Global nonconvex optimization with Gurobi

Robert Luce

OLC workshop, Princeton, June 2024
What is MIQCP?

We consider the problem

\[
\min_{x \in \mathbb{R}^n} \quad x^T Q_0 x + c^T x \\
\text{s.t.} \quad Ax = b \\
\quad x^T Q_k x + p_k^T x \geq d_k, \quad k = 1, \ldots, q \\
\quad l \leq x \leq u \\
\quad x_i \in \mathbb{Z}, \quad i \in I
\]

with \( A \in \mathbb{R}^{m,n} \), and all \( Q_k \in \mathbb{R}^{n \times n} \) symmetric.

- Our goal: find a provably global optimal solution.
- Our solution strategy: Branch-and-bound (BnB).
Branch-and-bound, relaxations
Basic idea of BnB for MIP

Let’s forget about all quadratic constraints, think plain MIP!

- Basic idea: Forget integrality constraints, enlarge the feasible space ("relaxation")
  - This is now a convex problem! Denote optimal solution by $x^*$.
- If all $\{x^*_i \mid i \in I\}$ are integral: done
- Otherwise: Pick fractional $x^*_i$, create two child problems by enforcing
  - $x_i \leq \lfloor x^*_i \rfloor$ in one branch, and
  - $x_i \geq \lceil x^*_i \rceil$ in the other branch.
  - Recurse on both subproblems.
  - Stop when relaxation objective value exceeds objective value of a known solution.

- Naive algorithm may implicitly enumerate all integer points.
- Not all subtrees need to be explored though.
- Practical implementations of this idea are surprisingly effective.
- A lot of algorithmic machinery is needed.
Relaxations for nonconvex quadratic constraints

Goal: BnB subproblems must be solvable efficiently.

- **Ingredient 1**: Break quadratic constraints into a set of elementary, bilinear constraints $z = xy$.
- **Ingredient 2**: Replace such bilinear constraints by their convex envelope.
- **Ingredient 3**: Branch not only on integer variables, but also on variables that tighten the envelope.
Breaking apart quadratic constraints

Consider the quadratic constraint

\[ 3x_1^2 - 7x_1x_2 + 2x_1x_3 - x_2^2 + 3x_2x_3 - 5x_3^2 = 12. \]

Introduce auxiliary variables and bilinear constraints:

\[ 
\begin{align*}
  z_{11} &= x_1^2 \\
  z_{12} &= x_1x_2 \\
  z_{13} &= x_1x_3 \\
  z_{22} &= x_2^2 \\
  z_{23} &= x_2x_3 \\
  z_{33} &= x_3^2 
\end{align*}
\]

And add replace the quadratic constraint by a linear one:

\[ 3z_{11} - 7z_{12} + 2z_{13} - z_{22} + 3z_{23} - 5z_{33} = 12. \]
McCormick relaxation for $z = xy$

$-z + xy = 0$

$-z + l_x y + l_y x \leq l_x l_y$

$-z + u_x y + u_y x \leq u_x u_y$

$-z + u_x y + l_y x \geq u_x l_y$

$-z + l_x y + u_y x \geq l_x u_y$
McCormick relaxation, cont’d

\[-z + l_x y + l_y x \leq l_x l_y\]
\[-z + u_x y + u_y x \leq u_x u_y\]
\[-z + u_x y + l_y x \geq u_x l_y\]
\[-z + l_x y + u_y x \geq l_x u_y\]

- Inequalities depend on the *bounds* of $x, y$ (only!).
- The smaller the domains, the better approximation from the envelope.
- Picking a variable to branch on balances:
  - Total reduction in envelope volume.
  - Number of participations in violated bilinear constraints.
- Picking a branching value for the variable considers:
  - Midpoint of the variable’s local domain.
  - Variable value in the current relaxation.
A useful cutting plane
Simplified setup

We consider the problem

\[
\min_{x \in \mathbb{R}^n} x^T Q x + c^T x \\
\text{s.t. } Ax = b \\
\quad x \geq 0 \\
\quad x_I \in \mathbb{Z}
\]

- We are interested in the case where \( x^T Q x \) is nonconvex.
- Problem: Relaxing \( x_I \in \mathbb{Z} \) gives us only a nonconvex continuous problem.
- Relaxing the quadratic function as just seen gives us a convex relaxation.
McCormick relaxation as extended formulation

For notational convenience, we will rephrase the McCormick relaxation:

- For each appearing quadratic term $x_i x_j$ introduce an auxiliary variable $X_{ij}$.
- Add some polyhedral constraints $(x, X) \in S$ that connect $x_i x_j$ with $X_{ij}$ (linear envelope of $x_i x_j$).
- The envelope becomes tighter in the course of branching, bound changes for $x_i, x_j$ propagate to bound changes for $X_{ij}$.

Challenge: We may need to branch many times until the relaxation solution satisfies

$$xx^T = X.$$
What is a cutting plane?

- Let $\mathcal{F} \subset \mathbb{R}^n$ be the feasible set of the original optimization problem.
- Let $\mathcal{R} \subset \mathbb{R}^n$ be the feasible set of some relaxation during BnB.
- Assume that the optimal relaxation solution $(x^*, X^*) \in \mathcal{R}$ satisfies $x^*x^T \neq X^*$. 
What is a cutting plane?

- Let $\mathcal{F} \subset \mathbb{R}^n$ be the feasible set of the original optimization problem.
- Let $\mathcal{R} \subset \mathbb{R}^n$ be the feasible set of some relaxation during BnB.
- Assume that the optimal relaxation solution $(x^*, X^*) \in \mathcal{R}$ satisfies $x^* x^*^T \neq X^*$.

Can we find a linear inequality

$$d^T x + \text{tr}(DX) \leq g, \quad d \in \mathbb{R}^n, D \in \mathbb{R}^{n,n}, g \in \mathbb{R}$$

such that

- $d^T x^* + \text{tr}(DX^*) > g$, but
- $\mathcal{F} \cap \{x \in \mathbb{R}^n \mid d^T x + \text{tr}(DX) \leq g\} = \mathcal{F}$?
What is a cutting plane?

- Let $\mathcal{F} \subset \mathbb{R}^n$ be the feasible set of the original optimization problem.
- Let $\mathcal{R} \subset \mathbb{R}^n$ be the feasible set of some relaxation during BnB.
- Assume that the optimal relaxation solution $(x^*, X^*) \in \mathcal{R}$ satisfies $x^* x^T \neq X^*$.

Can we find a linear inequality

$$d^T x + \text{tr}(DX) \leq g,$$

such that

- $d^T x^* + \text{tr}(DX^*) > g$, but
- $\mathcal{F} \cap \{x \in \mathbb{R}^n \mid d^T x + \text{tr}(DX) \leq g\} = \mathcal{F}$?

Then we can get a better relaxation through $\mathcal{R} \cap \{x \in \mathbb{R}^n \mid d^T x + \text{tr}(DX) \leq g\}$!
Cuts from SDP outer approximation 1

We will use the $xx^T = X$ to derive globally valid cutting planes for the relaxed extended formulation.
Cuts from SDP outer approximation 1

We will use the $xx^T = X$ to derive globally valid cutting planes for the relaxed extended formulation.
For any $x \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times n}$ We have

$$
xx^T = X \Rightarrow xx^T \preceq X
\Rightarrow 0 \preceq X - xx^T
\Rightarrow 0 \preceq \begin{bmatrix} 1 & 0 \\ 0 & X - xx^T \end{bmatrix}
\Rightarrow 0 \preceq \begin{bmatrix} 1 & 0 \\ x & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & X - xx^T \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix}
\Rightarrow 0 \preceq \begin{bmatrix} 1 & xx^T \\ x & X \end{bmatrix} =: \hat{X}
$$

How do we derive cuts from $0 \preceq \hat{X}$?
Cuts from outer approximation 2

Recall

\[
\begin{bmatrix}
1 & x^T \\
x & X
\end{bmatrix} =: \hat{X}
\]

From the variational characterization

\[
\hat{X} \succeq 0 \iff \mathbf{v}^T \hat{X} \mathbf{v} \geq 0 \quad \forall \mathbf{v} \in \mathbb{R}^n
\]

we see that a solution \((x^*, X^*)\) for the relaxation is cut off by the \textit{linear} cutting plane \(\mathbf{v}^T \hat{X} \mathbf{v} \geq 0\) by any \(\mathbf{v} \in \mathbb{R}^n\) satisfying

\[
\mathbf{v}^T \hat{X}^* \mathbf{v} < 0.
\]
Characterization of cut-defining vectors

- Let \((\lambda, v)\) be a normalized eigenpair with \(\lambda < 0\), then

\[ v^T \hat{X}^* v = \lambda v^T v = \lambda < 0. \]

- More generally, let \(\mathcal{U} := \text{span}\{v_1, \ldots, v_s\}\) be the subspace generated from eigenvectors corresponding to all negative eigenvalues. Then any \(v \in \mathcal{U}\) defines a cut.

- Reverse: any cut-defining \(v\) satisfies \(\text{proj}_{\mathcal{U}}(v) \neq 0\)

- Even better: If \(v \notin \mathcal{U}\), and \(w = \text{proj}_{\mathcal{U}}(v)\), then \(w^T \hat{X}^* w \leq v^T \hat{X} v\).

Conclusion: \(\mathcal{U}\) is the right place to look for cuts.

Problems: \(\mathcal{U}\) is expensive to compute for large \(n\), and the number of nonzeros in the cut are \(\frac{n(n+1)}{2} + n\).
Cuts from submatrices

For $\mathcal{I} \subseteq [n]$ we define the submatrix of $\hat{X}$ induced by $\mathcal{I}$ by

$$
\hat{X}_\mathcal{I} := \begin{bmatrix} x(\mathcal{I})^T & X(\mathcal{I}, \mathcal{I}) \end{bmatrix}.
$$

Passing to subsets is a way around computational burden, but since

$$
\min_{\mathbf{v} \in \mathbb{R}^n} \mathbf{v}^T \hat{X} \mathbf{v} \leq \min_{\mathbf{v} \in \text{span}\{e_i\}_{i \in \mathcal{I}}} \mathbf{v}^T \hat{X} \mathbf{v} = \min_{\mathbf{v} \in \mathbb{R}^{\mathcal{I}}} \mathbf{v}^T \hat{X}_\mathcal{I} \mathbf{v}
$$

a cut may be quite a bit weaker than the best possible cut on $\hat{X}$. 
Sparse extended formulations

Typically we will not add *all* the variables $X_{ij}$ in our extended formulation. For simplicity assume that we have added all variables corresponding to the incidence graph $G_Q = (V, E) := G(Q)$ though.

Simple heuristic 1:
▶ Pick any “small” clique $C$ in $G_Q$.
▶ Apply cut heuristic to $G_Q[C]$.

Simple heuristic 2:
▶ Compute a chordal completion $C$ of $G_Q$.
▶ For each maximal clique of $C$ (that is still small enough...) fill entries in $X^*$ by $X^*_{ij} = X_{ij}$ if $(i, j) \in E$ and relax “missing” variables in the cut by an upper bound.
▶ If cut still cuts off $(x^*, X^*)$, take it!
Sparse extended formulations

Typically we will not add all the variables $X_{ij}$ in our extended formulation. For simplicity assume that we have added all variables corresponding to the incidence graph $G_Q = (V, E) := G(Q)$ though.

Simple heuristic 1:

- Pick any "small" clique $C$ in $G_Q$.
- Apply cut heuristic to $G_Q[C]$.

Simple heuristic 2:

- Compute a chordal completion $C$ of $G_Q$.
- For each maximal clique of $C$ (that is still small enough...) fill entries in $X^*_{ij}$ by $X^*_{ij} = X_{ij}$ if $(i, j) \in E$, $x^*_{i} x^*_{j}$ otherwise, and relax “missing” variables in the cut by an upper bound.
- If cut still cuts off $(x^*, X^*)$, take it!
Sparse extended formulations

Typically we will not add all the variables $X_{ij}$ in our extended formulation. For simplicity assume that we have added all variables corresponding to the incidence graph $G_Q = (V, E) := G(Q)$ though.

Simple heuristic 1:

▶ Pick any "small" clique $C$ in $G_Q$.
▶ Apply cut heuristic to $G_Q[C]$.

Simple heuristic 2:

▶ Compute a chordal completion $C$ of $G_Q$.
▶ For each maximal clique of $C$ (that is still small enough...) fill entries in $X^*$ by

$$[X^*]_{ij} = \begin{cases} 
X^*_{ij} & \text{if } (i, j) \in E \\
x^*_i x^*_j & \text{otherwise},
\end{cases}$$

and relax “missing” variables in the cut by an upper bound.
▶ If cut still cuts off $(x^*, X^*)$, take it!
Eigenspace guided submatrix selection

Now consider the setting where $G_Q$ is large and sparse. We can compute an $s$-dimensional approximation to $\mathcal{U}$ (e.g., Lanczos, Krylov-Schur).

▶ Basic operation: Matrix vector products with $\hat{X}^*$, cost $\mathcal{O}(n + |E|)$ each, and a few eigensolves of size $s$.

▶ *If the method converges*, we obtain a $U \in \mathbb{R}^{n,s}$ with orthonormal columns, such that $\text{span}(U) \subseteq \mathcal{U}$. (Or a certificate that no cuts can be separated.)
Now consider the setting where $G_Q$ is large and sparse. We can compute an $s$-dimensional approximation to $U$ (e.g., Lanczos, Krylov-Schur).

- Basic operation: Matrix vector products with $\hat{X}^*$, cost $\mathcal{O}(n + |E|)$ each, and a few eigensolves of size $s$.
- If the method converges, we obtain a $U \in \mathbb{R}^{n,s}$ with orthonormal columns, such that $\text{span}(U) \subseteq U$. (Or a certificate that no cuts can be separated.)

With $U$ at hand, we could:

1. Generate dense cuts as before.
2. Project $U$ on a matrix with smaller support, resulting in sparse cuts.
Interlude 1: NMF

A nasty nonconvex quadratic problem
NMF is a structured, low-rank matrix factorization

Given $X \in \mathbb{R}^{m,n}_+$, find

- “small” $r \in \mathbb{N}$,
- $W \in \mathbb{R}^{m,r}_+$,
- $H \in \mathbb{R}^{r,n}_+$

such that $X = WH$ (or $X \approx WH$). $W, H$ often decompose $X$ while preserving interpretability (face pictures / facial features, measured spectrum of a mixture / spectral signatures of species, etc.)

Given an NMF $X = WH$ of rank $r$, is it unique

- up to permutation $WH = (WP^T)(PH)$, and
- up to nonnegative, diagonal scaling $WH = (WD^{-1})(DH)$?

Typically all questions one may want to ask around NMF result in NP-hard optimization problems.
The sufficiently scattered condition (SSC)\textsuperscript{1}

Let $H \in \mathbb{R}_{+}^{r \times n}$ and

$$cone(H) = \{x \mid x = Hy, y \geq 0\}$$

$$cone(H)^* = \{x \mid H^T x \geq 0\}$$

$$C = \{x \in \mathbb{R}^r \mid e^T x \geq \sqrt{r - 1}\|x\|_2\} \subseteq \mathbb{R}_{+}^r$$

$$C^* = \{x \in \mathbb{R}^r \mid e^T x \geq \|x\|_2\} \supseteq \mathbb{R}_{+}^r$$

Then $H$ satisfies the SCC if

**SSC1** $C \subseteq cone(H)$

**SSC2** $cone(H)^* \cap \{x \mid e^T x = \|x\|_2\} \subseteq \bigcup_i \text{span}\{e_i\}$

**Theorem**

*Let $X = WH$ and NMF where $W^T$ and $H$ satisfy the SSC. Then this NMF is unique up to permutation and scaling.***

Checking the SSC, computationally

How do we check a given \( H \in \mathbb{R}_{+}^{r,n} \) for the SCC? A few subtleties set aside\(^2\), Gurobi can solve

\[
\begin{align*}
\max_x \quad & \|x\|_2 \\
\text{s.t.} \quad & e^T x = 1 \\
& H^T x \geq 0
\end{align*}
\]

If the optimal objective value is strictly greater than one, then \( H \) does not satisfy SSC1. Further, the set of optimal solutions must be equal to \( \{e_i\}_{i=1}^{r} \) – a condition Gurobi can check by enumerating the optimal solutions.

Going nonlinear in BnB
Relaxations for nonlinear constraints

Consider the constraint

\[ f(\theta) := \sqrt{1 + \theta^2} + \ln(\theta + \sqrt{1 + \theta^2}) \leq 2, \quad \theta \geq 0 \]

Goal: BnB subproblems must be solvable efficiently.

- **Ingredient 1:** Decompose \( f(\theta) \leq 2 \) into a set of \textit{atomic} nonlinear constraints \( y_k = f_k(x) \).
- **Ingredient 2:** Replace epigraph of each such atomic constraint by its convex envelope.
- **Ingredient 3:** Branch not only on integer variables, but also on variables that tighten the envelopes.
Decomposing a nonlinear constraint

\[ f(\theta) := \sqrt{1 + \theta^2} + \ln \left( \theta + \sqrt{1 + \theta^2} \right) \leq 2, \quad \theta \geq 0 \]

Introduce auxiliary variables \( y_k \) and the atomic constraints:

\[
\begin{align*}
    y_1 &= 1 + \theta^2 \\
    y_2 &= \sqrt{y_1} \\
    y_3 &= \theta + y_2 \\
    y_4 &= \ln y_3
\end{align*}
\]

And replace the original nonlinear constraint by a linear one:

\[ y_2 + y_4 \leq 2 \]

We only need to deal with these atomic \( y = g(x) \) constraints in BnB.
From bilinear constraints to $y = f(x)$

- Goal: Find a convex relaxation to the epigraph of $f(x) - y = 0$.
- Basic technique: Polyhedral envelopes
  - Fixed number of hyperplanes
  - Upper and/or lower envelope (as needed)
  - Adaptive adjustment of envelope coefficients
- As branching reduces the domain of $x$, the relaxation becomes tighter
- Hyperplane generation takes into account the specific properties of each atomic function $f$
- In effect this is *roughly* equivalent to a dynamic PWL approximation in the BnB tree traversal

All this becomes is pretty heavy in notation if spelled out in detail, we’ll just give a few examples
An easy case

If \( \sin \) is convex within the bounds of \( x \) ...

- Upper envelope is given by secant through \( f(\text{lb}) \) and \( f(\text{ub}) \)
- Lower envelope constructed by tangents to \( \sin \), viz. \( \sin(x_0) + \frac{d}{dx}\sin(x_0)(x - x_0) \)
- Resulting hyperplanes added to LP
- Shaded in red: Relaxation of the epigraph of \( f(x) - y = 0 \)

- Similar if \( \sin \) is concave on the domain of \( x \)
- Adding more tangents at various points improves the relaxation
Neither convex, concave

- If $\sin$ is neither convex nor concave on the domain of $x$...
  - Lower envelope
    - Compute leftmost solution $x_0$ to
      \[
      \frac{d}{dx} \sin(x) = \frac{\sin(x) - \sin(lb)}{x - lb}
      \]
      - Computed $x_0$ defines one tangent
      - Remaining part is convex, use some tangent
  - Similar: Upper envelope
    - Compute rightmost solution $x_1$ to
      \[
      \frac{d}{dx} \sin(x) = \frac{\sin(ub) - \sin(x)}{ub - x}
      \]
      - Remaining part is concave, use some tangent
“Large” domains

- Not much to get from the relaxation if $x$’s domain is large
- Again, branching on $x$ tightens the relaxation quickly!
Handling poles

- Example: $f(x) = x^{-2}$
- If no bounds on $x$ are given, the only possible convex relaxation of $f(x) - y = 0$ is $y \geq 0$
Handling poles

- Example: $f(x) = x^{-2}$
- We have bounds on $x$ with $lb < 0 < ub$
- Lower envelope
  - Unique tangent with $x_0 > 0$ that passes $f(lb)$
- Upper envelope
  - Only trivial envelope is valid
  - Branching at the pole needed
Handling poles

- Example: \( f(x) = x^{-2} \)
- We have bounds on \( x \) with \( 0 < lb < ub \)
- Lower envelope
  - Convex, use tangent(s) at will
- Upper envelope
  - Use secant between \( f(lb) \) and \( f(ub) \)
- Caution if \( lb \) is close to pole
  - Coefficients in tangents/secant become very large!
- Caution if \( ub \) is large
  - Coefficients in tangents/secant become very small
Branching and convergence

Branching on the argument variable $x$ “improves” the relaxation; pointwise quadratic convergence in convex/concave domains
Our direction is set

- Gurobi 9: Extend BnB to nonconvex quadratic functions
- Gurobi 9,10: Computes static PWL approximations to atomic functions $f(x)$
  - Atomic functions used in heuristics
- Gurobi 11: Implements dynamic envelopes for atomic functions $f(x)$, nonlinear constraints must be given already decomposed
- Gurobi 12 (Q4/24): Will (hopefully) accept any nonlinear constraint
  - Automatically decomposition for BnB
  - Original formulation used in heuristics
Interlude 2: Trained predictors as constraints
From linear constraints to trained predictors

Think about $x \in \mathbb{R}^n$ as *input* variables, and $y \in \mathbb{R}^m$ as *output* variables.

$$\min_{x,y} \ f(x, y)$$

s.t. $Ax = y$

plus bounds, integrality of any $x_i, y_j$

So the input and output are linearly related through $A$. 

---

GUROBI OPTIMIZATION
From linear constraints to trained predictors

Think about $x \in \mathbb{R}^n$ as *input* variables, and $y \in \mathbb{R}^m$ as *output* variables.

$$\min_{x, y} f(x, y)$$

s.t. $Ax = y$

plus bounds, integrality of any $x_i, y_j$

So the input and output are linearly related through $A$. Let’s replace $A$ by a trained predictor $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$:

$$\min_{x, y} f(x, y)$$

s.t. $g(x) = y$

plus bounds, integrality of any $x_i, y_j$
Meet Gurobi Machine Learning

- Gurobi ML is a Python package to formulate trained predictors in Gurobi models
- Beyond linear regression most ML models rely on some nonlinear function:
  1. Sigmoid/Logistic function: \( \sigma(z) = \frac{1}{1 + e^{-z}} \)
  2. \( \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}} \)
  3. SoftMax: \( \sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \) for \( i = 1, 2, \ldots, K \)
  4. ReLU: \( y = \max(0, x) \)
  5. Discrete choice for decision trees (Piecewise Constant functions)
- Currently Gurobi ML can formulate models that use ReLU and Logistic

https://github.com/Gurobi/gurobi-machinelearning
Advantages of Gurobi 11’s dynamic approach

- Gurobi 10: Logistic regression models solved through a static PWL approximation
- Gurobi 11: BnB approach and nonlinear barrier
- Better Solutions Faster. E.g. Janos model (Bergman et.al. 2020):
  - Solution with static approach violates function by $3.0434\times 10^{-4}$
  - Solution with dynamic approach violates by $6.46901614\times 10^{-7}$
  - Running time similar (£1 sec.)
  - Requiring a static PWL approximation with same accuracy, builds prohibitive model solved in 700 sec.
- Using this can do add-hoc models for SoftMax. On an example of adversarial ML
  - Gurobi 11 is $13\times$ faster
  - Significantly less violated solutions
Advantages of Gurobi 12’s nonlinear function support

- Nonlinear functions can be handled better:
  - We know the function as whole (not just the decomposition!), better error control
  - Capitalize on nonlinear barrier to get locally optimal solutions
- It also becomes possible to write an entire neural network within one single expression:
  - Advantage for global optimality unclear (the spatial BnB algorithm will still need to use a decomposition)
  - But barrier can be used to get locally optimal solutions for much larger networks
Conclusions

- Nonconvex global optimization is fun
- Integration in a grown BnB-MIP-framework requires touching a lot of wheels, big and small
- Can we repeat the success story of MIP?
Conclusions

- Nonconvex global optimization is fun
- Integration in a grown BnB-MIP-framework requires touching a lot of wheels, big and small
- Can we repeat the success story of MIP?

Thanks!