Implementation of interior-point methods for positive semi-definite Toeplitz cone programming

Matt Staib, Dieterich Lawson

March 21, 2015

1 Introduction

Consider the cone of positive-semidefinite Toeplitz matrices

$$\mathcal{T}_n = \{x \in \mathbf{R}^n \mid T_n(x) \geq 0\}$$

with $T_n(x)$ defined as

$$T_n(x)_{ij} = \begin{cases} x_{|i-j|+1} & 0 \leq |i-j| \leq n-1 \\ 0 & \text{otherwise} \end{cases}.$$

In words, $T_n(x)$ is the square Toeplitz matrix resulting from taking the ith entry of x as the ith sub- and super-diagonal. This cone appears in various applications in signal processing

Flesh out applications

. In this paper we investigate methods for cone programming over $\mathcal{T}$, i.e. we present methods for solving problems of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \in \mathcal{T}_n. \end{array} \tag{1}$$

where $x \in \mathbf{R}^n$ is the optimization variable, $c \in \mathbf{R}^n$, $A \in \mathbf{R}^{m \times n}$, and $b \in \mathbf{R}^m$. The central difficulty for problems of this form is that $\mathcal{T}_n$ is not symmetric, so much of the standard primal-dual machinery becomes unweildly.

The remainder of this paper is organized as follows. We first explore the properties of $\mathcal{T}_n$, deriving its dual and showing that it is not symmetric. To motivate new algorithms we then investigate how un-symmetric cones present a problem for a standard primal-dual interior point method. We discuss a suitable barrier for $\mathcal{T}_n$ and its use in two different algorithms that make speed gains over the standard primal-dual interior-point method by forgoing pieces of the dual picture. Finally we present some numeric results from these algorithms. Note that nothing presented here is original, and in many cases follows our sources literally.

2 The Positive Semi-Definite Toeplitz Cone

To characterize the dual of $\mathcal{T}_n$ we show that $\mathcal{T}_n$ is the dual of the cone of finite autocorrelation sequences. A finite autocorrelation sequence is a vector $x \in \mathbf{R}^n$ with a $y \in \mathbf{R}^n$ such that

$$x_k = \sum_{i=1}^{n-k} y_i y_{i+k} \quad k = 1, \ldots, n.$$

For any positive scalar α, αx can be represented as above by $\alpha^{1/2} y$, so the set of finite autocorrelation sequences forms a cone. The dual of $\mathcal{C}_n$ is

$$\mathcal{C}_n^* = \{z \in \mathbf{R}^n \mid z^T x \geq 0 \text{ for } x \in \mathcal{C}_n\}.$$

If $z^T x \geq 0$ for all $x \in \mathcal{C}_n$, then for all $y \in \mathbf{R}^n$

$$z^T x \geq 0 \iff \sum_{i=1}^{n} z_i \left(\sum_{i=1}^{n-k} y_i y_{i+k} \right).$$

probably don't want that iff symbol in there

The summation above can be expressed in terms of the Toeplitz form of z — the above is true if and only if $y^T T_n(z) y \geq 0$ for all $y \in \mathbf{R}^n$. Thus $\mathcal{C}_n^*$ is exactly the positive semi-definite Toeplitz cone, $\mathcal{T}_n$. Nonempty, closed, convex cones are reflexive, so $\mathcal{T}_n^* = \mathcal{C}_n$, and $\mathcal{T}_n$ is not symmetric.

Do we need more than this? Is it clear that these aren't the same thing or should we provide an example?

Do we need to show that it is nonempty, closed, and convex? Or is that too obvious?

3 The Barrier Method

We turn now to interior point methods and their suitability for solving un-symmetric cone problems. Consider the following problem, which is slightly more general than (1)

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \in K \end{array} \tag{2}$$

where all variables are as in (1), and K is a proper (closed, convex, solid, pointed) cone possessing a self-concordant and ν-logarithmically homogenous barrier function, ϕ. A ν logarithmically homogenous barrier satisfies

$$\phi(tx) = -\nu \log(t) + \phi(x).$$

The *barrier method* is an interior-point method that solves (2) by using ϕ to make the cone constraint implicit in the objective. Using equality-constrained Newton's method, the barrier method solves a series of problems of the form

$$\begin{array}{ll} \text{minimize} & tc^T x + \phi(x) \\ \text{subject to} & Ax = b \end{array} \tag{3}$$

where $t \in \mathbf{R},\ t > 0$. For each $t > 0$, (3) has a unique minimizer TODO:ref and that series of minimizers define the *primal central path*

$$x^*(t) = \arg\min_x \{tc^T x + \phi(x) \mid Ax = b\}.$$

Each point on the primal central path is associated with a dual-feasible point which can be used to lower-bound the optimal value of (2). To see this note that any minimizer of (3) $x^*(t)$ must satisfy the optimality conditions

$$tc + \nabla\phi(x^*(t)) - y^T A = 0$$

for some $y \in \mathbf{R}^m$. It can be shown TODO:cite that if $x \in \mathbf{int} K$ then $\nabla\phi(x) \in K^*$. Thus, from the optimality conditions we see that $\nabla\phi(x^*(t))/t$, y/t are feasible for the dual of (1). The dual is:

$$\begin{array}{ll} \text{maximize} & -b^T\lambda \\ \text{subject to} & c + A^T\lambda + \nu = 0 \\ & \nu \in K^* \end{array} \tag{4}$$

where $\lambda, \nu \in \mathbf{R}^n$ and K^* is the dual cone of K. Now we can plug y/t into the dual objective, giving:

$$c^T x^*(t) + b^T(y/t) = \text{MAGIC} = \nu/t$$

Which bounds the sub-optimality of

$$\text{something} \le \nu/t$$

This suggests the following algorithm:

Algo goes here

In fact in TODO:Cite Vaderberghe's paper just such a barrier is proposed for the cone we're interested in. He does lots of stuff to blah

Section on vanderberghe paper

paragraph on: we would like to do primal-dual IPMs but can't (just give KKT)

4 A Predictor-Corrector Method

Nesterov's method goes here.

5 Numerical Results

Our implementation is hosted on Github at github.com/mstaib/toeplitz-ipm. Unfortunately we were unable to produce a fully working variation of Nesterov's algorithm. We encountered difficulties with poorly-conditioned Hessians among other things. We tested the specific problem with $c = \mathbf{1} \in \mathbb{R}^4$, $A \in \mathbb{R}^{3\times 4}$, and $b = A \cdot (8,4,2,1)^T$. Given a starting feasible point, our method managed 10 iterations in about 2 seconds, approximately converging to an objective value of 12.83 before this inexplicably jumped up to 14.7 on the 10th iteration. Each point produced was strictly feasible. In contrast, CVX took only 0.3 seconds to produce a better objective value of 11.04.