CS 229T Paper Review

Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embeddings

Paper by O'Donoghue, et al.
Review by Akshay Agrawal
Stanford University
May 2017
`akshayka@cs.stanford.edu`

Abstract

These notes review and comment upon a first-order method for solving large convex cone programs, first introduced by O'Donoghue, et al. in [7]. The method is available as a software package dubbed SCS (Splitting Conic Solver), and the convex programming domain specific languages CVXPY and Convex.jl employ it by default when solving semidefinite programs. We will refer to both the software package and the mathematical method as SCS, as it will be clear from context which of the two we are speaking of.

SCS reduces cone programs to convex feasibility problems. For an arbitrary cone program, the associated feasibility problem is a homogeneous self-dual embedding of the cone program's optimality conditions, hence the title of the paper. SCS uses the alternating direction method of multipliers (ADMM) to solve the feasibility problem, but *any* algorithm for the feasibility problem can be used to solve the embedding.

In this review, we briefly present the paper's primary contributions, surfacing and discussing assumptions made by its authors (for example, the assumption that strong duality holds for the cone program to be solved). Note that we will at times suppress technical details when they obscure intuition, and will more generally attend our focus selectively, so of course those readers who are looking for a complete treatment of the topic should read instead or in addition to these notes the full paper. We then highlight some of its limitations; in particular, the authors did not benchmark ADMM against other first-order methods for convex feasibility problems, which is problematic because ADMM converges sub-linearly under general assumptions [5]. Finally, we close by proposing a method to accelerate a classical algorithm for the feasibility problem. Indeed, SCS provides the primary motivation for our research into this method.

1 Problem Statement: Solving Cone Programs

SCS is a first-order convex optimization method for cone programs. It is noteworthy because of its generality and its scalability: it is not tied to any particular cone, and, because it is a first-order method, it scales to large problems.

The input to SCS is a cone program of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax + s = b \\ & (x, s) \in \mathbb{R}^n \times \mathcal{K}, \end{array} \tag{1.1}$$

where x and s are the optimizationv ariables and $\mathcal{K}$ is a nonempty, closed, convex cone. We refer to (1.1) as the *primal* problem. A dual of the primal problem can be formulated as

$$\begin{array}{ll} \text{maximize} & -b^T y \\ \text{subject to} & -A^T y + r = c \\ & (r, y) \in \{0\}^n \times \mathcal{K}^*, \end{array} \tag{1.2}$$

with variables r and y.

The output returned by SCS falls into one of the following three categories.

- A primal-dual optimal tuple $(x^\star, y^\star, s^\star)$. If strong duality obtains and the primal (or dual) problem is feasible, then SCS is guaranteed to return such a tuple.
- A certificate of primal or dual infeasibility. If strong duality obtains and the primal or dual is infeasible, then SCS is guaranteed to return such a certificate.
- An indeterminate status. This is a pathological scenario that may happen when strong duality does not obtain.

Note how the value returned depends upon whether strong duality obtains. Indeed, for reasons that will be made clear in a later section, **SCS assumes that strong duality obtains** for the problem (1.1). We will discuss this assumption further in the next section.

A remark: though it may not seem to be the case at first glance, the primal formulation (1.1) is *incredibly expressive.* Indeed, most practical cone programs, including those involving the second-order, semidefinite, exponential, and power cones, can be canonicalized to this form. The paper in discussion provides some examples of some such programs, including ℓ_1-regularized logistic regression and robust principal components analysis [7]. For more details about the canonicalization process, we refer readers to the CVXPY paper [4] or the more classical paper on disciplined convex programming and graph implementations for nonsmooth convex programs [6]. The bottom line is that modern domain specific languages allow us to write convex cone programs in their most natural forms, as the software handles the canonicalization to the primal problem (1.1) for us.

1.1 Derivation of the Dual Problem

For a pedagogical purpose, we derive the dual problem (1.2) using the calculus detailed in [3, Sect. 5.9]; [7] omits this derivation.

The Lagrangian of the primal problem is

$$\begin{aligned}\mathcal{L}(x,s,r,y) &= c^T x - r^T x - y^T s \\ &= c^T x - r^T x + y^T(Ax - b) \\ &= (A^T y + c - r)^T x - b^T y.\end{aligned}$$

The infimum over x of the Lagrangian is bounded above negative infinity only if $A^T y + c - r$ is zero, giving rise to the problem (1.2).

2 A Reduction to Convex Feasibility, Solved via ADMM

In this section, we describe and comment upon the method by which SCS solves cone programs. The main result is a reduction from a cone program for which strong duality obtains to a convex feasibility problem; it is this latter problem that SCS solves.

2.1 The KKT system

If strong duality obtains, then the KKT conditions below are both necessary and sufficient for optimality for problem (1.1):

$$Ax^\star + s^\star = b, \quad s^\star \in \mathcal{K}, \quad A^T y^\star + c = r^\star, \quad r^\star = 0, \quad y^\star \in \mathcal{K}^*, \quad c^T x + b^T y = 0. \tag{2.1}$$

Indeed, note that a tuple (x, y, r, s) satisfies (2.1) if and only if it lies in the intersection of an affine set and a convex cone. In this sense, (2.1) is a *convex feasibility problem*, defined as a problem in which the input is some number of convex sets and the output is any point in the intersection of the sets, if one exists, or an indication that no such point exists.

If all the authors wished to do were to obtain solutions to feasible primal problems, then they could simply stop here and apply their algorithm of choice to the KKT system. But note that if the primal were in fact infeasible, then applying an iterative algorithm to solve (2.1) would be unsatisfactory, in the following sense: the KKT system does not admit a principled way to obtain a *certificate of infeasibility* from it. From a practical perspective, it is valuable to know when a specified problem is infeasible, as it means that we are asking for the impossible and should carefully reconsider our objective and constraints.

2.2 Handling Infeasibility: A Homogeneous Self-Dual Embedding

In order to handle infeasibility, SCS solves an augmentation of (2.1), which happens to be a *self-dual and homogeneous* problem. The self-dual and homogeneous embedding of a problem was first introduced in [8], and a variety of methods target this embedding already, so the authors use of it is not a new one. What is new is that the authors use a first-order method to solve the embedding, which had not been attempted previous to the publication of SCS.

The details of the embedding are technical and not particularly enlightening, so we omit them here. Suffice it to say that the embedding introduces two nonnegative variables, κ and τ, whose values at a solution are used to either (1) recover an optimal solution to (1.1), (2) produce a certificate of infeasibility, or (3) if a nonzero solution does not exist, then return an indeterminate status. In particular, the embedding produces the convex feasibility problem

$$\begin{array}{ll} \text{find} & (u, v) \\ \text{subject to} & v = Qu \\ & (u, v) \in \mathcal{C} \times \mathcal{C}^* \end{array}, \tag{2.2}$$

where

$$u = \begin{bmatrix} x \\ y \\ \tau \end{bmatrix}, \quad v = \begin{bmatrix} r \\ s \\ \kappa \end{bmatrix}, \quad Q = \begin{bmatrix} 0 & A^T & c \\ -A & 0 & b \\ -c^T & -b^T & 0 \end{bmatrix},$$

and $\mathcal{C} = \mathbb{R}^n \times \mathcal{K}^* \times \mathbb{R}_+$. Note how closely (2.2) resembles (2.1).

The key is to find a *nonzero* solution to (2.2) with at least one of $\tau > 0$ or $\kappa > 0$, if such a solution exists; if $\tau > 0$, then case (1) is triggered, and if $\kappa > 0$, then case (2) is triggered. If no such solution exists, then the pathological case (3) is triggered. See Section 3.1 for a discussion about when the pathological case may occur; the authors are not clear about this, so we attempt to be precise.

2.3 Solving the Embedding via ADMM

The authors employ ADMM (see, e.g., [2] for a survey) to solve the self-dual embedding (2.2). The application is straightforward, and the primary result is that, by choosing the initial iterate carefully, it is possible to guarantee that if a nonzero solution to (2.2) exists, then it will be found. The authors assume that either τ or κ is non-zero at a solution, an assumption that we will discuss in Section 3.2. The key facts used in the proof are the nonexpansivity of ADMM and the homogeneity of the problem, and the key technique used is simply Cauchy-Schwarz.

The upshot is that, so long as a nonzero solution exists to (2.2), ADMM will find it (or some other nonzero solution). Because ADMM is a first-order method, this means that the authors' work enables us to solve very large cone programs. For example, they solve a regularized logistic regression problem with almost a billion nonzeros in the data matrix in just a few hours [7]; interior-point methods simply cannot scale to such sizes.

3 An Analysis of Stated Assumptions

In this section, we attend our focus to a couple of strong assumptions that the authors impose upon their problem instances, teasing out their subtleties and implications.

3.1 Strong Duality Obtains

For the entirety of the paper, the authors assume that strong duality obtains for the primal problem (1.1) [7]. This assumption is not justified nor analyzed; indeed, it is certainly possible to invoke the software package SCS with a problem instance that does not obtain strong duality. What goes wrong, if anything at all, when we don't have strong duality?

If strong duality is not obtained, then the KKT conditions (2.1), while sufficient for optimality of (1.1), are not necessary [3, Sect. 5.9]. Therefore, at least in theory, if we do not have strong duality, it is possible that the primal problem is feasible, but that its solutions do not satisfy the KKT conditions. If this occurs, then SCS will fail to find a solution. In particular, it will return an indeterminate status. The fact that SCS will not erroneously produce a certificate of infeasibility means that it is acceptable to apply it to problems that do not have strong duality.

3.2 A Nonzero Solution Exists

The assumption that a nonzero solution to (2.2) exists is tied to the assumption that strong duality obtains for (1.1).

If strong duality obtains, then exactly one of three scenarios will hold to be true: the primal problem is feasible, the primal is infeasible and the dual problem is unbounded above, or the primal is unbounded below and the dual is infeasible. In particular, because (2.2) was constructed to encode two theorems of strong alternatives ([7, Sect. 2.2]), it must be the case that there exists a solution with either $\tau > 0$ or $\kappa > 0$. This is true because for each theorem of strong alternatives, exactly one set will be nonempty. If the sets for primal and dual feasibility are non-empty, then a point in that set can be encoded as a solution to (2.2) with $\tau > 0$; otherwise, if the set for primal infeasibility or dual infeasibility is non-empty, then a point in either of those sets can be encoded as a solution to (2.2) with $\kappa > 0$.

4 Limitations

An immediately obvious limitation is that SCS assumes that strong duality obtains for the primal problem. It is unclear to us how often strong duality fails to obtain in practice, and the authors do not address this question. Moreover, when strong duality does not obtain, the authors do not further characterize the scenarios in which the KKT system may nonetheless admit a solution to the problem. It is stated informally in [3, Sect. 5.2.3] that strong duality usually obtains for convex problems, so perhaps this limitation is not actually a limitation in practice.

Another limitation is comes from the choice to use ADMM, a first-order method, to solve the convex feasibility problem. The authors consciously trade-off the rate of convergence and the accuracy of solutions obtained in favor of the capability to solve very large cone programs. Indeed, under the most general assumptions, ADMM converges sub-linearly [5].

5 Future Directions

The homogeneous, self-dual embedding provides compelling motivation for algorithms that solve the convex feasibility problem, and SCS in particular motivates the use of first-order algorithms. A future direction that we are interested in is exploring whether it is possible to accelerate first-order algorithms like ADMM or even the classical alternating projections algorithm [1], due to von Neumann, by interleaving a series of small, cheap-to-solve quadratic programs into them. Such an algorithm would, in the ideal, retain the scalability properties of first-order methods while interpolating between the computational complexity and convergence rates of first-order and interior-point methods, where the interpolations would be tuned by making the quadratic programs more or less expressive. The character of these interpolations is an open question, one that we plan to explore for the final project of this course.

References

[1] Heinz H Bauschke and Jonathan M Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996.

[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[4] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[5] Jonathan Eckstein. *Splitting methods for monotone operators with applications to parallel optimization.* PhD thesis, Massachusetts Institute of Technology, 1989.

[6] Michael C Grant and Stephen P Boyd. Graph implementations for nonsmooth convex programs. In *Recent advances in learning and control*, pages 95–110. Springer, 2008.

[7] Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 2016.

[8] Yinyu Ye, Michael J Todd, and Shinji Mizuno. An o($\sqrt{nL}$)-iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research*, 19(1):53–67, 1994.