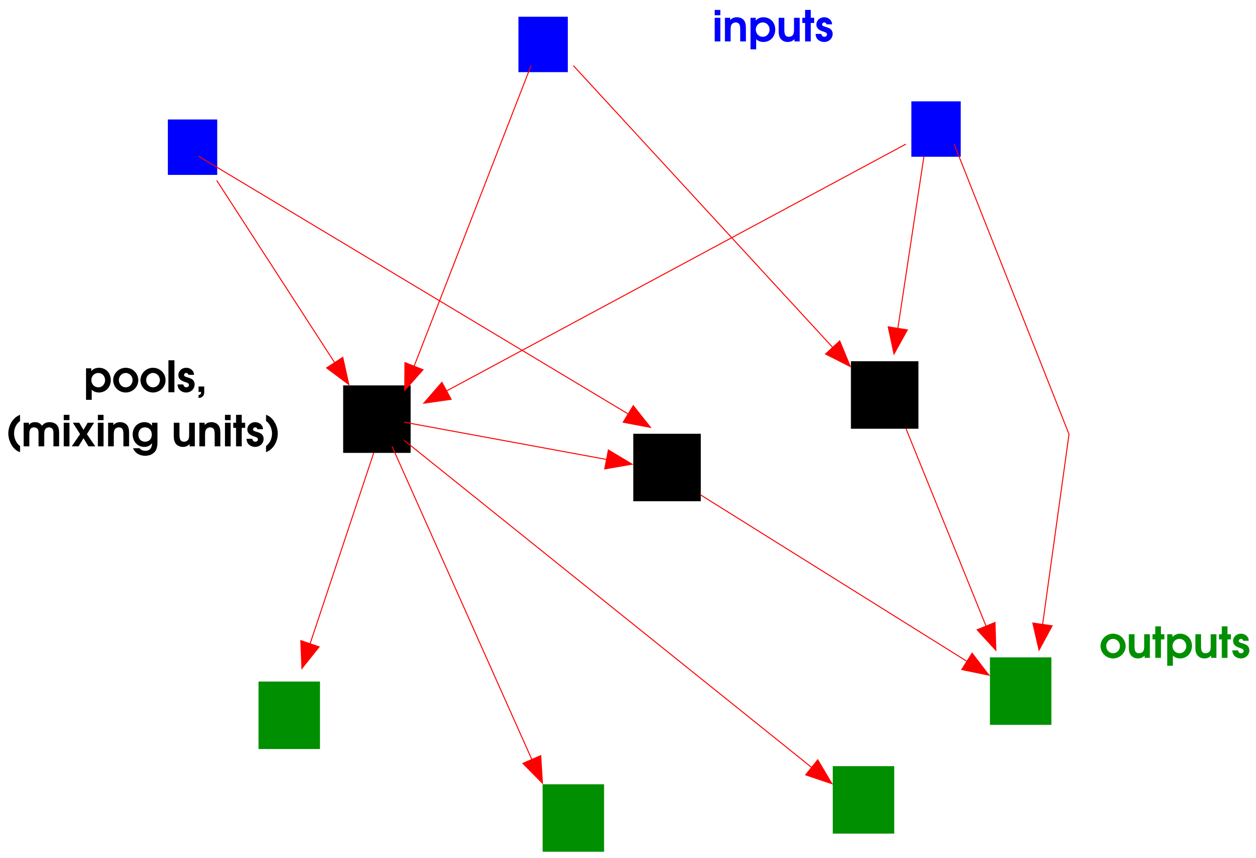
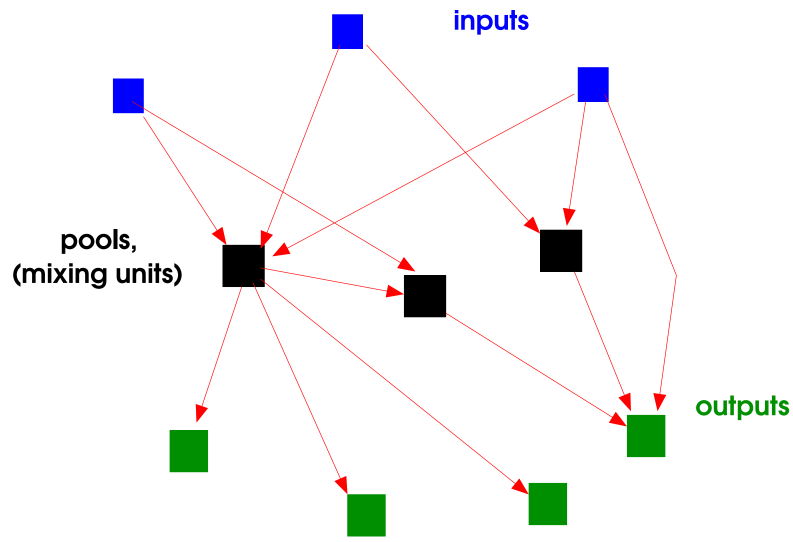
Talk 1: my review of nonlinear nonconvex optimization

Back to the pooling problem

We are given a directed, acyclic graph with three classes of vertices





1. We have $\boldsymbol{K}$ commodities ('specs') present at the inputs in different amounts.
2. Flows have to be routed to the outputs subject to flow conservation and capacity constraints.
3. Flows that reach a pool become **mixed**, and the **proportion** of each spec is upper- and lower-bounded.
4. Optimize a linear function of the flows.

Usual version: capacity constraints and costs are on total flows, not per-spec

Formulation

- $\mathfrak{I}$ = set of inputs, $\mathfrak{M}$ = set of pools,
- $\boldsymbol{\lambda_{ik}}$ = fraction of spec $\boldsymbol{k}$ at input $\boldsymbol{i}$ (data)

$$\min \quad \sum_{ij \in \mathcal{A}} c_{ij} y_{ij} \qquad \leftarrow y_{ij} = \text{total flow on } ij$$

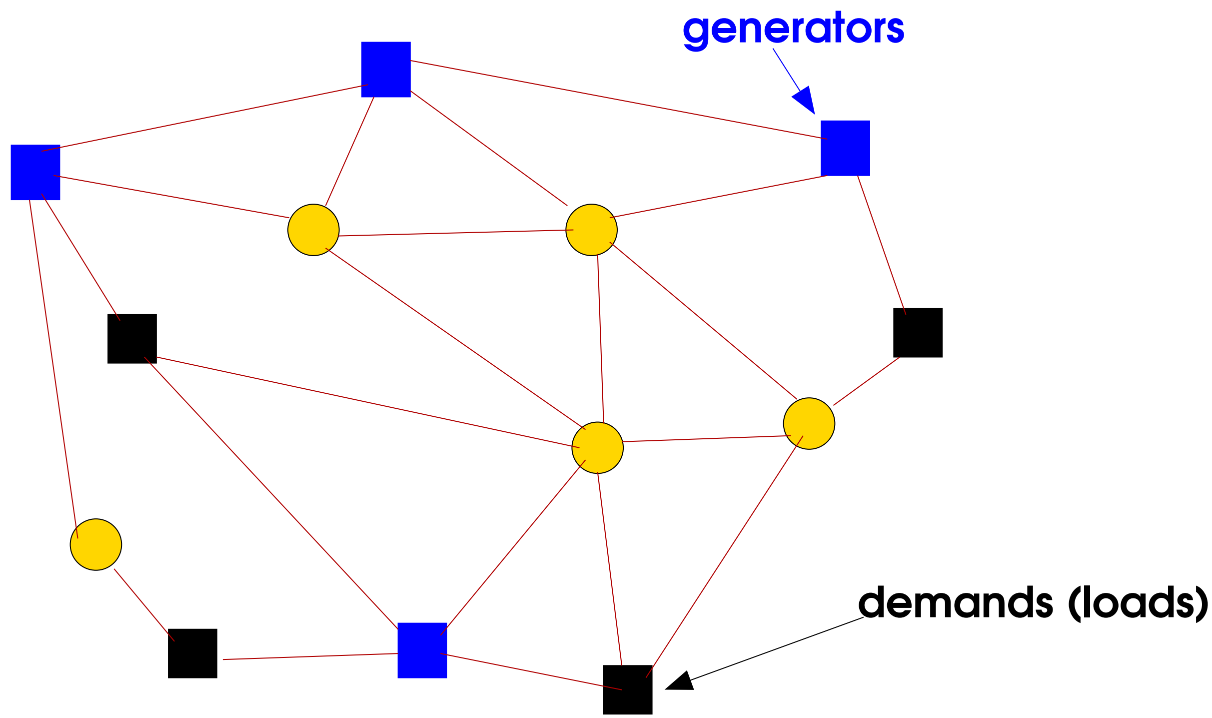
$$\text{s.t.} \quad \text{flow conservation,} \quad \text{capacity constraints on } y_{ij}$$

and for all spec $\boldsymbol{k}$, pool $\boldsymbol{j}$,

$$p_{jk} = \frac{\sum_{i \in \mathfrak{I}} \lambda_{ik} y_{ij} + \sum_{m \in \mathfrak{M}} p_{mk} y_{mj}}{\sum_{i \in \mathfrak{I} \cup \mathfrak{M}} y_{ij}} \qquad \leftarrow p_{jk} = \text{fraction of spec } k \text{ in pool } j$$

$$p_{jk}^{\min} \le p_{jk} \le p_{jk}^{\max}$$

Problem 2: AC-PF and -OPF problems on power grids



- Graph is undirected
- Each power line has a (complex) **admittance**
- Send power from generators to loads, subject to laws of physics and equipment constraints

Physics

- Each bus (node) $\boldsymbol{k}$ has a complex **voltage** $\boldsymbol{V_k}$.
  Voltage = potential energy
- Line (directed version of edge) $\boldsymbol{km}$ $\rightarrow$ complex **current** $\boldsymbol{I_{km}}$

$$I_{km} \;=\; y_{km}(V_k - V_m)$$

  (y = admittance)
- Line (directed version of edge) $\boldsymbol{km}$ $\rightarrow$ complex **power** $\boldsymbol{S_{km}}$

$$S_{km} \;=\; V_k I^*_{km} \;=\; y^*_{km} V_k (V_k - V_m)^*$$

  this is the *complex power injected into* $\boldsymbol{km}$ *at* $\boldsymbol{k}$
- Generators produce current at a certain voltage
- Demands (loads) expressed in units of complex power
- This is a time-averaged (steady-state) representation

Formulation

- Must choose voltage $\boldsymbol{V_k}$ at every bus $\boldsymbol{k}$

- Network constraints: total net power injected by each bus is constrained

$$S_k^{\min} \ \leq \ \sum_{km \in \delta(k)} y_{km}^* V_k (V_k - V_m)^* \ \leq \ S_k^{\max}$$

(two ranged inequalities)

1. At a generator, this says that total generated complex power is upper and lower bounded
2. At a load, $S_k^{\min} = S_k^{\max} = \ -$ (complex) demand

- Line constraints: e.g. $|y_{km}^* V_k (V_k - V_m)^*| \ \leq \ L_{km}$

- Voltage constraints: $U_k^{\min} \ \leq \ |V_k| \ \leq \ U_k^{\max}$

- $\boldsymbol{V_k}$= voltage bus $\boldsymbol{k}$
- Network constraints: total net power injected by each bus is constrained

$$S_k^{\min} \ \le \ S_k \ \doteq \sum_{km \in \delta(k)} y_{km}^* V_k (V_k - V_m)^* \ \le \ S_k^{\max}$$

- Line constraints: $|y_{km}^* V_k (V_k - V_m)^*| \ \le \ L_{km}$
- Voltage constraints: $U_k^{\min} \ \le \ |V_k| \ \le \ U_k^{\max}$

1. Feasibility version: **PF** or power flow problem

2. Optimization version, or **OPF**:

$$\min \sum_{g \in \mathcal{G}} c_g \left(\Re\mathrm{e}(S_g) \right)$$

($\mathcal{G}$ = set of generator nodes)

Each function $\boldsymbol{c_g}$ is convex quadratic. Want to minimize total cost of generation.

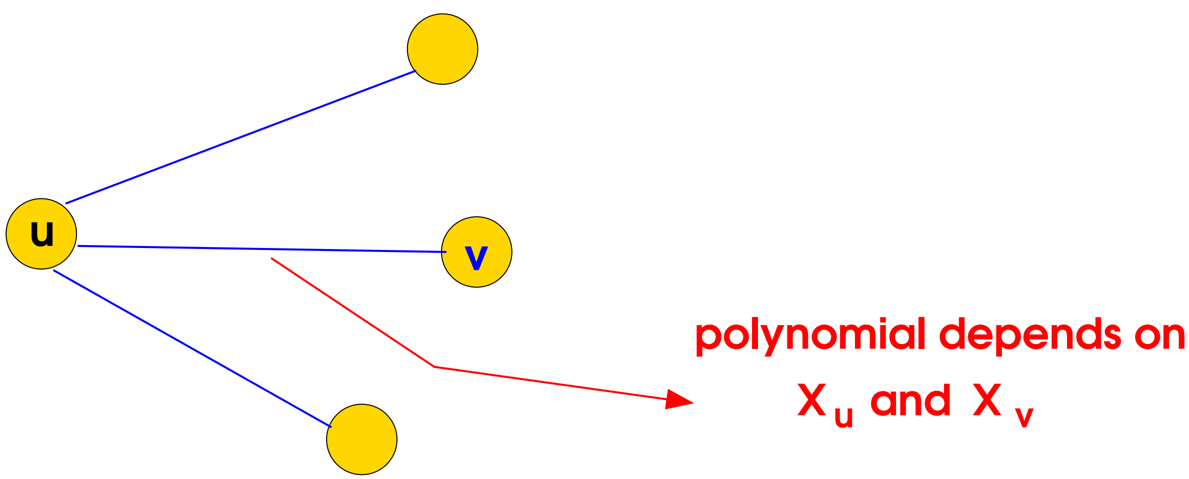
A generalization - network polynomial problems

Both the pooling problem and ACOPF are special cases of a general problem

- We are given an undirected graph $\mathcal{G}$
- For each node $u \in \mathcal{G}$ there is an associated set of variables, $\boldsymbol{X_u}$. Assume pairwise-disjoint.
- Likewise each constraint is associated with some node. A constraint associated with u takes the form:

$$\sum_{\{u,v\} \in \delta(u)} p_{u,v}(X_u \cup X_v) \ \geq \ 0$$

where each $\boldsymbol{p_{u,v}}$ is a polynomial function.



How to solve QCQPs?

How to solve QCQPs?

$\rightarrow$ **IPOPT?** (Wächter, Biegler, Laird)

How to solve QCQPs?

$\rightarrow$ **IPOPT?** (Wächter, Biegler, Laird)

$$\begin{array}{rl} \min & f(x) \\ \text{s.t.} & g(x) \; = \; 0 \\ & x \geq 0 \end{array}$$

How to solve QCQPs?

→ **IPOPT?** (Wächter, Biegler, Laird)

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \; = \; 0 \\ & x \geq 0 \end{aligned}$$

$$\rightarrow \quad \min \quad f(x) \; - \; \mu \sum_i \log(x_i) \tag{3a}$$

$$\text{s.t.} \quad g(x) \; = \; 0 \tag{3b}$$

Here $\boldsymbol{\mu > 0}$ is the barrier parameter, and we want $\boldsymbol{\mu \rightarrow 0}$.

How to solve QCQPs?

$\rightarrow$ **IPOPT?** (Wächter, Biegler, Laird)

$$\begin{array}{rl} \min & f(x) \\ \text{s.t.} & g(x) \;=\; 0 \\ & x \geq 0 \end{array}$$

$$\rightarrow \quad \min \quad f(x) \;-\; \mu \sum_i \log(x_i) \tag{4a}$$

$$\text{s.t.} \quad g(x) \;=\; 0 \tag{4b}$$

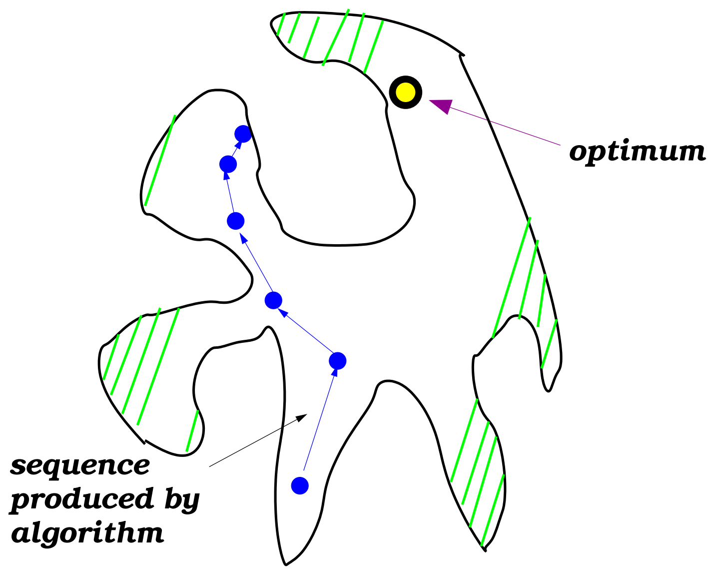
Here $\boldsymbol{\mu > 0}$ is the barrier parameter, and we want $\boldsymbol{\mu \rightarrow 0}$.

Algorithm

1. For given $\boldsymbol{\mu}$ approximately solve problem (4a), (4b).
2. Effectively, attempt to find a solution to the first-order optimality conditions for (4a), (4b): (damped) Newton method
3. Then decrease $\boldsymbol{\mu}$ and go to 1.
4. But a lot of cleverness employed in Step 3 (filter method).

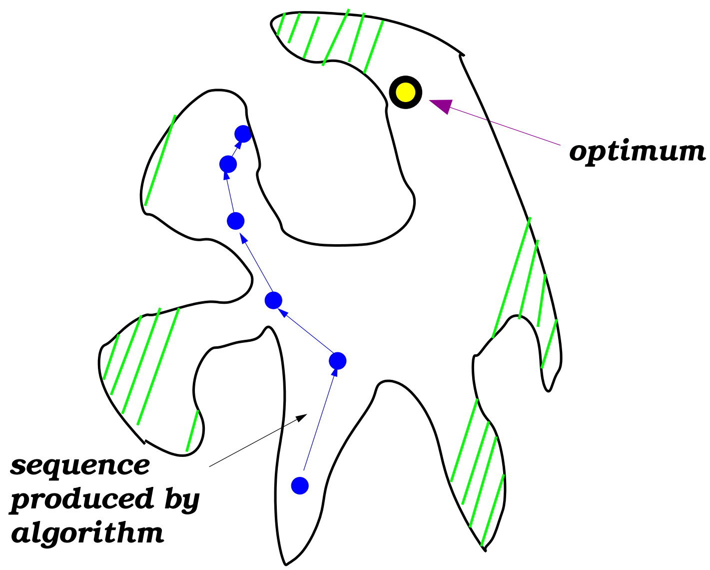
How to solve QCQPs?

→ **IPOPT?** (Wächter, Biegler, Laird)



How to solve QCQPs?

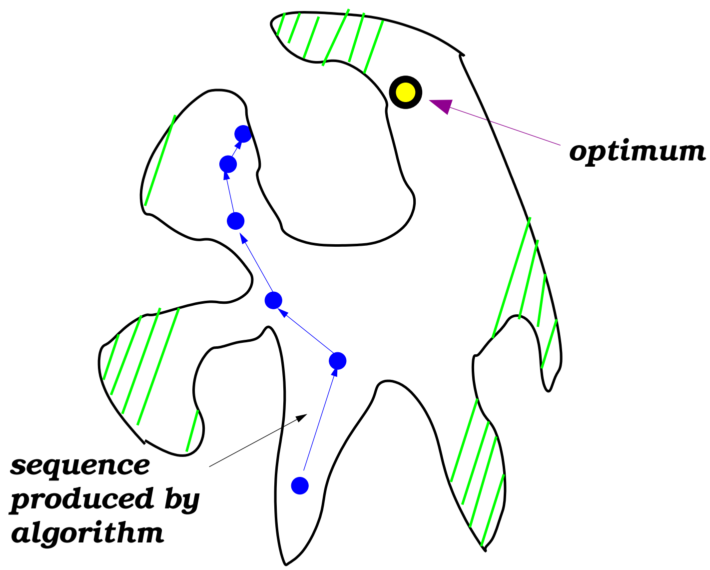
→ **IPOPT?** (Wächter, Biegler, Laird)



Claim: IPOPT globally solves all ACOPF instances

How to solve QCQPs?

→ **IPOPT?** (Wächter, Biegler, Laird)



Claim: IPOPT globally solves all ACOPF instances

What does this mean?

Three basic techniques

1. McCormick relaxation

2. Spatial branch-and-bound

3. RLT: lifting to higher-dimensional representation

McCormick relaxation: a very widely used technique

McCormick (1976), Al-Khayal and Falk (1983)
given:

$$\boldsymbol{x} \in [\boldsymbol{\ell^x}, \boldsymbol{u^x}], \quad \boldsymbol{y} \in [\boldsymbol{\ell^y}, \boldsymbol{u^y}], \quad \boldsymbol{z} = \boldsymbol{xy}$$

The convex hull of $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ in this set is given by

$$\begin{aligned} z &\geq \max\{\, u^y x + u^x y - u^y u^x \,,\, \ell^y x + \ell^x y - \ell^y \ell^x \,\} \\ z &\leq \min\{\, u^y x + \ell^x y - u^y \ell^x \,,\, \ell^y x + u^x y - \ell^y u^x \,\}. \end{aligned}$$

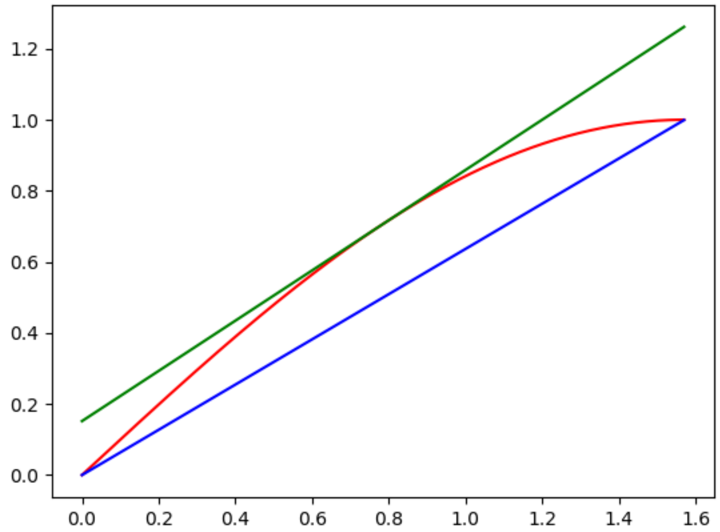
- Can be used directly to reformulate any polynomial optimization problem
- But some codes avoid this so as to not introduce the variables $\boldsymbol{w}$
- And the quality of the relaxation is in general poor
- Unless the bounds $\boldsymbol{\ell^x}, \boldsymbol{u^x}$ or $\boldsymbol{\ell^y}, \boldsymbol{u^y}$ are tight

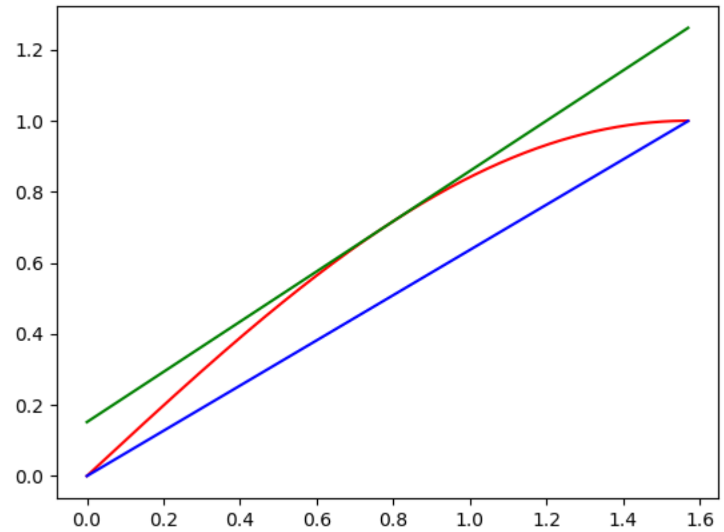
Spatial Branch-and-Bound: a very widely used technique

Tuy, 1998

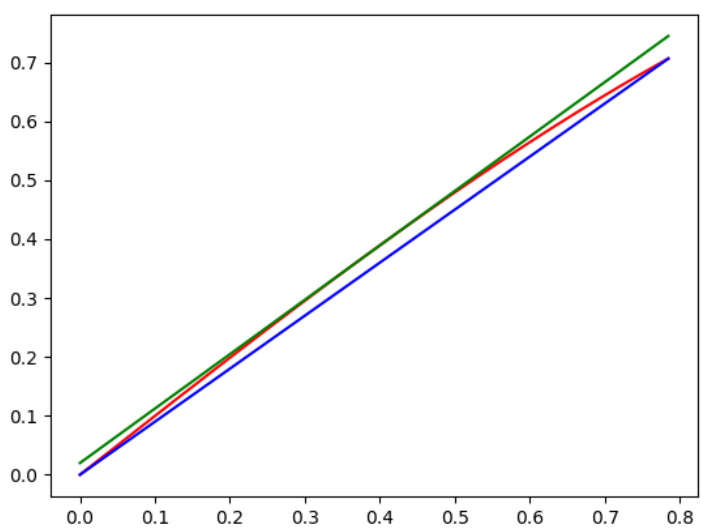
- Used in many codes, e.g. BARON
- Directly applicable to McCormick relaxations

Example: approximate $\sin(x)$ for $0 \leq x \leq \pi/2$

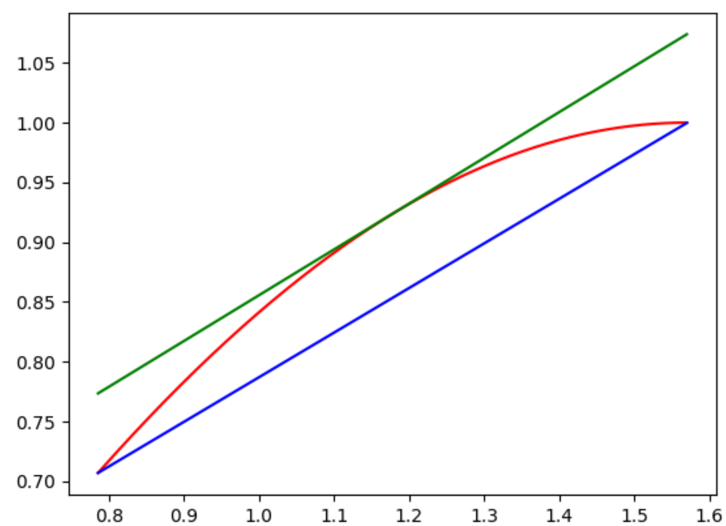




Branch at $\boldsymbol{x = \pi/4}$:



$\boldsymbol{0 \le x \le \pi/4}$



$\boldsymbol{\pi/4 \le x \le \pi/2}$

RLT: another very widely used technique

Sherali and Adams (1992)

Example:

Suppose $\mathbf{5x_1^2 + 2x_2 - 4 \geq 0}$ and $\mathbf{0 \leq x_3 \leq 10}$ are valid inequalities

Then:
$\mathbf{(5x_1^2 + 2x_2 - 4)x_3 \geq 0}$ and $\mathbf{(5x_1^2 + 2x_2 - 4)(10 - x_3) \geq 0}$ also valid

- Any nonlinear terms, e.g. $\mathbf{x_1^2 x_3}$ are *linearized* via McCormick
- It may be the case that the nonlinear terms are already found elsewhere
- General idea: multiplication of valid inequalities
- **Which** inequalities: using all is too expensive
- (Misener): *scan* possible products, keep if estimate of relaxation improves

Back to McCormick:

$$\mathbf{x \in [\ell^x, u^x]}, \quad \mathbf{y \in [\ell^y, u^y]}, \quad \mathbf{z = xy}$$

e.g. can do $\mathbf{(x - \ell^x)(u^y - y) \geq 0}$ or $\mathbf{u^y x + \ell^x y - \ell^x u^y \geq xy}$

Hierarchies

$$\textbf{(QCQP):} \quad \begin{array}{ll} \min & x^T Q x \,+\, 2c^T x \\ \text{s.t.} & x^T A_i x \,+\, 2b_i^T x \,+\, r_i \ \geq 0 \qquad i = 1, \ldots, m \\ & x \in \mathbb{R}^n. \end{array}$$

$\rightarrow$ form the semidefinite relaxation

$$\textbf{(SR):} \quad \begin{array}{ll} \min & \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X \\ \text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \ \geq 0 \qquad i = 1, \ldots, m \\ & X \succeq 0, \quad X_{00} = 1. \end{array}$$

Here, for symmetric matrices $\boldsymbol{M}$, $\boldsymbol{N}$,

$$M \bullet N \ = \ \sum_{h,k} M_{hk} N_{hk}$$

So if **SR** has a **rank-1 solution**, the lower bound is **exact**.

Unfortunately, **SR** **typically does not** have a rank-1 solution. **Why?**

- $\rightarrow$ Lavaei and Low (2010): on **ACOPF**, the semidefinite relaxation is often strong
- And it may even have a rank-1 solution.
- There remains the issue of solving the d***n SDP

Moment relaxations and polynomial optimization

Consider the polynomial optimization problem

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where each $f_i(x)$ is a **polynomial** i.e. $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^\pi$.

- Each π is a tuple $\pi_1, \pi_2, \ldots, \pi_n$ of **nonnegative integers**, and $x^\pi \doteq x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}$
- Each $S(i)$ is a finite set of **tuples**, and the $a_{i,\pi}$ are reals.

We know $f_0^* = \inf_\mu \mathbb{E}_\mu f_0(x)$, over all measures μ over $K \doteq \{x \in \mathbb{R}^n : f_i(x) \geq 0,\ 1 \leq i \leq m\}$.

$$\text{i.e.} \quad f_0^* = \inf\left\{ \sum_{\pi \in S(0)} a_{0,\pi} y_\pi \;:\; y \text{ is a } K\text{-moment} \right\}$$

Here, y is a K-moment if there is a measure μ over K with $y_\pi = \mathbb{E}_\mu x^\pi$ for each tuple π

Polynomial optimization

Consider the polynomial optimization problem

$$\boldsymbol{f_0^*} \quad \doteq \quad \min \{ \boldsymbol{f_0(x)} \ : \ \boldsymbol{f_i(x) \geq 0}, \quad \boldsymbol{1 \leq i \leq m}, \quad \boldsymbol{x} \in \mathbb{R}^n \},$$

where each $\boldsymbol{f_i(x)}$ is a **polynomial** i.e. $\boldsymbol{f_i(x)} = \sum_{\pi \in S(i)} \boldsymbol{a_{i,\pi}} \, \boldsymbol{x^\pi}$.

- Each $\boldsymbol{\pi}$ is a tuple $\boldsymbol{\pi_1, \pi_2, \ldots, \pi_n}$ of **nonnegative integers**, and $\boldsymbol{x^\pi} \doteq \boldsymbol{x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}}$
- Each $\boldsymbol{S(i)}$ is a finite set of **tuples**, and the $\boldsymbol{a_{i,\pi}}$ are reals.

We know $\boldsymbol{f_0^*} = \inf_\mu \mathbb{E}_\mu \boldsymbol{f_0(x)}$, over all measures $\boldsymbol{\mu}$ over $\boldsymbol{K} \doteq \{\boldsymbol{x} \in \mathbb{R}^n \ : \ \boldsymbol{f_i(x) \geq 0, \ 1 \leq i \leq m}\}$.

$$\text{i.e.} \quad \boldsymbol{f_0^*} \quad = \quad \inf \left\{ \sum_{\pi \in S(0)} \boldsymbol{a_{0,\pi} y_\pi} \ : \ \boldsymbol{y} \text{ is a } \boldsymbol{K}\text{-moment} \right\}$$

Here, $\boldsymbol{y}$ is a $\boldsymbol{K}$-moment if there is a measure $\boldsymbol{\mu}$ over $\boldsymbol{K}$ with $\boldsymbol{y_\pi} = \mathbb{E}_\mu \boldsymbol{x^\pi}$ for each tuple $\boldsymbol{\pi}$

(**Cough!** Here, $\boldsymbol{y}$ is an **infinite-dimensional** vector).

Polynomial optimization

Consider the polynomial optimization problem

$$f_0^* \doteq \min \{ f_0(x) \ : \ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n \},$$

where each $f_i(x)$ is a **polynomial** i.e. $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^{\pi}$.

- Each π is a tuple $\pi_1, \pi_2, \ldots, \pi_n$ of **nonnegative integers**, and $x^{\pi} \doteq x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}$
- Each $S(i)$ is a finite set of **tuples**, and the $a_{i,\pi}$ are reals.

We know $f_0^* = \inf_{\mu} \mathbb{E}_{\mu} f_0(x)$, over all measures μ over $K \doteq \{x \in \mathbb{R}^n \ : \ f_i(x) \geq 0, \ 1 \leq i \leq m\}$.

$$\text{i.e.} \quad f_0^* = \inf \left\{ \sum_{\pi \in S(0)} a_{0,\pi} y_{\pi} \ : \ y \text{ is a } K\text{-moment} \right\}$$

Here, y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π

(**Cough!** Here, y is an **infinite-dimensional** vector). Can we make an easier statement?

Polynomial optimization

$$f_0^* \doteq \min\{ f_0(x) \ : \ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n \},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^{\pi}$,

Thus $f_0^* = \inf_{\mu} \mathbb{E}_{\mu} \, f_0(x)$, over all measures μ over $K \doteq \{x \in \mathbb{R}^n \ : \ f_i(x) \geq 0, \ 1 \leq i \leq m\}$.

Polynomial optimization

$$f_0^* \doteq \min\{ f_0(x) \ : \ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

So $f_0^* = \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \ : \ f_i(x) \geq 0, \ 1 \leq i \leq m\})$.

Polynomial optimization

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) \;=\; \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

So $f_0^* \;=\; \inf_y \sum_\pi a_{0,\pi} y_\pi$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_\pi \;=\; \mathbb{E}_\mu x^\pi$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \;:\; f_i(x) \geq 0,\; 1 \leq i \leq m\})$.

So: $y_0 = 1$.

Polynomial optimization

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \ge 0, \quad 1 \le i \le m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) \;=\; \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

So $f_0^* \;=\; \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} \;=\; \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \;:\; f_i(x) \ge 0, \; 1 \le i \le m\})$.

So: $y_0 = 1$. Can we say more?

Polynomial optimization

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) \;=\; \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

So $f_0^* \;=\; \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} \;=\; \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \;:\; f_i(x) \geq 0,\; 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v \;=\; (x^{\pi})$ (all monomials).

Polynomial optimization

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

So $f_0^* = \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \;:\; f_i(x) \geq 0, \; 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^{\pi})$ (all monomials). Also define $M[y] \doteq E_{\mu} v v^T$.

Polynomial optimization

$$f_0^* \doteq \min\{\, f_0(x) \ : \ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^{\pi}$.

So $f_0^* = \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \ : \ f_i(x) \geq 0, \ 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^{\pi})$ (all monomials). Also define $M[y] \doteq E_{\mu} v v^T$.

So for any tuples π, ρ, $M[y]_{\pi,\rho} = \mathbb{E}_{\nu} \, x^{\pi} x^{\rho} = E_{\nu} x^{\pi+\rho} = y_{\pi+\rho}$

Polynomial optimization

$$f_0^* \doteq \min\{\ f_0(x)\ :\ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

So $f_0^* = \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n\ :\ f_i(x) \geq 0,\ 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^{\pi})$ (all monomials). Also define $M[y] \doteq E_{\mu} v v^T$.

So for any tuples π, ρ, $M[y]_{\pi,\rho} = \mathbb{E}_{\nu}\, x^{\pi} x^{\rho} = E_{\nu} x^{\pi+\rho} = y_{\pi+\rho}$

So for any (∞-dimensional) vector z, **indexed by tuples**, i.e. with entries z_{π} for each tuple π,

Polynomial optimization

$$f_0^* \doteq \min\{ f_0(x) \ : \ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^{\pi}$.

So $f_0^* = \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \ : \ f_i(x) \geq 0, \ 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^{\pi})$ (all monomials). Also define $M[y] \doteq E_{\mu} v v^T$.

So for any tuples π, ρ, $M[y]_{\pi,\rho} = \mathbb{E}_{\nu} \, x^{\pi} x^{\rho} = E_{\nu} x^{\pi+\rho} = y_{\pi+\rho}$

So for any (∞-dimensional) vector z, **indexed by tuples**, i.e. with entries z_{π} for each tuple π,

$$z^T M[y] z = \sum_{\pi,\rho} \mathbb{E}_{\mu} \, z_{\pi} x^{\pi} x^{\rho} z_{\rho} = \mathbb{E}_{\mu} \left(\sum_{\pi} z_{\pi} x^{\pi} \right)^2 \geq 0$$

Polynomial optimization

$$f_0^* \doteq \min\{ f_0(x) : f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^\pi$.

So $f_0^* = \inf_y \sum_\pi a_{0,\pi} y_\pi$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_\pi = \mathbb{E}_\mu x^\pi$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n : f_i(x) \geq 0, \; 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^\pi)$ (all monomials). Also define $M[y] \doteq E_\mu v v^T$.

So for any tuples π, ρ, $M[y]_{\pi,\rho} = \mathbb{E}_\nu\, x^\pi x^\rho = E_\nu x^{\pi+\rho} = y_{\pi+\rho}$

So for any (∞-dimensional) vector z, **indexed by tuples**, i.e. with entries z_π for each tuple π,

$$z^T M[y] z = \sum_{\pi,\rho} \mathbb{E}_\mu\, z_\pi x^\pi x^\rho z_\rho = \mathbb{E}_\mu \left(\sum_\pi z_\pi x^\pi\right)^2 \geq 0$$

so $M[y] \succeq 0$!!

Polynomial optimization

$$f_0^* \doteq \min\{\, f_0(x) \,:\, f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^\pi$.

So $f_0^* = \inf_y \sum_\pi a_{0,\pi} y_\pi$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_\pi = \mathbb{E}_\mu x^\pi$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n \,:\, f_i(x) \geq 0,\ 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^\pi)$ (all monomials). Also define $M[y] \doteq E_\mu v v^T$.

So for any tuples π, ρ, $M[y]_{\pi,\rho} = \mathbb{E}_\nu\, x^\pi x^\rho = E_\nu x^{\pi+\rho} = y_{\pi+\rho}$

So for any (∞-dimensional) vector z, **indexed by tuples**, i.e. with entries z_π for each tuple π,

$$z^T M[y] z = \sum_{\pi,\rho} \mathbb{E}_\mu\, z_\pi x^\pi x^\rho z_\rho = \mathbb{E}_\mu \left(\sum_\pi z_\pi x^\pi\right)^2 \geq 0$$

$$\textbf{so } M[y] \succeq 0 \text{ !!}$$

so

$$
\begin{aligned}
f_0^* \;\geq\; & \min \sum_\pi a_{0,\pi}\, y_\pi \\
\text{s.t.} \quad & y_0 = 1, \\
& M \succeq 0, \\
& M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\
& \text{the zeroth row and column of } M \text{ both equal } y. \text{ (redundant)}
\end{aligned}
$$

Polynomial optimization

$$f_0^* \doteq \min\{ f_0(x) : f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} x^{\pi}$.

So $f_0^* = \inf_y \sum_{\pi} a_{0,\pi} y_{\pi}$, over all K-moment vectors y;

(y is a K-moment if there is a measure μ over K with $y_{\pi} = \mathbb{E}_{\mu} x^{\pi}$ for each tuple π)

$(K \doteq \{x \in \mathbb{R}^n : f_i(x) \geq 0, \ 1 \leq i \leq m\})$.

So: $y_0 = 1$. Can we say more? Define $v = (x^{\pi})$ (all monomials). Also define $M[y] \doteq E_{\mu} v v^T$.

So for any tuples π, ρ, $M[y]_{\pi,\rho} = \mathbb{E}_{\nu}\, x^{\pi} x^{\rho} = E_{\nu} x^{\pi+\rho} = y_{\pi+\rho}$

So for any (∞-dimensional) vector z, **indexed by tuples**, i.e. with entries z_{π} for each tuple π,

$$z^T M[y] z = \sum_{\pi,\rho} \mathbb{E}_{\mu}\, z_{\pi} x^{\pi} x^{\rho} z_{\rho} = \mathbb{E}_{\mu} \left(\sum_{\pi} z_{\pi} x^{\pi}\right)^2 \geq 0$$

$$\textbf{so } M[y] \succeq 0 \text{ !!}$$

so

$$\begin{aligned} f_0^* \ \geq \ & \min \sum_{\pi} a_{0,\pi}\, y_{\pi} \\ \text{s.t.} \quad & y_0 = 1, \\ & M \succeq 0, \\ & M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

An **infinite-dimensional** semidefinite program!!

$$\boldsymbol{f_0^*} \quad \doteq \quad \mathbf{min}\,\{\, \boldsymbol{f_0(x)} \;:\; \boldsymbol{f_i(x) \geq 0}, \quad \mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{m}, \quad \boldsymbol{x} \in \mathbb{R}^n \},$$

where $\boldsymbol{f_i(x)} = \sum_{\pi \in S(i)} \boldsymbol{a_{i,\pi}\, x^{\pi}}$.

$$\begin{aligned}
\boldsymbol{f_0^*} \quad &\geq \quad \min \sum_{\pi} a_{0,\pi}\, y_{\pi} \\
\text{s.t.} \quad & \quad y_0 = 1, \\
& \quad M \succeq 0, \\
& \quad M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\
& \quad \text{the zeroth row and column of } M \text{ both equal } y.
\end{aligned}$$

$$f_0^* \doteq \min \{ f_0(x) \; : \; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n \},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^{\pi}$.

$$\begin{aligned} f_0^* \;\geq\; & \min \sum_{\pi} a_{0,\pi} \, y_\pi \\ \text{s.t.} \quad & y_0 = 1, \\ & M \succeq 0, \\ & M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

Restrict: pick an integer $\boldsymbol{d \geq 1}$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where $f_i(x) \;=\; \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}$.

$$\begin{array}{lll} f_0^* & \geq & \min \sum_{\pi} a_{0,\pi}\, y_{\pi} \\ \text{s.t.} & & y_0 \;=\; 1, \\ & & M \;\succeq\; 0, \\ & & M_{\pi,\rho} \;=\; y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & & \text{the zeroth row and column of } M \text{ both equal } y. \end{array}$$

Restrict: pick an integer $d \geq 1$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

Example: $d = 8$. So we will consider the monomial $x_1^2\, x_2^4\, x_3$ because $2 + 4 + 1 \leq 8$.

But we will not consider $x_3 x_5^7 x_8$, because $1 + 7 + 1 > 8$.

$$\boldsymbol{f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\}},$$

where $\boldsymbol{f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^\pi}$.

$$\begin{aligned} \boldsymbol{f_0^*} \;\geq\; & \min \sum_\pi a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 \;=\; 1, \\ & M \;\succeq\; 0, \\ & M_{\pi,\rho} \;=\; y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

Restrict: pick an integer $\boldsymbol{d \geq 1}$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

$$\begin{aligned} \boldsymbol{f_0^*} \;\geq\; & \min \sum_\pi a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 \;=\; 1, \\ & \text{the rows and columns of } M\text{, and the entries in } y\text{, indexed by tuples of size} \leq d \\ & M \;\succeq\; 0, \\ & M_{\pi,\rho} \;=\; y_{\pi+\rho}, \quad \text{for all appropriate tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y, \end{aligned}$$

$$f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n \},$$

where $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^\pi$.

$$\begin{aligned} f_0^* \;\geq\; & \min \sum_\pi a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 = 1, \\ & M \succeq 0, \\ & M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

Restrict: pick an integer $d \geq 1$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

$$\begin{aligned} f_0^* \;\geq\; & \min \sum_\pi a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 = 1, \\ & \text{the rows and columns of } M\text{, and the entries in } y\text{, indexed by tuples of size} \leq d \\ & M \succeq 0, \\ & M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all appropriate tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y \end{aligned}$$

A **finite-dimensional** semidefinite program!!

$$\boldsymbol{f_0^* \doteq \min\{\, f_0(x) \;:\; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\}},$$

where $\boldsymbol{f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi}\, x^{\pi}}$.

$$\begin{aligned} \boldsymbol{f_0^*} \;\geq\; & \min \sum_{\pi} a_{0,\pi}\, y_{\pi} \\ \text{s.t.} \quad & y_0 = 1, \\ & M \succeq 0, \\ & M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

Restrict: pick an integer $\boldsymbol{d \geq 1}$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

$$\begin{aligned} \boldsymbol{f_0^*} \;\geq\; & \min \sum_{\pi} a_{0,\pi}\, y_{\pi} \\ \text{s.t.} \quad & y_0 = 1, \\ & \text{the rows and columns of } M\text{, and the entries in } y\text{, indexed by tuples of size} \leq d \\ & M \succeq 0, \\ & M_{\pi,\rho} = y_{\pi+\rho}, \quad \text{for all appropriate tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y \end{aligned}$$

A **finite-dimensional** semidefinite program!! But could be very large!!

$$\boldsymbol{f_0^*} \quad \doteq \quad \boldsymbol{\min\{\ f_0(x)\ :\ f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\}},$$

where $\boldsymbol{f_i(x)} \ = \ \sum_{\pi \in S(i)} \boldsymbol{a_{i,\pi}\, x^\pi}$.

$$\begin{aligned} \boldsymbol{f_0^*} \ \geq \ & \min \sum_\pi a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 \ = \ 1, \\ & M \ \succeq \ 0, \\ & M_{\pi,\rho} \ = \ y_{\pi+\rho}, \quad \text{for all tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

Restrict: pick an integer $\boldsymbol{d \geq 1}$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

$$\begin{aligned} \boldsymbol{f_0^*} \ \geq \ & \min \sum_\pi a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 \ = \ 1, \\ & \text{the rows and columns of } M, \text{ and the entries in } y, \text{ indexed by tuples of size} \leq d \\ & M \ \succeq \ 0, \\ & M_{\pi,\rho} \ = \ y_{\pi+\rho}, \quad \text{for all appropriate tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y \end{aligned}$$

A **finite-dimensional** semidefinite program!! But could be very large!!

- Can be strengthened to account for the constraints $\boldsymbol{f_i(x) \geq 0}$.

$$\boldsymbol{f_0^* \doteq \min\{\, f_0(x) \,:\, f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\}},$$

where $\boldsymbol{f_i(x)} = \sum_{\pi \in S(i)} \boldsymbol{a_{i,\pi}\, x^{\pi}}$.

$$\begin{aligned} \boldsymbol{f_0^*} \;\geq\; & \min \sum_{\pi} a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 \;=\; 1, \\ & M \;\succeq\; 0, \\ & M_{\pi,\rho} \;=\; y_{\pi+\rho}, \\ & \text{the zeroth row and column of } M \text{ both equal } y. \end{aligned}$$

Restrict: pick an integer $\boldsymbol{d \geq 1}$. Restrict the SDP to all tuples π with $|\pi| \leq d$.

$$\begin{aligned} \boldsymbol{f_0^*} \;\geq\; & \min \sum_{\pi} a_{0,\pi}\, y_\pi \\ \text{s.t.} \quad & y_0 \;=\; 1, \\ & \text{the rows and columns of } M\text{, and the entries in } y\text{, indexed by tuples of size} \leq d \\ & M \;\succeq\; 0, \\ & M_{\pi,\rho} \;=\; y_{\pi+\rho}, \quad \text{for all appropriate tuples } \pi, \rho \\ & \text{the zeroth row and column of } M \text{ both equal } y \end{aligned}$$

A **finite-dimensional** semidefinite program!! But could be very large!!

- Can be strengthened to account for the constraints $\boldsymbol{f_i(x) \geq 0}$. **How? e.g. use RLT**
- This is the level- $\boldsymbol{d}$ Lasserre relaxation (abridged).

Solving SDP relaxations of QCQPs

$$\textbf{(QCQP):} \quad \begin{array}{ll} \min & x^T Q x + 2c^T x \\ \text{s.t.} & x^T A_i x + 2b_i^T x + r_i \geq 0 \qquad i = 1, \ldots, m \\ & x \in \mathbb{R}^n. \end{array} \tag{6}$$

$$\textbf{(SR):} \quad \begin{array}{ll} \min & \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X \\ \text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \geq 0 \qquad i = 1, \ldots, m \\ & X \succeq 0, \quad X_{00} = 1. \end{array} \tag{7}$$

Solving SDP relaxations of QCQPs

$$\textbf{(QCQP):}\quad \begin{array}{ll} & \min \ x^T Q x + 2c^T x \\ \text{s.t.} & x^T A_i x + 2b_i^T x + r_i \geq 0 \qquad i = 1, \ldots, m \\ & x \in \mathbb{R}^n. \end{array} \tag{8}$$

$$\textbf{(SR):}\quad \begin{array}{ll} \min & \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X \\ \text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \geq 0 \qquad i = 1, \ldots, m \\ & X \succeq 0, \quad X_{00} = 1. \end{array} \tag{9}$$

Matrix completion theorem.

- Form a graph, $\mathcal{G}$ with vertex set $\mathbf{0, 1, \ldots, \boldsymbol{n}}$
- Include an edge $\{\boldsymbol{i}, \boldsymbol{j}\}$ if the $(\boldsymbol{i}, \boldsymbol{j})$ entry of some constraint (9) (or objective) is nonzero
- Suppose there is a **chordal supergraph** $\mathcal{H}$ of $\mathcal{G}$ such that:

  $\mathcal{H}$ is the union of $\boldsymbol{k}$ maximal cliques $\boldsymbol{Q_1, \ldots, Q_k}$
- Then $\boldsymbol{X} \succeq \mathbf{0}$ is equivalent to:

$$\boldsymbol{X}|_{\boldsymbol{Q_1}} \succeq \mathbf{0}, \ldots, \boldsymbol{X}|_{\boldsymbol{Q_k}} \succeq \mathbf{0}$$

($\boldsymbol{X}|_{\boldsymbol{Q_j}}$: submatrix of $\boldsymbol{X}$ indexed by vertices of $\boldsymbol{Q_j}$).

- $\rightarrow$ If the submatrices are small this approach can be effective
- Current SDP-based methods for ACOPF rely on this paradigm

Can we do anything else involving SDP?

Chen, Atamtürk and Oren (2016):
For $n > 1$ a nonzero $n \times n$ Hermitian psd matrix has rank one iff all of its 2×2 principal minors are zero.

$\longrightarrow$ use this criterion to drive *branching*:

- Minimum eigenvalue of any 2×2 principal submatrix should be zero
- Choose submatrix with largest deviation from this constraint
- Can then (spatially) branch on any of the three values

Can we do anything else involving SDP?

Chen, Atamtürk and Oren (2016):
For $n > 1$ a nonzero $n \times n$ Hermitian psd matrix has rank one iff all of its 2×2 principal minors are zero.

$\rightarrow$ use this criterion to drive *branching*:

- Minimum eigenvalue of any 2×2 principal submatrix should be zero
- Choose submatrix with largest deviation from this constraint
- Can then (spatially) branch on any of the three values

Kocuk, Dey, Sun (2017):
For $n > 1$ a nonzero $n \times n$ Hermitian matrix is psd of rank one iff its diagonal is nonnegative and all the 2×2 minors are zero.

- Also, any $k \times k$ principal submatrix should be psd ($k \geq 2$)
- Use $k = 3$ or $k = 4$ and *cycles*
- Use SDP duality (whiteboard) to generate cuts
- **Let's think about it.** Why cycles? $\rightarrow$ use chordal extensions

Digitization and Discretization

Glover, (1975)

Given an **integer** variable $\mathbf{0 \le x \le u}$ (integral), we can reformulate

$$x \;=\; \sum_{i=1}^{k} 2^i y_i, \quad \text{where each } y_i \text{ is binary, and } \boldsymbol{k = \log_2 u}, \text{ or}$$

$$x \;=\; \sum_{i=1}^{u} z_i, \quad \text{where each } z_i \text{ is binary, or}$$

$$x \;=\; \sum_{i=1}^{u} i\, w_i, \quad \sum_{i} w_i \le 1, \quad \text{where each } w_i \text{ is binary}$$

Digitization and Discretization

Glover, (1975)

Given an **integer** variable $\mathbf{0 \le x \le u}$ (integral), we can reformulate

$$x = \sum_{i=1}^{k} 2^i y_i, \quad \text{where each } y_i \text{ is binary, and } \boldsymbol{k = \log_2 u}\text{, or}$$

$$x = \sum_{i=1}^{u} z_i, \quad \text{where each } z_i \text{ is binary, or}$$

$$x = \sum_{i=1}^{u} i\, w_i, \quad \sum_i w_i \le 1, \quad \text{where each } w_i \text{ is binary}$$

And if we have a bilinear expression $\boldsymbol{x f}$ $(0 \le f \ge F)$ then we get an exact linear representation for e.g. each $\boldsymbol{w_i f}$ through RLT

$$0 \le P_i \le F w_i$$

$$f - F(1 - w_i) \le P_i \le f$$

Digitization and Discretization

B., (2006), Dash, Günlük, Lodi (2007):
Discretization to approximate a bilinear form on **continuous** variables:

Consider a bilinear expression $\boldsymbol{xy}$ where $\mathbf{0} \leq \boldsymbol{x} \leq \boldsymbol{u^x}$, $\mathbf{0} \leq \boldsymbol{y} \leq \boldsymbol{u^y}$.

Then we write:

$$\boldsymbol{x} = u^x \left(\sum_{j=1}^{L} 2^{-j} \, \boldsymbol{z_j} + \boldsymbol{\delta} \right),$$

$$\text{each } \boldsymbol{z_j} \text{ binary}, \quad 0 \leq \boldsymbol{\delta} \leq 2^{-L}$$

And so we can represent

$$\boldsymbol{xy} = u^x \left(\sum_{j=1}^{L} 2^{-j} \, \boldsymbol{w_j} + \boldsymbol{\gamma} \right)$$

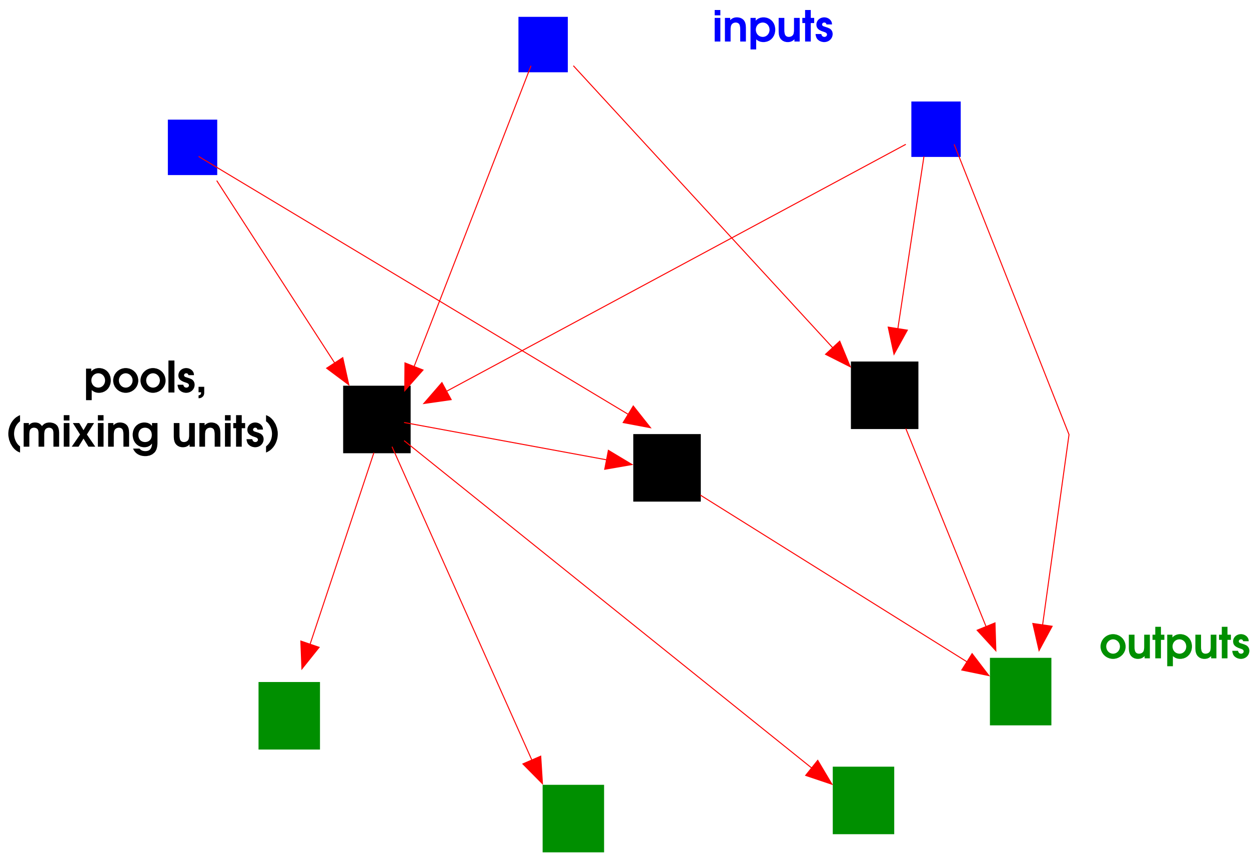
$$0 \leq \boldsymbol{\gamma} \leq \min\{2^{-L} \boldsymbol{y}, \; \boldsymbol{\delta} u^y\} \quad \text{(RLT)}$$

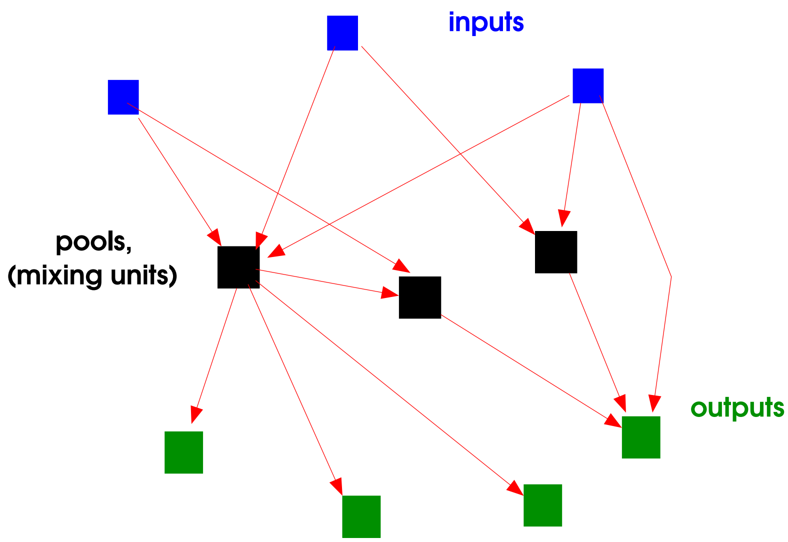
$$\text{each } \boldsymbol{w_j}\text{: RLT of } \boldsymbol{z_j y}$$

→ A valid relaxation. **We will come back to this later.**

Back to the pooling problem

We are given a directed, acyclic graph with three classes of vertices





1. We have $\boldsymbol{K}$ commodities ('specs') present at the inputs in different amounts.
2. Flows have to be routed to the outputs subject to flow conservation and capacity constraints.
3. Flows that reach a pool become **mixed**, and the **proportion** of each spec is upper- and lower-bounded.
4. Optimize a linear function of the flows.

Usual version: capacity constraints and costs are on total flows, not per-spec

Formulation

- $\mathfrak{I}$ = set of inputs, $\mathfrak{M}$ = set of pools,
- $\boldsymbol{\lambda_{ik}}$ = fraction of spec $\boldsymbol{k}$ at input $\boldsymbol{i}$ (data)

$$\min \quad \sum_{ij \in \mathcal{A}} c_{ij}\, \boldsymbol{y_{ij}} \qquad \leftarrow \quad \boldsymbol{y_{ij}} = \text{total flow on } ij$$

s.t. flow conservation, capacity constraints on y_{ij}

and for all spec $\boldsymbol{k}$, pool $\boldsymbol{j}$,

$$\boldsymbol{p_{jk}} = \frac{\sum_{i \in \mathfrak{I}} \lambda_{ik}\, \boldsymbol{y_{ij}} + \sum_{m \in \mathfrak{M}} \boldsymbol{p_{mk} y_{mj}}}{\sum_{i \in \mathfrak{I} \cup \mathfrak{M}} \boldsymbol{y_{ij}}} \leftarrow p_{jk} = \text{fraction of spec } k \text{ in pool } j$$

$$p_{jk}^{\min} \leq \boldsymbol{p_{jk}} \leq p_{jk}^{\max}$$

Digitization and Discretization in the Pooling Problem

Ahmed, Dey, Gupte, Jeon (2015, 2017)
Consider a bilinear expression $\boldsymbol{xy}$ where $\mathbf{0} \le \boldsymbol{x} \le \boldsymbol{u^x}$, $\mathbf{0} \le \boldsymbol{y} \le \boldsymbol{u^y}$.

Then we **approximate**

$$\boldsymbol{x} \;=\; u^x \sum_{j=1}^{L} 2^{-j}\, \boldsymbol{z_j},$$

$$\text{each } \boldsymbol{z_j} \text{ binary}, \quad 0 \le \boldsymbol{\delta} \le 2^{-L}$$

And so one can **approximate**

$$\boldsymbol{xy} \;=\; u^x \sum_{j=1}^{L} 2^{-j}\, \boldsymbol{w_j}$$

$$\text{each } \boldsymbol{w_j}\text{: RLT of } \boldsymbol{z_j y}$$

- An **approximation**, not a **relaxation**
- In some cases, the **best** upper bounds for larger pooling problems are obtained this way

“Take-away” and next talk

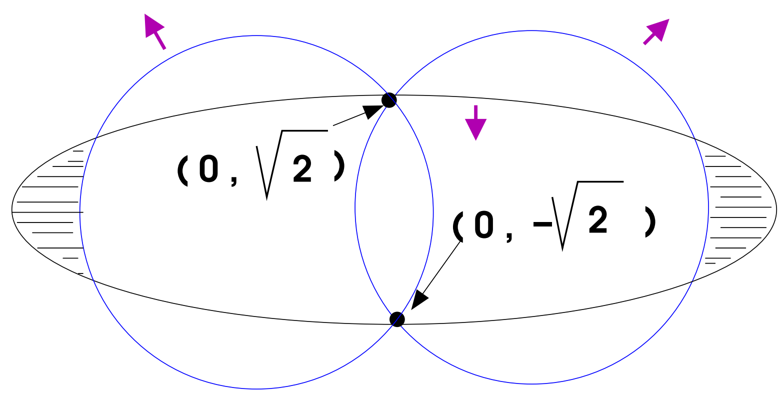
- We want strong relaxations, but the relaxations can be hard to solve
- A challenge: come up with strong branching, cutting and reformulation mechanisms that are robust across problem classes
- And how about accuracy and numerical stability?
- Local search for nonconvex nonlinear optimization?

Crimes against computers

$$
\begin{aligned}
\max \quad & x_2 - 20s_5 - 20s_6 + 2s_7 + s_5^2 \\
\text{s.t.} \quad & (x_1 - 1)^2 + x_2^2 \geq 3 + \frac{\phi}{10} && \text{(11a)} \\
& (x_1 + 1)^2 + x_2^2 \geq 3 && \text{(11b)} \\
& \frac{1}{10}x_1^2 + x_2^2 \leq 2 && \text{(11c)} \\
& 10\,\delta + 10\,\phi^2 \geq 1 && \text{(11d)} \\
& -10\,a + \delta + 10\,\phi^2 \leq 0 \\
& -10\,b + a + 10\,\phi^2 \leq 0 \\
& -10\,c + b + 10\,\phi^2 \leq 0 \\
& -10\,d + c + 10\,\phi^2 \leq 0 \\
& -10\,e + d + 10\,\phi^2 + 10\,s_5^2 = 0 && \text{(11e)} \\
& -10\,f + e + 10\,\phi^2 + 10\,s_6^2 = 0 \\
& -10\,g + f + 10\,\phi^2 + 10\,s_7^2 = 0 \\
& -10\,\phi + g + 10\,\phi^2 \leq 0 && \text{(11f)}
\end{aligned}
$$

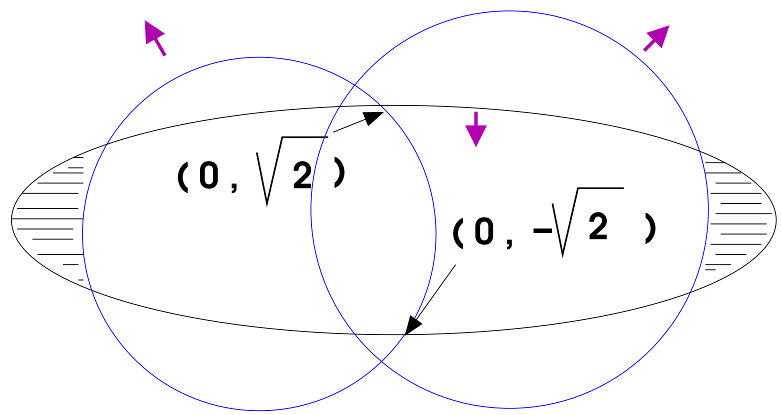
What's going on?

$$\begin{aligned} \max \quad & x_2 \\ \text{s.t.} \quad (x_1-1)^2 \; + \; & x_2^2 \; \geq \; 3 \\ (x_1+1)^2 \; + \; & x_2^2 \; \geq \; 3 \\ \frac{x_1^2}{10} + & x_2^2 \; \leq \; 2 \end{aligned}$$



What's going on?

$$\begin{array}{rrcl} & \max & x_2 & \\ \text{s.t.} & (x_1-1)^2 + x_2^2 & \geq & 3 + \phi \qquad (\phi > 0) \\ & (x_1+1)^2 + x_2^2 & \geq & 3 \\ & \dfrac{x_1^2}{10} + x_2^2 & \leq & 2 \end{array}$$



S-free Sets for Polynomial Optimization and Oracle-Based Cuts

B., Chen Chen and Gonzalo Muñoz, 2017

Consider:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & x \in S \cap P. \end{aligned}$$

$\boldsymbol{P := \{x \in \mathbb{R}^n | Ax \leq b\}}$ is a polyhedral set, and $\boldsymbol{S} \subset \mathbb{R}^{\boldsymbol{n}}$ is a closed set.

Can we strengthen the description of P with cuts?

$\boldsymbol{S}$-free Sets for Polynomial Optimization and Oracle-Based Cuts

B., Chen Chen and Gonzalo Muñoz, 2017

Consider:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & x \in S \cap P. \end{aligned}$$

$\boldsymbol{P := \{x \in \mathbb{R}^n | Ax \le b\}}$ is a polyhedral set, and $\boldsymbol{S \subset \mathbb{R}^n}$ is a closed set.

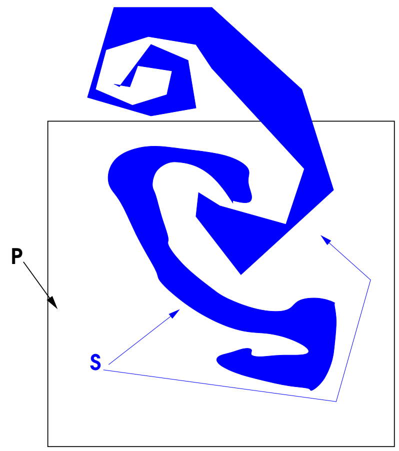
Can we strengthen the description of P with cuts?

We will focus on the geometric approach: cuts via **$\boldsymbol{S}$-free sets**.

(Many other ways to generate cuts, e.g. disjunctions, algebraic arguments, combinatorics, convex cuts, etc.)

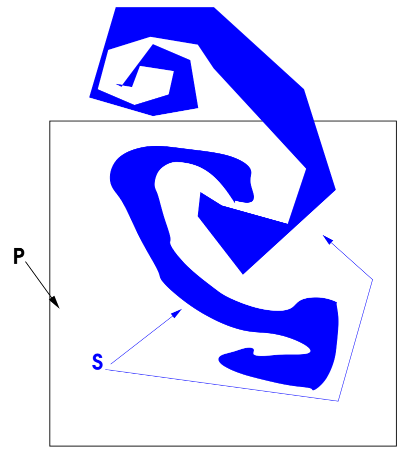
(McCormick, RLT)

Tightening P with an S-free set C

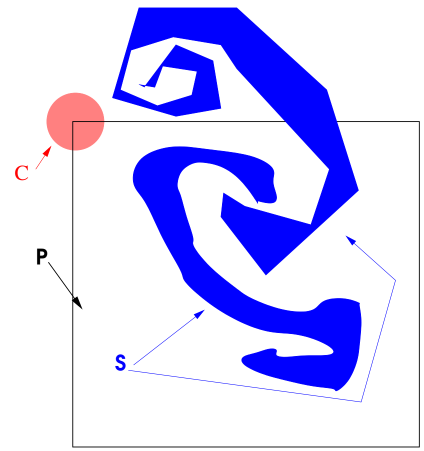


$\boldsymbol{C}$ = closed convex, $\boldsymbol{C} \cap \boldsymbol{X} = \emptyset$

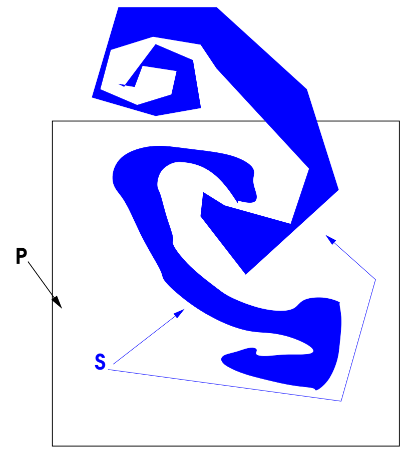
Tightening P with an S-free set C



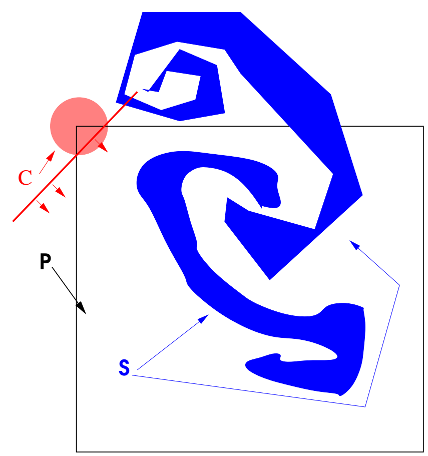
$\boldsymbol{C}$ = closed convex, $\boldsymbol{C} \cap \boldsymbol{X} = \emptyset$



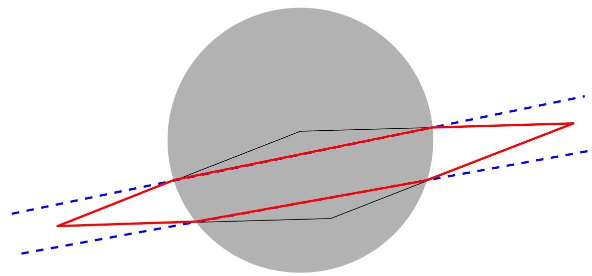
Tightening P with an S-free set C



$\boldsymbol{C}$ = closed convex, $\boldsymbol{C \cap X = \emptyset}$. $\mathbf{conv}(\boldsymbol{P \setminus C})$:



Could be more complex:



- Might need an infinite number of cuts to get $\mathbf{conv}(\boldsymbol{P} \cap \boldsymbol{S})$.

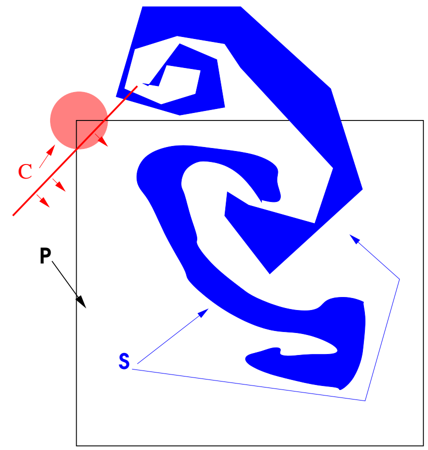
- The problem: given a polytope $\boldsymbol{P}$ and a ball $\boldsymbol{B}$, is $\boldsymbol{P} \subseteq \boldsymbol{B}$? is strongly NP-complete (Freund and Orlin, 1985).

- Given a polyhedral cone $\boldsymbol{C}$ and a ball $\boldsymbol{B}$ it is strongly NP-hard to minimize a convex quadratic over $\boldsymbol{C} \cap \bar{\boldsymbol{B}}$ (B. 2010)

Recent work on the geometry of convex quadratics in the complement of a convex quadratic region

- B. 2010, B and Michalka (2014)
- Belotti, Goez, Pólik, Ralphs, Terlaki (2013)
- Modaresi, M. Kilinc, Vilema (2015)
- F. Kilinc (2015)

From a polyhedral perspective



- Balas (1971), Tuy (1964): if $\boldsymbol{Q}$ is a simplicial cone then the **intersection cut** guarantees separation over $\boldsymbol{conv(Q \setminus \mathrm{int}(C))}$.

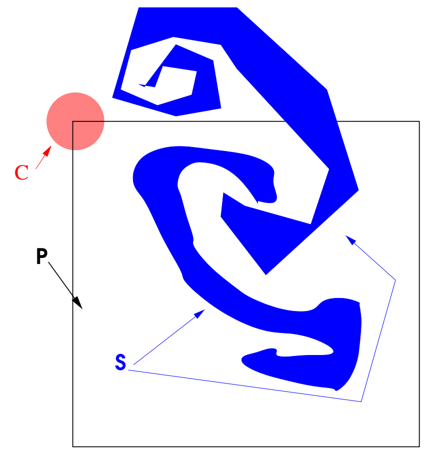
- (Simplicial cone: $\boldsymbol{n}$ linearly independent linear inequalities)

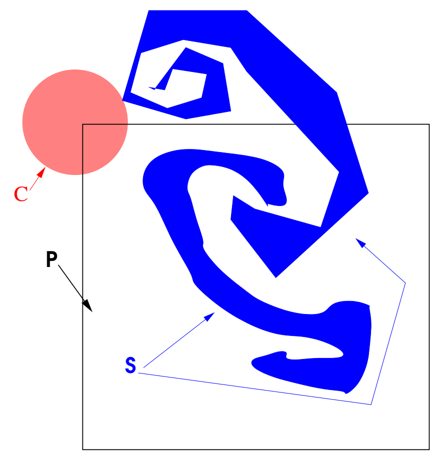
- Simplicial conic relaxation $\boldsymbol{P' \supseteq P}$ is easily obtained from a basic solution of $\boldsymbol{P}$

- And so we could attempt to get $\mathbf{conv}(\boldsymbol{P'} \setminus \mathbf{int}\, \boldsymbol{C}$.

- Intersection cut (w.r.t. $\boldsymbol{P'}$) is described in closed form $\rightarrow$ fast separation of extreme points of $\boldsymbol{P}$ using $\boldsymbol{P'}$

Larger $\boldsymbol{C}$, $\textcolor{red}{\rightarrow}$ deeper cut





Def: $\boldsymbol{S}$-free *maximal* set.

(Some) additional literature

- Maximal S-free sets and minimal valid inequalities: [Basu et al. 2010], [Conforti et al. 2014], [Cornuejols, Wolsey, Yildiz, 2015], [Kilinc-Karzan 2015]

- Intersection cuts and for mixed-integer conic programs programming: [Atamturk and Narayanan 2010], [Belotti et al., 2013], [Andersen and Jensen, 2013], [Dadush, Dey, Vielma 2011], [Modaresi, Kilinc, Vielma 2015/2016]

- Intersection cuts for bilevel optimization: [Fischetti, Monaci, Sinnl, 2016].

- Generalized intersection cut procedures: [Balas and Margot, 2013], [Balas, Kazachkov, Margot 2016].

- Huge literature on split cuts.

This talk

1. A simple, generic way to generate S-free sets that ensures separation. Also, a corresponding cutting plane method for arbitrary closed sets, guaranteed to converge on bounded problems.

2. A study of maximal S-free sets for polynomial optimization

3. Experiments with a resulting cutting-plane procedure that solves LPs only.

4. Joint work with a couple of characters in the audience.

Distance Oracle

We assume we have an oracle for a closed set $\boldsymbol{S}$ that gives us the distance $\boldsymbol{d(x, S)}$ from any point $\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}$ to the nearest point in $\boldsymbol{S}$.

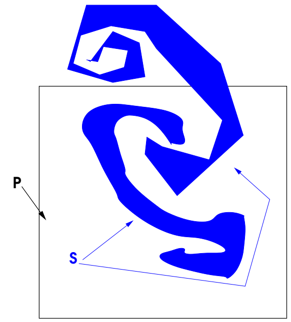
Examples:

- **Integer programming**: if $\boldsymbol{S}$ is the integer lattice, then one can round.
- **Cardinality constraint** nearest vector of cardinality $\leq \boldsymbol{k}$ can be obtained by rounding.
- **Semidefinite cone:** we will see this later

Observation. The ball centered around x with radius $d(x, S)$ is S-free. Call it $\boldsymbol{\mathcal{B}(x, d(x, S))}$.

We will call the corresponding intersection cut an oracle ball cut.

Convergence



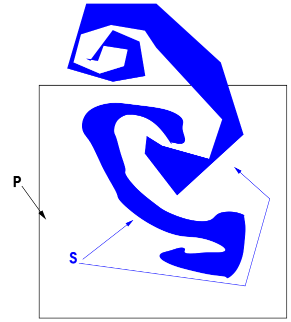
- Start with polytope $P_0 = P$.

- Let $P_{k+1} \doteq \cap_{v \in V_k} \mathbf{conv}(P_k \setminus \text{int}(\mathcal{B}(v\,,\ d(v, S))))$

  V_k = set of extreme points of P_k.

- P_k = rank k closure of P_0.

Convergence



- Start with polytope $P_0 = P$.

- Let $P_{k+1} \doteq \cap_{v \in V_k} \mathbf{conv}(P_k \setminus \mathrm{int}(\mathcal{B}(v\,,\ d(v, S))))$

  V_k = set of extreme points of P_k.

- P_k = rank k closure of P_0.

Theorem: $\lim_{k\to\infty} P_k = \mathbf{conv}(S \cap P)$.

Corollary: iven an inexact but arbitrarily accurate distance oracle, we can obtain arbitrarily close (in terms of Hausdorff distance) polyhedral approximation to $\mathbf{conv}(S \cap P)$ in finite time.

Borrows from proof technique used in [Averkov 2011].

Application: Polynomial Optimization

$$\begin{aligned} z^* \; := \; & \inf \; p_0(x) \\ & \text{s.t.} \quad x \in \boldsymbol{S} \doteq \{x \in \mathbb{R}^n | p_1(x) \geq 0, ..., p_m(x) \geq 0\} \end{aligned}$$

- Saxena, Bonami, Lee 2010-2011: Disjunctive cuts from MILP inner-approximation + convex cuts. Applies to bounded polynomial optimization.

- Ghaddar, Vera, Anjos 2011: Projections of moment relaxations. Generalizes Balas, Ceria, Cornuejols lifting. Separation not guaranteed in general.

- Other literature on convex envelopes of functions, e.g. multilinear. McCormick, spatial branching, RLT.

- Our intersection cuts guarantee polynomial-time separation without boundedness assumptions.

How, 1: lifted polynomial representation

→ this takes us to the moment relaxation we saw before.

[Shor 1987], [Lovasz and Schrijver 1991]

- Define a vector of monomials, $\boldsymbol{m} \doteq [\mathbf{1}, \boldsymbol{x_1}, ..., \boldsymbol{x_n}, \boldsymbol{x_1 x_2}, \boldsymbol{x_1 x_3}, ..., \boldsymbol{x_n^k}]$. Let $\boldsymbol{X} \doteq \boldsymbol{m}\boldsymbol{m}^{\boldsymbol{T}}$.

- Polynomial optimization can be formulated as

$$\begin{aligned} &\min P_0 \bullet X \\ &\text{s.t.} \;\; P_i \bullet X \leq b_i, \; i = 1, ..., m. \end{aligned}$$

  ($\boldsymbol{P_i}$ appropriately defined from the coefficients of $\boldsymbol{p_i}$)

- This is a linear programming relaxation with variables $\boldsymbol{X}$. $P_i \bullet X \doteq \sum p_{ij} m_{ij}$ is the inner product.

- Equivalency when $\boldsymbol{X} \succeq \mathbf{0}$ and rank$(\boldsymbol{X}) = \mathbf{1}$ and consistency constraints (among entries of $\boldsymbol{X}$). Dropping the rank constraint gives the moment relaxation [Lasserre, 2001].

How, 2: S-free sets for Polynomial Optimization

$\rightarrow$ this takes us to the moment relaxation we saw before.

[Shor 1987], [Lovasz and Schrijver 1991]

- Define a vector of monomials, $\boldsymbol{m} \doteq [\mathbf{1}, \boldsymbol{x_1}, \ldots, \boldsymbol{x_n}, \boldsymbol{x_1 x_2}, \boldsymbol{x_1 x_3}, \ldots, \boldsymbol{x_n^k}]$.
  Let $\boldsymbol{X} \doteq \boldsymbol{m}\boldsymbol{m}^{\boldsymbol{T}}$.

- Polynomial optimization can be formulated as

$$\begin{aligned} &\min P_0 \bullet X \\ &\text{s.t. } \; P_i \bullet X \leq b_i,\ i = 1, \ldots, m. \end{aligned}$$

  ($\boldsymbol{P_i}$ appropriately defined from the coefficients of $\boldsymbol{p_i}$)

- This is a linear programming relaxation with variables $\boldsymbol{X}$.
  $P_i \bullet X \doteq \sum p_{ij} m_{ij}$ is the inner product.

- Equivalency when $\boldsymbol{X} \succeq \mathbf{0}$ and $\text{rank}(\boldsymbol{X}) = \mathbf{1}$ and consistency constraints (among entries of $\boldsymbol{X}$). Dropping the rank constraint gives the moment relaxation [Lasserre, 2001].

Three types of $\boldsymbol{S}$-free condtions or cuts

Notation: always over vectorized matrices, e.g.

$$\boldsymbol{M} \in \mathbb{S}^{2\times 2} \rightarrow \{\boldsymbol{M}_{11}, \boldsymbol{M}_{12}, \boldsymbol{M}_{22}\} \in \mathbb{R}^3$$

$\boldsymbol{S}^{2\times 2} = 2 \times 2$ symmetric matrices

- $\mathbf{2 \times 2}$ minors. **Theorem** (Chen et al 2016):
  A psd matrix $\boldsymbol{M}$ is of rank one iff every principal $\mathbf{2 \times 2}$ minor is zero.
  So, given $\bar{\boldsymbol{X}}$, if $\bar{\boldsymbol{X}}_{i,j} \succ \mathbf{0}$ for some $\boldsymbol{i, j}$ we have a violation.
  $\boldsymbol{S}$-free set: $\boldsymbol{M}_{i,j} \succeq \mathbf{0}$, which is *maximal* $\boldsymbol{S}$-free.

- Positive-semidefiniteness: of $\bar{\boldsymbol{X}}$ is not psd, i.e. $\boldsymbol{c}^T\bar{\boldsymbol{X}}\boldsymbol{c} < \mathbf{0}$ for some $\boldsymbol{c}$, then get cut $\boldsymbol{c}^T\boldsymbol{X}\boldsymbol{c} \geq \mathbf{0}$ (also defines a maximal set, but we have a cut anyway)

- Oracle (rank-1) ball, and shifted oracle ball. $\boldsymbol{EYM}$ theorem gives **distance** from a psd matrix to the nearest rank one matrix (Modification by Dax for non-psd case).

Numerical Experiments

- Python
- All the cuts mentioned above
- Gurobi 7.0.1 to solve LPs
- 20-core server, but only Gurobi uses more than one

- 26 QCQP problems from GLOBALLib (6-63 variables)

- BoxQP instances (21-126 variables)

Results

Cut Family	Initial Gap	End Gap	Closed Gap	# Cuts	Iters	Time (s)	LPTime (%)
OB	1387.92%	1387.85%	1.00%	16.48	17.20	2.59	2.06%
SO		1387.83%	8.77%	18.56	19.52	4.14	2.29%
OA		1001.81%	8.61%	353.40	83.76	33.25	7.51%
2x2 + OA		1003.33%	32.61%	284.98	118.08	30.40	15.03%
SO+2x2+OA		1069.59%	31.91%	174.79	107.16	29.55	12.56%

Table 1: Averages for GLOBALLib instances

Comparison with V2: BoxQP

V2: second-order conic outer-approximation of PSD constraint; MIP to derive disjunctive cuts (Saxena, Bonami, Lee)