Second-order Cone Programming (SOCP)

Second-order cone programming (SOCP) offers robust and efficient way of solving several types of convex problems, such as convex quadratically constrained quadratic programming (QCQP), robust linear programming (LP), parameter fitting and various norm-related optimization problems. That is the reason why it can be highly utilized in a broad range of fields including finance, engineering and control. NAG introduces a new SOCP solver at Mark 27 of the NAG Library based on interior point method (IPM).

1 Introduction

Second-order cone programming is convex optimization in which a linear function is minimized subject to linear constraints and the intersection of second-order (Lorentz or the ice cream) cones. In contrast to LP, second-order cones allow users to bring curvature information into the model to solve more complicated problems. The standard form SOCP problem is

$$\begin{array}{ll} \underset{x\in\mathbb{R}^n}{\text{minimize}} & c^T x \\ \text{subject to} & l_A \leq Ax \leq u_A, \\ & l_x \leq x \leq u_x, \\ & x \in \mathcal{K}, \end{array} \tag{1}$$

where $A \in \mathbb{R}^{m\times n}$, $l_A, u_A \in \mathbb{R}^m$, $c, l_x, u_x \in \mathbb{R}^n$ are the problem data, and $\mathcal{K} = \mathcal{K}^{n_1} \times \cdots \times \mathcal{K}^{n_r} \times \mathbb{R}^{n_l}$ where $\mathcal{K}^{n_i}$ is either a second-order cone

$$\mathcal{K}_q^{n_i} := \left\{ x = (x_1, \ldots, x_{n_i}) \in \mathbb{R}^{n_i} \;:\; x_1^2 \geq \sum_{j=2}^{n_i} x_j^2,\; x_1 \geq 0 \right\}, \tag{2}$$

or a rotated second-order cone

$$\mathcal{K}_r^{n_i} := \left\{ x = (x_1, \ldots, x_{n_i}) \in \mathbb{R}^{n_i} \;:\; 2x_1x_2 \geq \sum_{j=3}^{n_i} x_j^2,\; x_1 \geq 0,\; x_2 \geq 0 \right\}. \tag{3}$$

As shown in Figure 1, the feasible region of an SOCP problem that has 3 variables is the intersection of the green polyhedron (corresponding to the linear inequality and bound constraints) and a cone. The curve represents the quadratic information added which makes SOCP more complicated yet powerful at the same time than LP.

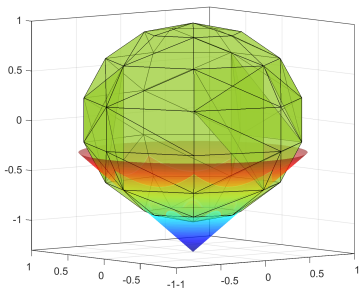

Figure 1: Feasible region of an SOCP problem with 3 variables

Second-order cone programming has been well-researched since 1990s and the most popular methods to solve it may be the interior point methods due to their theoretical polynomial complexity and practical performance. The main computational cost of IPM comes from solving a (potentially sparse) linear system for computing the Newton search direction. For more theoretical background and results, see [1, 2, 3]. NAG implements a path-following homogeneous self-dual algorithm that is both robust and efficient and is able to detect infeasibility and unboundedness of the model. Also NAG exploits the sparsity pattern of the problem thoroughly which makes the solver efficient especially on large and sparse problems. For more details on implementation, see the documentation of NAG SOCP solver [4].

2 Reformulation examples

It is rare that second-order cone (SOC) would appear naturally in the model and is more frequent that a reformulation is needed. Here we show only the most common cases, for more examples see [5].

2.1 Optimization with norms

Several problems including data/parameter fitting will use norms in the objective function or as a constraint. Here we mention three cases that might appear frequently in models from various applications:

- l_1-norm:
$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \leq t \quad \Longleftrightarrow \quad \sum_{i=1}^{n} s_i \leq t, \;\; (s_i, x_i) \in \mathcal{K}_q^2, \;\; i = 1, \ldots, n.$$
- l_2-norm:
$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2} \leq t \quad \Longleftrightarrow \quad (t, x) \in \mathcal{K}_q^{n+1}.$$
- l_∞-norm:
$$\|x\|_\infty = \max\{|x_i| : i = 1, \ldots, n\} \leq t \quad \Longleftrightarrow \quad (t, x_i) \in \mathcal{K}_q^2, \;\; i = 1, \ldots, n.$$

The above norms can be viewed as special cases of general p-norm constraint on a vector $x \in \mathbb{R}^n$ defined as

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \leq t, \tag{4}$$

where $p = l/m$ is a positive rational number with $l \geq m$. Inequality constraints (4) have second-order cone representation since they are equivalent to inequalities involving rational powers. Details on how to transform a general p-norm constraint can be found in [2].

2.2 Quadratic constraint

The convex inequality

$$\frac{1}{2} x^T P x + q^T x + r \leq 0 \tag{5}$$

where $P \in \mathcal{S}^n$ is positive semidefinite matrix arises from applications such as portfolio optimization. Assume the matrix P has factorization $P = F^T F$, (5) has an equivalent second-order cone representation as

$$t + q^T x + r = 0, \;\; (t, 1, Fx) \in K_r^{n+2}$$

since $(1/2)x^T P x \leq t$ is equivalent to $\|Fx\|_2^2 \leq 2t$. Thus, adding auxiliary variables $y = Fx$ will transform the quadratic constraint into a standard cone constraint.

2.3 Second-order cone representable functions

We have mentioned several important SOC-representable functions above. Many more functions and sets such as hyperbolic constraints, harmonic mean of positive affine functions, sum of quadratic/linear fractions *etc.* are all SOC-representable [5]. In general, if functions $f_1(x)$ and $f_2(x)$ are SOC-representable, then $\alpha f_1(x)$ $(\alpha \geq 0)$, $f_1(x) + f_2(x)$ and $max\{f_1(x), f_2(x)\}$ are SOC-representable as well. We encourage users to exploit their models thoroughly to reach a SOC-equivalent problem, therefore make the most of the versatility and practical performance of the NAG SOCP solver.

3 Performance of NAG SOCP solver

In this section, we show the performance of NAG SOCP solver together with two state-of-the-art convex optimization solvers SEDUMI [3] and SDPT3 [6], which are general solvers for linear programming, second-order cone programming and semidefinite programming. We also solved the same models with NAG general nonlinear programming (NLP) solver since SOCP can be viewed also as a general NLP problem. All four solvers were at default settings and the tests were run on a Windows laptop of 16GB memory and Intel(R) Core(TM) i7-7500 2.7GHz CPU in single-threaded mode.

For SEDUMI and SDPT3, tests were run in MATLAB since both solvers are written in MATLAB. For NAG solvers we used NAG Python library to conduct the tests, however, the same solver is available in many other environments (C, Java, Fortran, *etc.*). The test data comes from 18 DIMACS problems that are widely used to test convex optimization solvers. A general problem statistics can be found in Table 1. The problem data is in SEDUMI format but can be transformed into SDPT3 and NAG input format easily.

Prob. class	No. of prob.	Avg. n	Avg. m	Avg. nc
nb	4	3098.75	321	906
nql	3	86102	49440	12300
qssp	3	99486	49471	24871
sched	8	17712.5	8448.5	1.5

Table 1: DIMACS problem statistics. 18 test problems in 4 classes. (n number of variables, m number of linear constraints, nc number of quadratic cone constraints.)

As shown in Table 2, NAG SOCP solver managed to solve all the problems to the required accuracy (10^{-8}) while other solvers failed on some of the test cases.

	NAG SOCP	SEDUMI	SDPT3	NAG NLP (IPM)
Problems solved	100%	77.78%	83.33%	94.44%

Table 2: Statistics on solvers' status after run, percentage of successful solve attempts.

The computational time comparison can be found in Figure 2. We picked 11 problems that all four solvers solved successfully and Table 3 shows the problem ID and its corresponding problem name in the dataset. As shown in Figure 2, even general NLP solvers could solve SOCP in some cases, it is still recommended to use specialized solver for SOCP to gain maximum performance.

Problem ID	Problem name
1	nb
2	nb_L1
3	nql30
4	nql60
5	qssp30
6	qssp60
7	sched_100_100_scaled
8	sched_100_50_scaled
9	sched_200_100_scaled
10	sched_50_50_orig
11	sched_50_50_scaled

Table 3: Indexing of problem ID in the test dataset

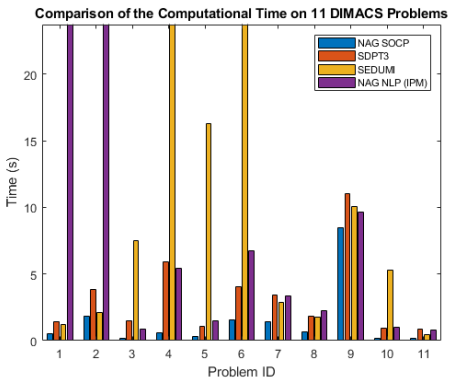

Figure 2: Execution time results

In conclusion, SOCP is widely used in a broad range of applications due to its powerful nature. NAG introduces, at Mark 27, a new robust and efficient SOCP solver that should be considered for problems with intrinsic quadratic information. Certain reformulation is required, yet it is worth the effort.

References

[1] F. Alizadeh and S. Schmieta, "Optimization with semidefinite, quadratic and linear constraints," in *RUTCOR, RUTGERS UNIVERSITY*. Citeseer, 1997.

[2] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Mathematical programming*, vol. 95, no. 1, pp. 3–51, 2003.

[3] J. F. Sturm, "Implementation of interior point methods for mixed semidefinite and second order cone optimization problems," *Optimization Methods and Software*, vol. 17, no. 6, pp. 1105–1154, 2002.

[4] NAG, "E04PT documentation," https://www.nag.co.uk/numeric/nl/nagdoc_27/flhtml/e04/e04ptf.html.

[5] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear algebra and its applications*, vol. 284, no. 1-3, pp. 193–228, 1998.

[6] K.-C. Toh, M. J. Todd, and R. H. Tütüncü, "On the implementation and usage of sdpt3–a matlab software package for semidefinite-quadratic-linear programming, version 4.0," in *Handbook on semidefinite, conic and polynomial optimization*. Springer, 2012, pp. 715–754.