INTERIOR-POINT METHOD FOR NUCLEAR NORM APPROXIMATION WITH APPLICATION TO SYSTEM IDENTIFICATION*

ZHANG LIU AND LIEVEN VANDENBERGHE†

Abstract. The nuclear norm (sum of singular values) of a matrix is often used in convex heuristics for rank minimization problems in control, signal processing, and statistics. Such heuristics can be viewed as extensions of ℓ_1-norm minimization techniques for cardinality minimization and sparse signal estimation. In this paper we consider the problem of minimizing the nuclear norm of an affine matrix valued function. This problem can be formulated as a semidefinite program, but the reformulation requires large auxiliary matrix variables, and is expensive to solve by general-purpose interior-point solvers. We show that problem structure in the semidefinite programming formulation can be exploited to develop more efficient implementations of interior-point methods. In the fast implementation, the cost per iteration is reduced to a quartic function of the problem dimensions, and is comparable to the cost of solving the approximation problem in Frobenius norm. In the second part of the paper, the nuclear norm approximation algorithm is applied to system identification. A variant of a simple subspace algorithm is presented, in which low-rank matrix approximations are computed via nuclear norm minimization instead of the singular value decomposition. This has the important advantage of preserving linear matrix structure in the low-rank approximation. The method is shown to perform well on publicly available benchmark data.

Key words. Semidefinite programming, interior-point methods, nuclear norm approximation, matrix rank minimization, system identification, subspace algorithm.

AMS subject classifications. 90C22, 90C25, 90C51, 93B30.

1. Introduction.

We discuss the implementation of interior-point algorithms for the *nuclear norm approximation* problem

$$\text{minimize} \quad \|\mathcal{A}(x) - B\|_* . \tag{1.1}$$

In this problem $B \in \mathbf{R}^{p\times q}$ is a given matrix,

$$\mathcal{A}(x) = x_1A_1 + x_2A_2 + \cdots + x_nA_n$$

is a linear mapping from $\mathbf{R}^n$ to $\mathbf{R}^{p\times q}$, and $\|\cdot\|_*$ denotes the *nuclear norm* on $\mathbf{R}^{p\times q}$. ($\|X\|_*$ is the sum of the singular values of X, and is also known as the *trace norm*, the *Ky Fan norm*, and the *Schatten norm*.) The techniques discussed in this paper also apply to problems with an added convex quadratic term in the objective,

$$\text{minimize} \quad \|\mathcal{A}(x) - B\|_* + (x - x_0)^TQ(x - x_0), \tag{1.2}$$

and nuclear norm approximation problems with convex constraints. However, we will limit most of the discussion to the unconstrained linear approximation problem for the sake of clarity.

The nuclear norm approximation problem is of interest as a convex heuristic for the *rank minimization* problem

$$\text{minimize} \quad \mathbf{rank}(\mathcal{A}(x) - B),$$

*Research supported by NSF under grants ECS-0524663 and ECCS-0824003, and a Northrop Grumman Ph.D. fellowship.

†Electrical Engineering Department, University of California, Los Angeles. Email: zhang@ee.ucla.edu, vandenbe@ee.ucla.edu.

which is NP-hard in general. The convex nuclear norm heuristic was proposed by Fazel, Hindi, and Boyd [20], and is motivated by the observation that the residual $\mathcal{A}(x)-B$ at the optimal solution of (1.1) typically has low rank. This idea is very useful for a variety of applications in control and system theory, including model reduction, minimum order control synthesis, and system identification; see [20, 19, 21, 35]. An important special case is the *minimum-rank matrix completion* problem, which arises in machine learning [39] and computer vision [43, 30]. In this problem the variables x are the nonzero entries of a sparse matrix $\mathcal{A}(x)$; the matrix B has the complementary sparsity pattern and contains the fixed entries in the completion problem. Another special case is nonnegative factorization, the problem of approximating a matrix by a nonnegative matrix of low rank. This problem arises in data mining [18, §9.2].

The nuclear norm of a diagonal matrix is the ℓ_1-norm of the vector formed by its diagonal elements: $\|\operatorname{diag}(u)\|_* = \|u\|_1$ for a vector u. The role of the nuclear norm in convex heuristics for rank minimization therefore parallels the use of the ℓ_1-norm in sparse approximation or cardinality minimization [41, 24, 12, 17, 10, 9, 8, 44]. The renewed interest in nuclear norm minimization is motivated by the remarkable success of ℓ_1-norm techniques in practice, and by theoretical results that characterize the possibility of exact recovery of sparse signals by ℓ_1-norm methods [17, 10, 9, 8, 44]. Parts of this theoretical analysis have recently been extended to nuclear norm methods, providing conditions for exact recovery of low-rank matrices via nuclear norm optimization [35, 7, 36, 11, 6].

Algorithms for nuclear norm approximation. This paper is concerned with numerical methods for problem (1.1), and for extensions of this problem that include convex contraints or regularized objectives as in (1.2). It is well known that the nuclear norm approximation problem can be cast as a semidefinite program (SDP)

$$\begin{array}{ll} \text{minimize} & (\mathbf{tr}\, U + \mathbf{tr}\, V)/2 \\ \text{subject to} & \left[\begin{array}{cc} U & (\mathcal{A}(x)-B)^T \\ \mathcal{A}(x)-B & V \end{array}\right] \succeq 0, \end{array} \tag{1.3}$$

with variables $x \in \mathbf{R}^n$, $U \in \mathbf{S}^q$, $V \in \mathbf{S}^p$ [20]. (We use $\mathbf{S}^n$ to denote the set of symmetric matrices of order n.) It can therefore be solved by interior-point methods available in general-purpose software for semidefinite programming [40, 46, 2, 49, 3]. Interior-point methods offer fast convergence (typically 10-50 iterations), robust performance across different problem instances, and high accuracy. However, the memory requirements and the volume of computation per iteration are often high. The SDP (1.3) has $n + p(p+1)/2 + q(q+1)/2$ variables and is very difficult to solve by general-purpose solvers, if p and q approach 100. This difficulty has limited the adoption of nuclear norm methods for rank minimization in practice. The main contribution of this paper is to describe a more efficient interior-point method that exploits the problem structure in (1.3), and is capable of solving problems with dimensions p, q on the order of several hundred.

Again, the parallel with ℓ_1-norm optimization is instructive. The vector counterpart of problem (1.1) is the ℓ_1-norm approximation problem

$$\text{minimize} \quad \|Px - q\|_1. \tag{1.4}$$

This problem can be solved as a linear program (LP)

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T y \\ \text{subject to} & -y \preceq Px - q \preceq y, \end{array} \tag{1.5}$$

where y is an auxiliary vector variable. Although the LP formulation requires the introduction of a large number of extra variables, it can be shown that the cost of solving the LP by an interior-point method is comparable to the cost of solving the least-squares counterpart, *i.e.*, to minimizing $\|Px - q\|_2$. To see why, note that an interior-point algorithm typically requires 10–50 iterations, almost independent of the problem size. Each iteration of an interior-point method applied to (1.5) reduces to solving a set of linear equations

$$P^T D P \Delta x = r$$

where D is a positive diagonal matrix with values that change at each iteration [4, p. 617]. This means that the cost per iteration of a custom interior-point method for solving (1.4) is roughly equal to the cost of solving a weighted least-squares problem with coefficient matrix P. In this paper we establish a similar result for nuclear norm approximation. By exploiting problem structure in the SDP (1.3), we show that the complexity of an interior-point method can be reduced to $O(pqn^2)$ per iteration, if $n \geq \max\{p, q\}$. This is comparable to solving the norm approximation problem in the least-squares sense, *i.e.*, to minimizing $\|\mathcal{A}(x) - B\|_F$.

As an alternative to interior-point methods one can consider methods that solve problem (1.1) directly without requiring the SDP formulation. The cost function $f(x) = \|\mathcal{A}(x) - B\|_*$ is nondifferentiable and convex, and can be minimized by the subgradient method [38]. A subgradient of f can be computed from the singular value decomposition (SVD) [35]. Let $\mathcal{A}(x)-B = P\Sigma Q^T$ be the singular value decomposition of $\mathcal{A}(x) - B$, with $P \in \mathbf{R}^{p\times r}$, $\Sigma \in \mathbf{R}^{r\times r}$, and $Q \in \mathbf{R}^{q\times r}$, where r is the rank of $\mathcal{A}(x) - B$. Then $\mathcal{A}_{\rm adj}(PQ^T)$ is a subgradient of f at x ($\mathcal{A}_{\rm adj}$ is the adjoint mapping of $\mathcal{A}$). Since the main computation in each iteration in the subgradient method involves one singular value decomposition, the time per iteration is lower, and grows more slowly with the problem dimensions, than one iteration of the interior-point method. However, the convergence of the subgradient method is often very slow, and the number of iterations to reach an accurate solution varies widely, depending on the problem data and step size rule.

Another possibility is to replace the cost function with a smooth approximation and then minimize the smooth approximation by a fast gradient method [31, 32, 45]. For example, a smooth approximation of the function $\|X\|_*$ is obtained by taking the SVD, $X = \sum_{i=1}^r \sigma_i u_i v_i^T$, replacing the singular values with

$$h_\mu(\sigma_i) = \begin{cases} \sigma_i^2/(2\mu) & \sigma_i \leq \mu \\ \sigma_i - \mu/2 & \sigma_i > \mu \end{cases}$$

where μ is a small positive parameter, and defining $f_\mu(X) = \sum_{i=1}^r h_\mu(\sigma_i) u_i v_i^T$. It can be shown that

$$f_\mu(X) = \sup_{\|Y\|\leq 1} \left(\mathbf{tr}(Y^T X) - \frac{\mu}{2}\|Y\|_F^2 \right), \tag{1.6}$$

where $\|Y\|$ denotes the standard matrix norm, *i.e.*, the maximum singular value of Y, and $\|Y\|_F$ is the Frobenius norm. Therefore f_μ is convex and differentiable, and its gradient is Lipschitz continuous with constant $1/\mu$ [25, p.121]. This method applies to much larger problems than the interior-point method, but is less accurate (see section 5).

Several algorithms have been proposed for the related problem

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & \mathcal{F}(X) = g, \end{array} \tag{1.7}$$

where $\mathcal{F}$ is a linear mapping from $\mathbf{R}^{p\times q}$ to $\mathbf{R}^m$, and the optimization variable is $X \in \mathbf{R}^{p\times q}$. One interesting approach is to use a factored parametrization $X = LR^T$ where $L \in \mathbf{R}^{p\times r}$ and $R \in \mathbf{R}^{q\times r}$. It can be shown that for sufficiently large r, minimizing $(\|L\|_F^2 + \|R\|_F^2)/2$ subject to $\mathcal{F}(LR^T) = g$, is equivalent to problem (1.7) [35]. Thus the problem is transformed to a smooth, nonconvex optimization problem. This method can be very effective if the rank of the optimal X is small. However it is not guaranteed to find the global optimum.

Recently, a new class of first-order algorithms for the nuclear norm minimization problem (1.7) has been proposed, extending Bregman iterative algorithms for large-scale ℓ_1-norm minimization [28, 5]. Again each iteration involves a singular value decomposition of X. The complexity can be further reduced by using an approximate or partial SVD. Although these algorithms were primarily developed for the low rank matrix completion problem, they also apply to the general problem (1.7), and are particularly attractive if $\mathcal{F}$ and its adjoint are easily evaluated.

Our focus on interior-point methods can be explained by the applications in system identification that motivated this work. While the optimization problems arising in this context are too large for general purpose SDP solvers, they are still orders of magnitude smaller than the machine learning applications that have inspired the recent work on first-order methods. As mentioned earlier, interior-point methods offer high accuracy and robust performance across different problem instances. (In particular, we will note that high accuracy is important if the results of the optimization are used for model order selection.) Moreover they readily handle variations on the basic problem format (1.1), including problems with general convex constraints. As we will see, the capabilities of the specialized interior-point method described in the paper are sufficient to investigate the effectiveness of the nuclear norm heuristic in system identification, an area where it has not been widely investigated due to lack of adequate software.

Applications to system identification. As a second contribution, we propose and test a subspace identification algorithm based on quadratically regularized nuclear norm minimization. This approach has important advantages over standard techniques based on the singular value decomposition, because it preserves linear (Hankel) structure in low rank approximations. Experiments on benchmark data sets indicate that the identification method based on nuclear norm minimization is competitive with existing subspace identification methods in terms of accuracy, and that it can greatly simplify the selection of the model order.

Outline. The paper is organized as follows. In section 2 we review some background on semidefinite programming. In section 3.1 we review different semidefinite programming formulations of the nuclear norm approximation problem and give the cost of solving them via general-purpose SDP solvers. Section 3.2 contains the main result in the paper. We show how standard interior-point methods for semidefinite programming can be customized to solve the nuclear norm approximation problem much more efficiently than by general-purpose methods. In section 4 we give numerical experiments that illustrate the effectiveness of the method. Section 5 describes in detail applications in system identification.

2. Semidefinite programming.

This short section provides some necessary background on semidefinite programming.

The following optimization problems are Lagrange duals:

$$\begin{array}{llll} \text{(P)} & \text{minimize} & \mathbf{tr}(CX) & \\ & \text{subject to} & \mathcal{G}(X) = h & \\ & & X \succeq 0, & \end{array} \qquad \begin{array}{lll} \text{(D)} & \text{maximize} & h^T z \\ & \text{subject to} & \mathcal{G}_{\rm adj}(z) \preceq C. \end{array} \tag{2.1}$$

The variable in the primal problem is a symmetric matrix $X \in \mathbf{S}^n$, and the variable in the dual problem is a vector $z \in \mathbf{R}^m$. The coefficients in the cost functions are $C \in \mathbf{S}^n$ and $h \in \mathbf{R}^m$. $\mathcal{G}$ is a linear mapping from $\mathbf{S}^n$ to $\mathbf{R}^m$, and $\mathcal{G}_{\rm adj}$ is the adjoint of $\mathcal{G}$, *i.e.*, it satisfies $u^T\mathcal{G}(V) = \mathbf{tr}(\mathcal{G}_{\rm adj}(u)V)$ for all $u \in \mathbf{R}^m$, $V \in \mathbf{S}^n$. The inequalities denote matrix inequalities. The primal problem is often called an SDP in *standard form*, and the dual problem an SDP in *inequality form*.

The most popular methods for solving SDPs are the primal-dual interior-point algorithms. To estimate the cost of solving the SDPs (2.1) by a primal-dual interior-point algorithm of the type used in the popular solvers [40, 46], it is sufficient to know that the number of iterations of an interior-point method is relatively small (usually less than 50) and grows slowly with problem size. Each iteration requires the solution of a set of linear equations

$$-T^{-1}\Delta X T^{-1} + \mathcal{G}_{\rm adj}(\Delta z) = R, \qquad \mathcal{G}(\Delta X) = r$$

where $T \succ 0$. The values of T and the righthand sides R, r change at each iteration. These equations can be interpreted as linearizations of the nonlinear equations that characterize the primal and dual central paths, and are therefore often referred to as the *Newton equations*.

The Newton equations are solved by eliminating ΔX from the first equation, and then solving

$$\mathcal{G}(T\mathcal{G}_{\rm adj}(\Delta z)T) = r + \mathcal{G}(TRT).$$

This is a positive definite set of equations of order m. It can be solved via a Cholesky factorization if we first express the left hand side as a matrix-vector product $H\Delta z$. The cost is $O(m^3)$, plus the cost of computing H. The latter part depends on the structure of $\mathcal{G}$, and often exceeds the cost of the Cholesky factorization. General-purpose software packages require that $\mathcal{G}$ is expressed as a vector of inner products

$$\mathcal{G}(X) = \left(\mathbf{tr}(G_1X),\, \mathbf{tr}(G_2X),\, \ldots,\, \mathbf{tr}(G_mX)\right).$$

In this case, the elements of H are given by

$$H_{ij} = \mathbf{tr}(G_iTG_jT), \quad i,j = 1,\ldots,m.$$

If no sparsity in the matrices G_i is exploited, this requires $O(\max\{mn^3, m^2n^2\})$ operations, since each matrix product G_iT requires $O(n^3)$ operations, and the m^2 inner products of $\mathbf{tr}(G_iTG_jT)$ cost $O(n^2)$ each.

3. Nuclear norm optimization via semidefinite programming. In this section we first present several semidefinite programming formulations of the nuclear norm approximation problem (1.1) and examine the cost of solving the problem by general-purpose semidefinite programming solvers. Then we investigate the possibility of a more efficient interior-point method implementation by exploiting the problem structure.

3.1. Standard interior-point method implementation. We already mentioned that problem (1.1) is equivalent to an SDP (1.3). This is an SDP in inequality form, with $p(p+1)/2 + q(q+1)/2 + n$ variables. Its dual is the standard form SDP

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}(B^T Z_{21}) \\ \text{subject to} & \mathcal{A}_{\text{adj}}(Z_{21}) = 0 \\ & Z_{11} = I \\ & Z_{22} = I \\ & Z = \left[\begin{array}{cc} Z_{11} & Z_{21}^T \\ Z_{21} & Z_{22} \end{array}\right] \succeq 0 \end{array} \tag{3.1}$$

with variable Z. The mapping $\mathcal{A}_{\text{adj}} : \mathbf{R}^{p\times q} \to \mathbf{R}^n$ is the adjoint of $\mathcal{A}$ with respect to the inner product $\langle X, Y\rangle = \mathbf{tr}(X^T Y)$ on $\mathbf{R}^{p\times q}$:

$$\mathcal{A}_{\text{adj}}(Y) = \left(\mathbf{tr}(A_1^T Y), \mathbf{tr}(A_2^T Y), \ldots, \mathbf{tr}(A_n^T Y)\right).$$

The large number of primal variables makes the pair of SDPs (1.3) and (3.1) very expensive to solve by general-purpose software. If we assume for simplicity that $p = O(n)$, $q = O(n)$, then the complexity per iteration grows at least as fast as $O(n^6)$, since every iteration involves the solution of a dense set of linear equations of the same order as the number of variables.

As an alternative, we can start from the dual of the norm approximation problem (1.1):

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}(B^T Z) \\ \text{subject to} & \mathcal{A}_{\text{adj}}(Z) = 0 \\ & \|Z\| \leq 1. \end{array} \tag{3.2}$$

The variable is $Z \in \mathbf{R}^{p\times q}$, and $\|Z\|$ denotes the standard matrix norm, *i.e.*, the maximum singular value of Z. Problem (3.2) can be written as an inequality form SDP with linear equality constraints

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}(B^T Z) \\ \text{subject to} & \left[\begin{array}{cc} I & Z^T \\ Z & I \end{array}\right] \succeq 0 \\ & \mathcal{A}_{\text{adj}}(Z) = 0. \end{array} \tag{3.3}$$

The dual of this SDP gives (1.3), expressed as a standard form SDP with free variables,

$$\begin{array}{ll} \text{minimize} & (\mathbf{tr}\, X)/2 \\ \text{subject to} & X_{21} - \mathcal{A}(x) = -B \\ & X = \left[\begin{array}{cc} X_{11} & X_{21}^T \\ X_{21} & X_{22} \end{array}\right] \succeq 0. \end{array}$$

We have pq variables in (3.3), which is less than the number of variables in (1.3). The second formulation is therefore recommended when using a general-purpose solver. Nevertheless the cost per iteration of a general-purpose method applied to this pair of problems is still very high. If we assume that $p = O(n)$, $q = O(n)$, then the number of variables is still $O(n^2)$, and the complexity per iteration grows at least as $O(n^6)$.

3.2. Custom interior-point method implementation. We now discuss the possibility of customizing an interior-point method to solve the SDP (1.3) more efficiently than by general-purpose solvers. We assume that $\mathcal{A}$ has zero nullspace (*i.e.*, the coefficient matrices A_i are linearly independent) and that $p \geq q$ (since otherwise we can minimize $\|\mathcal{A}(x)^T - B^T\|_*$).

The key step in each iteration of an interior-point method based on primal or dual scaling directions, or on the Nesterov-Todd primal-dual scaling direction [33, 42], is the solution of a set of linear equations

$$\mathcal{A}_{\rm adj}(\Delta Z) = r, \qquad \left[\begin{array}{cc} \Delta U & \mathcal{A}(\Delta x)^T \\ \mathcal{A}(\Delta x) & \Delta V \end{array}\right] + T \left[\begin{array}{cc} 0 & \Delta Z^T \\ \Delta Z & 0 \end{array}\right] T = R \tag{3.4}$$

to compute search directions Δx, ΔU, ΔV, ΔZ. The matrices

$$T = \left[\begin{array}{cc} T_{11} & T_{21}^T \\ T_{21} & T_{22} \end{array}\right] \in \mathbf{S}^{p+q}, \qquad R = \left[\begin{array}{cc} R_{11} & R_{21}^T \\ R_{21} & R_{22} \end{array}\right] \in \mathbf{S}^{p+q},$$

and the vector $r \in \mathbf{R}^n$ are given, with $T_{11} \in \mathbf{S}^q$, $R_{11} \in \mathbf{S}^q$. The matrix T is positive definite.

It is clear that if we know ΔZ and Δx in (3.4), then ΔU and ΔV follow from

$$\Delta U = R_{11} - T_{11}\Delta Z^T T_{21} - T_{21}^T \Delta Z T_{11} \tag{3.5}$$

$$\Delta V = R_{22} - T_{21}\Delta Z^T T_{22} - T_{22}\Delta Z T_{21}^T. \tag{3.6}$$

We can therefore focus on the reduced equations

$$\mathcal{A}_{\rm adj}(\Delta Z) = r, \qquad \mathcal{A}(\Delta x) + T_{22}\Delta Z T_{11} + T_{21}\Delta Z^T T_{21} = R_{21}, \tag{3.7}$$

with variables $\Delta x \in \mathbf{R}^n$, $\Delta Z \in \mathbf{R}^{p\times q}$. To simplify (3.7), we start by computing a block-diagonal congruence transformation that reduces the four blocks of T to diagonal form:

$$\left[\begin{array}{cc} W_1 & 0 \\ 0 & W_2 \end{array}\right] \left[\begin{array}{cc} T_{11} & T_{21}^T \\ T_{21} & T_{22} \end{array}\right] \left[\begin{array}{cc} W_1^T & 0 \\ 0 & W_2^T \end{array}\right] = \left[\begin{array}{cc} I & \Sigma^T \\ \Sigma & I \end{array}\right],$$

where

$$\Sigma = \left[\begin{array}{c} \mathrm{diag}(\sigma_1, \ldots, \sigma_q) \\ 0 \end{array}\right]$$

with $0 \leq \sigma_k < 1$, $k = 1, \ldots, q$. The matrices W_1 and W_2 can be computed as

$$W_1 = Q^T L_1^{-1}, \qquad W_2 = P^T L_2^{-1},$$

where L_1 and L_2 are Cholesky factors of $T_{11} = L_1 L_1^T$ and $T_{22} = L_2 L_2^T$, and the orthogonal matrices $P \in \mathbf{R}^{p\times p}$, $Q \in \mathbf{R}^{q\times q}$, and the diagonal matrix $\Sigma \in \mathbf{R}^{p\times q}$ follow from the singular value decomposition

$$L_2^{-1} T_{21} L_1^{-T} = P \Sigma Q^T.$$

If we now define $\Delta\tilde{Z} = W_2^{-T}\Delta Z W_1^{-1}$, $\tilde{\mathcal{A}}(\cdot) = W_2\mathcal{A}(\cdot)W_1^T$, and $\tilde{R}_{21} = W_2 R_{21} W_1^T$, then the equations (3.7) reduce to

$$\tilde{\mathcal{A}}_{\rm adj}(\Delta\tilde{Z}) = r, \qquad \tilde{\mathcal{A}}(\Delta x) + \Delta\tilde{Z} + \Sigma\Delta\tilde{Z}^T\Sigma = \tilde{R}_{21}. \tag{3.8}$$

This has the same form as (3.7), but with T_{11} and T_{22} replaced with identity matrices, and T_{21} with a positive diagonal matrix. It is important to note that the singular values σ_k are less than one, as a consequence of the positive definiteness of T.

The diagonal structure in (3.8) makes it easy to eliminate $\Delta\tilde{Z}$. To see this, consider the mapping

$$\mathcal{S} : \mathbf{R}^{p\times q} \to \mathbf{R}^{p\times q}, \qquad \mathcal{S}(X) = X + \Sigma X^T \Sigma.$$

The mapping $\mathcal{S}$ is self-adjoint and positive definite, and can be factored as

$$\mathcal{S}(X) = \mathcal{L}(\mathcal{L}_{\rm adj}(X)), \tag{3.9}$$

where $\mathcal{L} : \mathbf{R}^{p\times q} \to \mathbf{R}^{p\times q}$ is defined by

$$\mathcal{L}(X)_{ij} = \begin{cases} \sqrt{1-\sigma_i^2\sigma_j^2}X_{ij} + \sigma_i\sigma_j X_{ji} & i < j \\ \sqrt{1+\sigma_i^2}X_{ii} & i = j \\ X_{ij} & i > j, \end{cases}$$

and its adjoint $\mathcal{L}_{\rm adj} : \mathbf{R}^{p\times q} \to \mathbf{R}^{p\times q}$ by

$$\mathcal{L}_{\rm adj}(X)_{ij} = \begin{cases} X_{ij} & i > q \\ X_{ij} + \sigma_i\sigma_j X_{ji} & q \geq i > j \\ \sqrt{1+\sigma_i^2}X_{ii} & i = j \\ \sqrt{1-\sigma_i^2\sigma_j^2}X_{ij} & i < j. \end{cases}$$

Note that $\mathcal{L}$ is lower-triangular when represented as a $pq \times pq$ matrix that maps $\mathbf{vec}(X)$ (the matrix X stored in column major order as a pq-vector) to $\mathbf{vec}(\mathcal{L}(X))$. The factorization (3.9) can therefore be interpreted as a Cholesky factorization of $\mathcal{S}$. The inverse mappings of $\mathcal{L}$ and its adjoint are

$$\mathcal{L}^{-1}(X)_{ij} = \begin{cases} (X_{ij} - \sigma_i\sigma_j X_{ji})/\sqrt{1-\sigma_i^2\sigma_j^2} & i < j \\ X_{ii}/\sqrt{1+\sigma_i^2} & i = j \\ X_{ij} & i > j \end{cases}$$

and

$$\mathcal{L}_{\rm adj}^{-1}(X)_{ij} = \begin{cases} X_{ij} & i > q \\ X_{ij} - \sigma_i\sigma_j X_{ji}/\sqrt{1-\sigma_i^2\sigma_j^2} & q \geq i > j \\ X_{ii}/\sqrt{1+\sigma_i^2} & i = j \\ X_{ij}/\sqrt{1-\sigma_i^2\sigma_j^2} & i < j. \end{cases}$$

These expressions provide $O(q^2)$ algorithms for evaluating $\mathcal{S}$ and its inverse.

Returning to (3.8), we can use the second equation to express $\Delta\tilde{Z}$ in terms of Δx as

$$\Delta\tilde{Z} = \mathcal{S}^{-1}(\tilde{R}_{21} - \tilde{\mathcal{A}}(\Delta x)). \tag{3.10}$$

Substituting this in the first equation gives an equation in Δx:

$$H\Delta x = g, \tag{3.11}$$

where $g = \tilde{\mathcal{A}}_{\rm adj}(\mathcal{S}^{-1}(\tilde{R}_{21})) - r$ and H is the matrix that satisfies

$$H\Delta x = \tilde{\mathcal{A}}_{\rm adj}(\mathcal{S}^{-1}(\tilde{\mathcal{A}}(\Delta x)))$$

for all Δx, *i.e.*,

$$H_{ij} = \mathbf{tr}(\tilde{A}_i^T\mathcal{S}^{-1}(\tilde{A}_j)) = \mathbf{tr}(\mathcal{L}^{-1}(\tilde{A}_i)^T\mathcal{L}^{-1}(\tilde{A}_j)), \quad i,j = 1,\ldots,n. \tag{3.12}$$

Since $\mathcal{A}$ has full rank and $\mathcal{S}$ is positive definite, (3.11) is a positive definite set of linear equations. After solving for Δx, we first obtain $\Delta\tilde{Z}$ and $\Delta Z = W_2^T\Delta\tilde{Z}W_1$ from (3.10), and then ΔU, ΔV from (3.5)–(3.6).

Several methods can be used to solve (3.11). We can compute the matrix H by computing $\mathcal{L}^{-1}(\tilde{A}_i)$ for $i = 1,\ldots,n$ and forming the inner products (3.12), and then factor H using a Cholesky factorization. We can also note that $H = G^TG$, where

$$G = \left[\begin{array}{cccc} \mathbf{vec}(\mathcal{L}^{-1}(\tilde{A}_1)) & \mathbf{vec}(\mathcal{L}^{-1}(\tilde{A}_2)) & \cdots & \mathbf{vec}(\mathcal{L}^{-1}(\tilde{A}_n))) \end{array}\right].$$

We can therefore use a QR factorization of G to factor H. This is slower (by a factor of at most two) than the Cholesky factorization method and requires more memory. However it is also numerically more stable because it allows us to compute a triangular factorization of the matrix H without explicitly calculating H.

Complexity. The linear algebra complexity per iteration of the custom interior-point algorithm can be estimated as follows. The cost of computing the congruence transformation W and the singular value matrix Σ is $O(p^3)$ (recall our assumption that $p \geq q$). The cost of computing the coefficients of the scaled mapping

$$\tilde{A}(x) = x_1\tilde{A}_1 + x_2\tilde{A}_2 + \cdots + x_n\tilde{A}_n,$$

i.e., computing $\tilde{A}_i = W_2A_iW_1^T$ for $i = 1,\ldots,n$, is $O(np^2q)$ if the coefficient matrices A_i are dense and unstructured. Solving (3.11) by Cholesky factorization or QR factorization costs $O(n^2pq)$ (recall our assumption that the $\mathcal{A}$ has zero nullspace, and hence $n \leq pq$). If we make the reasonable assumption that $n \geq p$, we see that the overal cost is $O(n^2pq)$. If the coefficient matrices A_i are dense, this is comparable to solving the linear norm approximation problem in the Frobenius norm, *i.e.*, solving a least squares problem with n variables and pq equations. More generally, if $p = O(n)$, $q = O(n)$, the cost per iteration increases as $O(n^4)$, as opposed to $O(n^6)$ if we use general-purpose software.

4. Numerical experiments.

4.1. Randomly generated problems. We first consider a family of randomly generated nuclear norm approximation problems (1.1) with $n = p = q$. The matrices A_i and B have entries independently generated from $\mathcal{N}(0,1)$. The number of variables n ranges from 20 to 350. Table 4.1 shows the time per iteration in seconds on a 3.0 GHz processor with 3 GB of memory. A graph of the results is shown in Figure 4.1. All times are averaged over five randomly generated instances. The number of iterations ranges from 8 to 15 for all experiments. Blank entries in the table indicate that the simulation was aborted due to memory limitations. Columns 2 and 3 show the statistics for solving the problem using the general-purpose solvers SeDuMi (version 1.1R3) [40] and SDPT3 (version 4.0 beta) [46] in Matlab (version R2007a). The last column shows the results of a custom implementation of the algorithm described in Section 3.2 using CVXOPT (version 1.0) [13]. CVXOPT is written in Python (version

n	SeDuMi	SDPT3	CVXOPT
20	0.25	0.29	0.03
50	13	4.0	0.15
75	138	33	0.34
100			0.93
150			2.7
200			8.3
275			23
350			50

TABLE 4.1

Time per iteration in seconds for two standard SDP solvers and a custom interior-point method, for randomly generated problems with $n = p = q$.

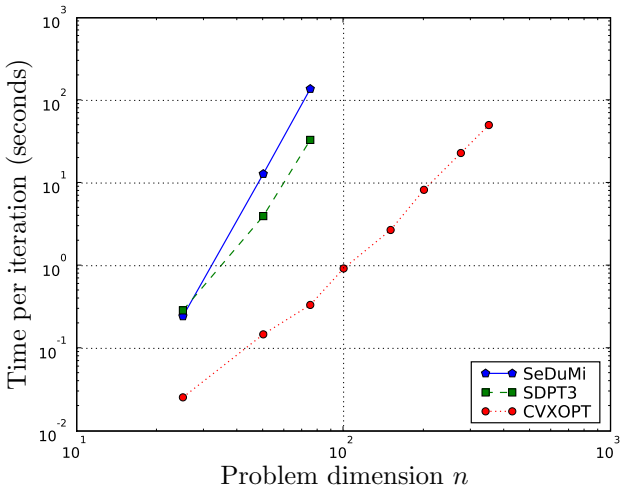

FIG. 4.1. *Graph of the results in table 4.1.*

2.5), with critical linear algebra computations implemented in C using LAPACK (version 3.1.1) and ATLAS (version 3.8.1). It includes an SDP solver that can be customized by providing functions to evaluate the constraints and to solve the Newton equations.

The results clearly illustrate the effectiveness of the fast algorithm of Section 3.2 in comparison with general-purpose solvers. Within the range of problem sizes considered here, the times per iteration grow roughly as n^5 for the general purpose solvers, and as n^3 for the custom solvers. For larger problem sizes, the times per iteration can be expected to increase as n^6, respectively, n^4, as predicted by the flop count analysis at the end of section 3.2.

4.2. An open-loop control problem. The second experiment is a variation on an example from [21]. The problem is to design a low-order discrete-time controller for a plant, so that the step response of the combined controller and plant lies within specified bounds. We denote the plant impulse response by $h(t)$, $t = 0, \ldots, N$, the controller impulse response by $x(t)$, $t = 0, \ldots, N$, and the step input by $u(t) = 1$ for $t = 0, \ldots, N$. The impulse response coefficients $x(t)$ are the variables in the problem,

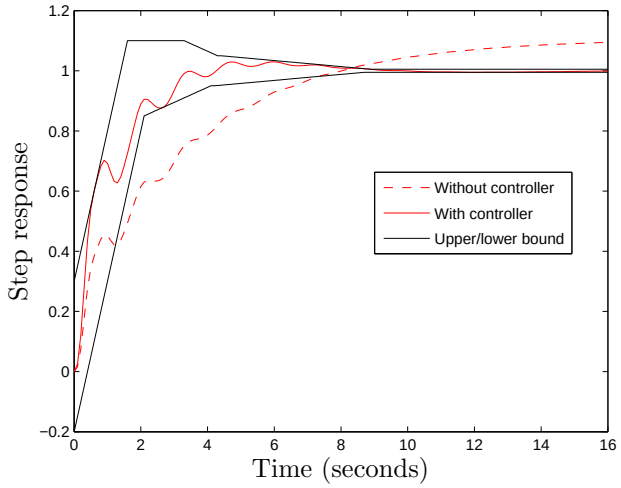

FIG. 4.2. *Open-loop controller design with time-domain constraints.*

and are computed by solving the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \|\mathcal{H}(x)\|_* \\ \text{subject to} & b_{\rm l}(t) \leq (h * x * u)(t) \leq b_{\rm u}(t), \quad t = 0, \ldots, N, \end{array} \tag{4.1}$$

where $b_{\rm l}$ and $b_{\rm u}$ are given lower and upper bounds on the step response, $*$ denotes convolution, and

$$\mathcal{H}(x) = \begin{bmatrix} x(1) & x(2) & x(3) & \cdots & x(N) \\ x(2) & x(3) & x(4) & \cdots & x(N+1) \\ \vdots & \vdots & \vdots & & \vdots \\ x(N) & x(N+1) & x(N+2) & \cdots & x(2N-1) \end{bmatrix}.$$

The variables are $x(t)$, $t = 0, \ldots, 2N-1$.

Figure 4.2 shows the result for a 6th-order plant with transfer function

$$P(z) = \frac{0.0158z^5 - 0.00292z^4 - 0.0284z^3 + 0.0177z^2 + 0.00816z - 0.00828}{z^6 - 4.03z^5 + 7.4z^4 - 8.06z^3 + 5.57z^2 - 2.31z + 0.434},$$

and $N = 160$. The step response of the original system has a long rise time and large biased steady state error. After solving the optimization problem (4.1), we obtain a second-order controller with transfer function

$$X(z) = \frac{1.58z^2 - 3.06z + 1.48}{z^2 - 1.93z + 0.931}.$$

The step response with the low-order controller satisfies all the time-domain constraints as shown in the figure. The singular values of $\mathcal{H}(x^*)$ at the optimal solution x^* are plotted in Figure 4.3, which clearly shows $\mathcal{H}(x^*)$ has rank 2.

The dimensions of the optimization (4.1) are $p = 160$, $q = 160$, $n = 320$, so the resulting SDP has more than 25000 variables. The custom interior-point method implemented in CVXOPT, reduces the relative duality gap and relative primal and dual residual below 10^{-6} in 15 iterations, and takes less than 2 minutes on a 3.0 GHz processor.

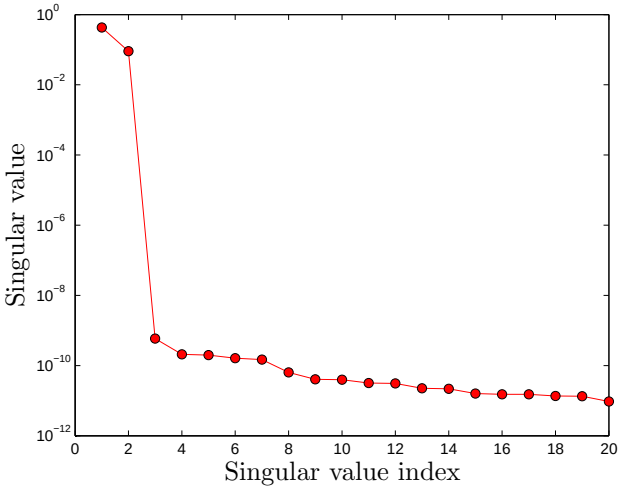

FIG. 4.3. *Largest singular values of the Hankel matrix $\mathcal{H}(x^*)$ at the optimal solution x^* of the open-loop controller design example.*

5. Application to system identification. In this section we present a system identification method based on nuclear norm minimization. The objective is to fit a discrete-time linear time-invariant state-space model

$$x(t+1) = Ax(t) + Bu(t), \qquad y(t) = Cx(t) + Du(t), \tag{5.1}$$

to a sequence of inputs $u(t) \in \mathbf{R}^m$ and outputs $y(t) \in \mathbf{R}^p$, $t = 0, \ldots, N$. The vector $x(t) \in \mathbf{R}^n$ denotes the state of the system at time t, and n is the order of the system model. The parameters to be estimated are the model order n, the matrices A, B, C, D, and the initial state $x(0)$.

Subspace algorithms [26, 47, 34, 48] use the singular value decomposition (SVD) of matrices constructed from observed inputs and outputs to estimate the system order, and to compute low-rank approximations from which the model parameters are subsequently determined. Nuclear norm minimization offers an alternative method for computing low rank matrix approximations. An advantage of this approach is that the matrix structure (for example, the block Hankel structure common in subspace identification) can be taken into account before the rank or model order is chosen. As a consequence the minimum nuclear norm approximation typically provides a more clear-cut criterion for model selection than the SVD low-rank approximation.

5.1. Input-output equation. Suppose we are given a sequence of inputs and outputs $u(t)$, $y(t)$, $t = 0, \ldots, N$. If we define block Hankel matrices

$$Y = \begin{bmatrix} y(0) & y(1) & y(2) & \cdots & y(N-r) \\ y(1) & y(2) & y(3) & \cdots & y(N-r+1) \\ \vdots & \vdots & \vdots & & \vdots \\ y(r) & y(r+1) & y(r+2) & \cdots & y(N) \end{bmatrix}$$

and

$$U = \begin{bmatrix} u(0) & u(1) & u(2) & \cdots & u(N-r) \\ u(1) & u(2) & u(3) & \cdots & u(N-r+1) \\ \vdots & \vdots & \vdots & & \vdots \\ u(r) & u(r+1) & u(r+2) & \cdots & u(N) \end{bmatrix},$$

we obtain from (5.1) the equation

$$Y = GX + HU \tag{5.2}$$

where

$$G = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^r \end{bmatrix}, \qquad H = \begin{bmatrix} D & 0 & 0 & \cdots & 0 \\ CB & D & 0 & \cdots & 0 \\ CAB & CB & D & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{r-1}B & CA^{r-2}B & CA^{r-1}B & \cdots & D \end{bmatrix},$$

and

$$X = \begin{bmatrix} x(0) & x(1) & x(2) & \cdots & x(N-r) \end{bmatrix}.$$

In the equation (5.2), the matrices G, H, and the state sequence X are unknown. We are given the matrices U and Y, or noisy approximations of them, and the goal is to estimate the system order, a system model A, B, C, D, and the initial state $x(0)$.

5.2. Subspace identification algorithm. An early idea in the subspace identification literature is to multiply (5.2) on the right with a matrix $U^\perp$ whose columns span the nullspace of U [16]. This gives

$$YU^\perp = GXU^\perp. \tag{5.3}$$

If the input and output measurements are exact, then the matrix $YU^\perp$ has rank n, provided X has rank n and there is no rank cancellation in the product $XU^\perp$. Both properties hold generically if the input sequence is chosen randomly. Assuming exact data, we can therefore determine a system realization as follows. We first compute a matrix $\hat{G}$ whose columns form a basis for the range of $YU^\perp$. From (5.3) we have $\hat{G} = GT$ where T is a nonsingular matrix. Therefore

$$\hat{G} = \begin{bmatrix} \hat{G}_0 \\ \hat{G}_1 \\ \vdots \\ \hat{G}_r \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^r \end{bmatrix} T,$$

i.e., $\hat{G}$ is the extended observability matrix for the system

$$\hat{x}(t+1) = \hat{A}\hat{x}(t) + \hat{B}u(t), \qquad y(t) = \hat{C}\hat{x}(t) + \hat{D}u(t),$$

with $\hat{A} = T^{-1}AT$, $\hat{B} = T^{-1}B$, $\hat{C} = CT$ and $\hat{D} = D$. This model is equivalent to (5.1) and differs only by a change of coordinates $x(t) = T\hat{x}(t)$. Given $\hat{G}$, we find $\hat{C}$ as the

first block of $\hat{G}$, *i.e.*, $\hat{C} = \hat{G}_0$. The matrix $\hat{A}$ can be computed from the equation

$$\begin{bmatrix} \hat{G}_1 \\ \hat{G}_2 \\ \vdots \\ \hat{G}_r \end{bmatrix} = \begin{bmatrix} \hat{G}_0 \\ \hat{G}_1 \\ \vdots \\ \hat{G}_{r-1} \end{bmatrix} \hat{A}. \tag{5.4}$$

Once $\hat{C}$ and $\hat{A}$ are available, the matrices $\hat{B}$, $\hat{D}$, and the initial state $\hat{x}(0)$ follow from the linear equations

$$\hat{C}\hat{A}^t\hat{x}(0) + \sum_{k=0}^{t-1} \hat{C}\hat{A}^{t-k}\hat{B}u(k) + \hat{D}u(t) = y(t), \quad t = 0, \ldots, N. \tag{5.5}$$

If the outputs or inputs are subject to error, the procedure outlined above can be used to estimate an approximate model. By examining the singular values of $YU^\perp$, we estimate the system order n. Truncating the singular value decomposition after n terms gives an estimate of the generalized observability matrix $\hat{G}$. This estimate does not possess the shift structure of the extended observability matrix, so the overdetermined set of equations (5.4) is not solvable. As an approximate solution we can take $\hat{C} = \hat{G}_0$, and compute a least-squares solution $\hat{A}$ to (5.4). Similarly, we can compute a least-squares solution of the overdetermined set of equations (5.5) to find estimates $\hat{B}$, $\hat{D}$ and $\hat{x}(0)$. We refer to [27, Section 10.6] for more details.

5.3. Identification by nuclear norm optimization. We now present a variation on the algorithm of the previous section based on nuclear norm minimization. We will use $y(t)$, $t = 0, \ldots, N$, as optimization variables, and distinguish the computed outputs $y(t)$ from the given measurements $y_{\rm meas}(t)$. As before, the matrix Y will denote the Hankel matrix constructed from $y(t)$. The proposed method is based on computing $y(t)$ by solving the convex optimization problem

$$\text{minimize} \quad \|YU^\perp\|_* + \gamma \sum_{t=0}^{N} \|y(t) - y_{\rm meas}(t))\|_2^2\,, \tag{5.6}$$

for a positive weighting parameter γ. This gives a point on the trade-off curve between the nuclear norm of $YU^\perp$ (as a proxy for $\mathbf{rank}(YU^\perp)$), and the deviation between the sequences $y(t)$ and $y_{\rm meas}(t)$. From the optimal solution $y(t)$ of (5.6), we can compute the singular value decomposition of $YU^\perp$, and proceed as in the method of section 5.2.

Obviously, many variations on this idea exist. For example, we can move the deviation between measured and computed output to the constraints, and solve

$$\begin{array}{ll} \text{minimize} & \|YU^\perp\|_* \\ \text{subject to} & \|y(t) - y_{\rm meas}(t)\| \leq \epsilon, \quad t = 0, \ldots, N. \end{array}$$

We can also use a weighted norm to measure the deviation, as in

$$\text{minimize} \quad \|YU^\perp\|_* + \sum_{t=0}^{N} \|W_t(y(t) - y_{\rm meas}(t))\|_2^2\,.$$

The method can also be extended to cases where the inputs are subject to error.

The idea outlined above is related to subspace algorithms based on structured total least squares [29, 14]. In these methods a low rank constraint on a structured matrix $\mathcal{A}(x)$ is imposed via nonlinear equality constraints $\mathcal{A}(x)y = 0$, $y^Ty = 1$. The resulting nonconvex optimization problem is solved locally via nonlinear optimization.

	Data set	Description	Inputs	Outputs	N_I	N_V
1	96-007	CD player arm	2	2	200	600
2	98-002	Continuous stirring tank reactor	1	2	250	700
3	96-006	Hair dryer	1	1	150	400
4	97-002	Steam heat exchanger	1	1	400	1000
5	99-001	SISO heating system	1	1	400	800
6	96-009	Flexible robot arm	1	1	100	300
7	96-011	Heat flow density	2	1	400	1000
8	97-003	Industrial winding process	5	2	250	600
9	96-002	Glass furnace	3	6	150	400
10	96-016	Industrial dryer	3	3	250	600

TABLE 5.1

Ten benchmark problems from the Daisy collection *[15]. N_I is the number of data points that will be used in the identification experiment. N_V is the number of points used for validation.*

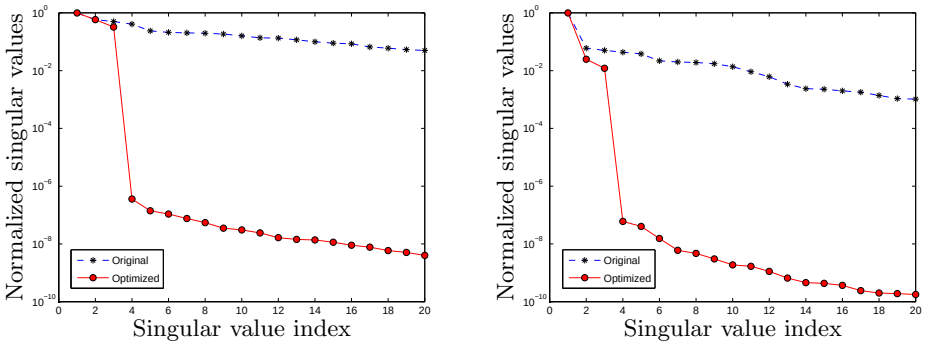

FIG. 5.1. *Largest singular values of $Y_{\text{meas}}U^\perp$ ('original') and $YU^\perp$ ('optimized') in example 1 (left) and example 2 (right). Y_{meas} and U are block-Hankel matrices constructed from the input-output data. Y is a block-Hankel matrix constructed from the optimal solution of the optimization problem (5.6).*

5.4. Experimental results. We have implemented the algorithm proposed in section 5.3 and tested it on ten benchmark problems from the DaISy collection (Database for the Identification of Systems) [15]. A brief description of the data sets is given in table 5.1. N_I is the number of data points used in the identification experiment, and N_V is the number of data points that will be used for model validation. This includes the N_I data points used in the identification.

We first apply the method of section 5.3, based on the nuclear norm minimization problem (5.6). Figures 5.1–5.5 show the singular values of $YU^\perp$ for the ten examples, where Y is the solution of the optimization problem (5.6) with $N = N_I$. The parameter γ was selected by examining different points on the trade-off curve, and choosing the value that gives approximately the smallest value of identification error. The singular values are normalized so that the largest singular value is one. As can be seen, in most cases the nuclear norm optimization provides a Hankel matrix Y with $YU^\perp$ very close to low rank, and the singular value plots suggest a distinct value for the model order. When this is not the case (as in example 9), we take n equal to the number of singular values greater than 10^{-3} times the first singular value. The figures also show the singular values of $Y_{\text{meas}}U^\perp$, where Y_{meas} is the Hankel matrix constructed from the output data. These singular values typically do not indicate a clear choice for the model order.

Columns 2–4 of table 5.2 summarize the results of the identification based on the

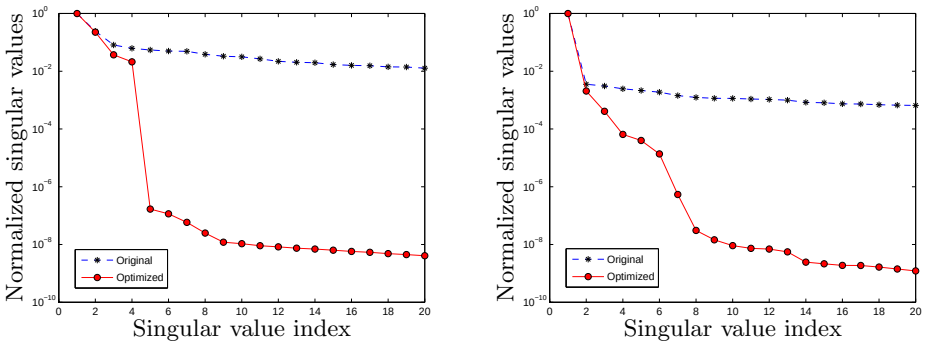

FIG. 5.2. *Largest singular values of* $Y_{\rm meas}U^\perp$ *('original') and* $YU^\perp$ *('optimized') in example 3 (left) and example 4 (right).*

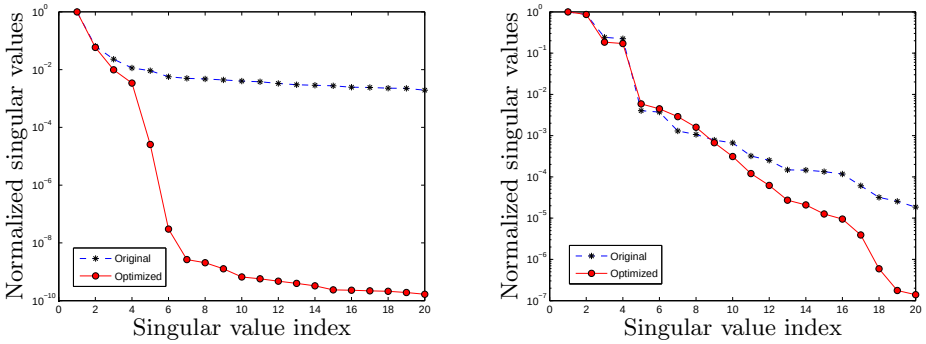

FIG. 5.3. *Largest singular values of* $Y_{\rm meas}U^\perp$ *('original') and* $YU^\perp$ *('optimized') in example 5 (left) and example 6 (right).*

nuclear norm approximation. n is the estimated system order. e_I and e_V are the identification error and validation error, computed as

$$e_I = \left(\frac{\sum_{t=0}^{N_I-1}\|y_{\rm meas}(t)-\hat{y}(t)\|_2^2}{\sum_{t=0}^{N_I-1}\|y_{\rm meas}(t)-\bar{y}_I\|_2^2}\right)^{1/2}, \qquad e_V = \left(\frac{\sum_{t=0}^{N_V-1}\|y_{\rm meas}(t)-\hat{y}(t)\|_2^2}{\sum_{t=0}^{N_V-1}\|y_{\rm meas}(t)-\bar{y}_V\|_2^2}\right)^{1/2}, \tag{5.7}$$

respectively, where $y_{\rm meas}(t)$ are the given output data,

$$\bar{y}_I = \frac{1}{N_I}\sum_{t=0}^{N_I-1} y_{\rm meas}(t), \qquad \bar{y}_V = \frac{1}{N_V}\sum_{t=0}^{N_V-1} y_{\rm meas}(t),$$

and $\hat{y}(t)$ is the output of the identified state-space model, starting at the estimated initial state. In table 5.2 we also give the results obtained with the subspace identification algorithm implemented in the Matlab Identification Toolbox. The results under 'N4SID default' are for the models computed by the Matlab command `n4sid` with the default settings (including the feedthrough matrix D). The results under 'N4SID' are from `n4sid` with a specified model order, equal to the order used in the nuclear norm method. We note that the identification errors for the nuclear norm optimization algorithm are comparable or slightly lower than `n4sid`. The main ad-

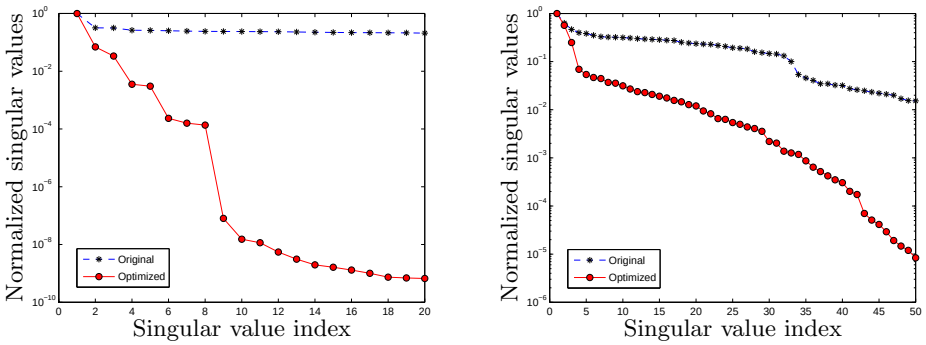

FIG. 5.4. *Largest singular values of* $Y_{\text{meas}}U^\perp$ *('original') and* $YU^\perp$ *('optimized') in example 7 (left) and example 8 (right).*

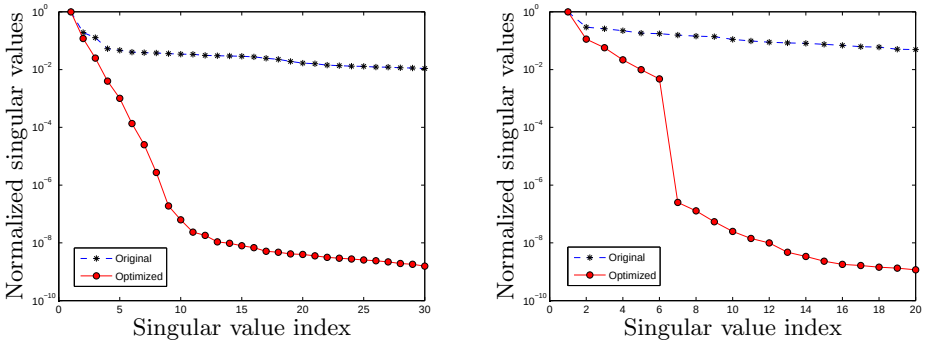

FIG. 5.5. *Largest singular values of* $Y_{\text{meas}}U^\perp$ *('original') and* $YU^\perp$ *('optimized') in example 9 (left) and example 10 (right).*

vantage of the nuclear norm technique appears to be that it makes the selection of an appropriate model order easier.

Figures 5.6 and Figure 5.7 provide some further illustration of the identification algorithm. Figure 5.6 shows the trade-off between the identification/validation error (e_I/e_V) and the nuclear norm of $YU^\perp$, for the first example. The trade-off curves were computed by solving problem (5.6) for a range of different values of γ. e_I and e_V are calculated using (5.7) based on the predictions of the identified state-space model. In Figure 5.7 we compare the measured outputs with the model predictions for example 1. The first $N_I = 200$ points were used for the identification.

5.5. Comparison with first-order method. In this section we compare the speed and accuracy of the interior-point method with a first-order method. We use Nesterov's optimal gradient method of 1983 [45, Algorithm 2] to minimize a smooth approximation of the problem (5.6), obtained by replacing the nuclear norm by the smooth matrix function (1.6). The problem data are from example 1, and the problem dimensions are $p = 81$, $q = 80$, and $n = 400$. We use three different values of the smoothing parameter μ, and a fixed step size t in the gradient steps (selected for best performance by trial and error).

The left plot in Figure 5.8 shows the relative accuracy versus the number of iterations. The relative error in this plot refers to the original, nonsmooth problem,

	Nuclear norm method			N4SID default			N4SID		
	n	e_I	e_V	n	e_I	e_V	n	e_I	e_V
1	3	0.17	0.18	10	0.15	0.24	3	0.16	0.19
2	3	0.23	0.23	7	0.22	0.20	3	0.26	0.23
3	4	0.069	0.12	3	0.079	0.13	4	0.080	0.13
4	6	0.12	0.20	2	0.21	0.47	6	0.22	0.38
5	4	0.02	0.085	2	0.038	0.094	4	0.032	0.090
6	4	0.028	0.038	7	0.046	0.075	4	0.13	0.26
7	8	0.14	0.14	1	0.18	0.17	8	0.14	0.13
8	3	0.17	0.17	10	0.17	0.18	3	0.18	0.17
9	5	0.21	0.36	10	0.31	0.47	5	0.54	0.59
10	6	0.28	0.27	10	0.31	0.32	6	0.37	0.26

TABLE 5.2

Identification results for the ten benchmark problems. *For each of the three methods, n is the estimated system order, e_I is the relative estimation error on the identification data, and e_V is the relative prediction error on the validation data. 'Nuclear norm method' refers to the algorithm described in section 5.3. The results under 'N4SID default' were obtained by the Matlab command* `n4sid` *with default values of the parameters. The results under 'N4SID' were obtained by* `n4sid` *with a specified model order.*

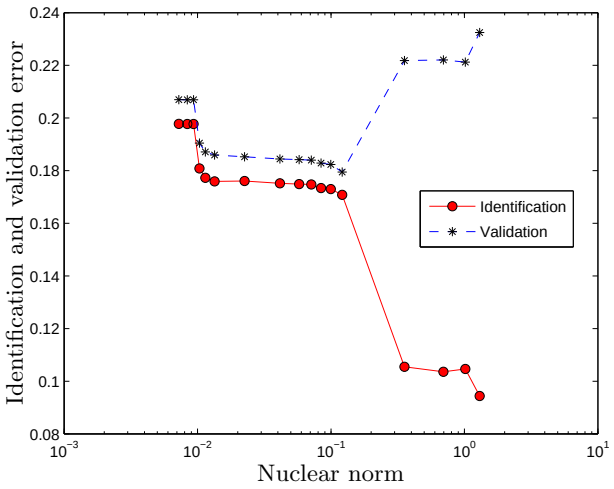

FIG. 5.6. *Trade-off between the identification/validation error (e_I/e_V) and the nuclear norm $\|YU^{\perp}\|_*$ in example 1.*

i.e., it is defined as $(f^{(k)} - f^\star)/f^\star$, where $f^{(k)}$ is the objective value of (5.6) after k iterations and f^* is the optimal value. (This explains why the error levels off at a relatively high value.) For the same problem, the interior-point method reaches an accurate solution in ten iterations. However the average CPU time per iteration of the interior-point method is 2.4 seconds, and is much larger than the 0.02 seconds of the gradient method. The right plot in Figure 5.8 shows the relative accuracy versus elapsed CPU time. The figure shows that when μ is sufficiently large, the first-order method can reach a moderate accuracy faster than the interior-point method. For larger problems, the difference will be even more pronounced. Figure 5.9 shows the singular values of the solutions for each value of μ. Here we note that the gap between zero and nonzero singular values rapidly decreases when the smoothing is increased. In combination, these three plots illustrate the trade-off between the quality

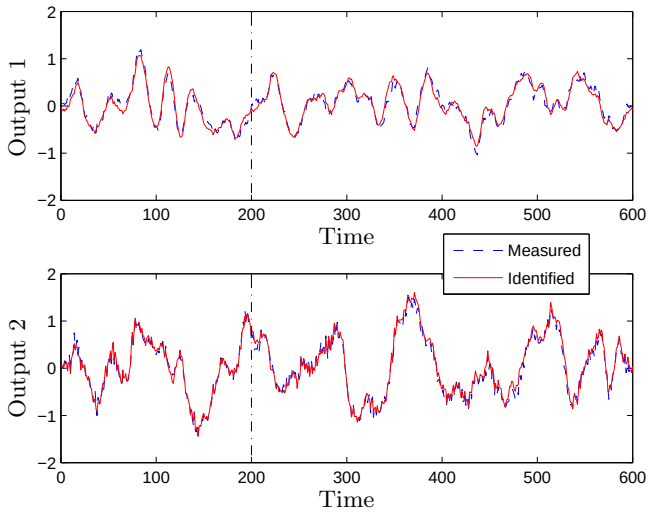

FIG. 5.7. *Measured and identified outputs in example 1.*

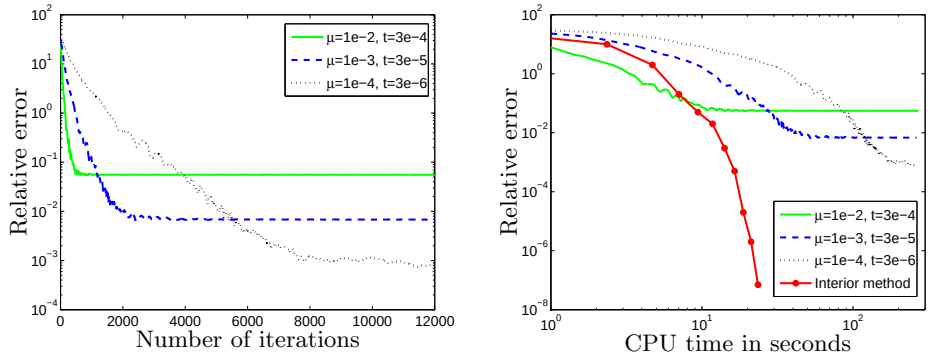

FIG. 5.8. *Convergence of interior-point method and a fast gradient method applied to a smooth approximation, with smoothing parameter μ. The parameter t is the step size in the gradient method.*

of the smooth approximation and the complexity of solving it. The experiment also demonstrates the importance of the robustness and high accuracy of interior-point methods when the results of the nuclear norm minimization are used for model order selection.

6. Conclusion. We have described techniques for improving the efficiency of interior-point methods for nuclear norm approximation. By exploiting problem structure in the linear equations that form the most expensive part of an interior-point method, we were able to reduce the cost per iteration to $O(n^2pq)$ operations, where n is the number of variables, p and q are the matrix dimensions, and we assume that $n \geq \max\{p, q\}$. The techniques can be used in combination with standard primal-dual interior-point algorithms of the type implemented in the general-purpose solvers SDPT3 and Sedumi. This results in algorithms with the same robustness and high accuracy as the state-of-the-art SDP solvers, but with a much lower computational

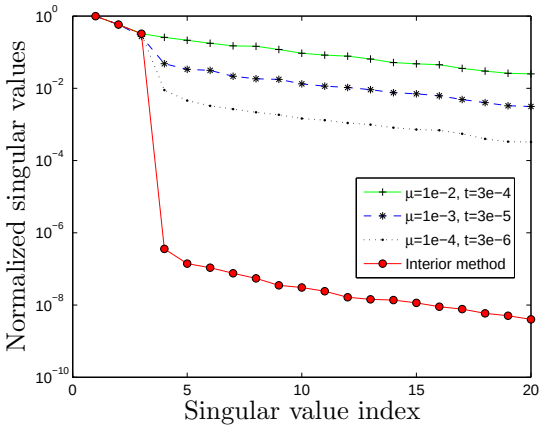

FIG. 5.9. *Normalized singular values from the fast gradient method and interior-point method.*

cost. The cost can be further reduced by exploiting structure in the mapping $\mathcal{A}(x)$. For example, some of the techniques developed for linear matrix inequalities with Hankel and Toeplitz structure in [23, 37] extend to the problem considered here.

As an application, we implemented and tested a subspace algorithm for system identification based on nuclear norm approximation. The experimental results on benchmark data sets suggest that the quality of the models obtained by this method is comparable with state-of-the-art subspace identification software. An important advantage of the nuclear norm method for computing low rank approximations is that, unlike the SVD, it preserves linear matrix structure. It also provides a less equivocal criterion for the selection of the system order.

We have not discussed the implementation of interior-point methods for the least-norm problem (1.7) or its SDP formulation

$$\begin{array}{ll} \text{minimize} & (\mathbf{tr}\, U + \mathbf{tr}\, V)/2 \\ \text{subject to} & \left[\begin{array}{cc} U & X^T \\ X & V \end{array}\right] \succeq 0 \\ & \mathcal{F}(X) = g. \end{array} \tag{6.1}$$

This SDP includes extra matrix variables U, V, and is therefore also very expensive to solve via general-purpose solvers. However the steps required for a custom implementation are more straightforward than for the nuclear norm approximation problem considered in the paper. The Newton equations for (6.1) take the form

$$T\left[\begin{array}{cc} 0 & \mathcal{F}_{\rm adj}(\Delta z)^T \\ \mathcal{F}_{\rm adj}(\Delta z) & 0 \end{array}\right]T - \left[\begin{array}{cc} \Delta U & \Delta X^T \\ \Delta X & \Delta V \end{array}\right] = \left[\begin{array}{cc} R_{11} & R_{21}^T \\ R_{21} & R_{22} \end{array}\right], \qquad \mathcal{F}(\Delta X) = r,$$

where

$$T = \left[\begin{array}{cc} T_{11} & T_{12} \\ T_{21} & T_{22} \end{array}\right]$$

is a positive definite scaling matrix. By eliminating ΔU, ΔV, ΔX from the first equation we obtain a smaller (and usually dense) system $H\Delta z = \tilde{r}$ where H is the matrix defined by

$$Hu = \mathcal{F}\left(T_{22}\mathcal{F}_{\rm adj}(u)T_{11} + T_{21}\mathcal{F}_{\rm adj}(u)^T T_{21}\right).$$

The key to improving the efficiency is then to exploit structure in $\mathcal{F}$ when assembling the matrix H. Properties that can be exploited are sparsity or low rank-structure (in the coefficients F_i, if we express $\mathcal{F}$ as $\mathcal{F}(X) = (\mathbf{tr}(F_1^T X), \ldots, \mathbf{tr}(F_m^T X)))$. These techniques are straightforward extensions of similar techniques for semidefinite programs in standard form [1, 22, 37].

REFERENCES

[1] S. J. Benson and Y. Ye, *DSDP5: Software for semidefinite programming*, Tech. Report ANL/MCS-P1289-0905, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, Sept. 2005. Submitted to ACM Transactions on Mathematical Software.

[2] ———, *DSDP5 user guide — software for semidefinite programming*, Tech. Report ANL/MCS-TM-277, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, September 2005.

[3] B. Borchers, *CSDP, a C library for semidefinite programming*, Optimization Methods and Software, 11 (1999), pp. 613–623.

[4] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004. `www.stanford.edu/~boyd/cvxbook`.

[5] J.-F. Cai, E. J. Candès, and Z. Shen, *A singular value thresholding algorithm for matrix completion*, (2008). Preprint available at `arXiv.org` (0810.3286).

[6] E. J. Candès and Y. Plan, *Matrix completion with noise*, (2009). Submitted to *Proceedings of the IEEE*.

[7] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, (2008). Preprint available at `arXiv.org` (0805.4471).

[8] E. J. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on Information Theory, 52 (2006), pp. 489–509.

[9] E. J. Candès, J. K. Romberg, and T. Tao, *Stable signal recovery from incomplete and inaccurate measurements*, Communications on Pure and Applied Mathematics, 59 (2006), pp. 1207–1223.

[10] E. J. Candès and T. Tao, *Decoding by linear programming*, IEEE Transactions on Information Theory, 51 (2005), pp. 4203–4215.

[11] E. J. Candès and T. Tao, *The power of convex relaxation: near-optimal matrix completion*, (2009). Submitted for publication.

[12] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Review, 43 (2001), pp. 129–159.

[13] J. Dahl and L. Vandenberghe, *CVXOPT: A Python Package for Convex Optimization*, `www.abel.ee.ucla.edu/cvxopt`, 2008.

[14] B. De Moor, *Total least squares for affinely structured matrices and the noisy realization problem*, IEEE Transactions on Signal Processing, 42 (1994), pp. 3104–3113.

[15] B. De Moor, P. De Gersem, B. De Schutter, and W. Favoreel, *DAISY: A database for the identification of systems*, Journal A, 38 (1997), pp. 4–5.

[16] B. De Moor, M. Moonen, L. Vandenberghe, and J. Vandewalle, *A geometrical approach for the identification of state space models with the singular value decomposition*, in Proceedings of the 1988 IEEE International Conference on Acoustics, Speech, and Signal Processing, 1988, pp. 2244–2247.

[17] D. L. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, 52 (2006), pp. 1289–1306.

[18] L. Eldén, *Matrix Methods in Data Mining and Pattern Recognition*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2007.

[19] M. Fazel, *Matrix Rank Minimization with Applications*, PhD thesis, Stanford University, 2002.

[20] M. Fazel, H. Hindi, and S. Boyd, *A rank minimization heuristic with application to minimum order system approximation*, in Proceedings of the American Control Conference, vol. 6, 2001, pp. 4734–4739.

[21] ———, *Rank minimization and applications in system theory*, in Proceedings of American Control Conference, 2004, pp. 3273–3278.

[22] K. Fujisawa, M. Kojima, and K. Nakata, *Exploiting sparsity in primal-dual interior-point methods for semidefinite programming*, Mathematical Programming, 79 (1997), pp. 235–253.

[23] Y. Genin, Y. Hachez, Yu. Nesterov, and P. Van Dooren, *Optimization problems over positive pseudopolynomial matrices*, SIAM Journal on Matrix Analysis and Applications, 25 (2003), pp. 57–79.

[24] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*, Springer-Verlag, 2001.

[25] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods*, vol. 306 of Grundlehren der mathematischen Wissenschaften, Springer-Verlag, New York, 1993.

[26] W. E. Larimore, *Canonical variate analysis in identification, filtering, and adaptive control*, in Proceedings of the 29th IEEE Conference on Decision and Control, vol. 2, 1990, pp. 596–604.

[27] L. Ljung, *System Identification: Theory for the User*, Prentice Hall, Upper Saddle River, New Jersey, second ed., 1999.

[28] S. Ma, D. Goldfarb, and L. Chen, *Fixed point and Bregman iterative methods for matrix rank minimization*, Mathematical Programming, Series A, (2008). Submitted.

[29] I. Markovsky, J. C. Willems, S. Van Huffel, B. De Moor, and R. Pintelon, *Application of structured total least squares for system identification and model reduction*, IEEE Transactions on Automatic Control, 50 (2005), pp. 1490–1500.

[30] T. Morita and T. Kanade, *A sequential factorization method for recovering shape and motion from image streams*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 19 (1997), pp. 858–867.

[31] Y. Nesterov, *Introductory Lectures on Convex Optimization*, Kluwer Academic Publishers, 2004.

[32] Yu. Nesterov, *Smooth minimization of non-smooth functions*, Mathematical Programming Series A, 103 (2005), pp. 127–152.

[33] Yu. E. Nesterov and M. J. Todd, *Self-scaled barriers and interior-point methods for convex programming*, Mathematics of Operations Research, 22 (1997), pp. 1–42.

[34] P. Van Overschee and B. De Moor, *N4SID: subspace algorithms for the identification of combined deterministic-stochastic systems*, Automatica, 30 (1994), pp. 75–93.

[35] B. Recht, M. Fazel, and P. A. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, (2007). Submitted to *SIAM Review*.

[36] B. Recht, W. Xu, and B. Hassibi, *Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization*, in Proceedings of the 47th IEEE Conference on Decision and Control, 2008, pp. 3065–3070.

[37] T. Roh and L. Vandenberghe, *Discrete transforms, semidefinite programming, and sum-of-squares representations of nonnegative polynomials*, SIAM Journal on Optimization, 16 (2006), pp. 939–964.

[38] N. Z. Shor, *Minimization Methods for Non-differentiable Functions*, Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 1985.

[39] N. Srebro, *Learning with Matrix Factorizations*, PhD thesis, Massachusetts Institute of Technology, 2004.

[40] J. F. Sturm, *Using SEDUMI 1.02, a Matlab toolbox for optimization over symmetric cones*, Optimization Methods and Software, 11-12 (1999), pp. 625–653.

[41] R. Tibshirani, *Regression shrinkage and selection via the Lasso*, Journal of the Royal Statistical Society. Series B (Methodological), 58 (1996), pp. 267–288.

[42] M. J. Todd, K. C. Toh, and R. H. Tütüncü, *On the Nesterov-Todd direction in semidefinite programming*, SIAM J. on Optimization, 8 (1998), pp. 769–796.

[43] C. Tomasi and T. Kanade, *Shape and motion from image streams under orthography: a factorization method*, International Journal of Computer Vision, 9 (1992), pp. 137–154.

[44] J. A. Tropp, *Just relax: Convex programming methods for identifying sparse signals in noise*, IEEE Transactions on Information Theory, 52 (2006), pp. 1030–1051.

[45] P. Tseng, *On accelerated proximal gradient methods for convex-concave optimization*, SIAM Journal on Optimization, (2008). submitted.

[46] R. H. Tütüncü, K. C. Toh, and M. J. Todd, *Solving semidefinite-quadratic-linear programs using SDPT3*, Mathematical Programming Series B, 95 (2003), pp. 189–217.

[47] M. Verhaegen, *Identification of the deterministic part of MIMO state space models given in innovations form from input-output data*, Automatica, 30 (1994), pp. 61–74.

[48] M. Viberg, *Subspace-based methods for the identification of linear time-invariant systems*, Automatica, 31 (1995), pp. 1835–1851.

[49] M. Yamashita, K. Fujisawa, and M. Kojima, *Implementation and evaluation of SDPA 6.0 (Semidefinite Programming Algorithm 6.0)*, Optimization Methods and Software, 18 (2003), pp. 491–505.