On the integration of LP folding into Mosek

Erling D. Andersen and Michal Adamaszek

Mosek ApS, Fruebjergvej 3, 2700 Copenhagen, Denmark

2026-06-01

Introduction

The problem

The linear optimization problem:

$$
\begin{aligned}
\text{minimize } & c^T x \\
\text{subject to } & Ax = b, \\
& x \geq 0,
\end{aligned}
$$

where $A \in R^{m \times n}$.

The coresponding dual problem:

$$
\begin{aligned}
\text{minimize } & b^T y \\
\text{subject to } & A^T y + s = c, \\
& s \geq 0.
\end{aligned}
$$

Presolve

Presolve:

- Reduce the size of problem before using an optimization algorithm e.g. an interior-point method.
- Simple reductions (e.g. remove fixed variable, remove obviously redundant constraints) .
- Advanced reductions (e.g. eliminating free variables).
- See [1].

A recent presolve technique:

- Problem folding.
- Continuos version of symmetry reduction (mixed-integer case).
- Proposed in [2].

Folding

Trivial example

$$\begin{aligned} \text{minimize } & (c^1)^T x^1 + (c^2)^T x^2 \\ \text{subject to } & A^1 x^1 = b^1 \\ & A^2 x^2 = b^2 \\ & x^1, x^2 \geq 0 \end{aligned}$$

where $A^1 \in R^{m^1 \times n^1}$, $A^2 \in R^{m^2 \times n^2}$.

Assuming

$$c^1 = c^2, \quad A^1 = A^2 \;\text{ and }\; b^1 = b^2$$

then there is an optimal solution such that

Trivial example

$$x^1 = x^2.$$

The folding idea

Partioning

Define the disjoint row and column sets I_r and J_c where

$$
\begin{aligned}
\mathcal{I} &= \{I_1, \ldots, I_{\overline{m}}\} \\
\{1, \ldots, m\} &= \cup_r I_r \\
I_1 \cap I_2 &= \emptyset, \forall I_1, I_2 \in \mathcal{I} \\
\mathcal{J} &= \{J_1, \ldots, J_{\overline{n}}\} \\
\{1, \ldots, n\} &= \cup_c J_c \\
J_1 \cap J_2 &= \emptyset, \forall J_1, J_2 \in \mathcal{J}.
\end{aligned}
$$

The folding idea

Equitable partition

The pair partitions $(\mathcal{I}, \mathcal{J})$ is called equitable if any $I \in \mathcal{I}$ and $J \in \mathcal{J}$ it holds

- $\forall j_1, j_2 \in J$

$$c_{j_1} = c_{j_2}$$

- $\forall j_1, j_2 \in J$

$$e^T A_{I,j_1} = e^T A_{I,j_2}$$

- $\forall i_1, i_2 \in I$

$$A_{i_1,J} e = A_{i_2,J} e$$

The folding idea

- $\forall i_1, i_2 \in I$

$$b_{i_1} = b_{i_2}$$

e is the vector of all ones of an appropriate dimension.

The folding idea

The main theorem

Theorem 0.1: Given $(\mathcal{I}, \mathcal{J})$ is a equitable partition and (P) has an optimal solution, then there is optimal solution (x^*, y^*) satisfying

- For any J in $\mathcal{J}$:

$$x^*_{j_1} = x^*_{j_2}, \forall j_1, j_2 \in J.$$

- For any I in $\mathcal{I}$:

$$y^*_{i_1} = y^*_{i_2}, \forall i_1, i_2 \in I.$$

The folding idea

Aggregated problem

Using the primal aggregation

$$\overline{x}_c = \sum_{j \in J_c} \frac{x_j}{\mid J_c \mid}$$

and dual aggregation

$$\overline{y}_r = \sum_{i \in I_r} \frac{y_i}{\mid I_r \mid}.$$

the aggregated problem

The folding idea

$$
\begin{aligned}
\text{minimize } & \overline{c}^T \overline{x} \\
\text{subject to } & \overline{A}\overline{x} = \overline{b} \\
& \overline{x} \geq 0
\end{aligned}
$$

is obtained.

Observations:

- Solve the aggregated problem instead of te original problem.
- [3] discusses how to find an equitable partition.
- An optimal solution to the aggregated problem can easily be used to obtain a solution to the original problem.
- The basic solution case requires a potential expensive basis identification(=crossover) step.

The folding idea

- Farkas type infeasibility certificates can also be computed when the aggregated problem is primal or dual infeasible.

Implementation in Mosek

Sketch

- Presolve (limited if folding seems worthwhile).
- Fold if deemed **worthwhile**.
- Presolve again if folded.
- Optimize.
- Postsolve if folded.
- Unfold if folded.
- Do basis identification if required.
- Postsolve.

Computational results

Setup

- OS: Linux 64 bit for X86.
- Hardware: Intel(R) Xeon(R) CPU E5-2687W v4 @ 3.00GH (espe)
- Software: Mosek 12.0.0.alpha
- Parameter setting
  - Basis identification: on
  - 12 threads
  - All other paramters default except mentioned explicit.

Defintion:

$$\text{Folding ratio} = \frac{m_{\text{fold}} + n_{\text{fold}}}{m_{\text{unfold}} + n_{\text{unfold}}}.$$

Representative examples

Folding	Off	On	On		
	Optimizer	Optimizer	Folding	Unfold	Fold
Run	time(s)	time(s)	time(s)	time(s)	ratio
z43490_hit_simple	146.84	404.88	14.26	322.33	0.12
nug20	51.08	14.80	0.03	12.27	0.26
richard-payment	1025.79	927.33	12.37	110.00	0.48
boyd600	2.58	2.40	0.11	0.08	0.68

- Fold and unfold time are included in optimizer time.
- A large folding percentage may not imply a time reduction (z43490...).
  - BI can kill the benefit.
- Folding is sometimes highly beneficial (nug20).

Extensive computational results

- Test: No folding allowed versus folding allowed.
- Benchmark set: MIPLIB 2017 complete collection.
  - Removed approximatly 10 instances due to MPS reader issues e.g LAZYCONSTRAINTS.
  - Removed aproximately 8 instances because one of the runs failed e.g. timed out.
- Max time limit: 7200
- Table headers:
  - Min: Smallest time for instances in set.
  - Max: Largest time for instances in set.
  - Count: Number of instances
  - Total time: Optimizer time incl. presolve etc.

Extensive computational results

- GMST: Geometric mean of shifted time. (shift=0.01)
- GMRST: Geometric mean of ratio of shifted time.
- # wins: Number of wins (a win is within 5% of the best).

Fold	Min	Max	Count	Total time(s)	GMST	GMRST	# wins
No	0.00	4375.74	1052	45417.63	0.65	1.00	764
Yes	0.00	3824.17	1052	40008.32	0.55	0.84	908
No	0.00	0.98	610	119.35	0.11	1.00	462
Yes	0.00	0.99	610	105.69	0.10	0.90	499
No	0.97	55.75	376	3176.92	4.28	1.00	257
Yes	0.23	56.79	376	2635.53	3.21	0.75	349
No	13.05	4375.74	66	42121.37	277.18	1.00	45
Yes	8.54	3824.17	66	37267.10	232.80	0.84	60

Extensive computational results

Comments:

- Average folding factor is 0.78 (not shown in the table). Quite high!
- Folding leads to a significant improvement in the geometric mean of the shifted time ratios and wins.

Conclusion

- Trying folding is worthwhile by default.
  - Using MIPLIB instances as benchmark set overestimates the on average benefit though.
- The requirement of basis identification (=crossover) limits the value of folding.
- Folding will be on as default in Mosek v12.

Bibliography

[1] E. D. Andersen and K. D. Andersen, “Presolving in linear programming,” *Mathematical Programming*, vol. 71, no. 2, pp. 221–245, 1995.

[2] M. Grohe, K. Kersting, M. Mladenov, and E. Selman, "Dimension Reduction via Colour Refinement," in *Algorithms - ESA 2014*, A. S. Schulz and D. Wagner, Eds., Springer, pp. 505–516.

[3] R. Paige and R. E. Tarjan, "Three Partition Refinement Algorithms," *SIAM Journal on Computing*, vol. 16, no. 6, pp. 973–989, 1987, [Online]. Available: https://doi.org/10.1137/0216062