_0) + f_{yy}(a,b)(y-y_0)^2]$$

Using ∇f and matrixes this is $f(x,y) = f(x_0, y_0) + \nabla f(x_0, y_0) \cdot [x-x_0, y-y_0] + \frac{1}{2} [x-x_0, y-y_0] \begin{bmatrix} f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{bmatrix} \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix}$

for $f: \mathbb{R}^n \rightarrow \mathbb{R}$, this is $f(\vec{x}) = f(\vec{x}_0) + \nabla f(\vec{x}_0)[\vec{x} - \vec{x}_0] + \frac{1}{2} [\vec{x} - \vec{x}_0]^T H(\vec{x}) [\vec{x} - \vec{x}_0]$ for some \vec{x}

Note: $\vec{x} - \vec{x}_0$ is a column vector that is a convex combination of \vec{x} and \vec{x}_0

Theorem: Suppose f has continuous second partial derivatives in \mathbb{R}^n . Then f is concave \Leftrightarrow its Hessian Matrix is negative semi-definite.

Defn: A is negative semi-definite ($\Leftrightarrow \vec{x}^T A \vec{x} \leq 0 \quad \forall \vec{x} \in \mathbb{R}^n$ (or \mathbb{C}^n)).

Proof: H is negative semi-definite $\Leftrightarrow \vec{z} \in \mathbb{R}^n \Leftrightarrow [\vec{x} - \vec{x}_0]^T H(\vec{z}) [\vec{x} - \vec{x}_0] \leq 0 \quad \forall \vec{x}, \vec{x}_0 \in \mathbb{R}^n$

$$\Leftrightarrow f(\vec{z}) \leq f(\vec{x}_0) + \nabla f(\vec{x}_0)[\vec{z} - \vec{x}_0] \quad \forall \vec{x}, \vec{x}_0 \in \mathbb{R}^n$$

$\Leftrightarrow f$ is concave on \mathbb{R}^n

$$\text{Ex: } f(x_1, x_2, x_3) = 4x_1^2 + x_2^2 + x_3^2 - 2x_2 x_3 \text{ is convex. (positive semi-definite Hessian)}$$

Details



$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} 8 & 0 & 0 \\ 0 & 2 & -2 \\ 0 & -2 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 8x_1^2 + 2x_2^2 + 2x_3^2 - 4x_2x_3$$

$$= 8x_1^2 + 2(x_2^2 - 2x_2x_3 + x_3^2) = 8x_1^2 + 2(x_2 - x_3)^2 \Rightarrow \text{PSD Hessian} \Leftrightarrow \text{convex!}$$

Other characteristics of NSD

① All eigenvalues are non-positive (iff)

② See text on Matrix Algebra for characteristic terms of principle minors.

Thm: If $f_i(\vec{x})$ is concave, then $\sum_{i=1}^n f_i(\vec{x})$ is also

Thm: If $f(\vec{x})$ is concave, and $g \geq 0$ then

$g \cdot f(\vec{x})$ is concave AND $-g \cdot f(\vec{x})$ is convex.

Thm: If $f(\vec{x})$ is concave and $g(y)$ is increasing, then $g(f(\vec{x}))$ is concave

Ex: $f(x_1, x_2, x_3) = e^{-(x_1^2 + x_2^2 + x_3^2)}$

e^x is increasing, $-x_1^2 - x_2^2 - x_3^2$ is concave

thus f is concave. Also

$$H_f(x, y) = \begin{bmatrix} -2 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -2 \end{bmatrix} \Rightarrow \text{NSD} \Rightarrow \text{concave.}$$

21.5.2. Nullstellensatz

21.5.3. Positivstellensatz

21.5.4. Putinar's Positivstellensatz

21.5.5. Handelman Decompositions

22. Some notes

22.0.1. Semi-Definite Programming (SDP)

$$\begin{aligned} \max \quad & \sum_{ij} c_{ij} x_{ij} = \text{Trace}(c \cdot x) \\ \text{s.t.} \quad & \text{Trac}(A_i \cdot x) = b_i \quad i \in I \\ & x \geq 0 \quad (x \text{ is a } n \times n \text{ matrix}) \end{aligned}$$

Theorem 22.1: Groichel-Lovesz - Schnver (GLS)

Optimization / test faribility over a convex set $S \subseteq \mathbb{R}^d$ is equivalent to separation over S . (i.e. we need a separation oracle).

Let $S = \{\Lambda \in \mathbb{R}^{n \times n} \mid \Lambda \geq 0\} \subseteq \mathbb{R}^{n \times n}$

Definition 22.2

Separation oracle for S - given a matrix Q , decide if Q belongs to S OR Find a Separating hyperplane.

For SDP cone:

- find all eigenvalues of Q
- if $\lambda_i \geq 0 \forall i$, then $Q \in S$
- otherwise $\exists y \in \mathbb{R}^n$ s.t.

$$\begin{aligned} Qy &= \lambda y, \quad \lambda < 0 \\ \Rightarrow y^\top Qy &= y^\top \lambda y = \lambda \|y\|_2^2 < 0 \end{aligned}$$

- Separating hyperplane $y^\top \Lambda y = 0$ ie. $y^\top \Lambda y \geq 0$ is valid for S .

Therefore, GLS \Rightarrow SDP can be solved (By Ellipsoid Method) (but we can also do this with interior point methods)

22.0.2. Application in Combinatorial optimization

Definition 22.3: Stable Set Problem (NP-complete)

Find a maximum size stable set of vertices where no two vertices are connect by an edge

Proposition 22.4

S is a stable set $\Leftrightarrow V/S$ is a vertex cover

$$\begin{aligned} \max \quad & \sum x_i \\ \text{sit.} \quad & x_i + x_j \leq 1 \quad \forall (i, j) \in E \\ & x_i \in \{0, 1\} \\ LP(G) = \{x \mid & x_i + x_j \leq 1 \quad \forall (i, j) \in E, \quad 0 \leq x_i \leq 1\} \end{aligned}$$

$$\text{stab}(G) = \text{conv}(\{x \mid x_i + x_j \leq 1, x_i \in \{0, 1\}\})$$

Definition 22.5

A clique is a set of vertices sit. $i, j \in E$ for all $i, j \in C$ (the clique)

Observation: For even clique C , at most 1 vertex from (C) belongs to a stable set.

$$\text{Clique}(G) = \left\{ x \mid \sum_{i=c} x_i \leq 1, 0 \leq x_i \leq 1 \right\}$$

Recall that we can write binary vesicles using quadratic constraints. One formulation is

$$\begin{aligned} x_i^2 - x_i &= 0 \\ x_i \cdot x_j &= 0 \quad \forall (i, j) \in E \end{aligned}$$

INTRODUCE "LINEARIZING VARIABLES" y_{ij}

$$\begin{aligned} y_{ij} &= x_i x_j, \quad y_{jj} = x_j^2 = x_j, \quad y_{0j} = y_{j0} = x_j, y_{00} = 1 \\ \Rightarrow Y &= \begin{pmatrix} 1 \\ x \end{pmatrix}^{(1x^\top)} \end{aligned}$$

* Y is a positive semidefinite matrix \Rightarrow Relaxation: $Y \leq 0$ i.e. need not be that $\gamma = ()^()$ for rank + matrices.
 \Rightarrow cannot recover x from the constraint.

$$\begin{aligned} TH(G) &= \left\{ x \mid Y \geq 0, y_{ij} = 0 \quad \forall (i, j) \in E, \text{ "Theta booty"} \right. \\ &\quad y_{0j} = y_{j0} = y_{ij} = x_j, \quad \text{from Lovasz} \\ &\quad \left. y_{00} = 1, \quad 0 \leq x_j \leq 1 \right\} \end{aligned}$$

Theorem 22.6

$$\text{Stab}(G) \subseteq TH(G) \subseteq \text{Clique}(G)$$

Proof. clearly $\text{Stab} \subseteq TH(G)$ by construction. We want to show that $TH(G) \subseteq \text{clique}(G)$.

Assume $x \in TH(G) \Rightarrow \exists Y \geq 0$ in $TH(G)$ (definition).

$$\Rightarrow v^\top Y v \geq 0 \quad \forall v \in \mathbb{R}^{n+1}$$

Take an clique C

$$\text{set } V_i = \begin{cases} i & \text{if } i = 0 \\ -1 & \text{if } i \in C \\ 0 & \text{otherwise} \end{cases}$$

Then $0 \leq v^\top Y v = \sum_{i,j} y_{ij} v_i v_j$

$$\#(y_{ij} = 0 \text{ it } (0, j) \in E)$$

$$= \sum_{i,j \in C} y_{ij} v_i v_j + \sum_{i=2}^n y_{i0} v_i v_0 + \sum_{j=1}^n y_{0j} v_0 v_j + y_{00} v_0^2$$

$$= 0 + \sum_{j=1}^n y_{jj} - \sum_{i=c} x_i - \sum_{i \in C} x_i + 1$$

$$\Rightarrow \sum_{i \in C} x_i \leq 1 \quad \text{i.e. } x \in \text{chique}(6)$$



22.0.3. Max-cut

Given a graph $G = (V, E)$ and weights on the edges, find a partition of $V = S \cup V/S$ such that the total weight of the edges crossing the partition is maximized.

Theorem 22.7: Goemans - Williamson Approximation Algorithm (SDP)

There is an efficient (poly time) algorithm which gets within 820% of the optimal max cut. - I any algorithms better then this unless $P = NP$

Formulation: $X_i = \begin{cases} 1 & \text{if } i \in S \\ -1 & \text{if } i \notin S, \end{cases}, i \in V.$

$$x_i \in \{\pm 1\} \Leftrightarrow x_i^2 = 1$$

$$x_i \neq x_j \Leftrightarrow \frac{1 - x_i x_j}{2} = 1$$

$$\Rightarrow \max w_{ij} \left(\frac{1 - x_i x_j}{2} \right)$$

$$\text{s.t. } x_i^2 = 1$$

22.0.4. SOS Theory

Definition 22.8

f is convex if $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$

$$\forall x, y \in \mathbb{R}^n, \lambda \in [0, 1]$$

Proposition 22.9

if f is twice-differentiable, then f is convex $\Leftrightarrow H_f = \left[\left(\begin{array}{c} \partial f \\ \partial x_i \partial x_j \end{array} \right) \right] \geq 0 \quad \forall x \in \mathbb{R}^n$ "the Hessian is PSD for all $x \in \mathbb{R}^n$ "

Polynomial Matrices $P(x) = [P_{ij}] . (m \times m)$ polynomial in x

Definition 22.10

$P(x)$ is a PSD matrix if $P(x)$ is PSD $\forall x \in \mathbb{R}^n$

Fact: $P(x)$ is PSD $\Leftrightarrow y^\top P(x)y \geq 0$ as an element of $\mathbb{R}[x, y]$

$$y \in \mathbb{R}^m$$

Def: $P(x)$ is an Sos matrix if \exists a polynomial matrix $\mu(x)$ such that $P(x) = \mu^T(x)\mu(x)$

Fact: $P(x)$ is sos $\Rightarrow P(x)$ is PSD

Fact: $P(x)$ is an SOS matrix $\Leftrightarrow y^\top P(x)y$ is a sum of squares polynomial.

Example PSD-Matrix not SOS matrix

$$A(x) = \begin{bmatrix} x_1^2 + 2x_2^2 & -x_1x_2 & -x_1x_3 \\ -x_1x_2 & x_2^2 + 2x_3^2 & -x_2x_3 \\ -x_1x_3 & -x_2x_3 & x_3^2 + 2x_1^2 \end{bmatrix}$$

$A(x)$ is PSD, but $A(x)$ is not sos

* $A(x)$ is not a Hessian matrix since $\frac{\partial}{\partial x_3} \left(\frac{\partial^2 p}{\partial x_1^2} \right) \neq \frac{\partial}{\partial x_1} \left(\frac{\partial p^2}{\partial x_1 \partial x_3} \right)$

Note: if $n = 1$, then PSD = SOS

22.0.5. Deciding complexity of a polynomial function

Defn: $P(x)$ is sos-convex if $H(x)$ is an Sos-matrix Clearly Sos -convex \Rightarrow Convex

Theorem (Amir Ali & Pablo Rarillo) \exists a polynomial that is convex, but not sos-convex.

Note: Deciding $H(x) = \text{sos}$ is easy since we just show that $y^\top H(x)y$ is a sos polynomial (use SDP to solve)

23. Algebraic and Geometric Ideas

Jesus Book

Jesus Book

Jesus book

Here we will discuss one more important application of Gröbner bases. We've seen how to solve polynomial equations, and now we're going to use Gröbner bases for integer programming. We consider the problem

$$\begin{array}{ll} \min & c \cdot x \\ \text{subject to} & Ax = b \\ & x \geq 0, x \in \mathbb{Z}^n \end{array}$$

where $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$.

Roughly speaking, the first goal is to understand the vectors in the kernel of A that are integer. That is to say, we want to analyze the set $\ker(A) \cap \mathbb{Z}^n = \{x \mid Ax = 0, x \in \mathbb{Z}^n\}$. For that purpose, we define the main object here, the *toric map* or *toric ideal*.

Definition 23.1: T

e map $\pi_A : \mathbb{N}^n \rightarrow \mathbb{Z}^m$ is defined by $u \mapsto Au$.

We will be interested in the following extension $\widehat{\pi}_A$ of the map π_A . The map is defined

$$\widehat{\pi}_A : K[x_1, \dots, x_n] \rightarrow K[t_1^\pm, \dots, t_m^\pm].$$

Every variable may appear with a positive or negative exponent. It's not the usual way to think of rings of polynomials. Here we have Laurent polynomials. For instance, we may have

$$\frac{t_1}{t_2^2} - \frac{7t^2}{t_1^7} - 3t_1t_2^7.$$

Laurent polynomials can be thought of as regular polynomials that are using variables of the form t_1 and $1/t_1$. We can now define a map by simply showing what happens to each variable x_i . The map $\widehat{\pi}_A$ maps

$$x_i \mapsto t_1^{a_{i1}} t_2^{a_{i2}} \cdots t_m^{a_{im}}$$

Here, a_i denotes the i th column of A . Suppose you have an arbitrary polynomial $f = \sum c_\alpha x^\alpha$. The map $\widehat{\pi}_A$ is linear, so

$$\widehat{\pi}_A(\sum c_\alpha x^\alpha) = \sum c_\alpha \widehat{\pi}_A(x^\alpha) = \sum c_\alpha (\widehat{\pi}_A(x_1))^{\alpha_1} \widehat{\pi}_A(x_2)^{\alpha_2} \cdots \widehat{\pi}_A(x_n)^{\alpha_n}.$$

Example 23.2: L

tA be the matrix

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix}.$$

Note that we could also write an example with some negative entries. Here the map is defined with spaces

$$K[x_1, \dots, x_4] \rightarrow K[t_1, t_2].$$

Suppose that the polynomial f is $7x_1^3 + x_2x_3 - 7x_4^3$. What is $\widehat{\pi}_A(x_1)$? It's just t_1 . What is $\widehat{\pi}_A(x_2)$? It's just t_1t_2 , by looking at the second column of A . What is $\widehat{\pi}_A(x_3)$? It's just $t_1t_2^2$. Finally, $\widehat{\pi}_A(x_4)$ is $t_1t_2^3$. So when applied to f , we obtain $7t_1^3 + (t_1t_2)(t_1t_2^2) - 7(t_1t_2^3)^3$ which simplifies to $7t_1^3 + t_1^2t_2^3 - 7t_1^3t_2^9$.

This map is going to tell us everything we want to know about the elements of the kernel of A that are integer. Once we understand the kernel of A , if we somehow obtain a single feasible solution, we can use the kernel of A as candidates for feasible directions to move in. Much like in the simplex method, we will pivot into feasible directions and attempt to find a more optimal value. The directions are found by the strange map. We will look at the kernel of the map $\widehat{\pi}_A$.

The question is, what is $\ker(\widehat{\pi}_A)$? That is, which polynomials map to zero under the projection $\widehat{\pi}_A$? The first fact we need to see is that this kernel is an ideal.

Lemma 23.3

The set $\ker(\widehat{\pi}_A)$ is an ideal inside the polynomial ring $K[x_1, \dots, x_n]$.

This ideal is called the *toric ideal*, which we will denote from now on by I_A . (The name of this ideal is unfortunately not suggestive/descriptive. There are mathematical reasons why these are called *toric*. The name comes from a torus action.) By the Hilbert Basis Theorem, there must be finitely many polynomials that generate this ideal. Our job is to compute these polynomials. Here is an example: The polynomial f earlier is not in the kernel of the map. Let's look at an example of a polynomial that does get mapped to zero.

Example 23.4

Let f be the polynomial $x_1x_3 - x_2^2$. When you apply the map $\widehat{\pi}_A$, we get $t_1(t_1t_2^2) - (t_1^2t_2^2) = 0$. So the polynomial f belongs to the kernel of the map $\widehat{\pi}_A$.

The point is that we should find all of the polynomials that are in the kernel. Here is the second observation: since I_A is an ideal,

Lemma 23.5

There exist finitely many polynomials f_1, \dots, f_s so that $I_A = \langle f_1, \dots, f_s \rangle$.

The polynomials f_i will actually end up being binomials: you're not going to believe it!

The following is the main theorem.

Theorem 23.6

For the toric ideal I_A associated to the matrix A , the following are true:

- (a) The ideal I_A is generated by finitely many binomials, i.e., by elements of the form

$$x_1^{u_1}x_2^{u_2}\cdots x_n^{u_n} - x_1^{v_1}x_2^{v_2}\cdots x_n^{v_n},$$

which we abbreviate by $x^u - x^v$, such that $Au = Av$.

- (b) For every monomial order \succ , there exists a Gröbner basis G of the ideal I_A which is composed only of binomials.

Continuing the running example,

Example 23.7: R

member the four columns of A are labeled by x_1, \dots, x_4 and the rows are indexed t_1 and t_2 . Then, the ideal I_A is generated as follows:

$$I_A = \langle x_1x_3 - x_2^2, x_2x_4 - x_3^2, x_1x_4 - x_2x_3, x_1^2x_4 - x_2^3, x_1x_4^2 - x_3^3 \rangle$$

The binomials belong in I_A , but they actually generate this ideal! We can show how to do this in Macaulay2.

For optimization, you might be thinking: how am I going to reconcile this with what I need to do to optimize? This is simple: you should think of this as a vector. For example, the square-free polynomial in the previous example should be associated with the vector $(1, -1, -1, 1)$. Notice that this vector is in the integer kernel of A . Similarly, the next binomial in the list is associated to the vector $(2, -3, 0, 1)$. We should think of these as oriented vectors.

The beautiful thing here is that these are integer kernels. In linear algebra, you compute real kernels. Before we prove the next theorem, we need the following lemma:

Lemma 23.8

The toric ideal I_A associated to the matrix A is a K -vector space generated by all binomials of the form $x^u - x^v$ such that $Au = Av$. In other words, every polynomial $f \in I_A$ can be written as a linear combination of the form

$$\sum c_{uv} (x^u - x^v), \quad (23.1)$$

where $c_{uv} \in K$ and $x^u - x^v \in I_A$.

In our example, the rank of the matrix A is two, but this is the dimension of the real kernel, in the sense of linear algebra. But, I'm not talking about that space. I need to make a basis for the integer points in the null space, which makes it much more difficult. What I'm saying is that, if you give me a vector in the integral kernel of the matrix A , I will write it as the integer linear combination of the five vectors associated to the binomials above.

Let's prove Lemma 23.8.

Proof. Note that $x^u - x^v \in I_A$ if and only if $Au = Av$. When you apply the map $\widehat{\pi}_A$, this is the same as doing $t^{Au} - t^{Av}$, which is 0. This follows from the definition of $(\widehat{\pi}_A)$.

We will prove this by contradiction. Suppose there is a polynomial $f \in I_A$ so that it cannot be written in the form (23.1). Using a monomial order \succ , assume that f is minimal in the sense that $LM_\succ(f)$ is the smallest possible. (We know this exists by the well-ordering property.) Our goal will be to find polynomial that is even “smaller” than f .

Now, we know that $f \in I_A$, which means that $\widehat{\pi}_A(f) = 0$. But that means that $f(t^{a_1}, t^{a_2}, \dots, t^{a_n}) = 0$.

Let x^u be the leading monomial of f with respect to \succ . So, $\widehat{\pi}_A(x^u) = t^{Au}$. The only way to become zero is: There exists x^v term inside f that cancels with x^u . In other words, $t^{Av} = t^{Au}$. Remember that x^u was the leading term. But $x^u = LM_\succ(f) \succ x^v$. Define f' to be the polynomial $f - \text{coefficient } x^u \text{ in } f(x^u - x^v)$.

By the earlier observation, the binomial $x^u - x^v$ belongs to I_A , by the observation $t^{Av} = t^{Au}$. By the property of ideals, f' is in I_A . Here is our contradiction, f' has a smaller leading term than f does. ♠

If you want, you can actually obtain your contradiction in several ways. Example: since you can write f' in the form (??), then you get a contradiction for f . Therefore, f' cannot be written in the form of (??).

Now, we are ready to prove the main theorem. Proving part (a) of the theorem is super easy. You guys are probably going to laugh.

Proof. [Proof of the theorem] To prove part (a), by Hilbert's Basis Theorem, there exist polynomials f_1, \dots, f_s in I_A such that $I_A = \langle f_1, \dots, f_s \rangle$. Recall here, that the coefficients are polynomials. But, by Lemma 23, each polynomial f_i is of the form

$$f_i = \sum_{Au=Av} c_{uv}(x^u - x^v).$$

Moreover, each sum above is finite. Take for each i the binomials $x^u - x^v$ in the above expressions. That's the proof of part (a). The proof of (b) is not that much harder. The proof requires Buchberger's Algorithm.

How do you compute a Gröbner basis? By part (a), we know that the toric ideal I_A can be written as

$$I_A = \langle x^u - x^v \rangle,$$

where the generation is over finitely-many pairs of vectors satisfying $Au = Av$ (or, that is, $A(u - v) = 0$). How did Buchberger's Alogrithm go? We needed to compute S-pairs. Let's remind ourselves about S-pairs. What happens when you take the S-pair of two binomials $x^u - x^v$ and $x^\alpha - x^\beta$?

Let's assume that both binomials here are written in order, so that the leading monomials are first. Recall $S_\succ(x^u - x^v, x^\alpha - x^\beta) = \frac{\text{lcm}(x^u, x^\alpha)}{x^u}(x^u - x^v) - \frac{\text{lcm}(x^u, x^\alpha)}{x^\alpha}(x^\alpha - x^\beta)$. But, we note that the inner terms here are

going to cancel. That is, we get $\frac{\text{lcm}(x^u, x^\alpha)}{x^u}(x^v) + \frac{\text{lcm}(x^u, x^\alpha)}{x^\alpha}(x^\beta)$. In particular, this expression is a binomial. (Note that this does not happen for general S-pairs. The punchline/moral of the story is that: S-pairs of binomials are themselves binomials.)

Now, when you do the Buchberger Algorithm, you need to divide the binomial by other binomials. When you divide binomials by binomials, you either get zero or another binomial. Therefore, when you apply the Buchberger Algorithm to the original set of binomials, you get a Gröbner basis that consists only of binomials. Therefore, there is a Gröbner basis that consists only of binomials. ♠

The moral of the story is that the vectors of $\ker(A) \cap \mathbb{Z}^n$ precisely correspond to binomials in the toric ideal I_A . Thus, the Gröbner basis G of I_A is enough to express *any* vector in $\ker(A) \cap \mathbb{Z}^n$ as an integer linear combination of the elements of G .

Example 23.9

Consider the set of all 2×3 matrices whose entries are all non-negative integers with the property that each row adds to 6 and each column adds to 4. An example of such a matrix is

$$\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}.$$

Another example is

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix}.$$

Finally, another is

$$\begin{bmatrix} 4 & 2 & 0 \\ 0 & 2 & 4 \end{bmatrix}.$$

How many of them are there? There are 19. If you want to know which matrix is “cheapest” with respect to a cost vector, we’re going to find out how to do that.

With toric ideals I_A , we will answer the following three fundamental questions for applications:

1. How many integer solutions are there for $\{x \mid Ax = b, x \in \mathbb{Z}^n, x \geq 0\}$?
2. Important for statistics: Can you find one such solution uniformly at random?
3. The question relevant to this class: Given a cost vector c , I wish to compute $x = \bar{x}$ in the set above that minimizes $c \cdot \bar{x}$.

Those questions depend on the previous theorem, but they also depend on the following definition and theorem.

Definition 23.10

Given a set of vectors $\mathcal{F} \subseteq \ker(A) \cap \mathbb{Z}^n$ and given the b -fiber $P_A(b) = \{\bar{x} \mid A\bar{x} = b, x \geq 0, x \in \mathbb{Z}^n\}$, I can define a graph $G_{\mathcal{F}}(P_A(b))$ as follows:

- The vertices of the graph are exactly the elements of the fiber $P_A(b)$.
- Two elements u, v in the b -fiber $P_A(b)$ are connected by an edge exactly when $u - v \in \mathcal{F}$ or $v - u \in \mathcal{F}$.

Let's look at an example of this graph. Let's do the example of the 2×3 matrices from before:

Example 23.11: T

The set \mathcal{F} should consist of matrices that are in the kernel of the 5×6 matrix A of 2×3 transportation polytopes.

Recall that we had feasible points

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 4 & 2 & 0 \\ 0 & 2 & 4 \end{bmatrix}.$$

Consider a vector such that the row and column sums are zero. Using a vector like this, we get from the first matrix above to the second matrix above. Here is an example:

$$\begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \end{bmatrix}$$

By adding this matrix to the the feasible solution on the left above, we obtain

$$\begin{bmatrix} 3 & 3 & 0 \\ 1 & 1 & 4 \end{bmatrix}.$$

There are more steps to get from the matrix on the left to the matrix on the right. The goal is to have a set \mathcal{F} of vectors that allows us to move between all feasible solutions via addition and subtraction.

For this problem, there are 3 basic moves. The moves are:

$$\pm \begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \end{bmatrix}, \pm \begin{bmatrix} -1 & 0 & 1 \\ 1 & 0 & -1 \end{bmatrix} \text{ and } \pm \begin{bmatrix} 0 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix}.$$

Let the positive versions of each of these be a, b , and c .

As mentioned, there are 19 tables, and here is a picture of the graph:

$$\begin{bmatrix} 0 & 4 & 2 \\ 4 & 0 & 2 \end{bmatrix} \quad \begin{bmatrix} 0 & 3 & 3 \\ 4 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 2 & 4 \\ 4 & 2 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 4 & 1 \\ 3 & 0 & 3 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 4 \\ 3 & 3 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 4 & 0 \\ 2 & 0 & 4 \end{bmatrix} \quad \begin{bmatrix} 2 & 0 & 4 \\ 2 & 4 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 3 & 0 \\ 1 & 1 & 4 \end{bmatrix} \xleftarrow[a]{\hspace{1cm}} \begin{bmatrix} 3 & 0 & 3 \\ 1 & 4 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 4 & 2 & 0 \\ 0 & 2 & 4 \end{bmatrix} \xrightarrow[c]{\hspace{1cm}} \begin{bmatrix} 4 & 1 & 1 \\ 0 & 3 & 3 \end{bmatrix} \xrightarrow{\hspace{1cm}} \begin{bmatrix} 4 & 0 & 2 \\ 0 & 4 & 2 \end{bmatrix}$$

24. Complexity

24.1 Main references

We will follow

1. Matthias Koeppe's Survey On the complexity of nonlinear mixed-integer optimization
2. Leo Liberti's Book - Chapter 5.
3. Leo Liberti's Paper Undecidability and hardness in mixed-integer nonlinear programming

24.2 Other references

1. Parameterized Integer Quadratic Programming: Variables and Coefficients
2. Integer Polynomial Optimization in Fixed Dimension
3. Information Complexity of Mixed-integer Convex Optimization

25. Recent topics

25.1 Exactness in SDP Relaxations

See Fatma's work

25.2 Quadratics and polytopes

fractional and other bivariate functions over general polytopes

25.2.1. Hidden Hyperplane Convexity

See Santanu's work

26. Decision Diagrams

Decision Diagrams for Optimization Authors: David Bergman , Andre A. Cire , Willem-Jan van Hoeve , John Hooker <https://link.springer.com/book/10.1007/978-3-319-42849-9>

Salemi H.* and Davarnia D. (2022) On the structure of decision diagram-representable mixed integer programs with application to unit commitment. Operations Research. In press. DOI: <https://doi.org/10.1287/opre.2022.2353>

Davarnia D. (2021) Strong relaxations for continuous nonlinear programs based on decision diagrams. Operations Research Letters. 49: 239-245. DOI: <https://doi.org/10.1016/j.orl.2021.01.011>

Davarnia D. and van-Hoeve W.-J. (2020) Outer approximation for integer nonlinear programs via decision diagrams. Mathematical Programming. 187: 111-150. DOI: <https://doi.org/10.1007/s10107-020-01475-4>

27. Sequential Quadratic Programming Methods

Sequential Quadratic Programming Methods Philip E. Gill & Elizabeth Wong

Leo Liberti - Chapter 9.1

28. Possible Projects

28.1 List

29. (Some) Topics Not covered

- (Spatial) Branch and Bound approaches
- Interior Point Methods (IPOPT and Knitro)
- Heuristics, Feasibility Pumps
- Algebraic Methods such as Cylindrical Algebraic Decomposition
- Submodular Minimization / Maximization

Index

Boolean Quadric Polytope, 83