On recent improvements in MOSEK

Erling D. Andersen
MOSEK ApS,
Fruebjergvej 3, Box 16,
2100 Copenhagen,
Denmark.
WWW: http://www.mosek.com

August 28, 2012

Introduction

MOSEK

- MOSEK is software package for large scale optimization.
- Version 1 released April 1999.
- Version 6 released March 2009.
- Linear and conic quadratic (+ mixed-integer).
- Conic quadratic optimization (+ mixed-integer).
- Convex(functional) optimization.
- C, JAVA, .NET and Python APIs.
- AMPL, AIMMS, GAMS, MATLAB, and R interfaces.
- Free for academic use. See **http://www.mosek.com**.

New in MOSEK version 7

- Interior-point optimizer
  - Rewritten factor routines.
  - Handling of intersection cones in conic quadratic optimization.
  - Handling of semi-definite optimization problems.
- New mixed integer optimizer for conic problems.
- Fusion, a modeling tool for conic problems.

Note:

- Semi-definite talk: J. Dahl, Thursday at 11, Room H0110.

- Version 7 is in early beta mode.
- Only limited results.

The factorization step

MOSEK solves:

$$\begin{array}{lll} (P) \quad \min & c^T x & \\ \quad \text{st} & Ax & = \quad b, \\ & x \geq 0. & \end{array}$$

where A is **large** and **sparse** using a homogenous interior-point algorithm.
Each iterations requires solution of multiple:

$$\begin{bmatrix} -H^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} f^1 \\ f^2 \end{bmatrix} = \begin{bmatrix} r^1 \\ r^2 \end{bmatrix} \tag{1}$$

where H is a diagonal matrix.

- Known as the **augmented system**.

Normal equation system

Is reduced to:

$$AHA^T f_2 = \tag{2}$$

- The **normal equation system**!
- Identical to the augmented system approach using a particular pivot order.
- System is reordered to preserve sparsity using approximative min degree (AMD) or graph partitioning (GP).
- No numerical pivoting and the pivot order is kept **FIXED** over the iterations.
- Just one dense columns in A cause a lot of fill-in.
- Known since the early days of interior-point methods.

Dense columns

Dense columns handling

Methods for dealing with dense columns:

- Pre interior-point. Tearing. (Duff, Erisman and Reid [2])
- Augmented system. (Fourer and Mehrotra [3])
  - Pivoting for stability and sparsity. Not fixed.
  - Stable and low flops.
  - Practice: Slow.
- Sherman-Morrison-Woodbury. (Cheap but unstable). (Karmarkar + many more)
- Stabilized SMW. (Cheaper and OK stability). (Andersen [1])
- Product form Cholesky. (Costly but stable). (Goldfarb and Scheinberg [4]].

Idea of SMW

- Let $\bar{\mathcal{S}}$ be the index set of the sparse columns in A. And $\bar{\mathcal{N}}$ the index set of the dense ones.
- Solve a system with the matrix

$$\begin{bmatrix} A_{\bar{\mathcal{S}}} H_{\bar{\mathcal{S}}} A_{\bar{\mathcal{S}}}^T & A_{\bar{\mathcal{N}}} \\ A_{\bar{\mathcal{N}}}^T & -H_{\bar{\mathcal{N}}}^{-1} \end{bmatrix}$$

- Pivot order is fixed.
- Requires $A_{\bar{\mathcal{S}}} H_{\bar{\mathcal{S}}} A_{\bar{\mathcal{S}}}^T$ to be of full rank.
  - Leads numerical instability.
- Consequence: Minimize number of dense columns.

Dense columns detection

- Density is only a sufficient condition for problems.
- Troublesome columns may not be that dense at all.
- Most published methods uses a simple threshold of around 30. Fairly naive.
- Meszaros [5] discuss a sophisticated method.
  - Hybrid approach.
  - Augmented but with a fixed pivot order.
  - Similar to Vanderbei's [6] quasi definite approach with a sophisticated dense column detection.

Example: karted

Dens.	Num.
8	8
9	81
10	544
11	3782
12	17227
13	48321
14	62561
15	1
16	3
17	15
18	27
19	127
20	224
21	193

- Has dense columns!
- Which cutoff to use?

A new graph partitioning based approach

- Idea.
  - Try to emulate the optimal ordering for the augmented system.
  - Fixed pivot order.
  - Keep detection cost down.
- Solve a linear system of the form:

$$\begin{bmatrix} A_{\bar{\mathcal{S}}} H_{\bar{\mathcal{S}}} A_{\bar{\mathcal{S}}}^T & A_{\bar{\mathcal{N}}} \\ A_{\bar{\mathcal{N}}}^T & -H_{\bar{\mathcal{N}}}^{-1} \end{bmatrix}$$

Let $(\bar{\mathcal{S}}, \bar{\mathcal{N}})$ be an initial guess for the partition. And choose a reordering P so

$$P \begin{bmatrix} A_{\bar{\mathcal{S}}} H_{\bar{\mathcal{S}}} A_{\bar{\mathcal{S}}}^T & A_{\bar{\mathcal{N}}} \\ A_{\bar{\mathcal{N}}}^T & -H_{\bar{\mathcal{N}}}^{-1} \end{bmatrix} P^T = \begin{bmatrix} M_{11} & 0 & M_{13} \\ 0 & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}$$

- M_{11} and M_{22} should be of about identical size.
- M_{33} should be as small a possible.
- Ordering can be locate using graph partitioning i.e. use MeTIS or the like.
- Nodes that appears in both $\bar{\mathcal{N}}$ and M_{33} are the dense columns.
- A refined $(\bar{\mathcal{S}}, \bar{\mathcal{N}})$ is obtained.

Refinements of the basic idea

- If too many dense columns are located then assume no dense columns.
- Can be applied recursively on M_{11} and M_{22} blocks.
- Graph partitioning is potentially expensive.
  - Skip dense detection when safe.
  - Do not recurse too much.

Other improvements to factor routines

- New graph partitioning based ordering.
- Remove dependency on OpenMP for parallelization.
  - Use native threads.
  - OpenMP is a pain in a redistribution setting.
- Made it possible to use external sequential BLAS.
  - Exploit hardware advances such as Intel AVX easily.

Computational results

Dense column handling

- Comparison of MOSEK v6 and v7.
- Hardware: Linux 64 bit, Intel(R) Xeon(R) CPU E31270 @ 3.40GHz.
- Using 1 thread unless otherwise stated.
- Problems
  - Private and public test problems

Results for linear problems

Name	Time (s) v6	R. time v7/v6	Iter. v6	R. iter. v7/v6	Num. dense v6	Num. dense v7
bas1	8.27	0.66	10	0.82	5	40
difns8t4	8.28	1.62	27	1.07	74	89
L1_nine12	8.54	1.62	15	1.00	29	0
bienstock-310809-1	10.39	1.06	20	2.19	400	625
net12	11.14	0.37	42	0.63	544	545
gonnew16	13.42	2.69	37	1.00	246	329
GON8IO	13.85	0.65	27	0.96	73	278
ind3	15.05	1.16	12	1.00	3	185
15dec2008	19.06	0.39	21	0.95	175	287
pointlogic-210911-1	20.82	1.14	45	0.83	451	175
_time_horizon_minimiser	23.89	0.37	15	1.00	23	0
lt	29.26	0.52	46	0.53	292	506
dray17	31.07	0.43	77	1.67	55	447
ind2	46.26	1.45	12	1.23	318	970
zhao4	48.32	0.26	31	0.91	680	0
neos3	60.93	1.14	9	1.80	1	2
c3	84.82	1.81	9	1.10	57	0
avq1	110.40	0.72	12	1.15	538	1
TestA5	117.18	0.72	14	1.00	1	1
ml2010-rmine14	139.17	1.64	24	2.40	28	28
rusltplan	140.50	1.01	41	1.02	718	2094
tp-6	142.91	2.06	49	0.98	776	742
dray5	171.21	0.44	53	0.69	0	1203
stormG2_1000	188.52	0.44	107	0.49	119	119
degme	229.51	1.01	62	0.54	883	890
karted	242.50	1.21	20	0.95	193	590
friedlander-6	266.44	0.11	24	1.04	0	721
ts-palko	320.57	0.65	212	0.11	210	210
scipmsk1	325.59	1.01	17	1.28	749	1
160910-2	326.86	0.72	80	4.95	291	1893
gamshuge	1266.58	0.77	98	1.15	270	44
G. avg		0.79		0.99		

Comments

- Other changes contributes to difference.
  - New dualization heuristic.
  - Better programming, new compiler etc.
- Many dense columns in v7.
  - Does not affect stability much.
- New method seems to work well.
  - Can be relatively expensive for smallish problems.

Results for conic quadratic problems

Name	Time (s) v6	R. time v7/v6	Iter. v6	R. iter. v7/v6	Num. dense v6	Num. dense v7
ramsey3	0.78	1.47	12	1.00	199	212
050508-1	1.02	1.80	25	1.15	72	198
msprob3	1.06	0.90	32	1.21	73	31
041208-1	1.80	0.61	25	1.19	5	93
230608-1	3.87	0.92	62	0.84	7	750
280108-1	4.92	1.37	42	1.09	67	15
pcqo-250112-1	5.67	0.82	17	1.11	1176	1260
211107-1	12.68	1.08	50	0.88	0	1725
autooc	15.43	2.21	28	1.03	41	201
msci-p1to	16.41	0.11	24	1.16	42	433
bleyer-200312-1	34.10	0.23	11	1.42	0	1828
201107-3	35.67	0.46	50	1.14	0	199
G. avg		0.77		1.09		

Comment:

- Results is similar to the linear case.

Factor speed improvements

- Very limited and selective test.
- Only Mittlemans conic quadratic benchmark.

Name	Time (s) v6	R. time v7/v6	Iter. v6	R. iter. v7/v6
firL2L1alph	1.08	0.87	11	1.09
firL1Linfeps	4.85	0.56	57	0.91
firL2Linfeps	7.12	0.56	17	1.00
firL1	15.02	0.55	18	0.78
firL2L1eps	15.67	0.56	18	0.94
firL2Linfalph	21.92	0.25	20	0.70
firLinf	32.93	0.38	21	1.00
firL2a	53.68	0.14	9	0.56
firL1Linfalph	63.79	0.31	22	1.05
G. avg		0.41		0.87

- Large speed-up.
- Dense problems!

Summary and conclusions

- A new method for detecting dense columns has been presented.
- Seems to work well.
- MOSEK v7 performs much better than v6 on Mittlemans CQO benchmark.
- Slide are at http://mosek.com/resources/presentations/.

References

[1] K. D. Andersen. A Modified Schur Complement Method for Handling Dense Columns in Interior-Point Methods for Linear Programming. *ACM Trans. Math. Software*, 22(3):348–356, 1996.

[2] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct methods for sparse matrices*. Oxford University Press, New York, 1989.

[3] R. Fourer and S. Mehrotra. Solving symmetric indefinite systems in an interior point method for linear programming. *Math. Programming*, 62:15–40, 1993.

[4] D. Goldfarb and K. Scheinberg. A product-form cholesky factorization method for handling dense columns in interior point methods for linear programming. *Mathematical Programming*, 99:1–34, 2004. 10.1007/s10107-003-0377-7.

[5] Cs. Mészáros. Detecting “dense” columns in interior