



Introduction to Mosek

DTU CEE Summer School, 15 June 2017

Michał Adamaszek

www.mosek.com





- Convex conic optimization package
- Mixed-integer capability
- LP, QP, SOCP, SDP, MIP
- Low-level optimization API
 - C, Python, Java, .NET, Matlab, R, (Julia)
- Object-oriented API *Fusion*
 - C++, Python, Java, .NET, Matlab
- 3rd party
 - GAMS, AMPL, CVXOPT, CVXPY, YALMIP, PICOS, GPkit
- Conda package
- Upcoming v8.1





A conic problem in canonical form:

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax + b \in \mathcal{K} \end{array}$$

where \mathcal{K} is a product of cones:

- linear:

$$K = \mathbb{R}_{\geq 0}$$

- quadratic:

$$K = \{x \in \mathbb{R}^n : x_1 \geq \sqrt{x_2^2 + \dots + x_n^2}\}$$

- semidefinite:

$$K = \{X \in \mathbb{R}^{n \times n} : X^T = X, z^T X z \geq 0 \text{ for all } z \in \mathbb{R}^n\}$$

- (possibly: power cones, exponential cone)



Lots of functions and constraints are representable using the three major types of cones.

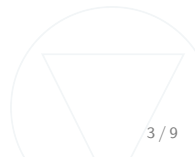
$$|x|, \|x\|_1, \|x\|_2, \|x\|_\infty, \|Ax + b\|_2 \leq c^T x + d$$

$$xy \geq z^2, x \geq \frac{1}{y}, x \geq y^{3/2}, x \geq y^{p/q}$$

$$t \leq (x_1 \cdots x_n)^{1/n}, t \leq n \left(\sum x_i^{-1} \right)^{-1}$$

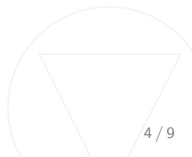
$$\det(X)^{1/n}, t \leq \lambda_{\min}(X), t \geq \lambda_{\max}(X)$$

$$\text{convex } (1/2)x^T Qx + c^T x + q$$





- primal simplex and dual simplex for linear problems
- primal-dual interior point method optimizer for all continuous problems
- automatic dualization of LPs and CQPs
- branch and bound and cut integer optimizer
- **exploits sparsity**



Example 1 (Dan)



$$\begin{aligned} \min_{\mathbf{W} \in \mathbb{R}^{3 \times 3}} \quad & \mathbf{W}_{11}^2 + \mathbf{W}_{22}^2 + \mathbf{W}_{33}^2 \\ \text{s.t.} \quad & 2\mathbf{W}_{12} + \mathbf{W}_{23} = 3 \\ & \mathbf{W}_{22} + 3\mathbf{W}_{13} \geq 7 \\ & \mathbf{W} \succeq 0 \end{aligned}$$

```
from mosek.fusion import *
M = Model()

W = M.variable(3, Domain.inPSDCone())
M.constraint(Expr.add(Expr.mul(2.0, W.index(0,1)), W.index(1,2)),
             Domain.equalsTo(3.0))
M.constraint(Expr.add(W.index(1,1), Expr.mul(3.0, W.index(0,2))),
             Domain.greaterThan(7.0))

t = M.variable()
M.constraint(Var.vstack(t, W.diag()), Domain.inQCone())
M.objective(ObjectiveSense.Minimize, t)
```



Example 1 (Dan)



```
M.solve()
```

```
w = numpy.reshape(W.level(), [3,3])
```

```
print "Eigvals:", sorted(numpy.linalg.eigvals(w))
```

```
[ ... ]
```

```
Optimizer - solved problem      : the primal
```

```
Optimizer - Constraints         : 5
```

```
Optimizer - Cones              : 1
```

```
Optimizer - Scalar variables   : 5          conic          : 4
```

```
Optimizer - Semi-definite variables: 1      scalarized       : 6
```

```
[ ... ]
```

```
ITE PFEAS   DFEAS   GFEAS   PRSTATUS   POBJ           DOBJ           MU           TIME
```

```
0  2.8e+00  1.0e+00  1.5e+00  0.00e+00  2.000000000e+00  0.000000000e+00  1.0e+00  0.00
```

```
1  6.9e-01  2.5e-01  4.0e-01  -4.78e-01  5.117111381e+00  6.046395705e+00  2.5e-01  0.00
```

```
[ ... ]
```

```
7  5.2e-08  1.9e-08  2.2e-04  1.00e+00  2.984810208e+00  2.984810149e+00  1.9e-08  0.00
```

```
8  2.5e-09  9.2e-10  4.9e-05  1.00e+00  2.984810039e+00  2.984810036e+00  9.2e-10  0.00
```

```
Optimizer terminated. Time: 0.01
```

```
Interior-point solution summary
```

```
Problem status : PRIMAL_AND_DUAL_FEASIBLE
```

```
Solution status : OPTIMAL
```

```
Primal.  obj: 2.9848100389e+00   nrm: 7e+00   Viol.  con: 3e-09   var: 0e+00   barvar: 0e+00   con
```

```
Dual.    obj: 2.9848100360e+00   nrm: 1e+00   Viol.  con: 6e-16   var: 1e-09   barvar: 2e-09   con
```

```
Eigvals: [9.488456837149494e-09, 0.64285538829761235, 4.4480504056675141]
```



$$\begin{aligned} \min \quad & \sum c_i P_{G_i} \\ \text{s.t.} \quad & P_{G_i}^{\min} \leq P_{G_i} \leq P_{G_i}^{\max} \\ & \mathbf{B} \cdot \boldsymbol{\theta} = \mathbf{P}_G - \mathbf{P}_D \\ & (\theta_i - \theta_j) / x_{ij} \leq P_{ij, \max} \end{aligned}$$

```
def dcopf(ng, nb, PGmin, PGmax, Pmax, B, x, PD, c):
    M = Model()
    PG = M.variable(ng, Domain.inRange(PGmin, PGmax))
    theta = M.variable(nb)

    M.constraint(Expr.sub(Expr.mul(B, theta), Expr.sub(PG, PD)),
                 Domain.equalsTo(0.0))
    M.constraint(Expr.sub(Var.hrepeat(theta, nb),
                          Var.vrepeat(theta.transpose(), nb)),
                 Domain.lessThan(Pmax*x))

    M.objective(ObjectiveSense.Minimize, Expr.dot(c, PG))
```




A basic version of the Unit Commitment Problem with:

- quadratic generation costs,
- ramp constraints,
- minimal uptime and downtime,
- startup costs,

as an example of MISOCP. See:

- <https://mosek.com/resources/doc>
 - The MOSEK notebook collection
 - Unit commitment





Licensing:

- Trial 30-day license
- Personal academic license
- Group academic license
- Commercial licenses

More:

- www.mosek.com
- support@mosek.com

Thank you!