Computation of Minimum Volume Covering Ellipsoids

Peng Sun* and Robert M. Freund†

July 30, 2002

Abstract

We present a practical algorithm for computing the minimum volume n-dimensional ellipsoid that must contain m given points $a_1, \ldots, a_m \in \mathbb{R}^n$. This convex constrained problem arises in a variety of applied computational settings, particularly in data mining and robust statistics. Its structure makes it particularly amenable to solution by interior-point methods, and it has been the subject of much theoretical complexity analysis. Here we focus on computation. We present a combined interior-point and active-set method for solving this problem. Our computational results demonstrate that our method solves very large problem instances ($m = 30,000$ and $n = 30$) to a high degree of accuracy in under 30 seconds on a personal computer.

Key Words: Ellipsoid, Newton's method, interior-point method, barrier method, active set, semidefinite program, data mining.

INFORMS subject classifications: Nonlinear programming algorithms, convexity, nonlinear programming applications.
Area of review: Optimization

*MIT Operations Research Center, 77 Massachusetts Avenue, Bldg. E40-149, Cambridge, MA 02139, USA, email: psun@mit.edu

†MIT Sloan School of Management, 50 Memorial Drive, Cambridge, MA 02142, USA, email: rfreund@mit.edu

1 Introduction

This paper is concerned with computing the minimum volume ellipsoid in n-dimensional space $\mathbb{R}^n$ containing m given points $a_1, a_2, \ldots, a_m \in \mathbb{R}^n$. This minimum volume covering ellipsoid (MVCE) problem is useful in a variety of different application areas. In computational statistics, the minimum volume ellipsoid covering k of m given points in $\mathbb{R}^n$ is well known for its affine equivariance and positive breakdown properties as a multivariate location and scattering estimator [2]. In the area of robust statistics and data mining, efficiently finding outliers is a challenge that has attracted much research interest [10]. Indeed, one can identify data outliers quickly if one can compute the minimum volume ellipsoid quickly, since outliers are essentially points on the boundary of the minimum volume covering ellipsoid. Another emerging research area in data mining is that of finding linear-transformation-invariant (or scale-invariant) clustering methods that work for very large data sets; invariance under linear transformation is important in a multi-dimensional setting where different coefficients have different units of measurement. Traditional distance-based clustering methods such as k-mean or k-median methods are not scale-invariant. However, clustering using minimum volume ellipsoids, which use the minimum volume covering ellipsoid to cover all points in each cluster and minimizes the total volume of these covering ellipsoids, has the linear-transformation-invariance property.

The minimum volume covering ellipsoid problem has been studied for over fifty years. As early as 1948, Fritz John [6] discussed this problem in his work on optimality conditions. Barnes [1] provides an algorithm for this problem based on matrix eigenvalue decomposition. Khachiyan and Todd [9] first used interior-point methods in developing an algorithm and a complexity bound for the closely related maximum volume inscribed ellipsoid problem (MVIE) together with a linear reduction from MVCE to MVIE; the complexity of their algorithm is $O(m^{3.5} \ln\left(\frac{mR}{\epsilon}\right) \ln\left(\frac{n \ln R}{\epsilon}\right))$ arithmetic operations. Here R is defined such that the convex hull of the given points contains the unit ball centered at 0 and is contained in the concentric ball of a given radius R. Nesterov and Nemirovski [11] obtain a complexity bound of $O(m^{3.5} \ln(\frac{mR}{\epsilon}))$ operations, and more recently Khachiyan [8] has reduced this to $O(m^{3.5} \ln(\frac{m}{\epsilon}))$ operations. Zhang [16] presents interior-point algorithms for MVIE, based on various equation system reduction schemes. In [17], Zhang and Gao extend their earlier results and compare different practical algorithms for the maximum volume inscribed ellipsoid problem. Boyd et. al [15] and Toh [12] both consider the minimum volume ellipsoid problem as a special case of the more general maximum determinant problem.

In contrast to the theoretical work on the MVCE problem, herein we develop a practical algorithm for the problem that is designed to solve very large instances ($m = 30,000$

and $n = 30$) such as those that arise in data mining contexts. We present a combined interior-point and active-set method for solving this problem. Our computational results demonstrate that our method solves these very large problem instances to a high degree of accuracy in under 30 seconds on a personal computer.

The paper is organized as follows. In the rest of this section, we present notation. In Section 2, we review different relevant primal and dual problem formulations. In Section 3 we present our interior-point algorithm for solving the MVCE. In section 4, we briefly review the Frank-Wolfe method for this problem, and in Section 5 we develop active set strategies. Computational results are presented in Section 6. Section 7 discusses some unsuccessful algorithmic approaches that we tried, and Section 8 contains concluding remarks. Last of all, we show in Section 9 the relationship between our interior-point algorithm and the theory of self-concordance of [11].

Acknowledgement The authors thank Kim Chuan Toh for his assistance with the software SDPT3 and for his guidance on computational matters. The authors also thank Yurii Nesterov for guidance on issues regarding self-concordant functions.

1.1 Notation

Let $\mathbb{S}^n_+$ and $\mathbb{S}^n_{++}$ denote the convex cone of $n \times n$ symmetric positive semidefinite matrices and symmetric positive definite matrices, respectively. We use $\succeq$ and $\succ$ to denote the partial ordering induced by the cones $\mathbb{S}^n_+$ and $\mathbb{S}^n_{++}$, respectively. The vector of ones is denoted by $e := (1, 1, \ldots, 1)^T$, where the dimension is dictated by context. The capital letters U and T are used to denote the diagonal matrices whose diagonal entries correspond to the entries of the vectors u and t: $U := \mathrm{diag}(u)$ and $T := \mathrm{diag}(t)$. The Euclidean norm $\sqrt{y^T y}$ is denoted by $\|y\|$. For a given symmetric positive definite matrix M, define the M-norm by $\|v\|_M := \sqrt{v^T M v}$.

2 Primal and Dual Formulations

Our concern is with covering m given points $a_1, a_2, \ldots, a_m \in \mathbb{R}^n$ with an ellipsoid of minimum volume. In order to avoid trivialities, we make the following assumption for the remainder of this paper:

Assumption 1 *The points $a_1, \ldots, a_m$ are affinely independent.*

Let A denote the $n \times m$ matrix whose columns are the vectors $a_1, a_2, \ldots, a_m \in \mathbb{R}^n$:

$$A := [a_1 | a_2 | \cdots | a_m] \ .$$

We point out that in most applications of the minimum volume covering ellipsoid problem, particularly those in data mining, one cannot presume much in the way of special structure of the data $a_1, a_2, \ldots, a_m$. In particular, the matrix A may be fairly dense, and in all likelihood $A^T A$ as well as AA^T will be completely dense.

For $c \in \mathbb{R}^n$ and $Q \in \mathbb{S}^n_{++}$, we define the ellipsoid

$$\mathbf{E}_{Q,c} := \{x \in R^n | \, (x-c)^T Q \, (x-c) \leq 1\} \ ;$$

here c is the center of the ellipsoid and Q determines its general shape. The volume of $\mathbf{E}_{Q,c}$ is given by the formula $\frac{\pi^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}+1\right)} \frac{1}{\sqrt{\det Q}}$, see [4] for example.

Under Assumption 1, a natural formulation of the minimum volume covering ellipsoid problem is:

$$\begin{array}{lll} (\text{MVCE}^1) \quad \min_{Q,c} & \det Q^{-\frac{1}{2}} & \\ \text{s.t.} & (a_i - c)^T Q \, (a_i - c) \leq 1 \ , \ i = 1, \ldots, m \\ & Q \succ 0 \ . \end{array}$$

As written, MVCE^1 is not a convex program. By the change of variables:

$$M = Q^{\frac{1}{2}} \quad \text{and} \quad z = Q^{\frac{1}{2}} c \ ,$$

we restate the problem as

$$\begin{array}{lll} (\text{MVCE}^2) \quad \min_{M,z} & \psi(M,z) & := -\ln \det M \\ \text{s.t.} & & (Ma_i - z)^T (Ma_i - z) \leq 1 \ , \ i = 1, \ldots, m \\ & & M \succ 0 \ , \end{array} \tag{1}$$

which is now a convex program. If $(\bar{M}, \bar{z})$ is a solution of MVCE^2, we recover the solution of MVCE^1 by setting $(\bar{Q}, \bar{c}) = (\bar{M}^2, \bar{M}^{-1}\bar{z})$.

Using standard Lagrange duality constructs, one can construct the following dual problem of MVCE^2:

$$\begin{array}{lll} (\text{D}^1) \quad \max_u & \phi(u) & := \left(\frac{n}{2} \ln 2 + \frac{n}{2}\right) + \frac{1}{2} \ln \det \left[AUA^T - \frac{Auu^T A^T}{e^T u} \right] - e^T u \\ \text{s.t.} & & u \geq 0 \ , \end{array} \tag{2}$$

and $\phi(u) \leq \psi(M,z)$ for all u and (M,z) feasible for (1) and (2), respectively.

Using the fact that

$$\det\left[AUA^T - \frac{Auu^TA^T}{e^Tu}\right] = \det\begin{pmatrix} AUA^T & Au \\ u^TA^T & e^Tu \end{pmatrix} \cdot \frac{1}{e^Tu} ,$$

we can re-write (D^1) as:

$$\begin{array}{lll} (\mathrm{D}^2) & \max_u & \left(\frac{n}{2}\ln 2 + \frac{n}{2}\right) + \frac{1}{2}\ln\det\begin{pmatrix} AUA^T & Au \\ u^TA^T & e^Tu \end{pmatrix} - \frac{1}{2}\ln(e^Tu) - e^Tu \\ & \text{s.t.} & u \geq 0 \ . \end{array} \tag{3}$$

If we then re-write u as $u = \tilde{\lambda}\tilde{u}$ where $\tilde{\lambda} = e^Tu$ and $\tilde{u} \in \mathbb{R}^m_+$, we can further re-write (D^2) as follows:

$$\begin{array}{lll} (\mathrm{D}^3) & \max_{\tilde{\lambda},\tilde{u}} & \left(\frac{n}{2}\ln 2 + \frac{n}{2}\right) + \frac{n+1}{2}\ln\tilde{\lambda} + \frac{1}{2}\ln\det\begin{pmatrix} A\tilde{U}A^T & A\tilde{u} \\ \tilde{u}^TA^T & e^T\tilde{u} \end{pmatrix} - \frac{1}{2}\ln\tilde{\lambda} - \tilde{\lambda} \\ & \text{s.t.} & e^T\tilde{u} = 1 \\ & & \tilde{\lambda} \geq 0, \tilde{u} \geq 0 \ . \end{array} \tag{4}$$

Gathering terms in the objective function and optimizing with respect to $\tilde{\lambda}$ yields $\tilde{\lambda} = \frac{n}{2}$, which when substituted yields the following refined dual problem:

$$\begin{array}{lll} (\mathrm{RD}) & \max_{\tilde{u}} & \left(\frac{n}{2}\ln n\right) + \frac{1}{2}\ln\det\begin{pmatrix} A\tilde{U}A^T & A\tilde{u} \\ \tilde{u}^TA^T & e^T\tilde{u} \end{pmatrix} \\ & \text{s.t.} & e^T\tilde{u} = 1 \\ & & \tilde{u} \geq 0 \ . \end{array} \tag{5}$$

We call RD a refinement of (D^3) because we have optimized with respect to the scalar variable $\tilde{\lambda}$.

2.1 Solution via Available Interior-Point Software

MVCE^2 can be re-written as a log-determinant maximization problem subject to linear equations and second-order cone constraints:

$$(\mathrm{MVCE}^3) \quad \min_{M,z,y,w} \quad -\ln\det M$$

$$
\begin{aligned}
\text{s.t.} \quad & Ma_i - z - y_i = 0 \ , \ i = 1, \ldots, m \\
& w_i = 1 \ , \ i = 1, \ldots, m \\
& (y_i, w_i) \in C_2^n \ , \ i = 1, \ldots, m \\
& M \succ 0 \ ,
\end{aligned}
$$

where C_2^n denotes the second-order cone $\{(y, w) \in \mathbb{R}^{n+1} \mid \|y\| \leq w\}$. The format of MVCE[3] is suitable for solution using a slightly modified version of the software SDPT3 (see [14], [13]), where the software is modified in order to handle the parameterized family of barrier functions:

$$
B_\theta(M, y, t) := -\ln \det M - \theta \sum_{i=1}^{m} \ln \left(w_i^2 - (y_i)^T (y_i) \right) \tag{6}
$$

for $\theta > 0$ (and where the parameter θ is of course absent from the first term above). However, because SDPT3 does not allow for unrestricted variables, the linear equations defining the variables $y_i, w_i, i = 1, \ldots, m$, cannot be eliminated, and as a result the Newton step at each iteration must unavoidably form and factorize an $m(n+1) \times m(n+1)$ Schur-complement matrix. Even for only reasonably large values n and m, say $n = 10$ and $m = 1,000$, the computational burden becomes prohibitive.

Notice that RD is also a determinant maximization problem subject to linear inequality constraints and can also be solved using a modified version of SDPT3. The computational bottleneck at each iteration of SDPT3 lies in forming and factorizing a Schur complement matrix of size $\left(\frac{n^2+3n+4}{2}\right) \times \left(\frac{n^2+3n+4}{2}\right)$.

In order to solve large practical instances of the minimum volume covering ellipsoid problem ($n \geq 20, m \geq 1,000$, for example), we develop our own specialized methodology, designed to take explicit advantage of the structure of the problem and the typical instance sizes that one might encounter in practice, particularly in the context of data mining. In Section 3, we present our basic algorithm, which we call the "dual reduced Newton" (DRN) algorithm; this algorithm is then applied and/or modified to work with active set strategies in Section 5.

2.2 Some More Duality

Taking the Lagrange dual of RD and further refining the resulting minimization problem yields the following problem:

$$
\begin{aligned}
\text{(PL)} \quad \min_Y \quad & \left(\tfrac{n}{2}\ln n\right) - \left(\tfrac{n+1}{2}\ln(n+1)\right) - \tfrac{1}{2}\ln\det Y \\
\text{s.t.} \quad & \begin{pmatrix} a_i \\ 1 \end{pmatrix}^T [Y] \begin{pmatrix} a_i \\ 1 \end{pmatrix} \leq 1 \quad , \; i = 1, \ldots, m \\
& Y \in \mathbb{S}^{n+1}_{++} \; .
\end{aligned}
\tag{7}
$$

Problem (PL) seeks to find the minimum volume ellipsoid in $\mathbb{R}^{n+1}$ centered at the origin that contains the lifted points $(a_i, 1)^T$ for $i = 1, \ldots, m$, where each point a_i has now been lifted into $\mathbb{R}^{n+1}$ onto the hyperplane $H := \{(x, x_{n+1}) \mid x_{n+1} = 1\}$. [9], [8], and [11] propose algorithms for solving minimum volume covering ellipsoids based on this lifting. The minimum volume ellipsoid of the original problem is recovered as the intersection of the hyperplane H and the minimum volume covering ellipsoid centered at the origin containing the lifted points $(a_i, 1)^T, i = 1, \ldots, m$.

3 Dual Reduced Newton Algorithm

In this section we describe and derive our basic algorithm for the minimum volume covering ellipsoid problem; we call this algorithm the "dual reduced Newton" algorithm for reasons that will soon be clear.

3.1 Newton Step

By adding a logarithmic barrier function to the problem formulation MVCE^2, we obtain the formulation

$$
\begin{aligned}
\text{(MVCE}^2_\theta\text{)} \quad \min_{M,z,t} \quad & -\ln\det M - \theta \sum_{i=1}^{m} \ln t_i \\
\text{s.t.} \quad & (Ma_i - z)^T (Ma_i - z) + t_i = 1 \; , \; i = 1, \ldots, m \\
& M \succ 0 \\
& t > 0 \; .
\end{aligned}
$$

The parameterized solutions to this problem as θ varies in the interval $(0, \infty)$ defines the central trajectory of the problem MVCE^2. Identifying dual multipliers $u_i, i = 1, \ldots, m$, with the first set of equations in MVCE$^2_\theta$, the optimality conditions for (MVCE$^2_\theta$)

can be written as:

$$\sum_{i=1}^{m} u_i \left[(Ma_i - z)\, a_i^T + a_i\, (Ma_i - z)^T \right] - M^{-1} = 0 \tag{8}$$

$$\sum_{i=1}^{m} u_i\, (z - Ma_i) = 0 \tag{9}$$

$$(Ma_i - z)^T (Ma_i - z) + t_i = 1 \quad , i = 1, \ldots, m \tag{10}$$

$$Ut = \theta e \tag{11}$$

$$u, t \geq 0 \tag{12}$$

$$M \succ 0\ . \tag{13}$$

We could attempt to solve (8)-(13) for (M, z, t, u) directly using Newton's method, which would necessitate forming and factorizing an $\left(\frac{n(n+3)}{2} + 2m\right) \times \left(\frac{n(n+3)}{2} + 2m\right)$ matrix. However, as we now show, the variables M and z can be directly eliminated, and further structural analysis will result in only having to form and factorize a single $m \times m$ matrix. To see how this is done, note that we can solve (9) for z and obtain:

$$z = \frac{MAu}{e^T u}\ . \tag{14}$$

Substituting (14) into (8), we arrive at the following Lyapunov equation for the matrix M:

$$\left(AUA^T - \frac{Auu^T A^T}{e^T u} \right) M + M \left(AUA^T - \frac{Auu^T A^T}{e^T u} \right) = M^{-1}\ . \tag{15}$$

The following proposition, whose proof is deferred to the end of this section, demonstrates an important property of the matrix arising in (15):

Proposition 2 *Under Assumption 1, if $u > 0$, then $\left(AUA^T - \frac{Auu^T A^T}{e^T u} \right) \succ 0$.* ∎

The following remark presents a closed form solution for Lyapunov equation systems like (15), see Corollary 2.6-3 of [7]:

Remark 3 *For a given $S \succ 0$, $X := S^{-\frac{1}{2}}$ is the unique solution of the equation system:*

$$\frac{1}{2} \left(X^T S + SX \right) = X^{-1}\ .$$ ∎

Utilizing Proposition 2 and Remark 3, the unique solution of (15) is easily derived:

$$M := M(u) := \left[2\left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right)\right]^{-\frac{1}{2}} , \tag{16}$$

and substituting (16) into (14), we conclude:

Proposition 4 *Under Assumption 1, if $u > 0$ the unique solution of (8) and (9) in M, z is given by:*

$$M := M(u) := \left[2\left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right)\right]^{-\frac{1}{2}} \tag{17}$$

and

$$z := z(u) := \frac{\left[2\left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right)\right]^{-\frac{1}{2}} Au}{e^Tu} . \blacksquare \tag{18}$$

Substituting (17) and (18) into the optimality conditions (8)-(13), we can eliminate the variables M and z explicitly from the optimality conditions, obtaining the following reduced optimality conditions involving only the variables (u, t):

$$\begin{aligned} h(u) + t &= e \\ Ut &= \theta e \\ u, t &\geq 0 , \end{aligned} \tag{19}$$

where $h_i(u)$ is the following nonlinear function of u:

$$\begin{aligned} h_i(u) &:= (M(u)a_i - z(u))^T (M(u)a_i - z(u)) \\ &= \left(a_i - \tfrac{Au}{e^Tu}\right)^T \left[2\left(AUA^T - \tfrac{Auu^TA^T}{e^Tu}\right)\right]^{-1} \left(a_i - \tfrac{Au}{e^Tu}\right) , \end{aligned} \tag{20}$$

for $i = 1, \ldots, m$, where $M(u)$ and $z(u)$ are specified by (17) and (18).

For a given value of the barrier parameter θ, we will attempt to approximately solve (19) using Newton's method. Let $\nabla_u h(u)$ denote the Jacobian matrix of $h(u)$. The Newton direction $(\Delta u, \Delta t)$ for (19) at the point (u, t) is then the solution of the following system of linear equations in $(\Delta u, \Delta t)$:

$$\begin{aligned} \nabla_u h(u)\Delta u + \Delta t &= r_1 := e - t - h(u) \\ T\Delta u + U\Delta t &= r_2 := \theta e - Ut . \end{aligned} \tag{21}$$

This system will have the unique solution:

$$\begin{array}{rcl} \Delta u & = & \left(\nabla_u h(u) - U^{-1}T\right)^{-1}\left(r_1 - U^{-1}r_2\right) \\ \Delta t & = & U^{-1}r_2 - U^{-1}T\Delta u \ , \end{array} \tag{22}$$

provided we can show that the matrix $(\nabla_u h(u) - U^{-1}T)$ is nonsingular.

In order to implement the above methodology, we need to explicitly compute $\nabla_u h(u)$, and we also need to show that $(\nabla_u h(u) - U^{-1}T)$ is nonsingular. Towards this end, we define the following matrix function:

$$\Sigma(u) := \left(A - \frac{Aue^T}{e^T u}\right)^T M^2(u) \left(A - \frac{Aue^T}{e^T u}\right) \tag{23}$$

as a function of the dual variables u. Let $A \otimes B$ denote the Hadamard product of the matrices A, B, namely $(A \otimes B)_{ij} := A_{ij}B_{ij}$ for $i, j = 1, \ldots, m$. The following result conveys an explicit formula for $\nabla_u h(u)$ and also demonstrates other useful properties.

Theorem 5 *Under Assumption 1,*

(*i*) $\nabla_u h(u) = -2\left(\frac{\Sigma(u)}{e^T u} + \Sigma(u) \otimes \Sigma(u)\right)$

(*ii*) $\nabla_u h(u) \preceq 0$

(*iii*) $h(u) = \mathrm{diag}\,(\Sigma(u))$.

The proof of this theorem is also presented at the end of this section.

Corollary 6 *Under Assumption 1, if $u > 0$ and $t > 0$, then $(\nabla_u h(u) - U^{-1}T) \prec 0$, and so is nonsingular.*

Proof: This follows immediately from part (*ii*) of Theorem 5 and the fact that $U^{-1}T \succ 0$. ∎

Now let us put all of this together. In order to compute the Newton direction $(\Delta u, \Delta t)$ for the reduced optimality conditions (19) at a given point (u, t), we compute according to the following procedure:

Procedure DRN-DIRECTION(u, t, θ)**:** Given (u, t) satisfying $u, t > 0$ and given $\theta \geq 0$,

(i) form and factorize the matrix $M^{-2}(u) = \left[2\left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right)\right]$

(ii) form the matrix $\Sigma(u) = \left(A - \frac{Aue^T}{e^Tu}\right)^T M^2(u)\left(A - \frac{Aue^T}{e^Tu}\right)$

(iii) form $\nabla_u h(u) = -2\left(\frac{\Sigma(u)}{e^Tu} + \Sigma(u) \otimes \Sigma(u)\right)$ and factorize $(\nabla_u h(u) - U^{-1}T)$

(iv) solve (22) for $(\Delta u, \Delta t)$.

The computational burden of each of the five computations in Procedure DRN-DIRECTION is dominated by the need to factorize the matrices in steps (i) and (iii) above. The matrix $\left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right)$ in step (i) is $n \times n$; it requires mn^2 operations to form and n^3 steps to factorize, while the matrix $(\nabla_u h(u) - U^{-1}T)$ in step (iv) is $m \times m$; it requires nm^2 steps to form and m^3 steps to factorize.

The direction $(\Delta u, \Delta t)$ given in Procedure DRN-DIRECTION is the solution to the linearization of the reduced optimality conditions (19), which were derived from the original optimality conditions (8)-(13) of MVCE_θ^2. We call $(\Delta u, \Delta t)$ the DRN direction for "dual reduced Newton." The reason for this is that we started with the optimality conditions (8)-(13), and reduced them by eliminating the primal variables M, z before linearizing the resulting equation system in defining the Newton step.

Notice that MVCE_θ^2 is itself an optimization problem of a self-concordant barrier function, see [11]. Because the theory of self-concordant functions is essentially a theory for Newton-based algorithms, it is natural to ask whether or not the Newton direction $(\Delta u, \Delta t)$ given in Procedure DRN-DIRECTION is the Newton direction of some self-concordant function. In Section 9, we give a negative answer to this question. However, we show that the Newton direction $(\Delta u, \Delta t)$ given in Procedure DRN-DIRECTION is the Newton direction of a function that is "almost" self-concordant.

Note from (14) that $c = M^{-1}z = \frac{Au}{e^Tu}$, which states that the center of the optimal ellipsoid is a convex weighting of the points $a_1, \ldots, a_m$, with the weights being the normalized dual variables $\frac{u_i}{e^Tu}, i = 1, \ldots, m$. It is also easy to see that when $\theta = 0$, the complementarity condition $u_i t_i = \theta = 0$ has a nice geometric interpretation: a point has positive weight u_i only if it lies on the boundary of the optimal ellipsoid. These observations are well-known. Another property is that if one considers the points $a_1, \ldots, a_m$ to be a random sample of m i.i.d. random variables, then with $u := \frac{e}{m}$ we have $M^{-2}(u) = \frac{2}{m}\left(A - \frac{Aee^T}{m}\right)\left(A - \frac{Aee^T}{m}\right)^T$ is proportional to the sample covariance matrix.

3.2 Algorithm DRN

Based on the Newton step procedure outlined in Subsection 3.1, we construct the following basic interior-point algorithm for solving the MVCE_θ^2 formulation of the minimum volume covering ellipsoid problem. We name this algorithm "DRN" for dual reduced Newton algorithm.

Algorithm DRN

> **Step 0: Initialization.** Set $r \leftarrow 0.99$. Choose initial values of (u^0, t^0) satisfying $u^0, t^0 > 0$. Set $(u, t) \leftarrow (u^0, t^0)$.
>
> **Step 1: Check Stopping Criteria.** If $\|e - h(u) - t\| \leq \epsilon_1$ and $u^T t \leq \epsilon_2$, STOP. Return u, $Q := [M(u)]^2$, and $c := [M(u)]^{-1} z(u)$.
>
> **Step 2: Compute Direction.** Set $\theta \leftarrow \frac{u^T t}{10m}$. Compute $(\Delta u, \Delta t)$ using Procedure DRN-DIRECTION(u, t, θ).
>
> **Step 3: Step-size Computation and Step.** Compute $\bar{\beta} \leftarrow \max\{\beta \mid (u, t) + \beta(\Delta u, \Delta t) \geq 0\}$ and $\tilde{\beta} \leftarrow \min\{r\bar{\beta}, 1\}$. Set $(u, t) \leftarrow (u, t) + \tilde{\beta}(\Delta u, \Delta t)$. Go to **Step 1**.

The algorithm is initialized in Step 0. Strategies for choosing (u^0, t^0) are discussed immediately below in Subsection 3.3. The quantity $r < 1$ is used to keep the iterate values of (u, t) strictly positive, see Step 3. The stopping criteria are checked in Step 1; the tolerances are ϵ_1 for feasibility and ϵ_2 for optimality. We discuss the stopping criterion in a bit more detail below in Subsection 3.4. In Step 2 the barrier parameter θ is updated and the DRN direction is computed; similar to standard interior-point methods for conic optimization, we use a rather aggressive shrinking factor of 10 when updating θ at each iteration. In step 3 we compute the step-size using a standard min-ratio test augmented by a fraction $r \in (0, 1.0)$ that keeps the new iterate values of (u, t) strictly positive. We found that setting $r = 0.99$ tended to work best.

3.3 Initialization Strategies

One way to start algorithm DRN is to choose any pair (u^0, t^0) that satisfies $u^0, t^0 > 0$, for example $(u^0, t^0) = (\alpha e, \alpha e)$ for some suitable positive scalar α. However, we found it preferable to choose (u^0, t^0) in a way that guarantees the initial primal feasibility of

$M(u^0), z(u^0)$. We start by setting $u^0 = \frac{n}{2m}e$ (where the factor $\frac{n}{2m}$ was chosen empirically). We then compute $M(u^0), z(u^0)$ via (17) and (18) and test for strict primal feasibility by checking if $h(u^0) \leq (0.95)e$, and if so we set $t^0 = e - h(u^0) > 0$, thus ensuring positivity of (u^0, t^0) as well as initial feasibility of the equations $h(u) + t = e$ at $(u, t) = (u^0, t^0)$. If $h(u^0) \not\leq (0.95)e$, observe that since $h(\alpha u) = \frac{1}{\alpha}h(u)$ from (20), we can rescale u^0 to ensure strict feasibility of the algorithm as follows: compute

$$\begin{array}{rcl} \alpha & = & \frac{\max\{h_1(u^0),\ldots,h_m(u^0)\}}{0.95} \\ u^0 & \leftarrow & \alpha u^0 \\ t^0 & \leftarrow & e - h(u^0)\ . \end{array} \tag{24}$$

This initialization strategy then guarantees strict positivity of (u^0, t^0) as well as initial feasibility of the equations $h(u) + t = e$ at $(u, t) = (u^0, t^0)$.

3.4 Stopping Criteria

The following result is the basis of the stopping criteria of Algorithm DRN:

Lemma 7 *Under Assumption 1, suppose that $u > 0$. If $h(u) \leq e$, then $(M, z) := (M(u), z(u))$ is feasible for MVCE^2 and u is feasible for D^1, and $\psi(M, z) - \phi(u) = u^T t$ where $t := e - h(u)$.* ∎

Lemma 7 states that the duality gap between feasible solutions of MVCE^2 and D^1 is simply $u^T t$, for $t = e - h(u) \geq 0$. The stopping criteria of Algorithm DRN, specified in Step 1, is to check that primal feasibility is satisfied to a pre-specified tolerance ϵ_1, and then to check if the duality gap is not greater than the pre-specified tolerance ϵ_2. In our computational tests, we set $\epsilon_1 = \epsilon_2 = 10^{-7}$. However, in practical applications in data mining where optimal objective function values are desirable but not critical, we might expect the optimality tolerance to be on the order of $\epsilon_2 = 10^{-4}$, for example.

3.5 Proofs of Proposition 2, Theorem 5, and Lemma 7

Proof of Proposition 2: Let $v = U^{\frac{1}{2}}e$. Then note that $(AUA^T - \frac{Auu^TA^T}{e^Tu}) = AU^{\frac{1}{2}}[I - \frac{vv^T}{v^Tv}]U^{\frac{1}{2}}A^T \succeq 0$ because $P := [I - \frac{vv^T}{v^Tv}] \succeq 0$ (P is a projection matrix, $P^2 = P$, $P = P^T$). Now suppose that $y^T(AUA^T - \frac{Auu^TA^T}{e^Tu})y = 0$, $y \neq 0$. This can only happen if $U^{\frac{1}{2}}A^Ty = \theta v$ for some scalar θ, i.e., $U^{\frac{1}{2}}A^Ty = \theta U^{\frac{1}{2}}e$. Therefore $A^Ty = \theta e$, $y \neq 0$, which violates Assumption 1. ∎

Proof of Theorem 5: Define $C(u) := M^2(u) = [2(AUA^T - \frac{Auu^TA^T}{e^Tu})]^{-1}$ and $\tilde{a}_i(u) := a_i - \frac{Au}{e^Tu}$. From the definition of $\Sigma(u)$, we have

$$\sigma_{ij}(u) := [\Sigma(u)]_{ij} = (\tilde{a}_i(u))^T M^2(u)(\tilde{a}_j(u)) \ , \tag{25}$$

and thus $h_i(u) = [\Sigma(u)]_{ii}$ from (20), which shows part (iii) of Theorem 5. In order to compute $\nabla_u h(u)$ we employ the chain rule. We have

$$\frac{\partial \tilde{a}_i(u)}{\partial u_j} = \frac{Au}{(e^Tu)^2} - \frac{a_j}{e^Tu} = \frac{-\tilde{a}_j}{e^Tu}$$

and

$$\begin{aligned} \frac{\partial C(u)}{\partial u_j} &= -2C(u)\left[a_j a_j^T + \frac{Auu^TA^T}{(e^Tu)^2} - \frac{a_j u^T A^T}{e^Tu} - \frac{Aua_j^T}{e^Tu}\right] C(u) \\ &= -2C(u)\tilde{a}_j(u)(\tilde{a}_j(u))^T C(u). \end{aligned}$$

Now invoke the chain rule on (25):

$$\begin{aligned} \frac{\partial h_i(u)}{\partial u_j} &= \frac{-2}{e^Tu}(\tilde{a}_i(u))^T C(u)\tilde{a}_j(u) - 2(\tilde{a}_i(u))^T C(u)\tilde{a}_j(u)(\tilde{a}_j(u))^T C(u)\tilde{a}_i(u) \\ &= -2\left(\frac{\sigma_{ij}(u)}{e^Tu} + (\sigma_{ij}(u))^2\right). \end{aligned}$$

Therefore $\nabla_u h(u) = -2\left(\frac{\Sigma(u)}{e^Tu} + \Sigma(u) \otimes \Sigma(u)\right)$, proving part (i). Now notice that $\Sigma(u) \succeq 0$, $e^Tu > 0$, and $\Sigma(u) \otimes \Sigma(u) \succeq 0$ since the Hadamard product of symmetric positive semi-definite matrices is also symmetric positive semi-definite, see Theorem 7.5.3 of [5]. Therefore $\nabla_u h(u) \preceq 0$, proving part (ii). ∎

Proof of Lemma 7: Let $(M, z) = (M(u), z(u))$. $M(u) \succ 0$ from Proposition 2, and from (20) we see that $h(u) \leq e$ is the same as $(Ma^i - z)^T(Ma^i - z) \leq 1$, $i = 1, ..., m$, whereby (M, z) is feasible for $MVCE^2$. Notice that u is feasible for the dual D^1, with duality gap

$$\begin{aligned} \psi(M, z) - \phi(u) &= -\ln\det M - \frac{n}{2}\ln 2 - \frac{n}{2} - \frac{1}{2}\ln\det\left[AUA^T - \frac{Auu^TA^T}{e^Tu}\right] + e^Tu \\ &= -\frac{n}{2} + e^Tu \qquad \text{(from(17))} \\ &= -\frac{n}{2} + u^Tt + u^Th(u). \end{aligned}$$

This yields the result provided we can show that $u^T h(u) = \frac{n}{2}$, which we do now:

$$\begin{aligned}
u^T h(u) &= \sum_{i=1}^{m} u_i (Ma_i - z)^T (Ma_i - z) \\
&= \sum_{i=1}^{m} u_i \left(a_i - \frac{Au}{e^T u} \right)^T M^2 \left(a_i - \frac{Au}{e^T u} \right) \\
&= \sum_{i=1}^{m} u_i \left(a_i - \frac{Au}{e^T u} \right) \left(a_i - \frac{Au}{e^T u} \right)^T \bullet M^2 \\
&= \left(A - \frac{Aue^T}{e^T u} \right) U \left(A - \frac{Aue^T}{e^T u} \right)^T \bullet M^2 \\
&= \left(AUA^T - \frac{Auu^T A^T}{e^T u} \right) \bullet M^2 = \frac{1}{2} M^{-2} \bullet M^2 = \frac{n}{2},
\end{aligned}$$

where " $\bullet$" denotes the trace operator $A \bullet B = \mathrm{trace}(A^T B) = \sum_{i=1}^{n} (A^T B)_{ii}$. ∎

4 The Frank-Wolfe Method

Interior-point methods and other second-order methods exhibit computational superiority in a number of settings in convex optimization. However, the work per iteration is necessarily large, which suggests that one might use a first-order method such as the Frank-Wolfe method [3] in the early solution stages. Khachiyan [8] analyzed the theoretical complexity of a first-order method for the solution of the minimum volume covering ellipsoid problem via formulation RD. Upon careful examination, the algorithm in [8] can be interpreted as a version of the Frank-Wolfe method applied to this problem. Here we re-state this algorithm in our notation and interpretation. Let $S^{(m-1)}$ denote the standard simplex in $\mathbb{R}^m$, namely $S^{(m-1)} := \{u \in \mathbb{R}^m \mid u \geq 0, e^T u = 1\}$, and let

$$V(u) := \begin{bmatrix} AUA^T & Au \\ u^T A^T & e^T u \end{bmatrix} .$$

Then problem RD can be cast as $\max_{u \in S^{(m-1)}} \ln \det V(u)$. It is straightforward to derive the partial derivatives of the objective function of RD:

$$g_i(u) := \frac{\partial \ln \det V(u)}{\partial u_i} = \begin{pmatrix} a_i^T & 1 \end{pmatrix} V(u)^{-1} \begin{pmatrix} a_i \\ 1 \end{pmatrix} , i = 1, \ldots, m .$$

Let $\bar{u} \in S^{(m-1)}$ be the current iterate value. At each iteration of the Frank-Wolfe method, we compute the gradient $g(\bar{u}) := (g_1(\bar{u}), \ldots, g_m(\bar{u}))$ of the objective function of RD and solve the subproblem $\max_{u \in S^{(m-1)}} g(\bar{u})^T u$, whose optimal solution is given by the j^{th} unit vector $e_j \in \mathbb{R}^m$ where $j = \arg\max_i g_i(\bar{u})$. The method then requires the solution of the line-search problem:

$$\max_{\alpha \in [0,1]} f_{\bar{u},j}(\alpha) := \ln\det V\left((1-\alpha)\bar{u} + \alpha e_j\right) = \ln\det\left((1-\alpha)V(\bar{u}) + \alpha \begin{pmatrix} a_j \\ 1 \end{pmatrix} \begin{pmatrix} a_j^T & 1 \end{pmatrix}\right) .$$

Khachiyan [8] cleverly observed that this line-search problem has a closed form solution, namely $\alpha = \frac{g_j(\bar{u})-(n+1)}{(n+1)(g_j(\bar{u})-1)}$ (see [8] for details). This leads to the following algorithm:

Algorithm FRANK-WOLFE

> **Step 0: Initialization.** Choose an initial values of (u^0) satisfying $u^0 \geq 0, e^T u^0 = 1$. Set $u \leftarrow u^0$.
>
> **Step 1: Solve subproblem.** Compute $g_i(u) = \begin{pmatrix} a_i^T & 1 \end{pmatrix} V(u)^{-1} \begin{pmatrix} a_i \\ 1 \end{pmatrix}, i = 1, \ldots, m$. Set $j \leftarrow \arg\max_i g_i(u)$.
>
> **Step 2: Step-size Computation and Step.** $\alpha \leftarrow \frac{g_j(u)-(n+1)}{(n+1)(g_j(u)-1)}$. $u \leftarrow (1-\alpha)u + \alpha e_j$. Go to **Step 1**.

The computational effort at each iteration of the Frank-Wolfe method is dominated by the gradient computation, which is $m(n+1)^2$ operations to form and factorize $V(u)$ and another $m(n+1)^2$ operations to then compute $g(u)$.

5 Active Set Strategies

It is easy to see from the optimality conditions (9)-(13) at $\theta = 0$ that the minimum volume covering ellipsoid is determined only by points a_i located on its boundary. The following well known result of John [6] states that the number of such boundary points is not too large:

Remark 8 *[6] The minimum volume covering ellipsoid is determined by a subset of at most $\frac{n^2+3n}{2}$ points.*

This motivates the design of active-set strategies for solving MVCE^2 wherein we try to make an intelligent guess of active points a_i at each iteration and we discard presumably inactive points from time to time. Let $\mathcal{M}$ denote the set of points that must be covered, namely $\mathcal{M} := \{a_1, \ldots, a_m\}$, and consider the following active-set method:

GENERIC ACTIVE SET METHOD

> **Step 0: Initialization.** Define some initial active set of points $\mathcal{S}_0 := \{a_{i_1}, a_{i_2}, \ldots, a_{i_l}\}$ for which $\mathcal{S}_0$ satisfies Assumption 1. Define an initial starting point u_0^0. $k \leftarrow 0$.
>
> **Step 1: Solve MVCE^2 for the set of points $\mathcal{S}_k$.** Use Algorithm DRN with starting point u_0^k. Let $(\bar{u}^k, Q_k, c_k)$ be the output returned.
>
> **Step 2: Check Feasibility.** If $\|a_j - c_k\|_{Q_k} \leq 1 + \epsilon_1$ for $i \in \mathcal{M} \setminus \mathcal{S}_k$, stop. Return $(u, Q, c) := (\bar{u}_k, Q_k, c_k)$. Otherwise go to **Step 3**
>
> **Step 3: Update Active Set.** Update the active set to $\mathcal{S}_{k+1}$. Determine a new starting point u_0^{k+1}. $k \leftarrow k + 1$. Go to **Step 1**.

In order to implement the above framework, we need to address the following specific questions: how to determine the initial active set $\mathcal{S}_0$ and the initial starting point u_0^0, how to update the active set from iteration to iteration, and how to choose the starting point u_0^k for all subsequent iterations.

5.1 Initial Active Set

One naïve approach is to randomly choose $l \geq n + 1$ points that satisfy Assumption 1. Not surprisingly, this method is inefficient in practice. Also, linear programming could be used to test and permanently eliminate all points a_i that lie in the convex hull of $\mathcal{M} \setminus \{a_i\}$. This also is inefficient.

We concentrated on developing heuristics for determining which points a_i were "far away" from the "center" of the data. We developed two main active set initialization schemes which we call Sample Covariance Initialization (SCI) and Frank-Wolfe Initialization (FWI), both of which we now describe.

5.1.1 Sample Covariance Initialization (SCI)

Following (16), the matrix $M^{-2}(e)$ is proportional to the sample covariance matrix $\frac{1}{(m-1)}\left[AA^T - \frac{Aee^TA^T}{m}\right]$ of the data points $a_1, \ldots, a_m$. The inverse of the sample covariance matrix can serve as a reasonable initial guess of the shape matrix of the covering ellipsoid and its induced norm $\|\cdot\|_{Q(e)}$ where $Q(e) := M^2(e)$ can serve as a natural initial distance metric to determine which points are far from the sample mean $\bar{a} := \frac{1}{m}Ae$. Following this idea, we define the initial active set to contain m_0 points whose distances from the sample mean $d_i := \|a_i - \bar{a}\|_{Q(e)}$ are the largest. In order to determine the cardinality of the initial set m^0, we need to trade off small size (for faster computation) against quality of information (which improves for larger m_0). We found that $m_0 := \min\{n^{1.5}, m\}$ worked well in practice. The computational burden of the SCI scheme is $O(mn^2)$ operations.

5.1.2 Frank-Wolfe Initialization (FWI)

The strategy in the Frank-Wolfe initialization scheme is to run the Frank-Wolfe algorithm for a small number of iterations starting at the barycenter $u = \frac{1}{m}e$ of $S^{(m-1)}$. At each iteration, we record the point a_j whose index j gave rise to the maximum partial derivative $g_j(u)$ at that iteration, see Step 1 of the algorithm in Section 4. We accumulate these points to form the initial active set $\mathcal{S}_0$. Although this method tended to produce initial active sets that were superior to those produced by the SCI scheme, the computational effort of this method is much greater than for SCI. Each Frank-Wolfe step needs $O(mn^2)$ operations (which is the same as the entire SCI scheme), and running the method for l steps then is $O(lmn^2)$ operations. To satisfy Assumption 1, we need to have at least $n+1$ affinely independent points in the initial active set to have a full dimensional ellipsoid, and so we must set $l \geq n+1$. Because of this, we chose $l = n+1$ as the number Frank-Wolfe steps to run.

We compare the computational performance of SCI versus FWI in Section 6.

5.2 Determining u_0^k

We first discuss the issue of determining u_0^0. If the initial active set $\mathcal{S}_0$ is chosen via SCI, we set $(u_0^0)_i$ proportional to the distance $d_i := \|a_i - \bar{a}\|_{Q(e)}$ for $i \in \mathcal{S}_0$, normalizing so that $e^T(u_0^0) = \frac{n}{2}$. If the initial active set is chosen via FWI, we set $(u_0^0)_i$ proportional to the output values u_i of the Frank-Wolfe algorithm for $i \in \mathcal{S}_0$, normalizing so that

$e^T(u_0^0) = \frac{n}{2}$. The reason for this normalization is that it follows from the formulation D^2 that u optimal for D^2 must satisfy $e^T u = \frac{n}{2}$.

We now discuss how u_0^k is determined for $k \geq 1$. At the end of the previous active set step, algorithm DRN has computed $\bar{u}_k$ for the active set $\mathcal{S}_k$. If the active set has just been expanded so that $\mathcal{S}_{k+1} = \mathcal{S}_k \cup \Delta\mathcal{S}$, we set u_0^{k+1} to be a combination $\alpha \begin{pmatrix} \bar{u}_k \\ 0 \end{pmatrix} + (1-\alpha) \begin{pmatrix} 0 \\ \bar{d} \end{pmatrix}$, where the indices are partitioned here into $\mathcal{S}_k$ and $\Delta\mathcal{S}$ and $\bar{d}_i = \|a_i - c_k\|_{Q_k}$. We found that $\alpha = 0.75$ worked well in practice. Then we normalize so that $e^T(u_0^{k+1}) = \frac{n}{2}$.

If the active set has just been shrunk, we simply re-normalize $\bar{u}_k$ so that the remaining indices satisfy $\sum_{i \in \mathcal{S}_{k+1}} (u_0^{k+1})_i = \frac{n}{2}$.

5.3 Updating the Active Set

5.3.1 Expanding the Active Set

Suppose that the current active set is $\mathcal{S}_k$ and that we have just run algorithm DRN on this set, obtaining $(\bar{u}_k, Q_k, c_k)$ as output. We consider expanding the active set to $\mathcal{S}_{k+1} = \mathcal{S}_k \cup \Delta\mathcal{S}$ for some set $\Delta\mathcal{S}$. When we expand the active set, we choose points $a_i \notin \mathcal{S}_k$ whose distances from the current center c_k are largest, using the current ellipsoidal norm to define the distances: $d_i := \|a_i - c_k\|_{Q_k}$. We would like to add a reasonable number of points to the active set whose distances d_i satisfy $d_i \geq 1$ (otherwise a_i would remain inactive in the current active set), and are large. We sort the d_i's to determine a priority ranking. The simple strategy of choosing the $l \geq 1$ farthest points to enter the active set does not work well, since for example the second farthest point may be nearby and dominated by the farthest point. Intuitively we want the points that we add to the active set to be spread around the current ellipsoid $\mathbf{E}_{Q_k, c_k}$. This is handled in our code as follows: after sorting points according to the d_i's and considering only points a_i with $d_i > 1$, if there are fewer than 30 such points we simply include all of them in $\Delta\mathcal{S}$. Otherwise, the first point to be included in $\Delta\mathcal{S}$ is the point a_i with the largest d_i. After that, we examine points one by one in descending order of d_i, and we include a_j into $\Delta\mathcal{S}$ if $\sum_{i \in \Delta\mathcal{S}} (a_j - c_k)^T Q_k (a_i - c_k) < 0$. In this way, the points that wind up in $\Delta\mathcal{S}$ will tend to make larger angles with other points in $\Delta\mathcal{S}$ (measured with respect to the matrix Q_k), and so will hopefully be spread around the ellipsoid $\mathbf{E}_{Q_k, c_k}$.

5.3.2 Shrinking the Active Set

There are several ways to delete points from the current active set with the guarantee that they will not enter the active set again. One way is to use linear programming to test if a given point in the active set lies in the convex hull of the other points in the active set, but this approach is obviously too expensive. Another possibility is to check if any of the points in the active set lie in the inscribed Löwner-John ellipsoid $\mathbf{E}_{(n^2Q_k),c_k}$, which is guaranteed to be contained in the convex hull of the current active set (see [6]). Checking this is relatively inexpensive, but is not effective in higher dimensions because it simply deletes too few points.

We used the following simple heuristic to delete points: when the cardinality of the active set first reaches $100, 150, 200, \ldots$, we delete all points a_i whose current distance from the current center (using the current ellipsoidal norm) satisfies $d_i < 0.9999$.

6 Computational Results

In order to perform computational tests, we generated data sets of varying dimension n and number of points m. The data sets were generated using independent random multinomial Gaussian distribution or several Gaussian distributions, in order to mimic the data points from one or more clusters as might be encountered in practice. All computation was done in MATLAB 6.1.0.450 on a Pentium IV 1.5GHz PC with 512M RAM running LINUX.

6.1 Small and Medium-Size Problems

Table 1 shows computational results for the solution of the minimum volume covering ellipsoid problem on small- and medium-sized problems, namely $4 \leq n \leq 20$ and $20 \leq m \leq 500$). We tested three different algorithms: (i) solution via the DRN algorithm described in Section 3, (ii) solution via formulation RD solved using a modified version of SDPT3 (modified to handle the parameterized family of barrier functions in (6) with θ absent from the first term), and (iii) solution via formulation MVCE^3 using the same modified version of SDPT3. In Table 1 and elsewhere we refer to these three approaches simply as DRN, RD-SDPT3, and MVCE^3-SDPT3. All three methods were run on the full problems, i.e., without any active-set methodology. The starting point used for the DRN algorithm was as described in Subsection 3.3. We tried a variety of different

ways to choose starting points for algorithms RD-SDPT3 and MVCE^3-SDPT3, but ultimately found no obvious advantage over the default starting point methodology built into SDPT3. All feasibility and duality gap tolerances were set to $\epsilon = 10^{-7}$. The "Iterations" columns in Table 1 shows the number of IPM/Newton iterations for each method. Table 1 also shows the geometric means of iterations and solution times. Notice in Table 1 that there were two problem instances of size $n = 10$ and $m = 500$ for which MVCE^3-SDPT3 terminated prematurely due to numerical problems.

Table 1: Performance of algorithms DRN, RD-SDPT3, and MVCE^3-SDPT3, on small and medium-sized problem instances of the minimum volume covering ellipsoid problem. (* indicates premature termination of SDPT3 with termination code "failure to solve the Shur-complement equation by using the Sherman-Morrison update.")

Dimensions		Algorithm					
		DRN		RD-SDPT3		MVCE^3-SDPT3	
n	m	Iterations	Solution Time (seconds)	Iterations	Solution Time (seconds)	Iterations	Solution Time (seconds)
4	20	10	0.02	19	0.59	16	1.00
4	20	10	0.03	18	0.56	12	0.77
4	20	12	0.03	17	0.53	16	1.02
4	20	10	0.02	19	0.58	12	0.76
4	20	10	0.02	18	0.56	14	0.88
4	20	11	0.03	18	0.55	14	0.89
4	20	10	0.02	20	0.61	17	1.09
4	40	10	0.04	17	0.64	15	1.65
4	40	10	0.04	15	0.56	11	1.23
4	40	10	0.05	15	0.57	13	1.45
4	40	10	0.04	17	0.62	16	1.75
4	60	11	0.11	17	0.91	12	2.16
4	60	11	0.09	17	0.75	11	2.00
4	60	11	0.09	18	0.81	13	2.33
4	60	11	0.10	18	0.81	13	2.34
Geometric Mean		10.45	0.04037	17.48	0.6343	13.54	1.3164
10	200	13	1.61	28	4.38	12	104.6
10	200	11	1.36	26	3.99	14	121.6
10	200	11	1.37	21	3.31	14	124.9
10	200	12	1.47	21	3.21	14	125.4
10	200	12	1.48	26	3.98	13	116.6
10	200	11	1.35	21	3.17	14	124
10	200	12	1.48	29	4.43	13	113.2
10	200	13	1.63	24	3.59	13	113.3
10	200	14	1.73	24	3.62	14	122
10	200	11	1.35	22	3.39	14	122.5
Geometric Mean		11.96	1.48	24.04	3.68	13.48	118.63
10	500	12	17.62	23	13.43	15*	4210.7*
10	500	14	20.59	27	15.71	16	3770.1
10	500	13	19.13	23	13.38	15	3704.9
10	500	15	22.09	33	18.53	17	4222.8
10	500	13	19.19	29	16.31	16	3892.3
10	500	15	22.1	30	17.24	16	3731.1
10	500	13	19.16	20	11.23	15*	4118.9*
10	500	14	20.65	19	10.89	15	3483.8
10	500	15	22.07	25	14.7	15	3354.3
10	500	14	20.57	28	16.09	15	3518.8
Geometric Mean		13.76	20.26	25.34	14.55	15.49	3789.73

Dimensions		Algorithm					
		DRN		RD-SDPT3		MVCE^3-SDPT3	
n	m	Iterations	Solution Time (seconds)	Iterations	Solution Time (seconds)	Iterations	Solution Time (seconds)
20	500	12	17.98	35	28.78		
20	500	12	17.92	21	16.65		
20	500	12	17.96	29	23.56		
20	500	13	19.53	27	22.23	OUT	
20	500	12	17.77	29	23.04	OF	
20	500	14	20.94	30	23.92	MEMORY	
20	500	12	17.97	29	22.99		
20	500	13	19.4	27	21.36		
20	500	12	17.92	31	24.56		
20	500	13	19.37	36	28.43		
Geometric Mean		12.48	18.65	29.11	23.31		

The first observation from Table 1 is that MVCE^3-SDPT3 has vastly inferior solution times to DRN and to RD-SDPT3. This is almost surely due to the very large Schur-complement matrix ($m(n+1) \times m(n+1)$) that must be formed and factorized to solve MVCE^3 via SDPT3.

The second observation from Table 1 is that DRN needs to take roughly one half as many Newton steps as RD-SDPT3, which shows that the directions generated by DRN are better than those for RD-SDPT3. Further analysis of the output of SDPT3 showed that this was particularly true in the first 10 iterations of RD-SDPT3, where RD-SDPT3 routinely had slow convergence to primal and/or dual feasibility. (In interior-point codes such as SDPT3, slow convergence to feasibility is indicated by step-sizes that are much less than 1.) However, in the last few iterations of RD-SDPT3, the iterates converged as quickly as for DRN. This could also indicate that SDPT3 is not as capable of capitalizing on good starting point information. Of course, the performance of RD-SDPT3 could potentially improve if a more successful starting point methodology is found, but so far such a methodology has eluded us even after testing of several different approaches.

The computational effort per iteration for DRN is dominated by factorizing and solving an $m \times m$ matrix, whereas for RD-SDPT3 it is dominated by factorizing and solving an $\left(\frac{n^2+3n+4}{2}\right) \times \left(\frac{n^2+3n+4}{2}\right)$ matrix. When $\frac{m}{n^2} << \frac{1}{2}$ we might expect DRN to dominate RD-SDPT3 due its superior choice of direction. However, when $\frac{m}{n^2} >> \frac{1}{2}$ we might expect RD-SDPT3 to dominate DRN. This intuition is verified by the numbers in Table 2. The right-most column in Table 2 shows the ratio of the DRN solution times to the RD-SDPT3 solution times. Table 2 shows the trend that the relative advantage of DRN over SDPT3 diminishes as $\frac{m}{n^2}$ grows. However, for $\frac{m}{n^2} \leq \frac{1}{2}$, DRN outperforms RD-SDPT3.

Table 2: Geometric mean of solution times of algorithms DRN and RD-SDPT3 as a function of the dimensions, for random samples of 10 problems.

			Geometric Mean of Solution Time (seconds)		Ratio
n	m	$\frac{m}{n^2}$	DRN	RD-SDPT3	$\frac{\text{DRN}}{\text{RD-SDPT3}}$
10	50	0.5	0.047	0.943	0.05
10	100	1	0.224	1.54	0.15
10	200	2	1.46	3.17	0.46
10	400	4	10.64	9.34	1.14
10	600	6	34.37	18	1.91
10	800	8	85.45	31.19	2.74
20	200	0.5	1.37	5.28	0.26
20	300	0.75	4.24	8.51	0.5
20	400	1	10.26	15.16	0.68
20	600	1.5	32.29	29.39	1.10
20	800	2	78.67	52.86	1.49
20	1000	2.5	166.2	81.25	2.05
20	1200	3	322.77	125.58	2.57
30	450	0.5	15.1	29.44	0.51
30	900	1	121.02	103.34	1.17
30	1350	1.5	425.68	258.65	1.65
30	1800	2	1031.65	519.89	1.98

6.2 Solving Large Problems using DRN and Active-Set Strategies

The computational results in Subsection 6.1 are for small- to medium-size problems; for larger-sized problems, an active-set strategy is necessary to achieve good computational performance. Recall from Remark 8 that the minimum-volume ellipsoid is determined by at most $\frac{n^2+3n}{2}$ points. Furthermore, our computational experience indicates that the number of points that determine the minimum-volume ellipsoid tends to be closer to $\frac{n^2}{4}$ in practice. Based on the analysis reported in Table 2, this suggests that the DRN algorithm should be used to solve the active-set subproblems at each major iteration, since its performance is superior to RD-SDPT3 when $\frac{m_k}{n^2} \leq \frac{1}{2} < \frac{1}{4}$, where m_k is the number of points in the active set at iteration k.

Table 3 shows the computational performance of the DRN algorithm coupled with the active-set strategy described in Section 5, for dimensions n and m in the ranges $10 \leq n \leq 30$ and $1000 \leq m \leq 30000$. The table presents results using the two initialization schemes FWI and SCI which were described in Subsections 5.1.1 and 5.1.2. The "Iterations" column reports the number of outer iterations, that is, the number of different subproblems solved, and the "Final Active Set" column reports the number of points present in the last active-set subproblem. (Note: the active-set is the current working set of points, as opposed to the set of points that lie on the boundary of the optimal ellipsoid, which we call the set of "binding points." Clearly the final active set is a superset of the set of binding points.) The "Initialization Time" columns report the time taken by the algorithm to initialize the active set using the FWI and the SCI initialization schemes. The "Total Solution Time" columns report the total time to solve the problems. As before, all subproblems were solved to a feasibility tolerance and a duality gap tolerance of $\epsilon = 10^{-7}$. Notice that the Final Active Set numbers are different for the two initialization schemes. This reflects the fact that the two initialization schemes start with different active sets, and hence terminate with different active sets as well.

Table 3: Performance of DRN algorithm with an active-set strategy using FWI and SCI initialization schemes on large problem instances of the minimum volume covering ellipsoid problem.

Dimensions		FWI				SCI			
n	m	Iterations	Final Active Set	Initialization Time (seconds)	Total Solution Time (seconds)	Iterations	Final Active Set	Initialization Time (seconds)	Total Solution Time (seconds)
10	10000	7	43	0.61	1.17	13	46	0.07	2.4
10	10000	9	71	0.6	2	7	46	0.07	1.35
10	10000	10	52	0.65	1.78	9	46	0.07	1.32
10	10000	8	68	0.61	1.86	10	42	0.07	1.62
10	10000	10	54	0.6	1.42	9	51	0.06	1.17
10	10000	5	41	0.61	1.16	12	36	0.07	1.88
10	10000	6	50	0.6	1.38	7	37	0.06	0.9

Dimensions		FWI				SCI			
n	m	Iterations	Final Active Set	Initialization Time (seconds)	Total Solution Time (seconds)	Iterations	Final Active Set	Initialization Time (seconds)	Total Solution Time (seconds)
10	10000	6	51	0.6	1.25	6	49	0.07	0.74
10	10000	12	74	0.6	2.65	15	58	0.07	2.62
10	10000	6	35	0.6	1.08	9	30	0.06	1.24
Geometric Mean		7.61	52.46	0.61	1.51	9.33	43.39	0.067	1.41
20	1000	12	91	0.25	1.78	7	73	0.03	1.43
20	1000	13	126	0.25	3.63	6	94	0.01	1.39
20	1000	11	94	0.25	1.98	7	87	0.01	1.58
20	1000	12	98	0.24	1.81	10	92	0.01	2.45
20	1000	8	69	0.24	0.95	5	66	0.02	1.2
20	1000	13	93	0.25	2.06	7	73	0.01	1.53
20	1000	16	144	0.25	4.71	5	99	0.02	1.14
20	1000	7	73	0.24	1.08	8	78	0.01	1.68
20	1000	12	95	0.25	2.05	8	83	0.02	1.79
20	1000	11	125	0.23	3.15	5	112	0.01	1.56
Geometric Mean		11.22	98.34	0.24	2.08	6.63	84.7	0.014	1.54
20	10000	22	138	2.14	11.09	14	105	0.13	5.67
20	10000	17	176	2.12	10.38	11	137	0.13	5.42
20	10000	22	177	2.12	12.13	18	138	0.13	8.27
20	10000	14	142	2.12	7.67	12	110	0.13	5.33
20	10000	15	147	2.12	8.64	7	136	0.13	3.26
20	10000	17	152	2.12	10.18	9	130	0.13	4.56
20	10000	19	214	2.12	14.99	11	148	0.13	5.42
20	10000	14	107	2.12	5.68	17	112	0.13	7.49
20	10000	14	133	2.12	7.16	15	101	0.13	6.81
20	10000	21	174	2.12	13.79	10	153	0.13	5.84
Geometric Mean		17.22	153.43	2.12	9.77	11.94	125.8	0.13	5.84
20	20000	20	126	4.25	13.1	18	102	0.26	10.63
20	20000	13	110	4.26	9.54	9	117	0.26	5.12
20	20000	15	163	4.26	12.29	17	148	0.26	9.37
20	20000	14	126	4.32	9.95	11	90	0.26	5.68
20	20000	15	167	4.25	12.07	13	110	0.26	7.04
20	20000	35	154	4.26	22.35	16	97	0.26	8.62
20	20000	18	147	4.27	12.49	9	111	0.26	4.52
20	20000	14	133	4.27	9.7	14	123	0.26	8.16
20	20000	12	148	4.26	10.04	8	110	0.25	3.85
20	20000	14	129	4.26	11.31	15	130	0.26	9.14
Geometric Mean		16.16	139.21	4.27	11.9	12.53	112.7	0.269	6.85
20	30000	11	122	6.46	12.06	11	116	0.38	7.13
20	30000	21	212	6.38	26.07	17	166	0.38	12.88
20	30000	14	152	6.39	14.72	16	155	0.39	11.23
20	30000	17	181	6.37	16.74	11	155	0.39	8.21
20	30000	17	159	6.36	16.66	14	125	0.39	10.06
20	30000	15	116	6.35	13.27	13	138	0.39	8.32
20	30000	71	220	6.36	88.15	14	138	0.39	9.69
20	30000	15	135	6.36	13.7	13	112	0.39	9.16
20	30000	15	136	6.36	13.79	16	107	0.39	11.96
20	30000	13	148	6.36	13.25	13	105	0.39	8.62
Geometric Mean		17.64	154.69	6.37	18.1	13.66	130.1	0.388	9.58
30	10000	18	202	5.36	22.18	11	188	0.22	9.64
30	10000	30	265	5.37	58.56	21	234	0.22	33.51
30	10000	53	274	5.36	102.4	20	210	0.22	23.44
30	10000	48	259	5.42	78.81	14	195	0.22	15.28
30	10000	23	221	5.36	30.39	18	182	0.22	18.68
30	10000	26	242	5.34	33.59	15	195	0.22	13.7
30	10000	18	201	5.38	21.87	20	162	0.22	22.01

Dimensions		FWI				SCI			
n	m	Iterations	Final Active Set	Initialization Time (seconds)	Total Solution Time (seconds)	Iterations	Final Active Set	Initialization Time (seconds)	Total Solution Time (seconds)
30	10000	16	182	5.44	17.01	16	173	0.22	15.96
30	10000	60	285	5.37	133.79	10	234	0.22	15.79
30	10000	41	268	5.38	86.59	12	209	0.22	19.26
Geometric Mean		29.96	237.36	5.38	46.39	15.23	196.9	0.22	19.26
30	20000	21	226	10.84	33.26	23	173	0.44	25.76
30	20000	34	280	10.82	61.29	22	182	0.43	25.56
30	20000	22	235	10.78	37.89	18	225	0.44	23.46
30	20000	64	255	10.81	131.28	19	202	0.44	25.06
30	20000	68	303	10.8	180	16	209	0.44	28.37
30	20000	26	264	10.76	49.61	18	213	0.43	26.21
30	20000	19	253	10.82	37.57	10	230	0.45	16.91
30	20000	22	283	10.77	49.21	12	219	0.44	21.5
30	20000	25	306	10.76	65.59	13	261	0.44	24.95
30	20000	20	233	10.93	34.73	16	206	0.43	18
Geometric Mean		28.64	262.42	10.81	57.3	16.20	210.7	0.438	23.29
30	30000	27	188	16.59	51.16	16	183	0.7	20.46
30	30000	54	331	15.98	181.31	14	229	0.66	30.53
30	30000	17	160	16.12	31.75	14	172	0.64	16.41
30	30000	48	208	16.05	79.31	20	170	0.63	20.39
30	30000	37	266	16.26	95.63	17	254	0.64	34.34
30	30000	21	209	16.13	38.75	16	194	0.66	19.87
30	30000	132	381	16.16	566.81	15	244	0.65	36.12
30	30000	24	232	16.19	53.05	19	197	0.64	26.69
30	30000	24	280	16.16	58.74	13	256	0.67	28.44
30	30000	22	245	16.13	47.18	21	222	0.63	36.61
Geometric Mean		33.03	242.32	16.18	76.88	16.30	209.8	0.652	26.04

The table indicates that SCI dominates FWI in terms of Total Solution Times, becoming more advantageous for larger problem dimensions. This is probably due to the fact that FWI requires roughly mn^3 operations as opposed to mn^2 for SCI, but also it appears from the numbers in Table 3 that the active set generated by FWI is just not as good, as evidence by the fact that if the initialization times are subtracted from the total times then SCI still wins by a large margin, especially on the larger problems.

The Total Solution Times reported in Table 3 for the largest problems ($n = 30$, $m = 30000$) clearly indicate that the DRN algorithm coupled with a suitable active-set strategy solves these problems to a high degree of accuracy ($\epsilon_1 = \epsilon_2 = 10^{-7}$) in under 30 seconds on a personal computer.

7 Some Unsuccessful Strategies

In developing the DRN algorithm and the active-set strategies discussed in Section 5, we tested and discarded a wide variety of computational strategies in favor of more efficient

methods. Some of these strategies were mentioned in Section 5; here is a short list of other strategies that were not previously discussed.

- Expanding the Active Set

  The strategy for expanding the active set was described in detail in Subsection 5.3.1, where we described how points are chosen to be added to the active set based on the idea of having the new points be far from the center c_k as measured in the ellpsoidal distance d_i for each point a_i (see Subsection 5.3.1 for details) and also spread out in all directions around the ellipsoid. This was originally accomplished by examining points one by one in descending order of d_i, and then including a_j into $\Delta\mathcal{S}$ if $(a_j - c_k)^T Q_k (a_i - c_k) < 0$ for all $i \in \Delta\mathcal{S}$. However, computational testing revealed that this resulted in too few points being added at each outer iteration.

- Hybrid of SCI and FWI

  We tested several hybrids of SCI and FWI. In one approach, SCI was used to obtain $\frac{m}{3}$ points, and then these points were used to initialize the Frank-Wolfe algorithm, which would then be run for $l = n + 1$ iterations to then produce the initial active set $\mathcal{S}_0$ and u_0^0. The motivation for this strategy was that this would essentially cut the computation time of FWI by one third. Another idea that we tested was to start the FWI from a point u_0 whose i^{th} component was proportional to the distance from a_i to the sample mean $\bar{a} := \frac{1}{m} Ae$. Neither of these approaches proved to be very effective.

- Frank-Wolfe algorithm for Solution of RD

  Khachiyan [8] provides the best known complexity bound for the solving the minimum volume covering ellipsoid problem, and his method can be interpreted as the Frank-Wolfe algorithm applied to the formulation RD. However, as one might expect, this algorithm is not effective in practice.

8 Concluding Remarks

Algorithms and associated software for conic formulations of convex optimization problems that use primal-dual interior-point methods are intended for general convex problems presented in such conic format. While these algorithms generally perform well in practice, they are not designed to be able to consider any special structure of certain classes of problems, such as the minimum volume covering ellipsoid problem. Herein, we have presented the DRN algorithm for solving the minimum covering ellipsoid problem,

which is itself an interior-point type algorithm, and which is designed around the optimality conditions of the problem augmented with a logarithmic barrier term, although it does not quite fall into the existing interior point algorithm theoretical framework of self-concordant functions. We have shown that this algorithm performs very well for problems of moderate size. When the number of points to be covered is large, we show how the DRN algorithm can be used with an active-set strategy (where the active-set strategy is also designed specifically for the minimum volume covering ellipsoid problem), and we report computational results on large problems which validate the efficiency of these approaches.

From a practical point of view, most applications of the minimum volume ellipsoid are based on the ideal situation in which there are no outliers in the data. To make the minimum volume ellipsoid problem more amenable in the presence of outliers, it is necessary to explore problem formulations that allow points that lie outside of the ellipsoid, such as in the following problem formulation which penalizes such points:

$$\begin{array}{lll} \text{(MVCEP)} & \min_{M,z,\xi} & -\ln\det M + Pe^T\xi \\ & \text{s.t.} & (Ma_i - z)^T (Ma_i - z) \leq 1 + \xi_i \ , \ i = 1, \ldots, m \\ & & \xi \geq 0 \\ & & M \succ 0 \ , \end{array}$$

in which P is a user-specified penalizing coefficient. Formulation (MVCEP) could also be solved by a slight modification of the DRN algorithm, with the active-set strategy if need be. This formulation has the potential of identifying outliers in the data, which has been an important focus in data mining, see [10]. However, from the point of view of determining a "robust" minimum volume ellipsoid, MVCEP still has the drawback that the shape of the optimal ellipsoid is potentially determined in part by points that lie outside of the optimal ellipsoid. Future work in this area could include developing formulations and solution methods for this problem that include non-convex penalizing terms such as $P \sum_{i=1}^{m} \text{sign}\,(\xi_i)$.

9 The DRN direction is not a Newton direction of a self- concordant function

In this section we show that the (Newton) direction produced by the DRN algorithm is not the Newton direction of a self-concordant function. However, it is the Newton direction of a function that is almost self-concordant.

For a given $u > 0$ and $t > 0$, the DRN direction is given by the formula:

$$\Delta u \;=\; \left(\nabla_u h(u) - U^{-1}T\right)^{-1}\left(e - \theta U^{-1}e - h(u)\right)\;, \tag{26}$$

see (22).

Let $D_f := \left\{u \in \mathbb{R}^m \;\mid\; \left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right) \succ 0\right\}$, and for $u \in D_f$ define the function:

$$f(u) := -\frac{1}{2}\ln\det\left(AUA^T - \frac{Auu^TA^T}{e^Tu}\right) \tag{27}$$

and recall the definition of $h(u)$ in (20).

Proposition 9 *Suppose that $u \in D_f$. Then $\nabla f(u) = -h(u)$.*

Proof: This follows by direct but tedious application of the chain rule. ∎

Our analysis relies on the non-self-concordance of $f(u)$, which is stated in the next proposition, and whose proof appears at the end of this section.

Proposition 10 *$f(u)$ is not a self-concordant function on D_f.* ∎

Now consider the following optimization problem, which is equivalent to the program D^1 with a logarithmic barrier term added:

$$\begin{array}{lll} (\mathrm{D}^1_\theta) & \min_u & f(u) + e^Tu - \theta\sum_{i=1}^m \ln(u_i) \\ & \text{s.t.} & u > 0\;. \end{array} \tag{28}$$

At any point $u > 0$, the Newton direction for D^1_θ is given by:

$$\tilde{\Delta}u \;=\; \left(\nabla^2 f(u) + \theta U^{-2}\right)^{-1}\left(-\nabla f(u) - e + \theta U^{-1}e\right)\;, \tag{29}$$

and notice from Proposition 10 that the objective function of D^1_θ is not a self-concordant function for $\theta > 0$ and sufficiently small. The following proposition shows that when u, t satisfy $Ut = \theta e$, then the directions Δu of (26) and $\tilde{\Delta}u$ of (29) are the same.

Proposition 11 *Suppose that $u, t > 0$ and that $Ut = \theta e$. Then $\Delta u = \tilde{\Delta}u$.*

Proof: From Proposition 9 we have $\nabla f(u) = -h(u)$ and so $\nabla^2 f(u) = -\nabla_u h(u)$. Substituting these equalities and the hypothesis of this proposition into (29) yields:

$$\tilde{\Delta} u = \left(-\nabla_u h(u) + U^{-1}T\right)^{-1} \left(h(u) - e + \theta U^{-1} e\right) = \Delta u \ . \blacksquare$$

From Proposition 11, we see that at points (u, t) satisfying $Ut = \theta e$, the DRN direction is exactly the Newton direction of the non-self-concordant objective function of D^1_θ for $\theta > 0$ and sufficiently small.

Notice that if the DRN direction were instead derived as a "dual only" direction, then it would correspond to the Newton direction of problem D^1_θ for all $u > 0$, and so would correspond to the Newton direction of a non-self-concordant function for all $u > 0$. To see this, let us re-write equations (19) as

$$h(u) + \theta U^{-1} e = e, u > 0 \ , \tag{30}$$

and the Newton direction $\bar{\Delta} u$ at a point $u > 0$ for (30) is:

$$\bar{\Delta} u = \left(\nabla_u h(u) - \theta U^{-2}\right)^{-1} \left(e - h(u) - \theta U^{-1} e\right) \ .$$

It then follows from Proposition 9 that $\bar{\Delta} u = \tilde{\Delta} u$, and so the "dual only" version of the DRN algorithm is the Newton direction of a non-self-concordant function for $\theta > 0$ and sufficiently small.

Finally, we point out that while $f(u)$ is not a self-concordant function, the function $f(u) - \frac{1}{2}\ln(e^T u) = -\frac{1}{2}\ln\det\left[AUA^T - Auu^T A^T\right]$, which is known to be a self-concordant function, see [11], and so $f(u)$ is very closely related to a self-concordant function.

Proof of Proposition 10: For each $u \in D_f$ and any direction d, define the univariate function $f_{u,d}(\alpha) := f(u + \alpha d)$. Then $f(\cdot)$ is self-concordant or not depending on whether the quantity

$$\frac{|f'''_{u,d}(0)|}{\left(f''_{u,d}(0)\right)^{\frac{3}{2}}}$$

can be bounded independent of u and d, see [11]. Here we will show that this quantity cannot be bounded, thereby demonstrating that $f(\cdot)$ is not self-concordant. We can write

$$\begin{aligned} f_{u,d}(\alpha) &= -\tfrac{1}{2}\ln\det\begin{pmatrix} A(U + \alpha D)A^T & A(u + \alpha d) \\ (u + \alpha d)^T A^T & e^T(u + \alpha d) \end{pmatrix} + \tfrac{1}{2}\ln\left(e^T(u + \alpha d)\right) \\ &= -\tfrac{1}{2}\ln\det\left(M + \alpha N\right) + \tfrac{1}{2}\ln\left(e^T u + \alpha e^T d\right) \end{aligned}$$

where

$$M = \begin{bmatrix} AUA^T & Au \\ u^TA^T & e^Tu \end{bmatrix} \quad \text{and} \quad N = \begin{bmatrix} ADA^T & Ad \\ d^TA^T & e^Td \end{bmatrix} .$$

It then follows from the rules of differentiation that

$$\begin{aligned} f'_{u,d}(0) &= -\tfrac{1}{2}\mathrm{Tr}\,(M^{-1}N) + \tfrac{e^Td}{e^Tu} \\ f''_{u,d}(0) &= \tfrac{1}{2}\mathrm{Tr}\,(M^{-1}NM^{-1}N) - \left(\tfrac{e^Td}{e^Tu}\right)^2 \\ f'''_{u,d}(0) &= -\left(\mathrm{Tr}\,(M^{-1}NM^{-1}NM^{-1}N) - \left(\tfrac{e^Td}{e^Tu}\right)^3\right) \end{aligned}$$

where "Tr(B)" denotes the trace of a matrix B.

Now let

$$A = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \end{bmatrix} , u = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix}^T , \text{and} \quad d = \begin{pmatrix} 0 & 0 & 0 & \delta & 1-\delta \end{pmatrix}^T$$

for a given scalar δ. Then

$$M = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 5 \end{bmatrix} \quad N = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \delta & \delta \\ 0 & \delta & 1 \end{bmatrix} \quad M^{-1}N = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{2}\delta & \frac{1}{2}\delta \\ 0 & \frac{1}{5}\delta & \frac{1}{5} \end{bmatrix}$$

and $\frac{e^Td}{e^Tu} = \frac{1}{5}$, and direct substiution of these equalities yields $f''_{u,d}(0) = \frac{9}{40}\delta^2$ and $f'''_{u,d}(0) = \frac{11}{40}\delta^3 + \frac{3}{50}\delta^2$. Therefore for $\delta > 0$ we have:

$$\frac{|f'''_{u,d}(0)|}{\left(f''_{u,d}(0)\right)^{\frac{3}{2}}} = \frac{\frac{11}{40}\delta^3 + \frac{3}{50}\delta^2}{\left(\frac{9}{40}\right)^{\frac{3}{2}}\delta^3}$$

which goes to $+\infty$ as $\delta \downarrow 0$. ∎

References

[1] E. Barnes. An algorithm for separating patterns by ellipsoids. *IBM J. Res. Develop.*, 26(6):759–764, 1982.

[2] C. Croux, G. Haesbroeck, and P.J. Rousseeuw. Location adjustment for the minimum volume ellipsoid estimator. *Statistics and Computing*, to appear.

[3] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

[4] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, 1998.

[5] R. Horn and C. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

[6] F. John. Extreme problems with inequalities as subsidiary conditions. *Studies and Essays presented to R. Courant on his 60th Birthday, January 8, 1948*, pages 187–204, 1948.

[7] T. Kailath. *Linear Systems*. Pretice-Hall, New Jersey, 1980.

[8] L. Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21(2):307–320, 1996.

[9] L. Khachiyan and M. Todd. On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming*, 61:137–159, 1993.

[10] E. Knorr, R. Ng, and R. Zamar. Robust space transformations for distance-based operations. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 126–135, 2001.

[11] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia, 1993.

[12] K. Toh. Primal-dual path-following algorithms for determinant maximization problems with linear matrix inequalities. *Computational Optimization and Applications*, 14:309–330, 1999.

[13] K. Toh, M. Todd, and R. Tütüncü. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, to appear.

[14] K. Toh, M. Todd, and R. Tütüncü. Sdpt3 — a matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

[15] L. Vandenberghe, S. Boyd, and S. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533, 1998.

[16] Y. Zhang. An interior-point algorithm for the maximum-volume ellipsoid problem. Department of Computational and Applied Mathematics, Rice University, Technical Report TR98-15, 1998.

[17] Y. Zhang and L. Gao. On numerical solution of the maximum volume ellipsoid problem. Department of Computational and Applied Mathematics, Rice University, Technical Report TR01-15, 2001.