Semidefinite Programming (SDP)

Mark 26 presents the first version of a linear and nonlinear *semidefinite programming (SDP)* solver in the NAG Library. This is a very brief overview of SDP and its applications, for further details refer to [1], [2] and [3].

Introduction

Linear semidefinite programming can be viewed as a generalization of linear programming. While keeping many good properties of LP (such as the duality theory and solvability in polynomial time), SDP introduces a new highly nonlinear type of constraint – *matrix inequality*. It is an inequality on the eigenvalues of a matrix which depends on the decision variables. Typically, the matrix inequality is written in the form to request all eigenvalues of the matrix to be non-negative, thus the matrix is to be positive semidefinite. Mathematically linear SDP might be written as a minimization (or maximization) of a linear objective function subject to linear matrix inequalities and possibly standard linear constraints and box bounds:

$$\begin{array}{ll} \min\limits_{x} & c^T x \\ \text{subject to} & \sum\limits_{i=1}^{n} x_i A_i^k + A_0^k \succcurlyeq 0, \quad k = 1, \dots, m_A \\ & l_B \leq Bx \leq u_B \\ & l_x \leq x \leq u_x \end{array}$$

Here the first constraint is the matrix inequality.

The new solver is based on a generalized Augmented Lagrangian method (see [4]) and as such allows extensions even to nonlinear (possibly nonconvex) SDP. The current release can deal with a formulation where both the objective and the matrix constraints are (possibly nonconvex) quadratic. To our knowledge this is the only nonlinear SDP solver offered commercially. The problem is formulated as follows:

$$\begin{array}{ll} \min\limits_{x} & c^T x + \frac{1}{2} x^T H x \\ \text{subject to} & \sum\limits_{i,j=1}^{n} x_i x_j Q_{ij}^k + \sum\limits_{i=1}^{n} x_i A_i^k + A_0^k \succcurlyeq 0, \quad k = 1, \dots, m_A \\ & l_B \leq Bx \leq u_B \\ & l_x \leq x \leq u_x \end{array}$$

Note that due to historical reasons such problems are often referred to as *bilinear matrix inequality (BMI)*.

Applications

It turns out that the matrix inequality is a very powerful instrument and various constraints in many diverse fields can be formed or approximated that way. Although the matrix constraints appear naturally in some applications a vast majority of applications for SDP come from a reformulation or relaxation of the original problem as demonstrated in examples accompanying the solver.

The **Nearest Correlation Matrix** problem is an example where the matrix constraint is formulated directly. In this problem the task is to find a correlation matrix (a symmetric positive semidefinite matrix with the unit diagonal) which is closest to the input matrix in some measure, such as in Frobenius norm. Despite the many specialized solvers for this problem in the G02 Chapter, a formulation via SDP might be appropriate if additional constraints are imposed on the new correlation matrix, for example, a certain sparsity structure, bounds on individual elements or condition number of the matrix.

Similarly, **eigenvalue optimization** can be directly written as a matrix constraint stated above. This includes problems such as minimization of the maximal eigenvalue or the condition number of the matrix. These problems appear in **engineering** and **structural optimization** where stability constraint translates into a bound for eigenvalues.

Many linear SDPs and BMIs originate in **control and system theory** as the stability of a linear differential equation can be expressed as a matrix inequality due to the Lyapunov theory. Imagine a simple linear system $\dot{x} = Ax$. It is stable if and only if there exist a Lyapunov function $V(x) = x^T P x$ with P symmetric positive definite such that $V(x)$ is decreasing (the time derivative of $V(x)$ is negative) for all $x \neq 0$:

$$\frac{d}{dt} V(x) = \dot{x}^T P x + x^T P \dot{x} = x^T (A^T P + PA) x < 0, \forall x \neq 0$$

which translates to a matrix inequality $A^T P + PA \prec 0$ with a matrix variable $P \succ 0$. Thus solving an SDP problem can be used to test if the system is stable. In addition, if the system matrix depends on certain control variables, the problem can be stated as how to choose the control variables so that the system remains stable.

Combinatorial optimization is a great source of problems (even NP-hard) which can be relaxed and approximated via SDP in polynomial time. For example, Lovász theta function of a graph gives a lower and upper bound on the chromatic number and the clique number of the complement of the graph, respectively. Despite these are both NP-complete, Lovász theta function can be formulated as an eigenvalue maximization problem and solved via SDP. Another nice example is the max-cut problem. It can be written as an integer quadratic problem with all variables $x_i \in \{-1, +1\}$. The trick of the relaxation is that the integrality constraint on the variables can be equivalently reformulated as a nonconvex quadratic constraint $x_i^2 = 1$. In turn this can be expressed via a matrix $X = xx^T$ requesting a unit diagonal $X_{ii} = 1$ or equivalently, matrix X such as $X_{ii} = 1$ should have rank one and be positive semidefinite. If the (nonconvex) constraint that the matrix is rank one is omitted, we have a relaxation of the original problem which is solvable as SDP.

Further fields with SDP problems include chemical engineering, statistics, pattern recognition, polynomial optimization, robust optimization and others.

The Solver

The new solver is named *e04sv* and it is located within the E04 Chapter of the NAG Library. It is a part of the *NAG Optimization Modelling Suite* which significantly simplifies the interface of the solver and related routines.

References

[1] Todd M J (2001) Semidefinite Optimization *Acta Numerica* **10** 515-560

[2] Vandenberghe L and Boyd S (1996) Semidefinite Programming *SIAM Review* **38** 49-95

[3] Wolkowicz H., Saigal R. and Vandenberghe L (2000) *Handbook on Semidefinite Programming: Theory, Algorithms and Applications* Kluwer Academic

[4] Kočvara M and Stingl M (2003) PENNON – a code for convex nonlinear and semidefinite programming *Optimization Methods and Software* **18(3)** 217-333