An Efficient Re-scaled Perceptron Algorithm for Conic Systems

Alexandre Belloni[1,2], Robert M. Freund[1], and Santosh S. Vempala[1,3]

[1] MIT
[2] IBM
[3] Georgia Tech
belloni@mit.edu, rfreund@mit.edu, vempala@cc.gatech.edu

Abstract. The classical perceptron algorithm is an elementary algorithm for solving a homogeneous linear inequality system $Ax > 0$, with many important applications in learning theory (e.g., [11,8]). A natural condition measure associated with this algorithm is the Euclidean width τ of the cone of feasible solutions, and the iteration complexity of the perceptron algorithm is bounded by $1/\tau^2$. Dunagan and Vempala [5] have developed a re-scaled version of the perceptron algorithm with an improved complexity of $O(n \ln(1/\tau))$ iterations (with high probability), which is theoretically efficient in τ, and in particular is polynomial-time in the bit-length model. We explore extensions of the concepts of these perceptron methods to the general homogeneous conic system $Ax \in \mathbf{int}\ K$ where K is a regular convex cone. We provide a conic extension of the re-scaled perceptron algorithm based on the notion of a *deep-separation oracle* of a cone, which essentially computes a certificate of strong separation. We give a general condition under which the re-scaled perceptron algorithm is theoretically efficient, i.e., polynomial-time; this includes the cases when K is the cross-product of half-spaces, second-order cones, and the positive semi-definite cone.

1 Introduction

We consider the problem of computing a solution of the following conic system

$$\begin{cases} Ax \in \mathbf{int}\ K \\ x \in X \end{cases} \tag{1}$$

where X and Y are n- and m-dimensional Euclidean subspaces, respectively, $A : X \to Y$ is a linear operator and $K \subset Y$ is a regular closed convex cone. We refer to this problem as the "conic inclusion" problem, we call K the *inclusion cone* and we call $\mathcal{F} := \{x \in X : Ax \in K\}$ the *feasibility cone*. The goal is to compute an interior element of the feasibility cone $\mathcal{F}$. Important special cases of this format include feasibility problem instances for linear programming (LP), second-order cone programming (SOCP) and positive semi-definite programming (SDP). These problems are often encountered in learning theory, e.g., to learn threshold functions and in support vector machines, to mention two well-known examples.

N. Bshouty and C. Gentile (Eds.): COLT 2007, LNAI 4539, pp. 393–408, 2007.

The ellipsoid method ([10]), the random walk method ([2]), and interior-point methods (IPMs) ([9], [12]) are examples of methods which solve (1) in polynomial-time. These methods differ substantially in their representation requirement as well as in their practical performance. For example, a membership oracle suffices for the ellipsoid method and the random walk method, while a special barrier function for K is required to implement an IPM. The latter is by far the most successful algorithm for conic programming in practice: for example, applications of SDP range over several fields including optimal control, eigenvalue optimization, combinatorial optimization and many others, see [18].

For the important special case of linear inequalities, when $X = \mathbb{R}^n$ and $K = \mathbb{R}^m_+$, an alternative method is the perceptron algorithm [17,13], developed primarily in learning theory. It is well-known that this simple method terminates after a finite number of iterations which can be bounded by the square of the inverse of the *width* τ of the feasibility cone $\mathcal{F}$. Although attractive due to its simplicity and its noise-tolerance [4,3], the perceptron algorithm is not considered theoretically efficient since the width τ can be exponentially small in the size of the instance in the bit-length model. Dunagan and Vempala ([5]) combined the perceptron algorithm with a sequence of re-scalings constructed from near-feasible solutions. These re-scalings gradually increase τ on average and the resulting re-scaled perceptron algorithm has complexity $O(n \ln(1/\tau))$ iterations (with high probability), which is theoretically efficient.

Here we extend the re-scaled perceptron algorithm proposed in [5] to the conic setting of (1). Although the probabilistic analysis is similar, this is not the case for the remainder of the analysis. In particular, we observe that the improvement obtained in [5] arises from a clever use of a *deep-separation oracle* (see Def. 3), which is stronger than the usual separation oracle used in the classical perceptron algorithm. In the case of a system of linear inequalities studied in [5], there is no difference between the implementation of both oracles. However, this difference is quite significant for more general cones.

We investigate, in detail, ways to construct a deep-separation oracle for several classes of cones, since it is the driving force of the re-scaled perceptron algorithm. We establish important properties of the deep-separation oracle and its implementation for several classes. Our main technical result is a general scheme that yields a polynomial-time deep-separation oracle using only a deep-separation oracle for the dual cone of K (which is readily available for many cones of interest such as the cone of positive semi-definite matrices). This implies that the re-scaled perceptron algorithm runs in polynomial time for any conic program, provided we have a suitable deep separation oracle. This captures the important cases of linear programs, second-order cone programs and semi-definite programs[1] and thus conveys the benefits of the perceptron algorithm to these problems.

We start in Section 2 with properties of convex cones, oracles, and the definition of a deep-separation oracle. Section 3 generalizes the classical perceptron

[1] There have been earlier attempts to extend the algorithm of [5], to SDPs in particular, but unfortunately these have turned out to be erroneous.