Mixed-Projection Conic Optimization: A New Paradigm for Modeling Rank Constraints

Ryan Cory-Wright
MIT, Operations Research Center
Paper available at: ryancorywright.github.io
Questions? Email ryancw@mit.edu

Joint work with
Dimitris Bertsimas (MIT)
Jean Pauphilet (LBS)

Problem Setting

Setting: General low-rank optimization problems with conic constraints

$$\min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}} \quad \lambda \cdot \mathrm{Rank}(\boldsymbol{X}) + \langle \boldsymbol{C}, \boldsymbol{X} \rangle \quad \text{s.t.} \quad \boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}, \ \mathrm{Rank}(\boldsymbol{X}) \leq k, \ \boldsymbol{X} \in \mathcal{K},$$

Where:

- Rank objective ➡ ***X*** of minimal complexity,
- ***AX=B*** defines a subspace,
- $\mathcal{K}$ is a *proper* cone, e.g., non-negative orthant, semidefinite cone.

Application: Matrix Completion

Formulation

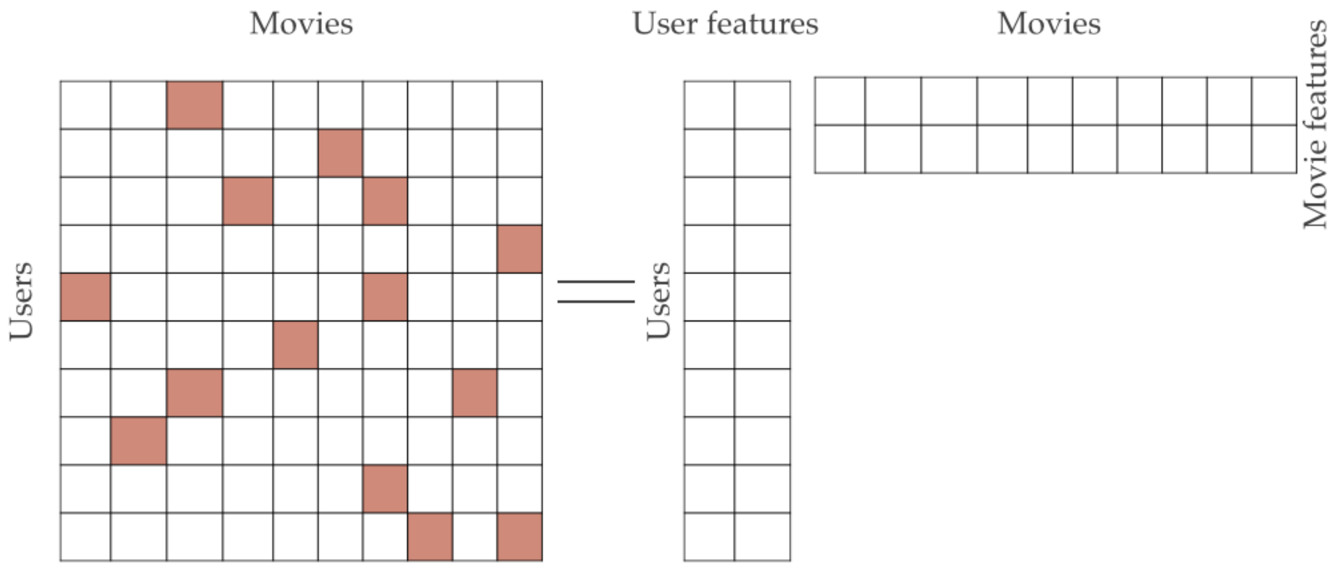
$$\min_{\boldsymbol{X} \in \mathbb{R}^{n \times p}} \quad \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} (X_{i,j} - A_{i,j})^2 \quad \text{s.t.} \quad \text{Rank}(\boldsymbol{X}) \leq k.$$

Decision variables/Problem data

$X_{i,j}$: Predicted rating movie j by user i
$A_{i,j}$: Reported rating movie j by user i

Movie Recommendation:

- Given user movie ratings, predict ratings for unseen movies.
- To make problem tractable, assume ratings depend on k factors (lead actor, lead actress, director, genre, year, ..)



Many Other Applications!

- **Machine Learning:** Decompose matrix into sparse plus low-rank matrix.
- **Control Theory:** Identify minimum complexity realization of a system.
- **Algebraic Geometry:** Minimum degree sum-of-squares decomposition of polynomial.
- **Computer Vision:** Decompose a matrix into a product of non-negative factors.
- **Many, many more…**

Low-Rank Optimization: How Hard Can it Be?

1. Rank constraints are hard, in a complexity-theoretic sense.
   - Known to be strongly NP-hard
   - **We show:** *existential theory of reals*-hard, i.e., as hard as deciding feasibility of semialgebraic set.

2. Rank constraints cannot be modeled using mixed-integer optimization.
   - How should we model rank constraints then?

We're gonna need a new approach!

Contributions

- We propose to model rank constraints using a projection matrix as an additional variable
- The resulting framework, mixed-projection optimization, naturally extends MIO to rank constraints
- We show that some of the most successful tools from MIO (e.g., big-M methods, branch-and-bound, outer-approximation, relaxation and rounding schemes) can be extended to MPO
- With the current technology, can already solve matrix completion w. 50s features to certifiable optimality,1000s to near optimality.

Modeling Rank with Projection Matrices

Cardinality constraints can be modeled using binary variables

$$\|\mathbf{x}\|_0 \leq k \iff \exists \mathbf{z} \in \{0,1\}^n : \mathbf{e}^\top \mathbf{z} \leq k, \ \mathbf{x} = \mathbf{z} \circ \mathbf{x},$$

Rank constraints can be modeled using projection matrices

$$\mathrm{Rank}(\mathbf{X}) \leq k \iff \exists \mathbf{Y} \in \mathcal{Y}_n : \mathrm{tr}(\mathbf{Y}) \leq k, \ \mathbf{X} = \mathbf{Y}\mathbf{X},$$

where $\mathcal{Y}_n := \{\mathbf{P} \in S^n : \mathbf{P}^2 = \mathbf{P}\}$

Similarities Between Binaries and Projections

- Cardinality on binaries is a linear function (inner product with the vector of all ones)
- Binary variables are idempotent scalars which satisfy $z^2=z$
- We model logical constraints via $x=zx$

- Rank on projection matrices is a linear function (inner product with identity matrix)
- Projection matrices are idempotent matrices which satisfy $\boldsymbol{Y}^2=\boldsymbol{Y}$
- We model rank constraints via $\boldsymbol{X}=\boldsymbol{Y}\boldsymbol{X}$

Mixed-Projection Formulation

$$\min_{\mathbf{X}\in\mathbb{R}^{n\times m}} \quad \lambda\cdot\mathrm{Rank}(\mathbf{X})+\langle\mathbf{C},\mathbf{X}\rangle \quad \text{s.t.} \quad \mathbf{A}\mathbf{X}=\mathbf{B},\ \mathrm{Rank}(\mathbf{X})\le k,\ \mathbf{X}\in\mathcal{K},$$

can be expressed as Mixed-Projection Conic Optimization problem

$$\min_{\mathbf{Y}\in\mathcal{Y}_n^k}\ \min_{\mathbf{X}\in\mathbb{R}^{n\times m}} \quad \lambda\cdot\mathrm{tr}(\mathbf{Y})+\langle\mathbf{C},\mathbf{X}\rangle \quad \text{s.t.} \quad \mathbf{A}\mathbf{X}=\mathbf{B},\ \mathbf{X}=\mathbf{Y}\mathbf{X},\ \mathbf{X}\in\mathcal{K}.$$

where ***Y*** is a projection matrix.

How to Deal With the Nonlinear Term ***X=YX***

We add a regularization term to the objective, **Ω(X)**, of the following form:

- Either a hard constraint on the spectral norm of **X**

$$\Omega(\mathbf{X}) = 0 \text{ if } \|\mathbf{X}\|_\sigma \leq M, \, +\infty \text{ otherwise}$$

(generalizing the big-M constraints from MIO)

- Or a penalty on the Frobenius norm of **X**

$$\Omega(\mathbf{X}) = \frac{1}{2\gamma}\|\mathbf{X}\|_F^2$$

(generalizing the perspective reformulation from MIO)

Min-max Formulation

Consider problem as a projection-only minimization

$$\min_{\boldsymbol{Y} \in \mathcal{Y}_n^k} \quad f(\boldsymbol{Y}) + \lambda \cdot \mathrm{tr}(\boldsymbol{Y})$$

with $\quad f(\boldsymbol{Y}) := \min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}} g(\boldsymbol{X}) + \Omega(\boldsymbol{X}) \quad \text{s.t.} \quad \boldsymbol{X} = \boldsymbol{Y}\boldsymbol{X}$

$$f(\boldsymbol{Y}) = \max_{\boldsymbol{\alpha}, \boldsymbol{V}_{11}, \boldsymbol{V}_{22}} \quad \lambda \cdot \mathrm{tr}(\boldsymbol{Y}) + h(\boldsymbol{\alpha}) - \Omega^\star(\boldsymbol{\alpha}, \boldsymbol{Y}, \boldsymbol{V}_{11}, \boldsymbol{V}_{22})$$ ⟵ strong duality

where $\Omega^\star(\boldsymbol{\alpha}, \mathbf{Y}, \mathbf{V}_{11}, \mathbf{V}_{22})$ $\boxed{\text{is linear in } \mathbf{Y}}$ ⟵ cutting-plane method

non-trivial part of the proof

Algorithms - Overview

- **Exact method:** Solve via outer-approximation w. spatial branch-and-bound.
- **Approximate method:** Solve semidefinite relaxation, round eigenvalues to obtain near-optimal solution.

Exact Outer-Approximation Algorithm

$$\min_{\boldsymbol{Y} \in \mathcal{Y}_n^k} \max_{\boldsymbol{\alpha}, \boldsymbol{V}_{11}, \boldsymbol{V}_{22}} \quad \lambda \cdot \mathrm{tr}(\boldsymbol{Y}) + h(\boldsymbol{\alpha}) - \Omega^\star(\boldsymbol{\alpha}, \boldsymbol{Y}, \boldsymbol{V}_{11}, \boldsymbol{V}_{22})$$

Algorithmic Implementation:

- Iteratively construct a piece-wise linear lower approximation
- Implemented using lazy constraint callbacks in Gurobi.
- To warm-start lower bound, run method on semidefinite relaxation.
- To warm-start upper bound, greedily round semidefinite relaxation.

Optimizing Over Projection Matrices

How to tell Gurobi that $\boldsymbol{Y}$ is a projection matrix?

→ Solution: *define* $\boldsymbol{Y}$ using non-convex quadratic constraints

How to tell Gurobi that $\boldsymbol{Y}$ is semidefinite ?

→ Solution: *strengthen* w. SOC constraints from outer-approximation of PSD cone.

Conclusion Master problem is non-convex quadratic w. SOCs.

$$\min_{\boldsymbol{Y}\in S^n, \boldsymbol{U}\in\mathbb{R}^{n\times k}, \theta} \quad \theta + \lambda \cdot \mathrm{tr}(\boldsymbol{Y}) \quad \text{s.t. } z_i\theta \geq h_i + \langle \boldsymbol{H}_i, \boldsymbol{Y} - \boldsymbol{Y}_i \rangle, \forall i \in [t],$$

$$\underbrace{\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{U}^\top\boldsymbol{U} = \mathbb{I},}_{\text{define}} \underbrace{Y_{i,i}Y_{j,j} \geq Y_{i,j}^2, \forall i,j \in [n], Y_{i,i} \geq \sum_{t=1}^{k} U_{i,t}^2, \forall i \in [n], \mathrm{tr}(\boldsymbol{Y})}_{\text{strengthen}} = k,$$

$$\underbrace{0 \geq \|\boldsymbol{U}_i - \boldsymbol{U}_j\|_2^2 + 2Y_{i,j} - Y_{i,i} - Y_{j,j}, \; 0 \geq \|\boldsymbol{U}_i - \boldsymbol{U}_j\|_2^2 + 2Y_{i,j} - Y_{i,i} - Y_{j,j}, \forall i,j \in [n].}_{\text{strengthen}}$$

Semidefinite Relaxation: Penalty interpretation

Outline of the analysis

- Consider the min-max formulation for the rank-penalized problem
- Relax $\mathbf{Y} \in \mathcal{Y}_n$ into $\mathbf{Y} \in \mathrm{conv}(\mathcal{Y}_n)$
- Take the dual of the inner-maximization problem
- Interpret the relaxation as a penalty on the primal variable **X**

$$\min_{\mathbf{Y} \in \mathbf{Y} \in \mathrm{conv}(\mathcal{Y}_n)} \min_{\mathbf{X}} \ {}'_{22}$$

With spectral regularization → recovers nuclear norm penalty (Fazel, 2002)

With Frobenius regularization → novel "Generalized Reverse Huber"

$$\min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}} \quad g(\boldsymbol{X}) + \sum_{i=1}^{n} \min\left(\sqrt{\frac{2\lambda}{\gamma}}\sigma_i(\boldsymbol{X}), \lambda + \frac{\sigma_i(\boldsymbol{X})^2}{2\gamma}\right).$$

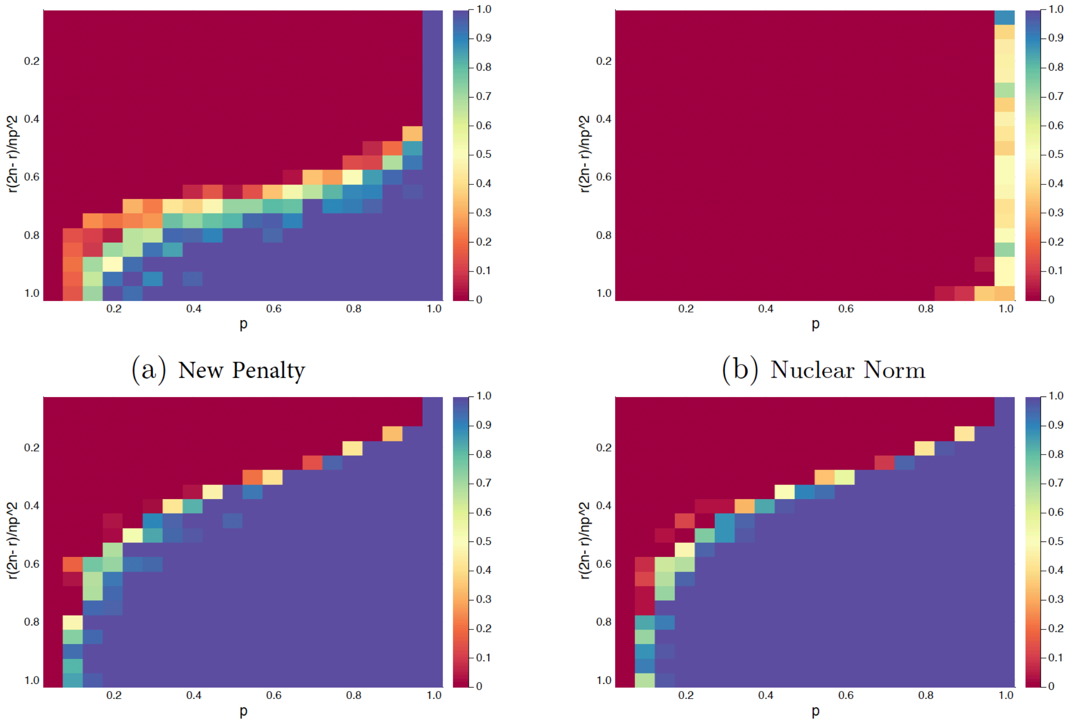
Note: generalizes perspective relaxation and the reverse Huber penalty from MIO

Semidefinite Penalty Comparison

Example:

Recover low-rank 100x100 matrix:

- Vary rank, proportion entries sampled
- Measure % time recover matrix to 1% MSE (more purple=better)
- Nuclear norm *by far* worst approach
- New penalty better, new penalty with rounding much better
- Combining greedy rounding with local search: solutions within 1% of optimality in practice, 10% in theory.



(a) New Penalty

(b) Nuclear Norm

(c) SVD+Local Improvement

(d) New Penalty+Local Improvement

- In practice, new penalty is viable and *often* more accurate

Matrix Completion Scalability

- Exact method: ***f(Y)*** evaluable by solving QP, scales to ***p=50***s.

n	p	γ	Time(s)	Nodes	Gap (%)	Cuts
20	0.1	$100/p$	1,266	8,792	0.87%	8.35
20	0.2	$100/p$	1,220	2,710	1.04%	7.80
20	0.3	$100/p$	1,272	1,837	0.64%	3.14

Sample solver-output from the exact method on matrix completion instances

Suggests improvement could come from dedicated branching strategies!

- Relax-then-round scales to ***p=1000***s of features

- On random matrix problems, both exact and approximate method supply lower out-of-sample MSE than state-of-the-art heuristics such as the Burer-Monterio method.

Conclusion

Mixed-Projection is a Natural Generalization of MIO

- Optimize convex function f(**z**) over non-convex $\{z: z^2=z\}$, derive tractable relaxations.
- Leads to algorithms which outperform state-of-the-art for central problems in OR/ML.
- Suggests this is a very general story, often useful to think about problems this way.

Two future directions:

1. Improve generality → extend framework to tensors, non-negative polynomials.
2. Improve scalability → custom solver which explicitly branches on proj. matrices

Thank you for listening!
Lingering questions? Email ryancw@mit.edu